

# PYTHON

## 확률과 통계 기초 이해

# 집합 기초

# 집합 정의

3

집합(集合, set)은 어떤 조건이 주어졌을 때, 그 조건이 가리키는 대상이 분명한 것들의 모임

집합은 서로 구별되는 대상들을 순서와 무관하게 모은 것이다. 이때 집합에 속하는 각각의 대상들은 집합의 원소라고 한다.

# 원소나열법 / 조건제시법

4

집합에 들어있는 원소들을 직접 나열하거나 조건을 제시하는 방식

$\{ \underline{\hspace{2cm}} \mid \underline{\hspace{2cm}} \}$   
대상      대상설명

원소나열법

- 집합 S는  $\{2, 4, 6, \dots\}$

원소들이 생성되는 방법을 재귀적으로 설명

- $2 \in S$
- 만약  $n \in S$  이라면  $(n+2) \in S$

집합 원소들의 특징을 표현하는 속성 P를 설명

- $S = \{x \mid x \text{는 양의 짝수인 정수}\}$

- $\{x \mid P(x)\}$ 
  - 속성 P를 갖는 특징적인 원소들로 구성된 임의의 집합
- 속성 P
  - 일항술어(unary predicate)
  - 술어
    - 변수가 가질 수 있는 성질
- $S = \{x \mid P(x)\}$ 
  - $(\forall x)[(x \in S \rightarrow P(x)) \wedge (P(x) \rightarrow x \in S)]$
  - 집합 S의 모든 원소는 속성 P를 가지며,
    - 속성 P를 갖는 모든 원소는 집합 S의 원소이다.

# Venn diagram 그리기

5

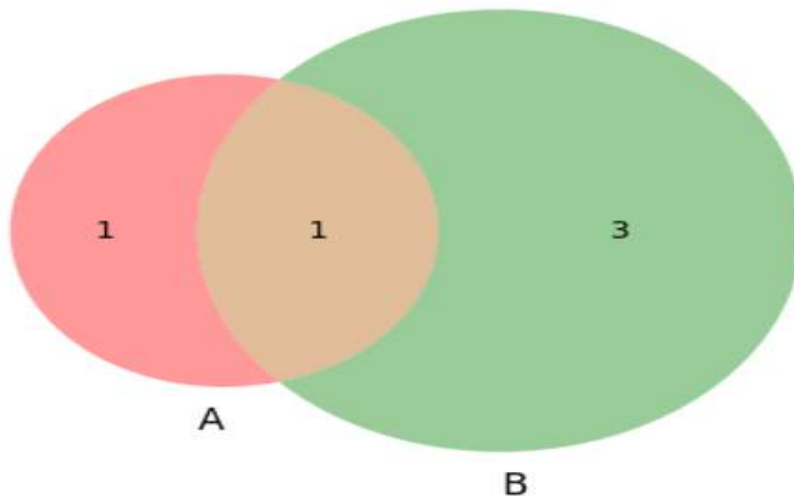
2개의 집합에 대한 숫자는 사이즈를 표시하는 벤 다이어그램 그리기

```
from matplotlib import pyplot as plt
from matplotlib_venn import venn2 |

v = venn2(subsets=[set([1,2]), set([2,3,4,5])], set_labels = ('A', 'B'))

plt.title(" Venn diagram")
plt.show()
```

Venn diagram



# Venn diagram : set\_text

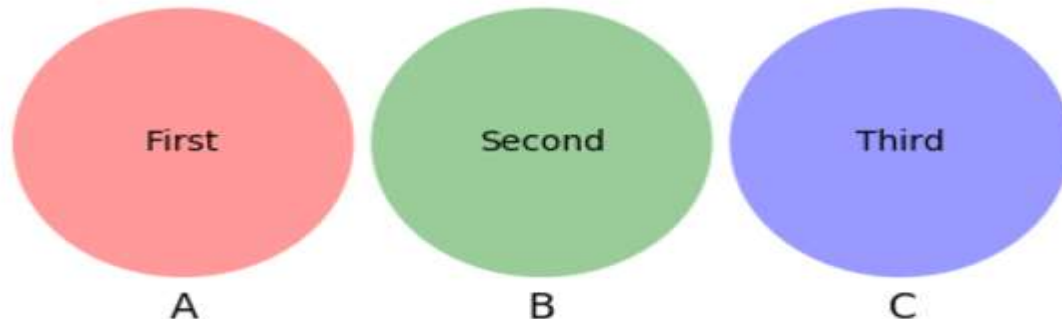
6

각 집합에 대한 내부 텍스트를 표시

```
from matplotlib import pyplot as plt
from matplotlib_venn import venn2, venn2_circles

v = venn3(subsets=(1,1,0,1,0,0,0),set_labels=('A', 'B','C'))
v.get_label_by_id('100').set_text('First')
v.get_label_by_id('010').set_text('Second')
v.get_label_by_id('001').set_text('Third')
plt.title("Not a Venn diagram")
plt.show()
```

