

# 통계 분석

2019-2020년 겨울 계절 학기  
강봉주

# 벡터와 행렬

# 벡터와 행렬

## [벡터와 행렬에 대한 이해가 필요한 이유]

- 행렬: 정형화된 데이터 형식
- 벡터: 데이터의 행 또는 컬럼(관측값 벡터, 변수 벡터)
- 머신러닝, 딥러닝의 모델 식:

$$y = X\beta + \epsilon,$$

$$h_{w,b} = w^T x + b,$$

$$z^{(l+1)} = W^{(l+1)} a^{(l)} + b^{(l+1)},$$

$$\sigma(z),$$

$$K \star I,$$

$$h_t = \sigma_h(W_h h_{t-1} + W_x x_t + b_h)$$

■ ...

# 파이썬으로 실무에 바로 적용하는 머신 러닝

강봉주 지음



# 벡터

# 벡터와 행렬

## [벡터의 표현]

- 벡터(vector)는 순서가 있는 숫자들의 목록이다. 즉 일종의 배열이다. 표현은 대괄호나 괄호를 통하여 표현한다.

$$\begin{bmatrix} 0.61 \\ 0.93 \\ 0.24 \\ 0.27 \end{bmatrix}, \quad \begin{pmatrix} 0.61 \\ 0.93 \\ 0.24 \\ 0.27 \end{pmatrix}$$

$$(0.74 \quad 0.75 \quad 0.93 \quad 0.46)$$

# 벡터와 행렬

## [벡터의 표현]

- 벡터(vector)는 순서가 있는 숫자들의 목록이다. 즉 일종의 배열이다. 표현은 대괄호나 괄호를 통하여 표현한다.

```
In      # 벡터의 생성
        v = np.array([0.61, 0.93, 0.24, 0.27])
        v
Out      array([0.61, 0.93, 0.24, 0.27])
```

# 벡터와 행렬

## [벡터의 원소]

- 벡터의 원소(element, entry, coefficient, component)는 배열의 값들을 의미한다.

```
In      # 벡터의 원소
        v = np.array([0.61, 0.93, 0.24, 0.27])
        v[0]
Out      0.61
```



# 벡터와 행렬

## [벡터의 크기]

- 벡터의 크기(size, dimension, length)는 벡터의 원소의 개수를 의미한다.

```
In      # 벡터의 크기
        v = np.array([0.61, 0.93, 0.24, 0.27])
        v.shape
Out      (4,)
```

# 벡터와 행렬

## [부분 벡터]

- 벡터의 부분벡터(sub vector)는 쌍점(colon) 기호를 사용하여 표현한다.

$$a_{r:s} = \begin{pmatrix} a_r \\ a_{r+1} \\ \dots \\ a_s \end{pmatrix}$$

# 벡터와 행렬

## [부분 벡터]

- 벡터의 부분벡터(sub vector)는 쌍점(colon) 기호를 사용하여 표현한다.

```
In      # 부분 벡터의 생성
        v = np.array([0.61, 0.93, 0.24, 0.27])
        v_sub = v[0:2]
        v_sub
Out      array([0.61, 0.93])
```

# 벡터와 행렬

## [특별한 벡터]

- 모든 원소가 0인 벡터를 0벡터

- $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

```
In      # 영 벡터
        size = 3
        zeros = np.zeros(shape=(size,))
        zeros
Out      array([0., 0., 0.] )
```

# 벡터와 행렬

## [특별한 벡터]

- 모든 원소가 1인 벡터

- $1_3 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

---

```
In      # 1 벡터
        size = 3
        one = np.ones(shape=(size, ))
        one
Out      array([1., 1., 1.]
```

# 벡터와 행렬

## [특별한 벡터]

- 하나의 원소만 1이고 나머지는 모두 0인 단위(unit)벡터

$$e_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, e_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, e_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

# 벡터와 행렬

## [특별한 벡터]

- 하나의 원소만 1이고 나머지는 모두 0인 단위(unit)벡터

```
In      # 단위 벡터
        size = 3
        I = np.identity(n=size)
        I
```

```
Out      array([[1., 0., 0.],
                [0., 1., 0.],
                [0., 0., 1.]])
```

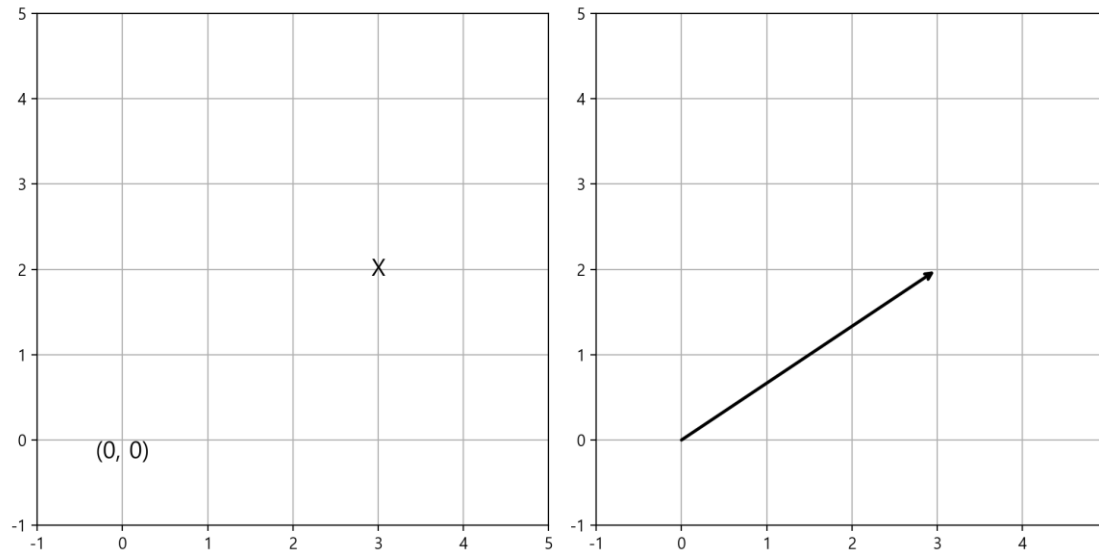
```
In      # 하나의 단위 벡터의 예
        e2 = I[:,1]
        e2
```

```
Out      array([0., 1., 0.])
```

# 벡터와 행렬

## [벡터의 기하학적인 의미]

- $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$  벡터는 좌표 상의 1개의 점으로 표현되거나, 오른쪽 그림과 같이  $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$  점을  $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ 로 옮기는 위치 이동 (displacement) 을 표현하기도 한다.

