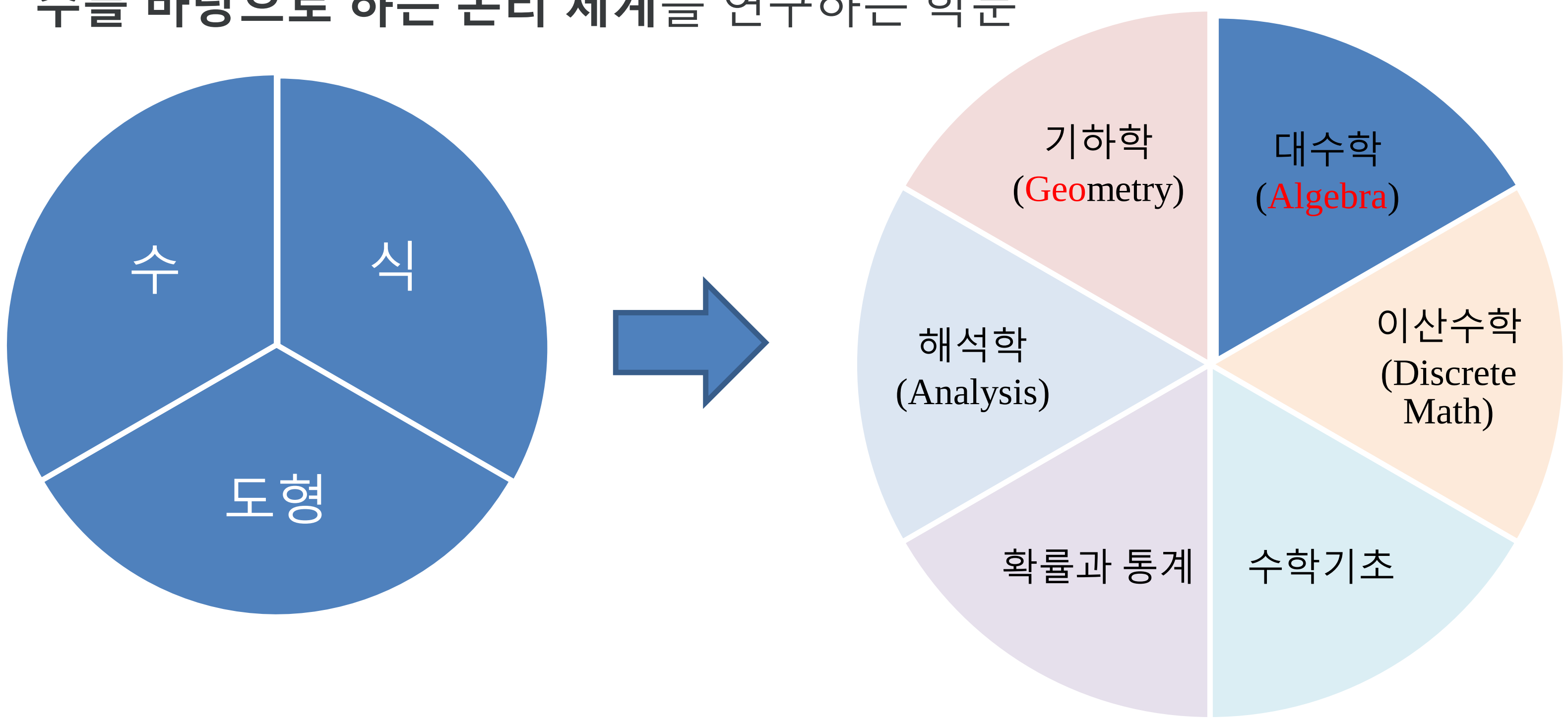
PBL기반 금융빅데이터 분석가 과정

부실예측모형연구2.2
(Linear Algebra)

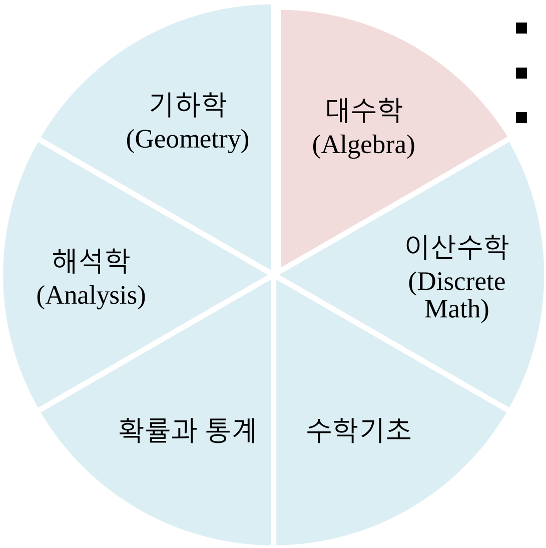
수학 분류

Mathematics

수를 바탕으로 하는 논리 체계를 연구하는 학문



수학 분류



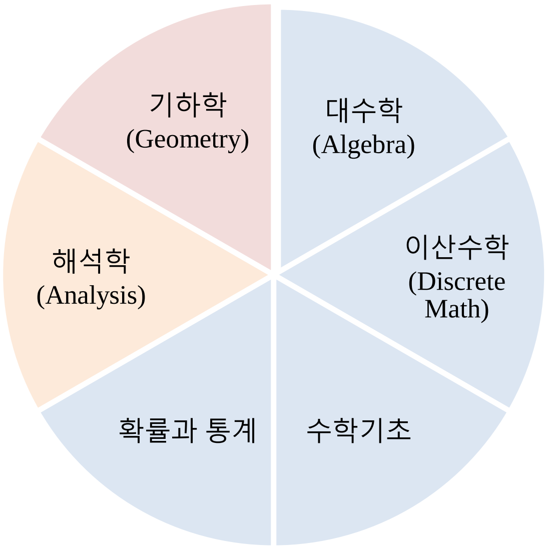
- 추상대수학 (Abstract Algebra) : Group, Ring , Field theory)
- 선형대수학 (Liner Algebra) : Vector , Matrix
- 함수론