Not a Venn diagram

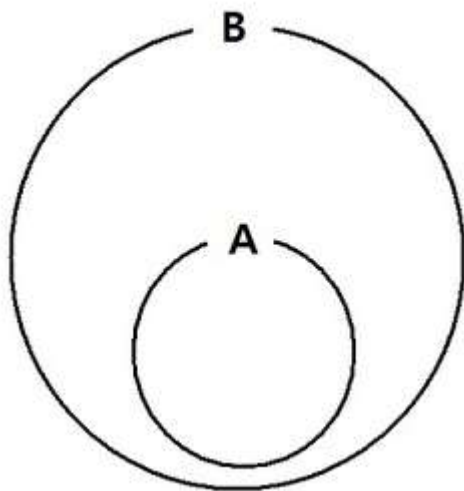


# 부분집합

7

집합 B의 부분집합(部分集合, subset) A는, 모든 원소가 B에도 속하는 집합, 관계를 주로  $A \subseteq B$ 라 표기

$A \subset B$

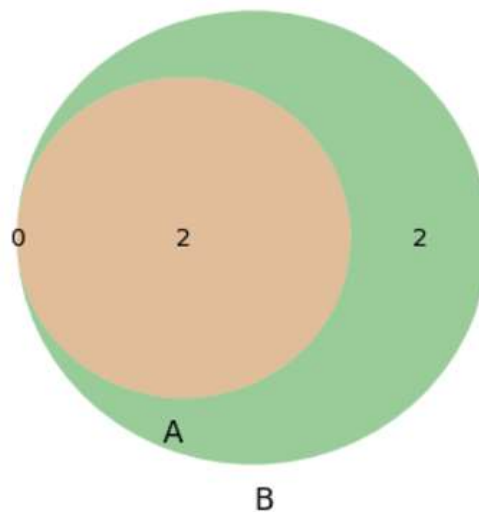


```
from matplotlib import pyplot as plt
from matplotlib_venn import venn2

v = venn2(subsets=[set([2,3]), set([2,3,4,5])], set_labels = ('A', 'B'))

plt.title(" Venn diagram")
plt.show()
```

Venn diagram



# 부분집합의 개수

8

## 집합에 대한 부분집합의 개수

[부분집합의 개수 구하는 공식]

$A = \{a_1, a_2, a_3, \dots, a_n\}$  에서

① A의 모든 부분집합의 개수

$$2^n$$

집합 A의 원소 개수

② 특정한 원소 m개를 포함하는 A의 부분집합의 개수

$$2^{n-m}$$

집합 A의 원소 개수  
- 포함된 원소 m개

③ 특정한 원소 l개를 포함하는 않는 A의 부분집합의 개수

$$2^{n-l}$$

집합 A의 원소 개수  
- 포함하지 않는 원소 l개

④ 특정한 원소 m개를 포함, 다른 l개를 포함하지 않는 A의 부분집합의 개수

$$2^{n-m-l}$$

집합 A의 원소 개수  
- 포함하는 원소 m개  
- 포함하지 않는 원소 l개

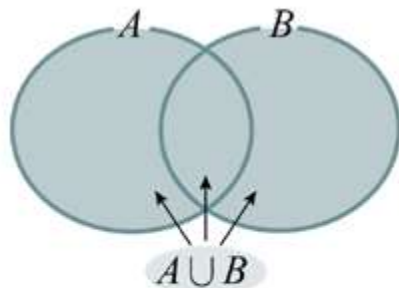


# 합집합

9

두 집합  $A$ ,  $B$ 의 합집합  $A \cup B$ 는,  $A$ 에 속하거나  $B$ 에 속하는 원소들로 이루어진 집합이다.

$$A \cup B = \{x : x \in A \vee x \in B\}$$



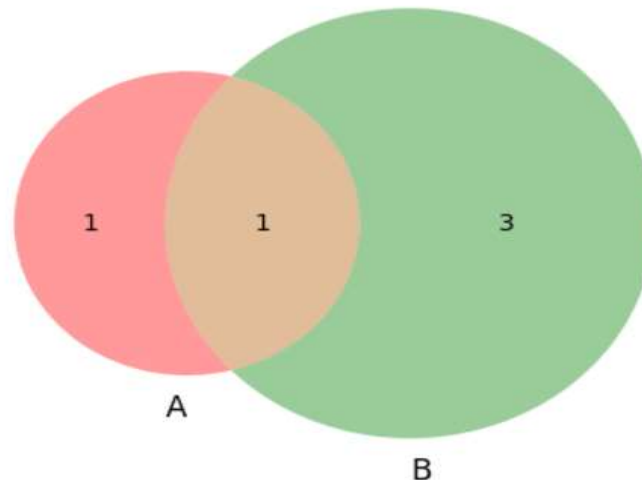
$A$ 와  $B$ 의 합집합  
 $A \cup B = \{x/x \in A \text{ 또는 } x \in B\}$

```
from matplotlib import pyplot as plt
from matplotlib_venn import venn2

v = venn2(subsets=[set([2,6]), set([2,3,4,5])], set_labels = ('A', 'B'))

plt.title(" Venn diagram")
plt.show()
```