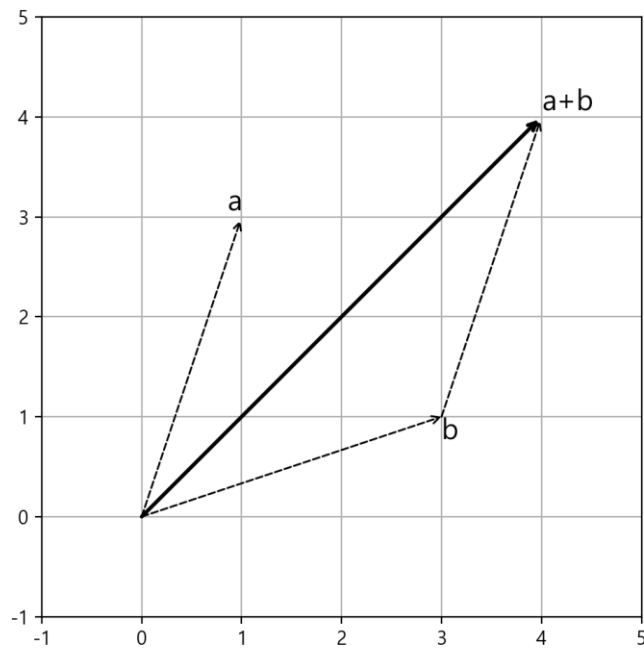




# 벡터와 행렬

## [벡터 덧셈]

- 대응되는 각 원소의 합으로 정의
- 2개의 변으로 구성된 평행사변형의 대각선 벡터



$$\begin{pmatrix} 1 \\ 3 \end{pmatrix} + \begin{pmatrix} 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \end{pmatrix}$$

# 벡터와 행렬

## [벡터 덧셈]

- 대응되는 각 원소의 합으로 정의
- 2개의 변으로 구성된 평행사변형의 대각선 벡터

```
In      # 벡터의 덧셈
        a = np.array([1, 3])
        b = np.array([3, 1])
        a + b
Out      array([1., 1., 1.] )
```

# 벡터와 행렬

## [벡터 덧셈]

- 교환 법칙:  $a + b = b + a$
- 결합 법칙:  $(a + b) + d = a + (b + d)$

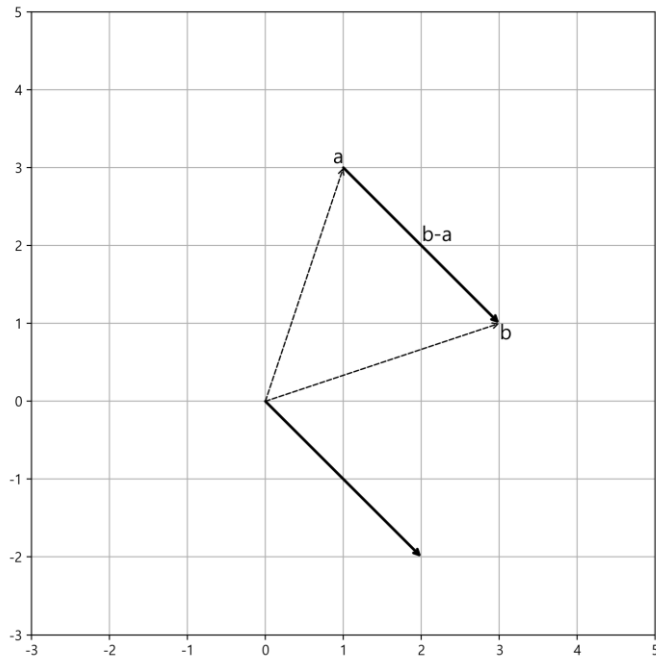
```
In      # 교환 법칙
        a + b == b + a
Out      array([ True,  True])
```

```
In      # 결합 법칙
        d = np.array([1, 2])
        (a + b) + d == a + (b + d)
Out      array([ True,  True])
```

# 벡터와 행렬

## [벡터 뺄셈]

- 대응되는 각 원소의 차(difference)로 정의
- 빼는 벡터를 시작점으로 연결한 벡터



$$\begin{pmatrix} 3 \\ 1 \end{pmatrix} - \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \end{pmatrix}$$