알콰리즈미
아라비아 수학의 위대한 영웅

1983년 9월 6일 소련에서 알콰리즈미 출생 1200주년을 기념하기 위해 만든 우표

https://ko.wikipedia.org/wiki/%EC%BD%B0%EB%A6%AC%EC%A6%88%EB%AF%B8
%EC%A6%88%EB%AF%B8

수학 분류



전자시계와 아나로그시계

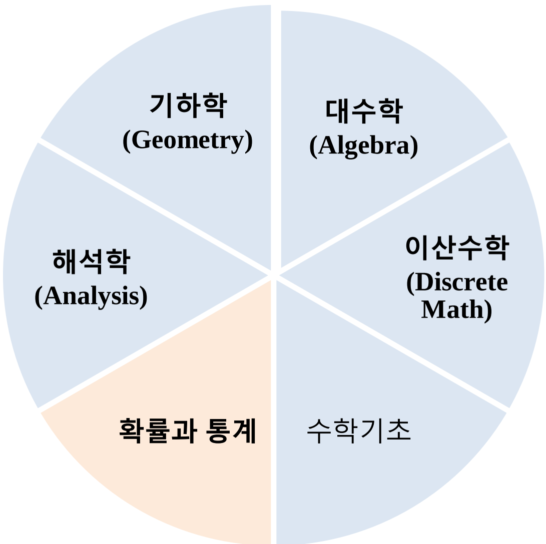
유클리드 기하학(Euclidean geometry)

원론을 명제나 정리부터 시작하지 않고 다음과 같은 정의(定義, definition)로 시작
수학에서 정의는 개인적인 사유 혹은 학자간 대화의 견고한 출발점

해석학(analysis)

대수학과 기하학에 대하여, 미분과 적분의 개념을 기초로 함수의 연속성에 관한 성질을 연구하는 수학의 분야

수학 분류



- **표본 공간, 사건, 순열**
- **확률, 조건부 확률, 베이즈 정리**
- **확률변수 : 누적분포함수, 확률밀도함수, 기대값, 분산**
- **확률분포 : 이산확률분포(베루누이, 이항.) 연속확률분포(정규, 표준정규, t분포,F분포 등)**
- **공분산, 상관관계**
- **모수 추정 : 확률분포, 추정량, 편향(biased),**
  불편추정량 (unbiased estimator)
  BLUE(Best Linear Unbiased Estimator)

- 표본공간 (sample space): 여기에 수식을 입력하십시오.가능한 모든 표본의 집합 (앞면, 뒷면)
- 사건(event) : 표본공간 부분집합
- 확률(probability): **사건(부분집합)을 입력하면 숫자(확률값)가 출력되는 함수**
- 베이즈 정리(Bayesian rule)
- 기술통계(descriptive statistics) : 분포의 표현 , 평균, 분산

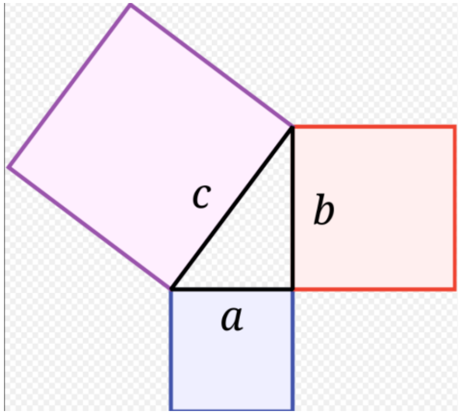
수학 체계 구성 용어

명칭	설명	사례
공리 **(axiom, 公理)**	▪ 증명할 필요가 없이 자명한 진리 ▪ 증명이 불가능 ▪ 참 이라고 가정하는 수학적 서술 ▪ 수학정리의 기반	평행선 공리 (유클리드 공리 중) A=B , B=C → A=C A=B 이면 A+C=B+C
정의 **(definition, 定義))**	▪ 수학적 특성을 모호하지 않고 정확하게 표현하는 것 ▪ 정확한 사고의 출발점 ▪ 수학적 증명을 기술하는 기본 용어	삼각형 정의 점의 정의 선의 정의
정리 **(theorem, 定理)**	▪ 수학에서 정의나 공리에 의해 가정(assumption)으로부터 이미 진리(참)로서 증명된 명제	피타고라스 정리
명제 **(proposition, 命題)**	▪ 논리적이고 수학적 추론으로 증명한 수학적 서술 ▪ 그 내용이 참인지 거짓인지를 명확하게 판단할 수 있는 식이나 문장을 의미	두 직선의 교차 여부 두 명제의 진릿값이 동일할 때, 두 명제식을 **논리적 동치 (Logically equivalent)**
보조정리 **(lemma)**	중요한 정리 증명을 위해 사용하는 보조적인 수학적 서술	유클리드 보조 정리 Ito's Lemma
따름정리 **(corollary)**	정리, 명제 보조정리를 이용해 쉽게 증명할 수 있는 수학적 서술	산술-기하 평균 부등식
가설 (hypothesis,假說)	과학적 자료들에 근거하여 논리적으로 유추하여 설정한 것	

피타고라스 정리

피타고라스의 정리Pythagoras' theorem

직각삼각형에서 빗변 길이의 제곱은 다른 두 변의 길이의 제곱의 합과 같다



유클리드 거리(Euclidean distance)

두 점 사이의 거리를 계산할 때 흔히 쓰는 방법

이 거리를 사용하여 유클리드 공간을 정의할 수 있으며, 이 거리에 대응하는 노름을 유클리드 노름(Euclidean norm)이라고 부른다.