Venn diagram

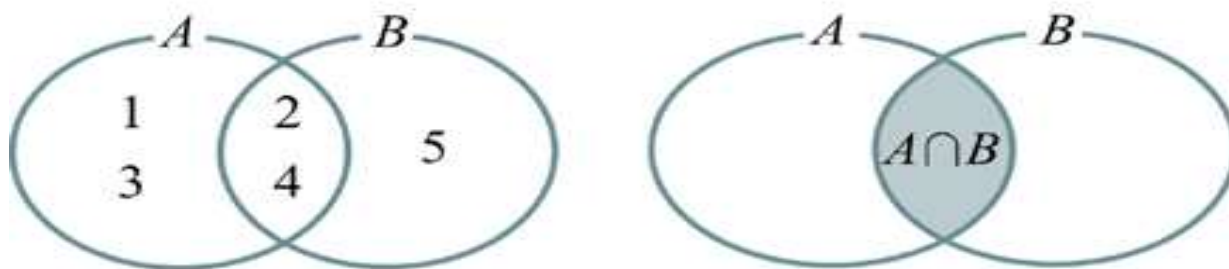


# 교집합

10

두 집합  $A$ ,  $B$ 의 교집합은  $A \cap B$ 로 표기하며,  $A$ 에도 속하고  $B$ 에도 속하는 원소들을 골라놓은 집합을 뜻한다.

$$A \cap B = \{x : x \in A \text{ and } x \in B\}$$



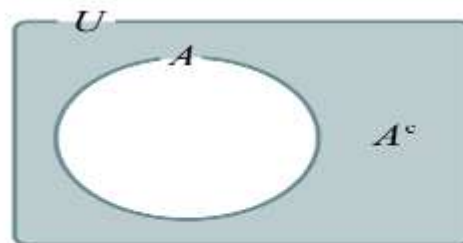
$A$ 와  $B$ 의 교집합  
 $A \cap B = \{x | x \in A \text{ 그리고 } x \in B\}$

# 여집합

11

여집합(餘集合, 또는 보집합(補集合), complement set)은 전체 집합  $U$ 가 정의되었을 때, 그의 부분집합 집합  $A$ 의 여집합은  $A^C$  표기

$$A^C = \{x \in U : x \notin A\}$$

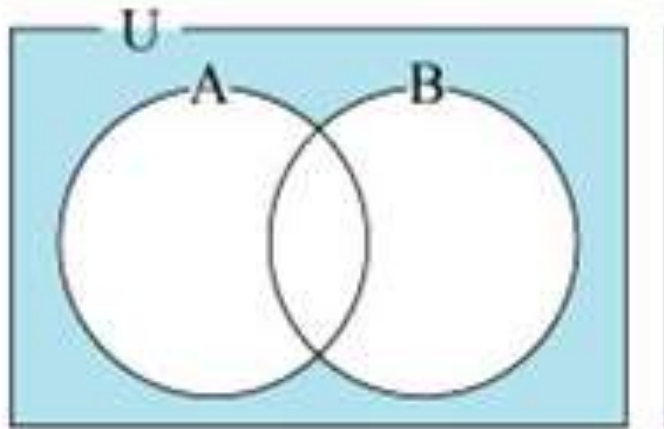


$$A^C = \{x \mid x \in U \text{이고, } x \notin A\}$$

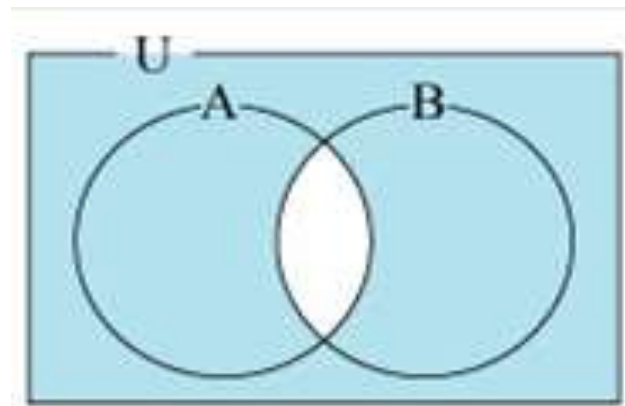
# 합집합과 교집합의 여집합

12

두 집합의 합집합의 여집합은 두 집합의 여집합의 교집합과 동일하며, 두 집합의 교집합의 여집합은 두 집합의 여집합의 합집합과 동일



$$(A \cup B)^c = A^c \cap B^c$$



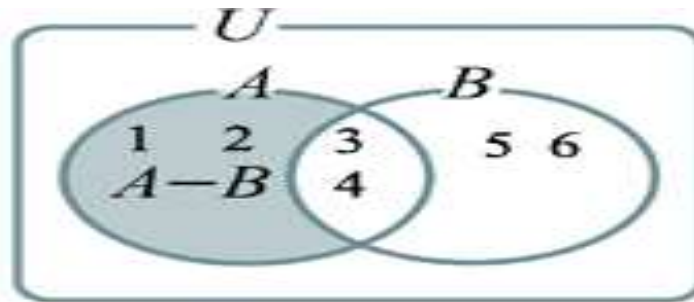
$$(A \cap B)^c = A^c \cup B^c$$

# 차집합

13

차집합(差集合, relative complement, set difference)은 집합  $B$ 에 대한  $A$ 의 차집합은  $B \setminus A$  또는  $B - A$ 로 표기

$$B \setminus A = \{x \in B : x \notin A\}$$



$A$ 에 대한  $B$ 의 차집합  
 $A-B = \{x | x \in A \text{ 그리고 } x \notin B\}$

# 추이적 관계

14

집합  $A, B, C$  대하여 정의된 이항관계 추이적 관계(推移的關係, transitive relation)

$$A \subseteq B \text{이고 } B \subseteq C \text{이면 } A \subseteq C$$

$$A = B \text{이고 } B = C \text{이면 } A = C$$

# Set 타입

15

중복을 허용하지 않고, 순서가 없음 (unordered)