# 벡터와 행렬

## [벡터 뺄셈]

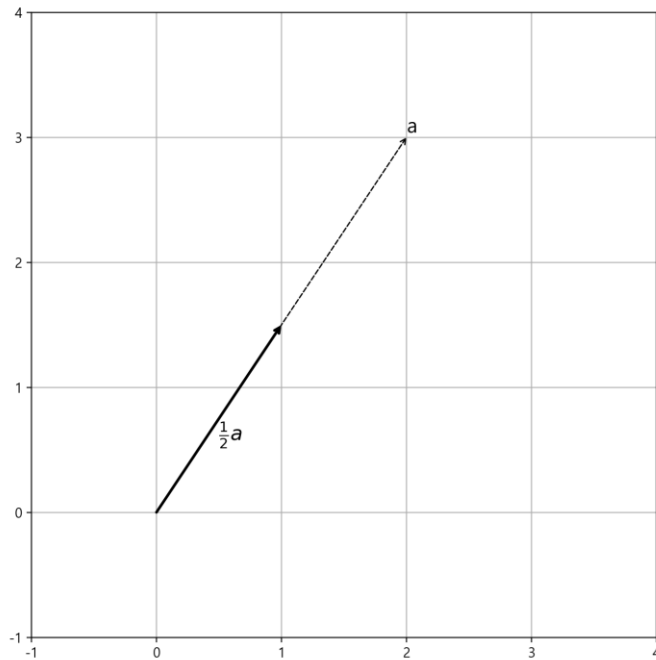
- 대응되는 각 원소의 차(difference)로 정의
- 빼는 벡터를 시작점으로 연결한 벡터

```
In      # 벡터의 뺄셈
        a = np.array([3, 1])
        b = np.array([1, 3])
        b-a
Out      array([-2,  2])
```

# 벡터와 행렬

## [스칼라 곱]

- 벡터에 대한 실수 곱(스칼라 곱: scalar multiplication)은 그 벡터의 모든 원소를 실수 만큼 곱한 것



$$\frac{1}{2} \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1.5 \end{pmatrix}$$

# 벡터와 행렬

## [스칼라 곱]

- 벡터에 대한 실수 곱(스칼라 곱: scalar multiplication)은 그 벡터의 모든 원소를 실수 만큼 곱한 것

```
In      # 스칼라-벡터 곱
        alpha = 1/2
        a = np.array([2, 3])
        alpha * a
Out      array([1. , 1.5])
```

# 벡터와 행렬

## [스칼라 곱]

- 스칼라 곱의 성질
- 교환 법칙:  $\beta a = a\beta$
- 분배 법칙:  $(\alpha + \beta)a = \alpha a + \beta a$

---

```
In      # 교환 법칙
        alpha * a == a * alpha
Out      array([ True,  True])
```

---

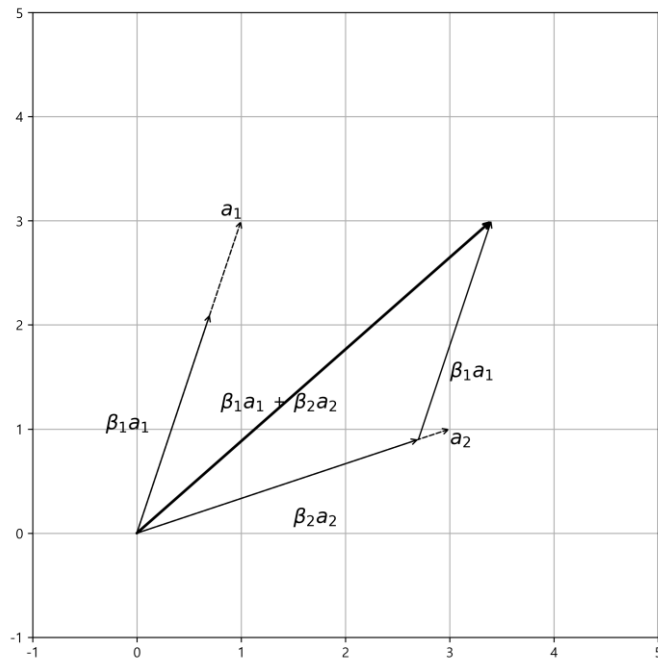
```
In      # 분배 법칙
        beta = 0.7
        (alpha + beta) * a == alpha * a + beta * a
Out      array([ True,  True])
```



# 벡터와 행렬

## [선형 결합]

- 벡터  $a_1, \dots, a_k$ 에 대한 선형결합(linear combination)
- $\beta_1 a_1 + \dots + \beta_k a_k$



$$0.7 \begin{bmatrix} 1 \\ 3 \end{bmatrix} + 0.9 \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

# 벡터와 행렬

## [내적]

- 내적(inner product) 또는 점적(dot product)
- $a^T b = \sum_i a_i b_i$

$$\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}^T \begin{pmatrix} 1 \\ -2 \\ 4 \end{pmatrix} = \text{sum} \begin{pmatrix} -1 \\ -4 \\ 12 \end{pmatrix} = 7$$

# 벡터와 행렬

## [내적]

- 내적(inner product) 또는 점적(dot product)
- $a^T b = \sum_i a_i b_i$

```
In      # 내적 계산
        a = np.array([-1, 2, 3])
        b = np.array([1, -2, 4])
```

```
        # 원소별 곱 벡터
        a * b
```

```
Out      array([-1, -4, 12])
```

```
In      # 원소별 곱 벡터 합
        np.sum(a*b)
```

```
Out      7
```

# 벡터와 행렬

## [내적]

- 내적(inner product) 또는 점적(dot product)
- $a^T b = \sum_i a_i b_i$

In	# 함수 이용 np.inner(a, b)
Out	7

In	np.dot(a, b)
Out	7

# 벡터와 행렬

## [내적]

- 내적의 성질

1)  $a^T b = b^T a$

2)  $(\alpha a)^T b = \alpha(a^T b)$

3)  $(a + b)^T c = a^T c + b^T c$

```
In      # 내적의 성질
        a = np.array([-1, 2, 3])
        b = np.array([1, -2, 4])

        # 교환 법칙
        np.dot(a, b) == np.dot(b, a)
Out      True
```

# 벡터와 행렬

## [내적]

- 내적의 성질

1)  $a^T b = b^T a$

2)  $(\alpha a)^T b = \alpha(a^T b)$

3)  $(a + b)^T c = a^T c + b^T c$

```
In      # 결합 법칙
        alpha= 0.5
        np.dot(alpha * a, b) == alpha * np.dot(a, b)
Out      True
```