직교 좌표계의 두 점 (x_1, y_1) , (x_2, y_2)

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

선형 대수학 : 스칼라

선형대수학(linear algebra)
벡터 공간, 벡터, **선형** 변환, 행렬, 연립 **선형** 방정식 등을 연구하는 **대수학** 한 분야

▪ 스칼라(scalar)

: 길이, 넓이, 질량, 온도 - 크기만 주어지만 완전히 표시되는 양

$x \in \mathbf{R}$

▪ : 크기 값을 가지는 양

→하나의 숫자만으로 이루어진 데이터

→ x와 같이 소문자로 표기하며 실수(Real number)인 숫자 중의 하나

→***상수, 변수, 함수 관계없이 1차원의 값을 나타내는 것은 모두* scalar**

스칼라(scalar) : a , b

- 덧셈의 교환법칙 : a + b = b + a
- 덧셈의 결합법칙 : (a+b) + c = a + (b+c)
- 곱셈의 교환법칙 : ab= ba
- 곱셈의 결합법칙 : a(bc) =(ab)c
- 분배법칙 : c (a+b) = ac + ab

벡터의 이해

- **벡터(vector)**
- 속도, 위치이동, 힘 - 크기뿐만 아니라 방향까지 지정하지 않으면 완전히 표현할 수 없는 양
- 2차원, 3차원 공간의 벡터는 화살표로 표현 가능
- 벡터는 공간에서 한점을 나타 남.
- 벡터는 원점에서부터 상대적 위치를 표현

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

$$x \in \mathbf{R}^N$$

→ 1 차원 배열(array)

- **Norm**

→벡터의 크기

하나의 벡터를 이루는 데이터의 개수가 n개 인 경우 → n-dimensional vector

→ Norm을 이용하면 두 점 사이의 **거리**(distance)를 쉽게 계산

→벡터의 노름은 원점에서부터 벡터까지의 거리

- L1 Norm : 각 성분의 변화량의 절대값을 모두 합 함.(멘하튼 거리, 택시거리)
- L2 Norm: 피타고라스 정리를 이용해 유클리드 거리를 계산 : (유클리드 거리)

벡터의 힘의 크기법 표시방법

$$\left|\overrightarrow{AB}\right| \quad \text{or} \quad \left\|\overrightarrow{AB}\right\|$$

벡터의 힘의 크기 계산

$$\|\vec{a}\| = \sqrt{a_x^{\ 2} + a_y^{\ 2}}$$

벡터의 종류

행벡터(row vector)

1 by n 행렬

$$\begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

열벡터(column vector)

n by 1 행렬

$$\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

n by 1 벡터 전체로 이루어지는 집합을 R에 이중선을 추가하여 표시

$$\mathbb{R}^n$$

단위벡터(unit vector)

벡터의 힘의 크기가 1인 벡터

$$|\overrightarrow{AB}| = |\vec{a}| = \|\vec{a}\| = 1$$

(공간)벡터(vector in space)

$\mathbf{x} = (x_1,\ x_2,\ x_3)$ 또는 $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_{3\times 1}$

x1, x2, x3 를 (공간)벡터 x의 성분(component)이라고 함.

n차원 벡터(-dimensional vector)

$$\mathbf{x} = (x_1,\ x_2,\ \ldots\ ,x_n) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{n\times 1}$$

일차결합(linear combination)

X_1 , X_2 …. X_n **벡터**, a_1 , a_2 …. a_n **실수**

→ $a_1X_1 + a_2X_2 + \ldots\ldots + a_nX_n$

벡터의 연산

벡터와 벡터의 덧셈

$\vec{\boldsymbol{a}}$ =$(\boldsymbol{a_1}, \boldsymbol{a_2})$, $\vec{\boldsymbol{b}}$ =$(\boldsymbol{b_1}, \boldsymbol{b_2})$

➔ $\vec{a}$ + $\vec{b}$ =($a_1 + b_1$, a_2 +b_2)

→ ($a_1 + b_1$, a_2 +b_2 + …….. + a_n +b_n)

벡터와 벡터의 뺄셈

$\vec{\boldsymbol{a}}$ =$(\boldsymbol{a_1}, \boldsymbol{a_2})$, $\vec{\boldsymbol{b}}$ =$(\boldsymbol{b_1}, \boldsymbol{b_2})$

➔ $\vec{a}$ - $\vec{b}$ =($a_1 - b_1$, a_2 −b_2)

→ ($a_1 - b_1$, a_2 −b_2 + …….. + a_n −b_n)

x = np.array([[1], [3], [5], [7]])
깃허브 참조 : 행렬 및 벡터 사칙연산

벡터의 연산

벡터와 벡터의 곱셈

내적**內積 → 곱한다.**
= dot product = scalar product
(inner product)

$\vec{a}$ =(a_1, a_2), $\vec{b}$ =(b_1, b_2)

→ $\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2$

→ 결과 값은 스칼라로!!

→ 벡터를 마치 수처럼 곱하는 개념 (표시 : ·)

→ 대응하는 성분끼리 곱한 다음 모두 더한 값이 내적 !!

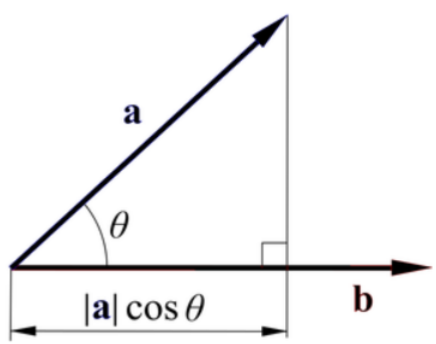
→ 벡터에는 방향이 있으므로, 방향이 일치하는 만큼만 곱한다.

→ 두 벡터의 방향이 같으면, 두 벡터의 크기를 그냥 곱한다.

→ 두 벡터가 이루는 각이 90도일 땐, 일치하는 정도가 전혀 없기 때문에 내적의 값은 0

$\vec{a} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ $\vec{b} = \begin{pmatrix} 0 \\ 3 \end{pmatrix}$

$\vec{a} \cdot \vec{b} = 0$



$\cos\theta = \dfrac{\text{밑변}}{\text{빗변}}$

내적 : 두 벡터의 각 성분끼리의 곱의 합산

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

→ 두 벡터의 크기 (Norm)와 두 벡터 사이의 각의 코사인 값(cos θ)을 곱한 것 : 내적으로 두 벡터간에 이루는 각도를 알 수 있다는 것

→ ***내적은 한 벡터를 다른 벡터로 projection 시켜서, 그 벡터의 크기를 곱한다.***

벡터의 연산

$\vec{a} = (a_1, a_2), \vec{b} = (b_1, b_2)$

➔ $\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2$

→ 결과 값은 스칼라로!!

$$\vec{a} \cdot \vec{b} = < \vec{a}, \vec{b} > = \vec{a}^T \vec{b}$$

두 열 벡터의 내적을 구하려고 할 경우 둘 중 하나의 벡터를 transpose 시켜 행 벡터의 형태로 바꾼 후, 나머지 벡터와 벡터곱을 시키는 것