교집합(&), 합집합(|), 차집합(-) 연산 처리

Set: mutable set

Frozenset: immutable set

```
for i in dir(set) :  
    if i[0:2] == "__" :  
        pass  
    elif i[0:1] == "_" :  
        pass  
    else :  
        print i,
```

add clear copy difference difference\_update discard intersection intersection\_update isdisjoint issubset issuperset pop remove symmetric\_difference symmetric\_difference\_update union update

# Set 타입 - 생성시 주의

16

내부 원소가 리스트 등 mutable 값이 들어갈 수 없음

```
s = set([1],[2])
```

```
-----  
TypeError                                 Traceback (most recent call last)  
<ipython-input-79-8e1820bbc476> in <module>()  
----> 1 s = set([1],[2])  
  
TypeError: set expected at most 1 arguments, got 2
```

Set 생성은 파라미터가  
1개만 가능

```
s1 = set(([1],[2]))
```

```
-----  
TypeError                                 Traceback (most recent call last)  
<ipython-input-80-24e087f69a30> in <module>()  
----> 1 s1 = set(([1],[2]))  
  
TypeError: unhashable type: 'list'
```

Set은 구조상 내부 요소  
가 변경이 가능한 값으  
로 구성할 수 없다



# Set 타입 - 생성

17

Set()으로 생성시 파라미터는 1개만 받는다. 리스트 등 mutable 객체를 요소로 처리할 수 없음

```
s = set(((1),(2)))  
print s  
  
sm = set({'a':1})  
print sm  
  
si = set((1,2,3))  
print si.pop()  
print si
```

```
set([1, 2])  
set(['a'])  
1  
set([2, 3])
```

# Set 타입 - 검색

18

Set 타입은 index/slice 검색을 제공하지 않으므로 내부 원소는 for문을 통해 원소를 조회

```
s = set([1,2,3])  
print(s[0])
```

```
-----  
TypeError                                 Traceback  
<ipython-input-53-17e4f1183b88> in <module>()  
      1 s = set([1,2,3])  
>>> 2 print(s[0])  
  
TypeError: 'set' object does not support indexing
```

```
s = set([1,2,3])  
for i in s :  
    print i
```

```
1  
2  
3
```

Index 검색 오류 발생

# 수열과 극한

Moon Yong Joon

# 급수(級數, series, $\sum a_n$ )

20

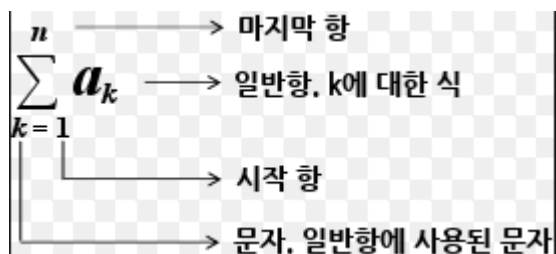
수학에서의 급수는 수열  $a_1, a_2, a_3, \dots, a_n$ 까지 주어졌을 때 이것들을 다 더해  $a_1 + a_2 + a_3 + \dots + a_n$ ,  $a_1 + a_2 + a_3 + \dots + a_n$ 로 나타낸 것, 즉 수열의 합을 의미한다.

급수의 예는 등차수열, 등비수열의 합, 자연수의 거듭제곱의 합 등이 있다

# 급수의 표현 1

21

$\Sigma$  는 일반항 식을 가지고 시작항과 마지막 항까지의 합을 표시하는 수학식



$\sum_{k=1}^n a_k$

- $n$  → 마지막 항
- $a_k$  → 일반항,  $k$ 에 대한 식
- $1$  → 시작 항
- $a$  → 문자, 일반항에 사용된 문자

$\Sigma$  의 뜻  $\rightarrow a_1 + a_2 + a_3 + \cdots + a_n = \sum_{k=1}^n a_k$

$$a_1^3 + a_2^3 + a_3^3 + \cdots + a_n^3 = \sum_{k=1}^n a_k^3$$

$$(a_1 + a_2 + a_3 + \cdots + a_n)^2 = \left( \sum_{k=1}^n a_k \right)^2$$

# 급수의 표현 2

22

급수는 연속된 숫자들이 합을 구하는 것에 대한 일반식과의 비교

$$(1) \sum_{k=1}^n k = 1 + 2 + 3 + \dots + n = \frac{n(n+1)}{2}$$

$$(2) \sum_{k=1}^n k^2 = 1^2 + 2^2 + 3^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

$$(3) \sum_{k=1}^n k^3 = 1^3 + 2^3 + 3^3 + \dots + n^3 = \left\{ \frac{n(n+1)}{2} \right\}^2$$

# 급수의 일반적인 성질

23

## 급수간의 계산을 위해 필요한 산식

$$(1) \sum_{k=1}^n (a_k + b_k) = \sum_{k=1}^n a_k + \sum_{k=1}^n b_k$$

$$(2) \sum_{k=1}^n (a_k - b_k) = \sum_{k=1}^n a_k - \sum_{k=1}^n b_k$$

배분의 가능

$$(3) \sum_{k=1}^n c a_k = c \sum_{k=1}^n a_k \text{ (단, } c \text{는 상수)}$$

상수는  $\Sigma$  밖으로 배출

$$c a_1 + c a_2 + \cdots + c a_n = c(a_1 + a_2 + \cdots + a_n) = c \sum_{k=1}^n a_k$$

$$(4) \sum_{k=1}^n c = cn \text{ (단, } c \text{는 상수)}$$

상수는  $n$ 번 덧셈

$$\sum_{k=1}^n c = c + c + c + \cdots + c = cn$$

# 급수의 일반적인 성질

24

급수 내에 곱과 나눗셈에 대해서는  $\Sigma$ 를 배분할 수 없음

$$\sum_{k=1}^{\infty} a_k b_k \neq \sum_{k=1}^{\infty} a_k \sum_{k=1}^{\infty} b_k$$