# 벡터와 행렬

## [내적]

- 내적의 성질

1)  $a^T b = b^T a$

2)  $(\alpha a)^T b = \alpha(a^T b)$

3)  $(a + b)^T c = a^T c + b^T c$

```
In      # 분배 법칙
        c = np.array([1, 2, 3])
        np.dot(a+b, c) == np.dot(a,c) + np.dot(b, c)
Out      True
```

# 벡터와 행렬

## [내적]

- 내적의 활용

$$1) 1^T a = a_1 + \cdots + a_n = \sum_{i=1}^n a_i$$

$$2) \frac{1}{n} 1^T a = \frac{1}{n} (a_1 + \cdots + a_n) = \bar{a}$$

$$3) a^T a = a_1^2 + \cdots + a_n^2$$

---

```
In      # 내적의 활용
        a = np.array([1, 2, 3])
        size = len(a)
        ones = np.ones(shape=(size,))
```

```
        # 합의 표현
        np.dot(ones, a)
```

```
Out     6.0
```



# 벡터와 행렬

## [내적]

- 내적의 활용

$$1) 1^T a = a_1 + \cdots + a_n = \sum_{i=1}^n a_i$$

$$2) \frac{1}{n} 1^T a = \frac{1}{n} (a_1 + \cdots + a_n) = \bar{a}$$

$$3) a^T a = a_1^2 + \cdots + a_n^2$$

In	# 평균의 표현 np.dot(ones, a) / len(a)
Out	2.0

# 벡터와 행렬

## [내적]

- 내적의 활용

$$1) 1^T a = a_1 + \cdots + a_n = \sum_{i=1}^n a_i$$

$$2) \frac{1}{n} 1^T a = \frac{1}{n} (a_1 + \cdots + a_n) = \bar{a}$$

$$3) a^T a = a_1^2 + \cdots + a_n^2$$

In	# 제공합의 표현 np.dot(a, a)
Out	14

# 벡터와 행렬

## [벡터 노름]

- 유클리디안 노름(Euclidean norm)
- $\|a\| = \sqrt{a^T a} = \sqrt{a_1^2 + \dots + a_n^2}$

$$\left\| \begin{bmatrix} 2 \\ 3 \end{bmatrix} \right\| = \sqrt{\begin{bmatrix} 2 \\ 3 \end{bmatrix}^T \begin{bmatrix} 2 \\ 3 \end{bmatrix}} = \sqrt{\text{sum} \begin{bmatrix} 4 \\ 9 \end{bmatrix}} = \sqrt{13} = 3.606$$

# 벡터와 행렬

## [벡터 노름]

- 유클리디안 노름(Euclidean norm)
- $\|a\| = \sqrt{a^T a} = \sqrt{a_1^2 + \dots + a_n^2}$

```
In      # 필요한 패키지
        from scipy import linalg as la

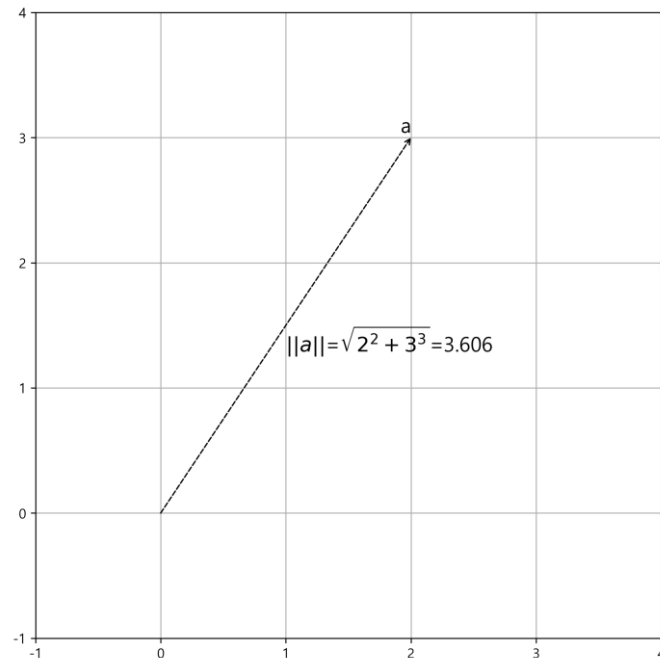
        # 벡터 노름의 계산
        a = np.array([2, 3])
        la.norm(a)

Out      3.606
```

# 벡터와 행렬

## [벡터 노름]

- 유클리디안 노름(Euclidean norm)의 의미
- $\|a\| = \sqrt{a^T a} = \sqrt{a_1^2 + \dots + a_n^2}$



# 벡터와 행렬

## [벡터 노름]

- 노름의 성질

1)  $\|\beta a\| = |\beta| \|a\|$

2)  $\|a + b\| \leq \|a\| + \|b\|$

3)  $\|a\| \geq 0$

4)  $\|a\| = 0 \Rightarrow a = 0$

---

```
In      # 벡터 노름의 성질
        a = np.array([-1, 2, 3])
        b = np.array([1, -2, 4])

        # 스칼라 곱
        beta = 0.5
        la.norm(beta * a) == np.abs(beta) * la.norm(a)

Out     True
```

# 벡터와 행렬

## [벡터 노름]

- 노름의 성질

1)  $\|\beta a\| = |\beta| \|a\|$

2)  $\|a + b\| \leq \|a\| + \|b\|$

3)  $\|a\| \geq 0$

4)  $\|a\| = 0 \Rightarrow a = 0$

In	# 삼각형의 한 변의 길이는 다른 두변의 길이의 합보다 작다 la.norm(a + b) <= la.norm(a) + la.norm(b)
Out	True