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^{n} a_i \, b_i = a_1 b_1 + a_2 b_2 + a_3 b_3 + \ldots\ldots\ldots\ldots\ldots\ldots + a_n b_n$$

1x3 행렬(row vector), 3x1 행렬(column vector) → 내적 : 1x1행렬

$$\begin{pmatrix} 1 & -3 & 5 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \\ 2 \end{pmatrix} = 2 - 3 + 10 = 9$$

벡터의 연산

벡터와 벡터의 곱셈

벡터 x와 벡터 y의 내적

두 벡터를 내적하려면 조건

1. 두 벡터의 차원(길이)이 같아야 한다.
2. 앞의 벡터가 행 벡터이고 뒤의 벡터가 열 벡터여야 한다.

$$w_1x_1 + \cdots + w_Nx_N = \sum_{i=1}^{N} w_ix_i$$

내적은 교환법칙이 성립, 분배법칙 성립

벡터의 연산

내적(dot product) 활용

두 벡터의 크기 (Norm)와 두 벡터 사이의 각의 코사인 값(cos θ)을 곱한 것

→ 내적으로 두 벡터간에 이루는 각도를 알 수 있다는 것

→ 두 벡터를 내적을 하고 나면 스칼라 값 : 두 벡터의 유사 정도를 알 수 있다.

크기가 1인 두 벡터가 있다고 한다면

(1) 두 벡터가 수직이라면 두 벡터가 이루는 각도가 90 도 → 내적값은 0

(2) 두 벡터가 평행하면서 동일 방향이면 두 벡터가 이루는 각도가 0 도 → 내적값은 1

(3) 두백터가 평행이면서 반대 방향이면 두 벡터가 이루는 각도가 180도 인 경우 → 내적값은 -1

(4) 두 벡터가 이루는 각도가 0도보다 크고 90도보다 작다면 → 내적값은 0 ~ 1사이값.

→ *두 벡터의 크기가 1 이라면, 두 벡터의 방향이 유사할수록 내적값은 1 에 가까워 진다.*

두 벡터 간의 유사도(similarity)를 계산하는 경우 사용

: 유사도(similarity) → 두 벡터가 닮은 정도를 정량적으로 나타낸 값

- 내적값 = 0 → 각도(θ) = 90도
- 내적값 > 0 → 각도(θ) < 90도
- 내적값 < 0 → 각도(θ) > 90도

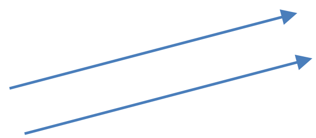
→ 위 관계를 이용하여 **두 벡터간의 위치관계를 파악 가능**

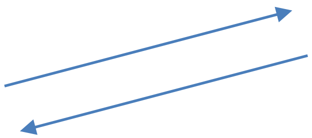
- ***두 벡터 사이의 거리를 측정하는데 이용***
- ***한 벡터의 norm 을 구하는 경우 활용***

벡터의 연산

$\vec{a} \cdot \vec{b}$ = ||a|| ||b|| cosθ

벡터 유사도 : cosine metric = cos θ = $\dfrac{\vec{a} \cdot \vec{b}}{||a||\ ||b||}$





(1)θ = 90도 : 내적 값은 0

$\vec{a} = \begin{pmatrix}1\\0\end{pmatrix}$ $\vec{b} = \begin{pmatrix}0\\3\end{pmatrix}$

$\vec{a} \cdot \vec{b}$ = 0

(2)θ = 0도 : 내적 값은 1

$\vec{a} = \begin{pmatrix}1\\0\end{pmatrix}$ $\vec{b} = \begin{pmatrix}1\\1\end{pmatrix}$

$\vec{a} \cdot \vec{b}$ = 1

(3)θ = 180도 : 내적 값은 -1

$\vec{a} = \begin{pmatrix}1\\0\end{pmatrix}$ $\vec{b} = \begin{pmatrix}-1\\1\end{pmatrix}$

$\vec{a} \cdot \vec{b}$ = -1

$\alpha°$	α rad	sin α	cos α	tan α	cot α	sec α	cosec α
0	0	0	1	0	∞	1	∞
30	$\frac{\pi}{6}$	$\frac{1}{2}$	$\frac{\sqrt{3}}{2}$	$\frac{1}{\sqrt{3}}$	$\sqrt{3}$	$\frac{2}{\sqrt{3}}$	2
45	$\frac{\pi}{4}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{2}}{2}$	1	1	$\sqrt{2}$	$\sqrt{2}$
60	$\frac{\pi}{3}$	$\frac{\sqrt{3}}{2}$	$\frac{1}{2}$	$\sqrt{3}$	$\frac{1}{\sqrt{3}}$	2	$\frac{2}{\sqrt{3}}$
90	$\frac{\pi}{2}$	1	0	∞	0	∞	1
120	$\frac{2\pi}{3}$	$\frac{\sqrt{3}}{2}$	$-\frac{1}{2}$	$-\sqrt{3}$	$-\frac{1}{\sqrt{3}}$	-2	$\frac{2}{\sqrt{3}}$
180	π	0	-1	0	∞	-1	∞
270	$\frac{3\pi}{2}$	-1	0	∞	0	∞	-1
360	2π	0	1	0	∞	1	∞

벡터의 연산

$\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}}$ = ||a|| ||b|| cosθ

벡터 유사도 : Euclidean metric *두 벡터 사이의 거리를 측정하는데 이용*

$\vec{a}=(a_1, a_2)$, $\vec{b}=(b_1, b_2)$

dist = $(\vec{a}, \vec{b}) = \| \vec{a} - \vec{b} \| = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}$

벡터의 연산

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

- 벡터 내적 : 벡터의 norm을 구하는 경우 활용

$$\vec{a} = (1 \;\; 0 \;\; 1) \quad \vec{b} = \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix}$$

$$\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = 0 + 0 + 1 = 1$$

$\|a\|\,\|b\| \cos\theta = \sqrt{2} \times \sqrt{2} \times \cos\theta \rightarrow \cos\theta = 0.5$