급수의 곱

$$\sum_{k=1}^{\infty} \frac{a_k}{b_k} \neq \frac{\sum_{k=1}^{\infty} a_k}{\sum_{k=1}^{\infty} b_k}$$

급수의 나눗셈



# 유한급수, 무한급수

25

수열의 모든 항을 더한 것이다. 항의 개수가 유한한 유한 급수(有限級數, finite series)와 항의 개수가 무한한 무한 급수(無限級數, infinite series)로 분류됨

유한 급수(有限級數, finite series)

급수  $\sum_{n=0}^{\infty} a_n$ 의 부분합(部分和, 영어: partial sum)  $\sum_{n=0}^N a_n$ 은 처음 오는 유한 개의 항에 대한 합이다. 즉,

$$\sum_{n=0}^N a_n = a_0 + a_1 + a_2 + \cdots + a_N,$$

무한 급수(無限級數, infinite series)

수열  $(a_n)_{n=0}^{\infty}$ 에 대한 (무한) 급수  $\sum_{n=0}^{\infty} a_n$ 은, 수열의 항들의 형식적인 합이다. 즉,

$$\sum_{n=0}^{\infty} a_n = a_0 + a_1 + a_2 + \cdots$$

# 등차급수

26

등차급수[ arithmetic series , 等差級數 ]

산술급수(算術級數)라고도 한다. 등차수열을 이루고 있는 것을 말 함

급수  $a_1 + a_2 + a_3 + \dots + a_n + \dots$ 에서  
 $a_n = a_{n-1} + d (n = 2, 3, \dots)$  인 관계식

$$\frac{n(a_1 + a_n)}{2} = \frac{n}{2} \{2a_1 + (n-1)d\}$$



$$\sum_{n=1}^{\infty} \{a + (n-1)d\}$$

급수로 표시

# 등비급수

27

기하급수(幾何級數)라고도 한다.

급수  $a_1 + a_2 + a_3 + \dots + a_n + \dots$ 에 있어  
 $a_n = ar^{n-1}$  ( $n = 2, 3, \dots$ )인 관계식

$$a_1 = a$$

$$a_2 = a_1 \times r = a \times r$$

$$a_3 = a_2 \times r = a \times r^2$$

.

.

.

$$a_n = a_{n-1} \times r = a \times r^{n-1}$$

# 수열

28

수에 대해 일렬로 구성하는 수열이 내부 구성 구조는 합과 곱으로 처리

수열의 합

$$\sum_{\text{문자=시작점의 수열값}}^{\text{끝점의 수열값}} \text{수열의 함수식}$$

수열의 곱

$$\prod_{\text{문자=시작점의 수열값}}^{\text{끝점의 수열값}} \text{수열의 함수식}$$

# 등차수열

29

초기값을 가지고 동일한 값으로 숫자가 증가되는 수열

일반항

$i$ 번째 항을  $a_i$ , 공차를  $d$ 라 하면 등차수열의 일반항은 다음과 같다.

$$a_n = a_i + (n - i)d$$

물론 여기에  $i = 1$ 을 대입하면 잘 알려진 일반항으로 다음을 얻는다.

$$a_n = a_1 + (n - 1)d$$

# 공차

30

등차수열에서 연속하는 두 수의 차이를 공차(公差. Common difference)라고 한다. 보통  $d$ 로 표시한다

등차수열: 첫째항에 일정한 수를 더해서 얻어진 항으로 이루어진 수열  
공차( $d$ ): 각 항에 더해지는 일정한 수

$$a_n = a_{n-1} + d$$

$$d = a_n - a_{n-1}$$

```
x = np.linspace(1,100,10)
cd = x[1] - x[0]

print(x)

print(cd)
print(np.sum(x))
s1 = len(x) * (x[0]+(x[9]))/2
print(s1)
```

```
[ 1.  12.  23.  34.  45.  56.  67.  78.  89. 100.]
11.0
505.0
505.0
```

# 등차 수열의 합 : 등차급수

31

**등차급수**( arithmetic series)는 등차수열의 합이다. 초항부터  $n$ 번째 항까지의 합  $S_n$ 는 다음과 같은 공식으로 나타난다

$$S_n = \frac{n(a_1 + a_n)}{2} = \frac{n[2a_1 + (n-1)d]}{2}$$

# 등차 수열의 합의 산식

32

하나의 등차수열과 이 등차수열을 역으로 변환해서 더하며 동일한 패턴이 값이 나오고 이를 다시 하나(2로 나눔)로 만들면 합산 값이 됨

등차수열을 역순으로 만들고 합산하면 동일한 결과를 가진 패턴이  
나옴

$$\begin{array}{r} S_n = a + (a + d) + (a + 2d) + \dots + (l - 2d) + (l - d) + l \\ +) S_n = l + (l - d) + (l - 2d) + \dots + (a + 2d) + (a + d) + a \\ \hline 2S_n = (a + l) + (a + l) + (a + l) + \dots + (a + l) + (a + l) + (a + l) \end{array}$$