# 벡터와 행렬

## [벡터 노름]

- 노름의 성질

1)  $\|\beta a\| = |\beta| \|a\|$

2)  $\|a + b\| \leq \|a\| + \|b\|$

3)  $\|a\| \geq 0$

4)  $\|a\| = 0 \Rightarrow a = 0$

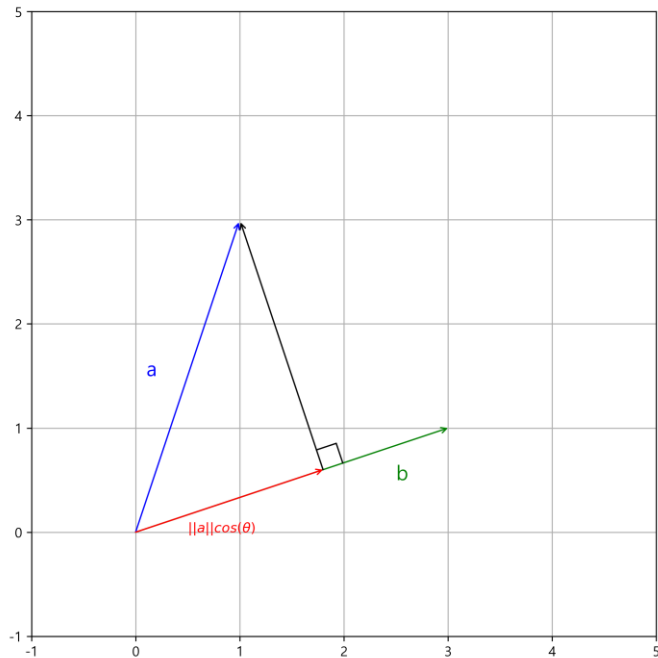
In	# 길이는 0보다 크거나 작다 la.norm(a) >= 0
Out	True



# 벡터와 행렬

## [내적과 노름]

- $a^T b = \|a\| \|b\| \cos(\theta)$
- $\cos(\theta)=1$  의 의미: 2개의 벡터가 일치 할 때



$$\cos(\theta) = \frac{a^T b}{\|a\| \|b\|}$$

# 벡터와 행렬

## [벡터 선형 종속]

- $n$  차원 벡터  $a_1, \dots, a_p$ 가 선형 종속(linear dependence)
- $\beta_1 a_1 + \dots, \beta_p a_p = 0$  가 0 벡터가 아닌 어떤  $\beta$ 들에 의하여 충족

$$a_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, a_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, a_3 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \quad \beta_1 = 1, \beta_2 = 1, \beta_3 = -1$$

# 벡터와 행렬

## [벡터 선형 종속]

- $n$  차원 벡터  $a_1, \dots, a_p$ 가 선형 독립(linear independence)
- $\beta_1 a_1 + \dots, \beta_p a_p = 0$  이면  $\beta = 0$ , 즉,  $\beta_i = 0, \forall i$

$$a_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, a_2 = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}, a_3 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \quad \beta_1 = 0, \beta_2 = 0, \beta_3 = 0$$

# 벡터와 행렬

## [기저]

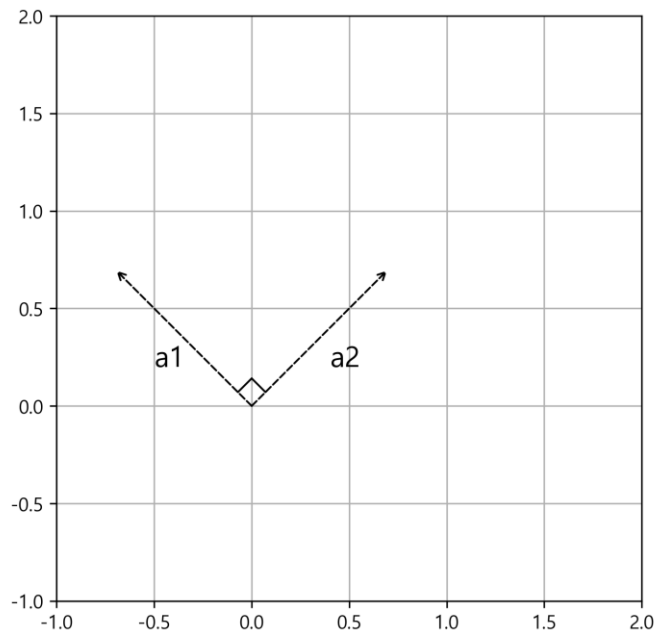
- 기저(base):  $n$  차원에서  $n$ 개의 선형 독립인 벡터
- 2차원 벡터에서는 선형 독립인 벡터의 수는 최대 2개
- $n$ 차원 벡터  $a_1, \dots, a_n$ 가 기저이면, 임의의  $n$ 차원 벡터  $x$ 는

$$x = \beta_1 a_1 + \dots + \beta_n a_n$$

# 벡터와 행렬

## [직교정규 벡터]

- $a_i \perp a_j$
- $a_i^T a_j = 0, \|a_i\| = 1, \forall i$



$$a_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix}, a_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

# 벡터와 행렬

## [직교정규 벡터]

- $a_i \perp a_j$
- $a_i^T a_j = 0, \|a_i\| = 1, \forall i$

$$a_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, a_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, a_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

# 벡터와 행렬

## [직교정규 벡터]

- $a_i \perp a_j$
- $a_i^T a_j = 0, \|a_i\| = 1, \forall i$

```
In      # 직교정규 벡터 확인
        a1 = np.array([0, 0, 1]).reshape(-1, 1)
        a2 = 1/np.sqrt(2) * np.array([1, 1, 0]).reshape(-1, 1)
        a3 = 1/np.sqrt(2) * np.array([1, -1, 0]).reshape(-1, 1)

        # 노름=1, 열들의 곱=0
        A = np.concatenate((a1, a2, a3), axis=1)
        np.round(A.T @ A, 3)
Out      array([[ 1.,  0.,  0.],
                [ 0.,  1., -0.],
                [ 0., -0.,  1.]])
```