$\|a\| = \sqrt{2}$

$\|b\| = \sqrt{2}$

벡터의 연산

내적(dot product) 활용

- **임의의 벡터의 특정 방향을 가진 성분의 크기를 알아내는데 유용**
- 내적은 벡터의 특정 방향, 성분, 투영(사영)의 크기, 일의 크기, 전류 밀도에 대한 전류의 크기 등을 구할 때 필요

1. 가중합/ 가중평균 사용
2. **다중선형회귀분석에서**
3. **딥러닝 자연어처리 유사도 분석**

벡터의 연산

벡터와 벡터의 곱셈

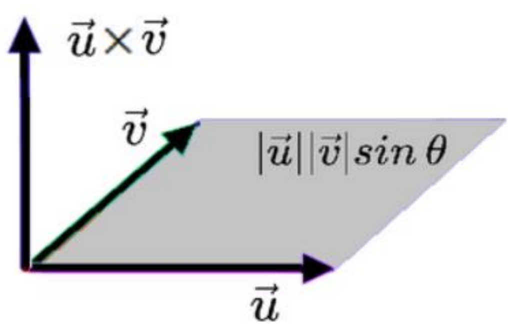
외적 (外積 , outer product)

= vector product 또는 cross product

(표시 : ×)

→ 외적의 크기는 면적

내적의 결과값은 스칼라, 외적의 결과값은 벡터



두 벡터의 크기와 두 벡터 사이의 각의 사인값(sin θ) 그리고 수직인 벡터의 곱으로 정의

외적은 교환법칙이 성립하지 않는다

행렬 이해

행렬

- 행렬은 복수의 차원을 가지는 데이터 레코드가 다시 여러 개 있는 경우의 데이터를 합쳐서 표기한 것
- 행렬은 행(row)과 열 (column)이라는 인덱스
- 벡터가 공간에서 한 점을 된다면, 행렬은 여러점들로 표시 됨.
- 행렬 : 대문자로 표기, 스칼라 : 이태리 소문자로 , 벡터 : 볼드체 소문자로 표기

- **스칼라와 벡터도 수학적으로는 행렬이다**.
- 스칼라는 열과 행의 수가 각각 1인 행렬이다.
- 벡터는 열 (column 의 수가 1인 행렬이다.

•**행렬**

•. 보통 3차원까지의 벡터는 그림 등으로 시각적 표현이 가능하지만 그 이상의 벡터는 벡터의 각 구성요소를 괄호 안에 나열함으로써 표기.

선형화 혹은 선형 근사를 통해, 복잡한 비선형 방정식 문제를 간단한 선형 방정식 문제로 변환해 문제를 해결할 수 있기 때문

텐서

같은 크기의 행렬이 여러 개 같이 묶여 있는 것

행렬의 종류

전치행렬(transposed matrix) : A^T

: **행렬의** 행과 열을 서로 맞바꾼 행렬 '

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

단위 행렬(unit matrix) 또는 항등 행렬(Identity matrix) : I

주대각선의 원소가 모두 1이며 나머지 원소는 모두 0인 정사각 행렬

$$I=\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix},\quad I=\begin{matrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{matrix}$$

역행렬(Inverse Matrix) : A^{-1}

AB=BA=I : B는 A의 역행렬

→ $AA^{-1}=A^{-1}A=\mathrm{I}$ −→ A^{-1}는 A의 역행렬

행렬에는 나누기 연산이 없기 때문에 역행렬을 곱해야 함.

정방행렬(square matrix)

행의 개수와 열의 개수가 같은 행렬

$\mathrm{m}\times m$

영행렬(Null Matrix or zero Matrix)

모든 원소가 0 이다.

$A=\begin{pmatrix}a & b\\c & d\end{pmatrix}$에 대한

$$\boldsymbol{A}^{-1}=\frac{1}{ad-bc}\begin{pmatrix}d & -b\\-c & a\end{pmatrix}$$

(예시)

$$A=\begin{pmatrix}1 & 2\\2 & 6\end{pmatrix} \rightarrow \boldsymbol{A}^{-1}=\frac{1}{6-4}\begin{pmatrix}6 & -2\\-2 & 1\end{pmatrix}$$

$$A\,\boldsymbol{A}^{-1}=\begin{pmatrix}1 & 2\\2 & 6\end{pmatrix}\begin{pmatrix}3 & -1\\-1 & \frac{1}{2}\end{pmatrix}=\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}$$

행렬의 연산

행렬의 덧셈과 뺄셈

행렬을 덧셈 , 뺄셈하기 위해서는 먼저 연산 하고자 하는 행렬끼리의 차원이 동일해야 한다.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} + \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} a+e & b+f \\ c+g & d+h \end{pmatrix}$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} - \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} a-e & b-f \\ c-g & d-h \end{pmatrix}$$

행렬의 연산

행렬의 곱셈

스칼라 곱셈

$$k\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} ka & kb \\ kc & kd \end{pmatrix}$$

→ 스칼라 곱셈은 어떤 차원의 행렬이든 가능

행렬끼리의 곱셈 : 적합성조건(conformability condition)이 충족되어야 한다.

→ 두 행렬 중 앞 행렬의 **열**과 뒷 행렬의 **행**의 갯수가 동일해야 함.

→ 결과 만들어지는 행렬의 차원은 (앞 행렬의 행의 개수 x 뒷 행렬의 열의 개수)가 된다.