패턴 찾을 것을 2로 나누면 수열  
의 합이 계산됨

$$\begin{aligned} 2S_n &= n(a + l) \\ S_n &= \frac{n(a + l)}{2} \end{aligned}$$



# 등차수열의 합 1

33

첫번째항과 마지막 항을 알고 둘 사이의 개수를  
알 경우

```
2S = (1 + 10) + (2 + 9) + (3 + 8) + ... + (8 + 3) + (9 + 2) + (10 + 1)
2S = 11 + 11 + 11 + ... + 11 + 11 + 11
2S = 11 × 10
S = 55
```

$$S_n = \frac{n(a_1 + a_n)}{2}$$

```
x = np.linspace(1,10,10)
y = np.linspace(1,10,10)
z = sorted(y,reverse=True)
```

```
print( x + z )
print( np.sum(x+z)/2)
```

```
[ 11.  11.  11.  11.  11.  11.  11.  11.  11.  11.]
55.0
```

# 등차수열의 합 2

34

첫째항과 공차만 있을 경우 처리하는 방식  
첫번째항(a), 마지막항(  $a*(n-1)*d$  )을 더해서 총  
개수만큼 곱하고 이를 2로 나눈면 됨

$$\text{첫째항이 } a, \text{ 공차가 } d \text{ 일 때: } S_n = \frac{n\{2a + (n - 1)d\}}{2}$$

```
a = 1
d = 1
n = 10

s1 = n*(a+(a*d*n))/2
print(s1)
s = (n*(2*a + (n-1)*d)) / 2
print(s)
```

55  
55

# 초기값이 중간부터 오는 경우

35

초기값이 중간부터 올 경우 계산하는 급수

$$\frac{n[2a_1 + (n-1)d]}{2}$$

초기값  $a = 11$  이고 공차  $d = 1$  이면서  
개수  $n = 5$  인 급수를 계산하기  
 $11 + 12 + 13 + 14 + 15$

$$\begin{aligned}\text{산식} &= na + n/2(n-1)d \\ &= 5*11 + 5/2(5-1)*1 \\ &= 55 + 10 \\ &= 65\end{aligned}$$

# 등비수열

36

초기값을 가지고 동일한 값으로 곱해서 얻어지는 수열

일반항

첫항이  $a$ 이며, 공비가  $r$ 인 등비수열의  $n$ 번째 항은 다음과 같다.

$$a_n = ar^{n-1}$$

등비수열은 다음과 같은 점화식으로 표현될 수 있다.

$$a_n r = a_{n+1}$$

$$a_n = a_{n-1} \times r$$

$$r = \frac{a_n}{a_{n-1}}$$

(단,  $a_1 \neq 0, r \neq 0$ )

# 공비

37

등비수열에서 연속하는 두 수의 차이를 공비 (Common ratio)라고 한다. 보통  $r$ 로 표시한다

$$r = \frac{a_n}{a_{n-1}}$$

(단,  $a_1 \neq 0, r \neq 0$ )

```
x = np.ones(10)
print(x)
n= np.linspace(1,10,10)
r = 2**n
y = x *r
print(y)
print("common ration ",y[1]/y[0])
```

```
[ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
[  2.   4.   8.  16.  32.  64. 128. 256. 512. 1024.]
('common ration ', 2.0)
```

# 등비 수열의 합 : 등비급수

38

$a_1$ 부터  $a_n$ 까지 더한 합인 **등비급수**(geometric series) 또는 **기하급수**  $S_n$ 은 다음과 같이 구할 수 있다.

$$\begin{aligned} S_n &= a + ar^1 + ar^2 + ar^3 + \cdots + ar^{n-1} \\ &= a(1 + r^1 + r^2 + \cdots + r^{n-1}) \end{aligned}$$

여기에서  $r$ 의 값이 1이 아니라면, 다음과 같이 정리할 수 있다.

$$\begin{aligned} S_n &= a \frac{(1 + r^1 + r^2 + \cdots + r^{n-1})(r - 1)}{r - 1} \\ &= a \frac{r^n - 1}{r - 1} = a \frac{1 - r^n}{1 - r} \end{aligned}$$

# 등비수열의 합

39

첫번째과 등비를 알 경우 계산이 가능 단 등비가 1 일 경우는 첫번째 항에 개수만큼 곱하면 됨

$$\begin{array}{r}
 S_n = a + ar + ar^2 + ar^3 + \dots + ar^{n-1} \\
 -) rS_n = ar + ar^2 + ar^3 + \dots + ar^{n-1} + ar^n \\
 \hline
 (1-r)S_n = a - ar^n
 \end{array}$$

$$\therefore (1-r)S_n = a - ar^n$$

$$S_n = \frac{a - ar^n}{1 - r} = \frac{ar^n - a}{r - 1} = \frac{a(r^n - 1)}{r - 1}$$

동일한 값으로 처리를 위해 등비로 양변을 곱하고 빼면

초항부터  $n$ 항까지의 합은 이 공식으로 나타낼 수 있다.

$\frac{a(1-r^n)}{1-r}$  인데, 편의상  $\frac{a(r^n-1)}{r-1}$  를 사용해도 된다.