# 행렬



# 벡터와 행렬

## [행렬 표현]

- 행렬(matrix)은 행(row)과 열(column)로 구분된 직사각형 배열

$$A = \begin{bmatrix} -1.09 & 1 & 0.28 & -1.51 \\ -0.58 & 1.65 & -2.43 & -0.43 \\ 1.27 & -0.87 & -0.68 & -0.09 \\ 1.49 & -0.64 & -0.44 & -0.43 \end{bmatrix}$$

$$B = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

# 벡터와 행렬

## [행렬 표현]

- 행렬(matrix)은 행(row)과 열(column)로 구분된 직사각형 배열

---

In	<pre># 정의: 2차원 배열 A = np.array(np.random.RandomState(123).normal(size=16)).reshape(4, 4) A.round(3)</pre>
Out	<pre>array([[ -1.086,   0.997,   0.283,  -1.506],        [ -0.579,   1.651,  -2.427,  -0.429],        [  1.266,  -0.867,  -0.679,  -0.095],        [  1.491,  -0.639,  -0.444,  -0.434]])</pre>

# 벡터와 행렬

## [행렬 표현]

- 행렬의 크기

- $A_{n \times p}$

- $A_{3 \times 4} = \begin{bmatrix} 3 & 3 & 7 & 2 \\ 4 & 11 & 10 & 7 \\ 2 & 1 & 2 & 10 \end{bmatrix}$

---

In	# 행렬의 크기 A = np.random.RandomState(123).randint(1, 12, size=12).reshape(3, 4) A.shape
Out	(3, 4)

# 벡터와 행렬

## [행렬 표현]

- $A_{3 \times 4} = \begin{bmatrix} 3 & 3 & 7 & 2 \\ 4 & 11 & 10 & 7 \\ 2 & 1 & 2 & 10 \end{bmatrix}$

- 행 벡터:  $[3 \quad 3 \quad 7 \quad 2]$

- 열 벡터:  $\begin{bmatrix} 3 \\ 4 \\ 2 \end{bmatrix}$

# 벡터와 행렬

## [행렬 표현]

- $A_{3 \times 4} = \begin{bmatrix} 3 & 3 & 7 & 2 \\ 4 & 11 & 10 & 7 \\ 2 & 1 & 2 & 10 \end{bmatrix}$
- 행 벡터:  $[3 \quad 3 \quad 7 \quad 2]$
- 열 벡터:  $\begin{bmatrix} 3 \\ 4 \\ 2 \end{bmatrix}$

In	# 열 벡터 <code>A[:,0]</code>
Out	<code>array([3, 4, 2])</code>

# 벡터와 행렬

## [행렬 표현]

- 행렬의 표기법

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1p} \\ a_{21} & a_{22} & \dots & a_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{np} \end{bmatrix} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1p} \\ a_{21} & a_{22} & \dots & a_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{np} \end{pmatrix} = (a_{ij}) \in \mathbb{R}^{n \times p}$$

원소(element, entry, coefficient)는  $a_{ij}, a_{i,j}, A[i,j], A_{i,j}, A_{ij}$

# 벡터와 행렬

## [행렬 표현]

- 행렬의 표기법

- $$A = \begin{bmatrix} 3 & 3 & 7 & 2 \\ 4 & 11 & 10 & 7 \\ 2 & 1 & 2 & 10 \end{bmatrix}$$

- $a_{11} = 3, a_{24} = 7$

In	# 행렬의 표기법 A[0, 0]
----	----------------------

Out	3
-----	---

In	A[1, 3]
----	---------

Out	7
-----	---

# 벡터와 행렬

## [행렬 표현]

- 행렬 연산: 덧셈
- $A = (a_{ij}), B = (b_{ij})$
- $A + B = (a_{ij} + b_{ij})$
- 대응하는 위치의 원소의 값을 더한 것

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix} + \begin{pmatrix} 3 & 5 & 7 \\ 4 & 6 & 8 \end{pmatrix} = \begin{pmatrix} 4 & 8 & 12 \\ 6 & 10 & 14 \end{pmatrix}$$

In	A + B
Out	array([[ 4,  6,  8], [10, 12, 14]])



# 벡터와 행렬

## [행렬 표현]

- 실수와 행렬 곱
- $A = (a_{ij}), c \in \mathbb{R}$
- $cA = (ca_{ij})$
- $cA = Ac$

$$0.1 \times \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \end{bmatrix}$$

```
In      # 스칼라와 행렬 곱: *  
        c = 0.1  
        c*A
```

```
Out     array([[0.1, 0.2, 0.3],  
              [0.4, 0.5, 0.6]])
```

# 벡터와 행렬

## [행렬 표현]

- 실수와 행렬 곱
- $A = (a_{ij}), c \in \mathbb{R}, cA = (ca_{ij})$
- $cA = Ac$

```
In      # 스칼라와 행렬 곱 성질
        c*A == A*c
Out      array([[ True,  True,  True],
               [ True,  True,  True]])
```

# 벡터와 행렬

## [행렬 표현]

- 행렬의 전치
- $A = (a_{ij})$
- $A^T = (a_{ji})$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

In	# 행렬의 전치 A.T
Out	array([[1, 4], [2, 5], [3, 6]])

# 벡터와 행렬

## [행렬 표현]

- 행렬의 전치 성질

- $A = (a_{ij})$

- $A^T = (a_{ji})$

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

2)  $(cA)^T = cA^T$

```
In      # 행렬 전치의 성질
        A = np.arange(1, 7).reshape(2, 3)
        B = np.arange(3, 9).reshape(2, 3)
```

```
        A.T + B.T == (A + B).T
Out      array([[ True,  True],
               [ True,  True],
               [ True,  True]])
```

# 벡터와 행렬

## [행렬 표현]

- 행렬의 전치 성질

- $A = (a_{ij})$

- $A^T = (a_{ji})$

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

2)  $(cA)^T = cA^T$

```
In      c = 0.1
        (c*A).T == c*A.T
Out      array([[ True,  True],
               [ True,  True],
               [ True,  True]])
```