행렬의 연산

행렬의 곱셈 사례

- 두 행렬 중 앞 행렬의 **열**과 뒷 행렬의 **행**의 갯수가 동일해야 함.
- $m\times n$ 행렬곱셈은 $n\times k$ 행렬이어야 함.
- →결과 만들어지는 행렬의 차원은 (앞 행렬의 행의 개수 x 뒤 행렬의 열의 개수)가 된다.
- $m\times n$ 행렬과 $n\times k$ 행렬 곱셈은 $m\times k$ 행렬이 됨.

$$[1\ 2\ 3]_{1\times 3} \times \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} = 1\times 4 + 2\times 5 + 3\times 6 = 32$$

→ 1×3 행렬과 3×1 행렬 곱셈 결과 스칼라

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \times (3\ 4\ 5) = \begin{pmatrix} 1\times 3 & 1\times 4 & 1\times 5 \\ 2\times 3 & 2\times 4 & 2\times 5 \end{pmatrix} = \begin{bmatrix} 3 & 4 & 5 \\ 6 & 8 & 10 \end{bmatrix}$$

→ 2×1 행렬과 1×3 행렬 곱셈 결과 2×3 행렬

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \quad B = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \rightarrow AB = \begin{bmatrix} 1\times 2 + & 2\times 3 \\ 3\times 2 + & 4\times 3 \\ 5\times 2 + & 6\times 3 \end{bmatrix} = \begin{bmatrix} 8 \\ 18 \\ 28 \end{bmatrix}$$

→ 3×3 행렬과 2×1 행렬 곱셈 결과 3×1 행렬

행렬의 연산

두 행렬

행렬A : m×n 행렬

행렬 B : n×r 행렬

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1r} \\ b_{21} & b_{22} & \cdots & b_{2r} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nr} \end{pmatrix}$$

행렬의 곱

AB : m×r 행렬

$$AB = \begin{pmatrix} \sum_k a_{1k}b_{k1} & \sum_k a_{1k}b_{k2} & \cdots & \sum_k a_{1k}b_{kr} \\ \sum_k a_{2k}b_{k1} & \sum_k a_{2k}b_{k2} & \cdots & \sum_k a_{2k}b_{kr} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_k a_{mk}b_{k1} & \sum_k a_{mk}b_{k2} & \cdots & \sum_k a_{mk}b_{kr} \end{pmatrix}$$

단, k=1,2, 3, ……. n

행렬의 연산

행렬의 덧셈

- 교환법칙 성립 : A + B = B +A
- 결합법칙 성립 : (A+B) + C = A + (B +C)

행렬의 곱셈

- 교환 법칙이 성립하지 않는다 : AB $\neq BA$
- --> 행렬의 곱셈에서 두 행렬의 순서 또한 중요한 것을 알 수 있다.

→ 단, 곱하려는 행렬 중 하나가 단위행렬이거나 영행렬인 경우는 교환법칙 성립

- 덧셈에 대한 분배법칙은 성립 : A (B + C) = AB + AC

행렬의 크기

행렬 Norm

표기 $$\|A\|_p = \left(\sum_{i=1}^{N} \sum_{j=1}^{M} |a_{ij}|^p \right)^{1/p}$$

p=2인 경우가 가장 일반적 $$\|A\| = \|A\|_2 = \|A\|_F = \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij}^2}$$

벡터의 놈의 제곱이 벡터의 제곱합과 같다는 것 $$\|x\|^2 = \sum_{i=1}^{N} x_i^2 = x^T x$$

예측분석을 위한 선형대수학 기초 사전지식

벡터와 행렬 이용하는 이유!

벡터와 행렬의 연산을 이용하면 대량의 데이터에 대한 계산을 간단한 수식으로 표시

깃허브 실습 참고

(벡터와 행렬 사칙연산)

행렬의 미적분학

https://en.wikipedia.org/wiki/Matrix_calculus

유형	스칼라	벡터	행렬
스칼라	$\frac{\partial y}{\partial x}$	$\frac{\partial \mathbf{y}}{\partial x}$	$\frac{\partial \mathbf{Y}}{\partial x}$
벡터	$\frac{\partial y}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$	
행렬	$\frac{\partial y}{\partial \mathbf{X}}$		

- 스칼라 함수: 그 결과값이 1차원의 값인 함수
- 벡터함수 : 결과값이 다차원인 함수

1) 스칼라 함수를 벡터로 미분(Gradient)

$$\frac{\partial y}{\partial \mathbf{x}} = \left[\frac{\partial y}{\partial x_1} \; \frac{\partial y}{\partial x_2} \cdots \frac{\partial y}{\partial x_n} \right]$$

2) 벡터를 스칼라로 미분

$$\frac{\partial \mathbf{y}}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \frac{\partial y_2}{\partial x} \\ \vdots \\ \frac{\partial y_m}{\partial x} \end{bmatrix}$$

행렬의 미적분학

3) 스칼라 함수를 행렬로 미분

$$\frac{\partial y}{\partial X} = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \cdots & \frac{\partial y}{\partial x_{m1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{1n}} & \cdots & \frac{\partial y}{\partial x_{mn}} \end{bmatrix}$$

4) 행렬을 스칼라로 미분

$$\frac{\partial Y}{\partial x} = \begin{bmatrix} \frac{\partial y_{11}}{\partial x} & \cdots & \frac{\partial y_{1n}}{\partial x} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_{m1}}{\partial x} & \cdots & \frac{\partial y_{mn}}{\partial x} \end{bmatrix}$$

5) 벡터를 벡터로 미분

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

로그(log) 이해

로그(log)
: logarithm의 줄임말

지수에 대비된다는 의미에서 '대수(對數)'

a^x =b 만족할 때
x = log_ab 로 정의 한다.

- 밑이 2인 경우 : 이진binary로그
- 밑이 10인 경우 : 상용common로그 log

→ 10을 생략하는 건 우리가 쓰는 수체계가 십진법이기 때문

→ 대중적인 분야의 로그에서는 대부분 상용로그로

- 밑이 e 인 경우 : 자연natural로그 ln

$$e = \lim_{n\to\infty}\left(1 + \frac{1}{n}\right)^n$$

로그(log) 이해

로그 함수 특징

상수 법칙	$\log_a 1 = 0, \log_a a = 1$
덧셈 법칙	$\log_a xy = \log_a x + \log_a y$
뺄셈 법칙	$\log_a \frac{x}{y} = \log_a x - \log_a y$
지수 법칙	$\log_a x^b = b \log_a x$
밑 변환 법칙	$\log_b x = \frac{\log_k x}{\log_k b}$ (단, $k > 0,\ k \neq 1$)
역수 법칙	$\log_b x = \frac{1}{\log_x b}$ (단, $b \neq 1$)

수열의 합과 곱

Sum $$\sum_{i=1}^{N} x_i = x_1 + x_2 + \ldots + x_N$$

product $$\prod_{i=1}^{N} x_i = x_1 \cdot x_2 \cdot \ldots \cdot x_N$$

확률과정 Stochastic Process

확률 과정(Stochastic Process)란 ?