단,  $r = 1$ 인 경우,  $na$ 로 표현한다.

# 등비수열의 합 예시

40

등비수열의  $S_n$  합은  $a_n$ (다음항) -  $a$ (초항)/(등비-1)로 처리

$$a_1 = a_1$$

$$a_2 = a_1 \times r$$

$$a_3 = a_2 \times r = (a_1 \times r) \times r = a_1 \times r^2$$

$$a_4 = a_3 \times r = (a_1 \times r^2) \times r = a_1 \times r^3$$

$$r = 2$$

$$a_1 = 2$$

$$a_2 = 4$$

$$a_3 = 8$$

$$a_4 = 16$$

$$\begin{aligned} S_n &= \frac{a(r^n - 1)}{r - 1} \\ &= \frac{ar^n - a}{r - 1} \\ &= \frac{ar^{n-1} \cdot r - a}{r - 1} \\ &= \frac{rl - a}{r - 1} \quad (\because ar^{n-1} = l) \end{aligned}$$

$$S_4 = 2 + 4 + 8 + 16 = 30$$

$S_5$ 는

$$\text{다음항 } a_5 = 32$$

$$S_5 = 32 - 2/(2-1)$$



# 등비수열의 합 예시

41

첫번째과 등비를 알 경우 계산이 가능 단 등비가 1 일 경우는 첫번째 항에 개수만큼 곱하면 됨

초항부터  $n$ 항까지의 합은 이 공식으로 나타낼 수 있다.

$\frac{a(1-r^n)}{1-r}$  인데, 편의상  $\frac{a(r^n-1)}{r-1}$  를 사용해도 된다.

단,  $r=1$ 인 경우,  $na$ 로 표현한다.

```
x = np.ones(10)
print(x)
n= np.linspace(1,10,10)
r = 2**n
y = x *r
print(y)
print(np.sum(y))
print("common ration ",y[1]/y[0])
cr = y[1]/y[0]
print(cr)
a = y[0]
print(a)
nn = len(n)
print(nn)

zz =(a*(1-cr**nn))/(1-cr)
print(zz)
```

[ 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]  
[ 2. 4. 8. 16. 32. 64. 128. 256. 512. 1024.]  
2046.0  
(common ration ', 2.0)  
2.0  
2.0  
10  
2046.0

# 수열의 극한

42

수열의 극한(數列-極限, limit of a sequence)은 수열이 한없이 가까워지는 값이다.

직관적으로, 수열  $(a_n)$ 이 수렴(收斂, convergent)한다는 것은  $n$ 이 커짐에 따라  $a_n$ 이 어떤 고정된 값  $a$ 에 한없이 가까워지는 것을 뜻한다. 이때  $a$ 를 수열  $\{a_n\}$ 의 극한이라고 한다.

수열이 수렴하지 않으면 발산(發散, divergent)한다고 한다.

# 부분합 급수

43

무한급수 중에 특정 위치까지의 유한개의 항에 대한 합을 나타내는 급수

급수  $\sum_{n=0}^{\infty} a_n$ 의 부분합(部分和, 영어: partial sum)  $\sum_{n=0}^N a_n$ 은 처음 오는 유한 개의 항에 대한 합이다. 즉,

$$\sum_{n=0}^N a_n = a_0 + a_1 + a_2 + \cdots + a_N,$$

# 무한수열과 극한

44

무한수열  $\{a_n\}$ 에서  $n$ 이 무한히 커짐에 따라  $a_n$ 이 일정한 값  $\alpha$ 에 한없이 가까워지면,  $\alpha$ 를 그 수열의 극한 또는 극한값(limit value)

$$\lim_{n \rightarrow \infty} \sum_{1}^n a_n = \lim_{n \rightarrow \infty} S_n$$

부분합

$$\lim_{n \rightarrow \infty} a_n = a \text{ 또는 } n \rightarrow \infty \text{일 때 } a_n \rightarrow a$$

$n$ 이 한없이 커진다:  $n \rightarrow$  무한대  $\infty$

$a_n$  한없이  $\alpha$ 에 가까워진다:  $a_n \rightarrow \alpha$

무한수열의 극한값  $\alpha$ 이다  $\rightarrow \lim_{n \rightarrow \infty} a_n = \alpha$

# 수렴

45

한 점으로 모인다는 뜻. 보통 의견 수렴이라든지 여론 수렴 등등으로 해서 한 점에 모인다는 의미로 사용하는 경우가 많은데, 이 뜻을 수학으로 빌려와서 여러 값이 기어코야 한 값으로 모이게 되었다는 의미로 사용한다.