# 벡터와 행렬

## [행렬 표현]

- 행렬 곱
- $A = (a_{ij}), i = 1, \dots, n, j = 1, \dots, p, B = (b_{kl}), k = 1, \dots, p, l = 1, \dots, m$
- $A_{n \times p}, B_{p \times m}$ : 앞 행렬의 컬럼 개수와 뒷 행렬의 행의 개수는 반드시 일치
- $AB = (c_{il}) = (\sum_{r=1}^p a_{ir}b_{rl})$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}_{2 \times 3} \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}_{3 \times 2} = \begin{bmatrix} 22 & 28 \\ \mathbf{49} & 64 \end{bmatrix}_{2 \times 2}$$

$$c_{2,1} = A_{2,:}^T B_{:,1} = 4 \times 1 + 5 \times 3 + 6 \times 5 = 49$$

# 벡터와 행렬

## [행렬 표현]

- 행렬 곱
- $AB = (c_{il}) = \left(\sum_{r=1}^p a_{ir}b_{rl}\right)$

```
In      # 행렬곱 연산자: @  
        A @ B
```

```
Out      array([[22, 28],  
                [49, 64]])
```

```
In      # 행렬곱 함수  
        np.matmul(A, B)
```

```
Out      array([[22, 28],  
                [49, 64]])
```

# 벡터와 행렬

## [행렬 표현]

- 행렬 곱의 성질
- $AB \neq BA$

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix} \neq \begin{pmatrix} 23 & 34 \\ 31 & 46 \end{pmatrix} = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

---

In      A @ B

Out     array([[19, 22],  
          [43, 50]])

---

In      B @ A

Out     array([[23, 34],  
          [31, 46]])



# 벡터와 행렬

## [행렬 표현]

- 행렬 곱의 전치
- $(AB)^T = B^T A^T$

$$(AB)^T = \begin{pmatrix} 19 & 43 \\ 22 & 50 \end{pmatrix}, B^T A^T = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix} \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} = \begin{pmatrix} 19 & 43 \\ 22 & 50 \end{pmatrix}$$

```
In      # 행렬 곱의 전치  
        (A @ B).T
```

```
Out     array([[19, 43],  
              [22, 50]])
```

```
In      B.T @ A.T
```

```
Out     array([[19, 43],  
              [22, 50]])
```

# 벡터와 행렬

## [행렬 표현]

- 부분 행렬(submatrix)

- $$A_{p:q,r:s} = \begin{pmatrix} A_{p,r} & \cdots & A_{p,s} \\ \vdots & \cdots & \vdots \\ A_{q,r} & \vdots & A_{q,s} \end{pmatrix}$$

$$A_{1:2,2:3} = \begin{bmatrix} 2 & 3 \\ 5 & 6 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

```
In      # 유형 1
        A[0:2, 1:3]
Out     array([[2, 3],
              [5, 6]])
```

# 벡터와 행렬

## [행렬 표현]

- 부분 행렬(submatrix)

- $A_{p:q,r:s} = \begin{pmatrix} A_{p,r} & \dots & A_{p,s} \\ \vdots & \dots & \vdots \\ A_{q,r} & \dots & A_{q,s} \end{pmatrix}$

$$A_{:,2:3} = \begin{bmatrix} 2 & 3 \\ 5 & 6 \\ 8 & 9 \\ 11 & 12 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

```
In      # 유형 2
        A[:, 1:3]
Out      array([[ 2,  3],
               [ 5,  6],
               [ 8,  9],
               [11, 12]])
```

# 벡터와 행렬

## [행렬 표현]

- 부분 행렬(submatrix)

- $$A_{p:q,r:s} = \begin{pmatrix} A_{p,r} & \dots & A_{p,s} \\ \vdots & \dots & \vdots \\ A_{q,r} & \vdots & A_{q,s} \end{pmatrix}$$

$$A_{(1,3),(2,3)} = \begin{bmatrix} 2 & 3 \\ 8 & 9 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

```
In      # 유형 3
        A[[0, 2], :][:, [1, 2]]
Out     array([[2, 3],
              [8, 9]])
```

# 벡터와 행렬

## [행렬 표현]

- 특별한 행렬: 영 행렬

- $$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

---

```
In      # 영 행렬
        size = 3
        np.zeros(shape=(size, size))
Out      array([[0., 0., 0.],
               [0., 0., 0.],
               [0., 0., 0.]])
```

# 벡터와 행렬

## [행렬 표현]

- 특별한 행렬: 항등 행렬

- $I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

```
In      # 항등 행렬
        np.identity(n=size)
Out      array([[1., 0., 0.],
               [0., 1., 0.],
               [0., 0., 1.]])
```

# 벡터와 행렬

## [행렬 표현]

- 특별한 행렬: 대각 행렬(diagonal matrix)
- 대각선 값을 제외한 모든 원소의 값이 0인 행렬

- $D_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$

```
In      # 대각 행렬
        np.diag([1, 2, 3])
Out      array([[1, 0, 0],
               [0, 2, 0],
               [0, 0, 3]])
```

# 벡터와 행렬

## [행렬 표현]

- 특별한 행렬: 삼각 행렬(triangular matrix)
- 대각선을 기준으로 하여 한 쪽이 모든 0인 행렬

- 상삼각행렬: 
$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & 7 \\ 0 & 0 & 5 \end{bmatrix}$$

```
In      # 상삼각 행렬
        np.triu([[1,2,3],[1, 2, 7],[7,8,5]], k=0)
Out      array([[1, 2, 3],
               [0, 2, 7],
               [0, 0, 5]])
```



# 벡터와 행렬

## [행렬 표현]

- 특별한 행렬: 직교 행렬(orthogonal matrix)
- $A^T A = A A^T = I$
- $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