→시간의 진행에 대해 확률적인 변화를 가지는 구조를 의미

Markov Process, Wieners process , Brownian motion process, Ito process, poisson process

random walk

- 임의 방향으로 향하는 연속적인 걸음을 나타내는 수학적 개념
- 1905년 칼 피어슨이 소개
- 생태학, 수학, 컴퓨터 과학, 물리학, 화학 등의 분야에서 광범위하게 사용
- 시간에 따른 편차의 평균이 0
- 분산은 시간에 비례하여 증가 → 앞뒤로 움직일 확률이 동일하다고 해도 시간이 흐름에 따라 평균에서 점차 벗어나는 경향
- 대표적인 예 : 브라운 운동

Brownian motion

- 1827년 스코틀랜드 식물학자 Robert Brown이 발견
- 액체나 기체 속에서 미소입자들이 불규칙하게 운동하는 현상

확률과정 Stochastic Process

랜덤 과정 / 확률 과정 (Random Process, Stochastic Process, Probabilistic Process)

Random / Stochastic

- 시간적으로 미리(사전에) 결과에 대해 정확히 예측,정의할 수 없다는 의미
- 단, 어떤 확률적 분포를 가질 수 있다는 통계적 규칙성은 있음 ☞ 랜덤성

-→ 여기서, `Random(무작위)` 및 `Stochastic(추계적)`를 같은 의미로 씀

Process (과정)

- 시간을 고려한 상태를 말할 때는 주로 `과정`
- 시간을 고려하지 않는 상태는 주로 `사건`이라고 함

확률 과정(Stochastic Process)

연속적 확률과정 : 시간 T가 어떠한 연속적인 범위를 두고 있는 경우

- 정상성(stationary) : 연속형 확률과정 X(t)가 같은 시간의 주기를 가지면 같은 확률분포를 가진다는 의미
- **독립성 :** 연속형 확률과정 X(t)가 서로 겹치지 않는 구간들에 대해서 전부 독립적인 확률 변수라는 의미

이산적 확률과정

Markov Process, Wieners process , Brownian motion process, Ito process, poisson process

확률과정 Stochastic Process

Markov stochastic process

- T+1시점은 과거의 값에는 전혀 영향을 받지 않고 오직 오늘의 값(T시점)에만 영향을 받는 확률과정
- 현재에 대한 조건부로 과거와 미래가 서로 독립인 확률 과정
- 1906년 러시아의 수학자 안드레이 마르코프가 도입

$$E\left(S_{t+1} \middle| S_t, S_{t-1}, \cdots, S_0\right) = E\left(S_{t+1} \middle| S_t\right)$$

확률과정 Stochastic Process

Wieners process

- 마르코프 과정 중에서도 특히, 변화량의 평균 0 →향후 예상되어지는 변화가 평균적으로 0이다.
- 년간 분산은 1을 따르는 확률과정
- Wieners process를 따르는 확률변수 z의 특징
- z의 변화량은 평균은 0, 분산은 t의 변화량을 값으로 갖는 정규분포를 따른다. T시점의 분산은 T 만큼의 값을 갖는다.

  → 즉 시간의 변화량이 커질수록, z의 변동성도 함께 커지게 된다.
- (위너 과정 특징)

  → 위너과정의 평균과 분산이 시간이 지남에 따라 커지는 선형 함수로 표현된다는 것
- (조건)
- → 정상성과 독립성을 만족하며 초기 시간대의 값이 0
- 시간에 따라 위너과정의 확률분포는 정규분포를 따른다.

*쉬어가기 : 확률과정 Stochastic Process

Wieners process : Arithmetic Brownian motion

평균변화율이 0이 아닌 확률 과정 : Generalized Wiener process , Arithmetic Brownian motion

- dx = $\mu\, dt + \sigma dW$
- μ : x의 변화율의 평균, σ : 표준편차를 나타내는 상수항 , dW: 위너 과정

Geometric Brownian motion

- 주식가격을 모델링하는 경우 주식의 가격이 이론적으로 음(-)의 값을 가질 수 있는 문제점 발생
- 주식의 가격을 S
- 기하브라운 운동
- $\frac{dS}{S} = \mu\, dt + \sigma dW$

블랙-숄즈 모형 , Geometric Brownian motion

- 주식의 가격을 브라운 운동으로 모델링
- 이러한 모델은 가격이 불규칙적(무작위적)으로 움직인다는 가정을 기반

확률과정 Stochastic Process

- 백색잡음과정 (White Noise Process)
- **확률보행과정 (Random Walk Process)**
- **정상확률과정 (Stationary Process)**

예측기법 : time series data 분석기법

1) **회귀분석법** : 인과형

2) **이동평균법** : 단순이동평균, 가중이동평균

→ MSE : t기의 관측값과 t기의 예측값간 차이 제곱값의 평균

,→ 이동평균의 기간 n이 짝수인 경우 이동평균법으로 계산하면 이동평균에 대응하는 시기에 문제가 발생함 → 이동평균의 기간이 짝수인 경우 인접한 두 이동평균의 평균을 계산하여 이동평균을 중심화하는 중심이동평균(centered moving average)을 구함

3) 지수평활법(exponential smoothing)

- 최근의 자료에 더 큰 가중치를 주고 현 시점에서 멀수록 작은 가중치를 주어 지수적으로 과거의 비중을 줄여 미래값을 예측하는 방법
- 통계적인 이론 배경이 있다기 보다는 경험적으로 해보니 예측이 잘 맞더라는 경험에 기반한 분석기법

단순 지수평활"(simple exponential smoothing, SES)

: 추세나 계절성 패턴이 없는 데이터를 예측할 때

- 지수평활법은 예측오차를 비교하여 예측오차가 작은 α값을 선택하는 것이 바람직함.
- 지수평활법에서 가중치는 과거로 갈수록 지수적으로 감소 하게 됨.
- 과거값(현재시점의 관측값)에 가장 큰 가중치를 부여하므로 일종의 가중이동평균 법이라 할 수 있음.

데이터가 어떤 확률과정(stochastic process)을 따라야 한다는 가정사항이 없다.