---

In	A.T @ A
Out	array([[1, 0], [0, 1]])

---

---

In	A @ A.T
Out	array([[1, 0], [0, 1]])

---

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- 선형 방정식

$$a_{11}x_1 + \cdots + a_{1p}x_p = b_1$$

■

$$\begin{matrix} & \cdots \\ a_{n1}x_1 + \cdots + a_{np}x_p = b_p \end{matrix}$$

$$x_1 + x_2 = 5$$

■

$$\frac{1}{2}x_1 - x_2 = 2$$

■

$$\begin{bmatrix} 1 & 1 \\ \frac{1}{2} & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

■

$$Ax = b$$

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- QR 분해
- $A_{n \times p} = Q_{n \times n} R_{n \times p}$
- Q는 직교정규행렬, R은 상삼각행렬

$$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 2 & -1 \end{bmatrix} = \begin{bmatrix} -0.89 & -0.45 \\ -0.45 & 0.89 \end{bmatrix} \begin{bmatrix} -1.12 & -0.45 \\ 0 & -1.34 \end{bmatrix} = QR$$

$$Ax = b$$

$$\Rightarrow QRx = b$$

$$\Rightarrow Q^T QRx = Q^T b$$

$$\Rightarrow Rx = Q^T b$$

$$\begin{bmatrix} -1.12 & -0.45 \\ 0 & -1.34 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -0.89 & -0.45 \\ -0.45 & 0.89 \end{bmatrix} \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \begin{bmatrix} -5.37 \\ -0.45 \end{bmatrix}$$

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- QR 분해
- $A_{n \times p} = Q_{n \times p} R_{p \times p}$
- Q는 직교정규행렬, R은 상삼각행렬
- $X^T X = \begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix}, X^T y = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$
- $(X^T X)\beta = X^T y$  인  $\beta = ?$

## 벡터와 행렬

### [선형 방정식과 QR 분해]

- $X^T X = \begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix}, X^T y = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$
- $(X^T X)\beta = X^T y$  인  $\beta = ?$
- $\begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

$$\begin{bmatrix} -1.35 & -1.21 \\ 0 & 1.67 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} -4.46 \\ 0.37 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} -1.35 * \beta_1 - 1.21 * \beta_2 \\ 0 * \beta_1 + 1.67 * \beta_2 \end{bmatrix} = \begin{bmatrix} -4.46 \\ 0.37 \end{bmatrix}$$

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- $X^T X = \begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix}, X^T y = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$
- $(X^T X)\beta = X^T y$  인  $\beta = ?$
- $\begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

```
In      # 선형 방정식과 후방 대입법
        A = np.array([[1, 1], [1/2, -1]])
        XTX = A.T@A
        XTX
Out      array([[1.25, 0.5 ],
               [0.5 , 2.   ]])
```

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- $X^T X = \begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix}, X^T y = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$
- $(X^T X)\beta = X^T y$  인  $\beta = ?$
- $\begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

```
In      # QR 분해
        Q, R = la.qr(XTX)
        R.round(2)
Out     array([[ -1.35, -1.21],
              [  0.   ,  1.67]])
```

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- $X^T X = \begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix}, X^T y = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$
- $(X^T X)\beta = X^T y$  인  $\beta = ?$
- $\begin{bmatrix} 1.25 & 0.5 \\ 0.5 & 2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

```
In      b = np.array([4, 2])  
        np.round(Q.T @ b, 2)  
Out     array([-4.46,  0.37])
```



# 벡터와 행렬

## [선형 방정식과 QR 분해]

- 예제

- 행렬  $A = \begin{bmatrix} 5 & 1 & 5 \\ 3 & 9 & 8 \\ 2 & 1 & 4 \end{bmatrix}$ 에 대하여 QR 분해를 실시하세요.

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- 예제

- 행렬  $A = \begin{bmatrix} 5 & 1 & 5 \\ 3 & 9 & 8 \\ 2 & 1 & 4 \end{bmatrix}$ 에 대하여 QR 분해를 실시하세요.

# 벡터와 행렬

## [역행렬]

- $ax = b \Rightarrow x = \frac{b}{a} = a^{-1} \times b$ : 조건은  $a \neq 0$
- $Ax = b \Rightarrow x = A^{-1}b = ?$ : 조건은 ?

# 벡터와 행렬

## [역행렬]

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

- A행렬이 가역이다.
- A의 열들은 서로 독립이다.
- A의 행들은 서로 독립이다.
- A는 좌 역행렬을 갖는다.
- A는 우 역행렬을 갖는다.

# 벡터와 행렬

## [역행렬]

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

```
In      # 역행렬의 몇 가지 성질 확인
        A = np.random.RandomState(123).randint(1, 9, 9).reshape
        (3, 3)
        A
Out      array([[7, 6, 7],
               [3, 5, 3],
               [7, 2, 4]])
```

```
In      # 행렬 계수 구하기
        np.linalg.matrix_rank(A)
Out      3
```

# 벡터와 행렬

## [역행렬]

- $Ax = b$
- $x = A^{-1}b$

```
In      # 역행렬 구하기
        A_inverse = la.inv(A)
        np.round(A_inverse, 2)
Out      array([[ -0.27,  0.2 ,  0.33],
               [ -0.18,  0.41,  0.  ],
               [ 0.57, -0.55, -0.33]])
```

# 벡터와 행렬

## [선형 방정식과 QR 분해]

- 과제 1

- $A = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 2 & -1 & 1 & -1 \\ 4 & 1 & -1 & 1 \\ 2 & 1 & -1 & -1 \end{bmatrix}, b = \begin{bmatrix} -1 \\ 6 \\ 0 \\ 2 \end{bmatrix}$ 에 대하여  $Ax = b$ 를 만족하는  $x$  값을

QR 분해를 통하여 구하세요.