(활용)

계절성 제거

$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha) y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots,$$

확률과정 Stochastic Process

ARIMA (AutoRegressive Integreated Moving Average) 모형

- 확률모형을 기반으로 한 시계열 분석 기법
- 정상확률과정(Stationary Process)을 가정
- 만약 시계열 데이터가 정상성(stationarity) 가정을 만족하지 않을 경우 데이터 전처리를 통해서 정상시계열 (stationary time series)로 변환해 준 다음에야 ARIMA 모형 적용 가능

- 백색잡음과정 (White Noise Process)
- **확률보행과정 (Random Walk Process)**
- **정상확률과정 (Stationary Process)**

- 백색잡음과정 (White Noise Process)

- 신호처리 분야 : 다른 주파수에서 동일한 강도를 가지고, 항상 일정한 스펙트럼 밀도를 발생시키는 무작위한 신호 → 물리학, 음향공학, 통신, 통계 예측 등 다양한 과학과 기술 분야에서 사용.
- 시계열 통계분석 :백색잡음과정(white noise process)은 평균이 0, 분산은 유한한 상수(σ^2)인 확률분포(random variables)로 부터 서로 상관되지 않게(uncorrelated) 무작위로 샘플을 추출한 이산 신호(discrete signal)

특히, 평균이 0 인 동일한 정규분포로부터 서로 독립적으로 샘플을 추출한 백색잡음을 **가법 백색 가우시언 잡음** (additive white Gaussian noise) 라고 함.

확률과정 Stochastic Process

- **확률보행과정 (Random Walk Process)**

시계열 과정 (python 참조)

- 정상확률과정 (Stationary Process)

[정상시계열의 조건 (conditions for stationary time series)]

(a) 평균이 일정

(b) 분산이 존재하며 상수

(c) 두 시점 사이의 자기공분산은 시차(time lag)에만 의존

→ ARIMA 시계열 통계모형은 시계열 데이터의 정상성(stationarity) 조건을 만족하는 경우에 사용 가능

→ 만약 추세(trend)가 있거나 시간이 갈 수록 분산이 커지거나 작아지는 시계열 데이터라면 정상확률과정 시계열이 아니다.

최적화 문제

- arguments of min : argmin
- arg min f(x)
- → 함수 f(x)를 최솟값으로 만들기 위한 x 값을 구한다

- arguments of max : argmax
- arg max f(x)
- f(x)를 최댓값으로 만들기 위한 x 값을 구한다.

- **f(x) : 목적함수(objective function), 비용함수(cost function), 손실함수(loss function)**

1) grid search 방법

가능한 x의 값을 여러 개 넣어 보고 그중 가장 작은 값을 선택

2) 수치적 최적화(numerical optimization)

- 단일 변수 함수인 경우: **미분값이 0**

$$\frac{df(x)}{dx} = 0$$

- 다변수 함수인 경우 : **모든 변수에 대한 편미분값이 0**

$$\frac{\partial f(x_1, x_2, \cdots, x_N)}{\partial x_1} = 0$$

$$\frac{\partial f(x_1, x_2, \cdots, x_N)}{\partial x_2} = 0$$

$$\vdots$$

$$\frac{\partial f(x_1, x_2, \cdots, x_N)}{\partial x_N} = 0$$

제한조건 있는 최적화(constrained optimization)

라그랑주 승수법 (Lagrange Multiplier Method) 을 사용하여 최적화

라그랑주 승수법 (Lagrange Multiplier Method)

- 프랑스의 수학자 조셉 루이 라그랑주 (Joseph-Louis Lagrange)가 제약 조건이 있는 최적화 문제를 풀기 위해 고안한 방법
- 어떠한 문제의 최적점을 찾는 것이 아니라, 최적점이 되기 위한 조건을 찾는 방법→ 최적해의 필요조건을 찾는 방법

제한조건 등식에 λ (≠ 0)라는 새로운 변수를 곱해서 더한 함수

$$
\begin{aligned}
h(x, \lambda) &= h(x_1, x_2, \ldots, x_N, \lambda_1, \ldots, \lambda_M) \\
&= f(x) + \sum_{j=1}^{M} \lambda_j g_j(x)
\end{aligned}
$$

sp.optimize.fmin_slsqp

선형계획법 문제

선형계획법(Linear Programming) 문제란?

방정식, 부등식 제한 조건을 가지는 선형 모형(linear model)의 값을 최소화하는 문제

- 최소화하려는 목적함수 설정 (아래 c 관련)
- 제한조건 관련하여 x1 , x2 등의 선형식으로 표현
- 제약조건 표현

scipy.optimize 패키지의 linprog() 명령을 사용하면 선형계획법 문제
linprog(c, A, b)

c: 목적함수의 계수 벡터

A: 등식 제한조건의 **계수 행렬**

b: 등식 제한조건의 **상수 벡터**

```
import scipy.optimize
A = np.array([[, ], [, ], [, ], [, ]])
b = np.array([, , , ])
c = np.array([, ])
result = sp.optimize.linprog(c, A, b)
result
```

이차계획법 문제

이차계획법(Quadratic Programming) 문제란?

방정식, 부등식 제한 조건을 가지는 일반화된 이차형식(Quadratic Form)의 값을 최소화하는 문제

- 이차 형식은 함수의 꼴이 이차 다항식인 것

$$f(\mathbf{x}) = \mathbf{x}^{\mathsf{T}}\mathbf{Q}\mathbf{x}$$

$$= q_{11} x_1^2 + q_{22} x_2^2 + \cdots + q_{nn} x_n^2 + q_{12} x_1 x_2 + \cdots + q_{n,n-1} x_n x_{n-1}$$

- Q는 nxn 크기의 대칭 행렬(symmetric matrix)이고, 이차 형식의 행렬(the matrix of the quadratic form)이라고 함.
- 대칭 행렬 : P와 D로 분해하면 P의 column이 A의 eigen vector로 이루어져 있고, 각 vector는 서로 직교한다는 것

```python
from cvxopt import matrix, solvers
```

수학 표기 이해

수학 기호
https://ko.wikipedia.org/wiki/%EC%88%98%ED%95%99_%EA%B8%B0%ED%98%B8

계산기 Microsoft Math Solver (이전 **Microsoft Mathematics** 및 **Microsoft Math**)

https://mathsolver.microsoft.com/en/algebra-calculator