즉  $x$ 가  $a$ 에 한없이 가까워지거나 한없이 커지거나 작아지면  $f(x)$ 도 어디로 한없이 가까워진다는 뜻.

$$\sum_{n=1}^{\infty} a_n \quad \text{이 수렴한다.} \Leftrightarrow \lim_{n \rightarrow \infty} S_n = \lim_{n \rightarrow \infty} \sum_{k=1}^n a_k \quad \text{이 존재하고 수렴한다.}$$

$$\text{수렴} \quad \lim_{n \rightarrow \infty} a_n = a \quad (\text{단, } a \text{는 실수})$$

# 발산

46

수렴하지 않으면 발산한다. 수열이 계속 커지는 양의 무한대로 발산, 수열이 계속 작아지는 음의 무한대로 발산이 있다.

$$\sum_{n=1}^{\infty} a_n \text{ 이 발산한다. } \Leftrightarrow \lim_{n \rightarrow \infty} S_n = \lim_{n \rightarrow \infty} \sum_{k=1}^n a_k \text{ 이 발산 (무한대 or 마이너스 무한대 or 진동) 한다.}$$

발산	{	양의 무한대로 발산: $\lim_{n \rightarrow \infty} a_n = \infty$
		음의 무한대로 발산: $\lim_{n \rightarrow \infty} a_n = -\infty$
		진동

# 극한의 기본 성질

47

수열이 수렴할 때 이 기본 성질이 사용됨

## 수열의 극한에 대한 기본 성질

두 수열  $\{a_n\}, \{b_n\}$ 이 수렴하고  $\lim_{n \rightarrow \infty} a_n = a, \lim_{n \rightarrow \infty} b_n = \beta$ 일 때,

①  $\lim_{n \rightarrow \infty} ca_n = c \lim_{n \rightarrow \infty} a_n = ca$  (단,  $c$ 는 상수)

②  $\lim_{n \rightarrow \infty} (a_n + b_n) = \lim_{n \rightarrow \infty} a_n + \lim_{n \rightarrow \infty} b_n = a + \beta$

③  $\lim_{n \rightarrow \infty} (a_n - b_n) = \lim_{n \rightarrow \infty} a_n - \lim_{n \rightarrow \infty} b_n = a - \beta$

④  $\lim_{n \rightarrow \infty} a_n b_n = \lim_{n \rightarrow \infty} a_n \cdot \lim_{n \rightarrow \infty} b_n = a\beta$

⑤  $\lim_{n \rightarrow \infty} \frac{a_n}{b_n} = \frac{\lim_{n \rightarrow \infty} a_n}{\lim_{n \rightarrow \infty} b_n} = \frac{a}{\beta}$  (단,  $b_n \neq 0, \beta \neq 0$ )

# 미적분

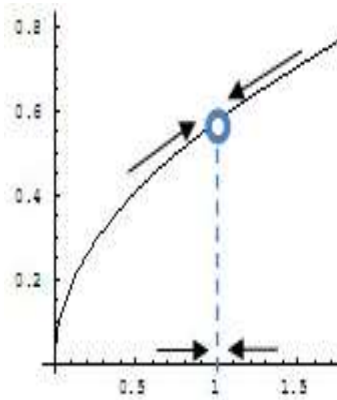
Moon Yong Joon



# 함수의 극한

49

함수의 극한( limit of a function)은 어떤 점에 가까이 다다름에 따른, 함수의 행태에 대한 개념



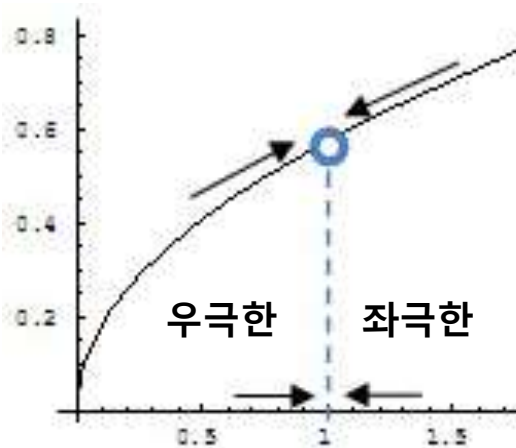
함수  $f = (x)$ 가  $x = a$ 에서 연속일 조건

- 1)  $f(a)$  가 존재함
- 2)  $\lim_{x \rightarrow a} f(x)$  가 존재함
- 3)  $\lim_{x \rightarrow a} f(x) = f(a)$

# 좌극한값 / 우극한값

50

그래프에서 좌측과 우측에서 접근하는 것에 따라  
값을 구분



좌측에서 접근하는 것을 수식으로 나타내면

$$\lim_{x \rightarrow a^-} f(x) : \text{좌극한값}$$

우측에서 접근하는 것을 수식으로 나타내면

$$\lim_{x \rightarrow a^+} f(x) : \text{우극한값}$$

# 함수의 연속

51

함수  $f(x)$ 가 정의역의 한점  $a$ 에 대해 다음 세 조건을 만족하면 함수  $f(x)$ 는  $x=a$ 에서 연속되므로 연속함수

- 함수  $f(x)$ 가  $x=a$ 에서 정의되어 있다.
- 극한값  $\lim_{x \rightarrow a} f(x)$ 가 존재한다.
- $\lim_{x \rightarrow a} f(x) = f(a)$

# 함수의 극한의 수렴

52

함수의 극한은 독립 변수가 일정한 값에 한없이 가까워질 때, 함수값이 한없이 가까워지는 값이 존재하면 수렴한다고 함

함수  $y = f(x)$  에 대하여  
 $x \neq a$  이면서 일정한 값  
 $\alpha$ 에 한없이 가까워질 때  
함수  $f(x)$ 는  $\alpha$ 에 수렴한다고 한다.

$$\lim_{x \rightarrow a} f(x) = \alpha$$

# 함수 극한의 성질

53

수렴하는 함수들이 존재할 때 함수들의 사칙연산을 처리할 수 있음

- $\lim_{x \rightarrow a} (f(x) + g(x)) = \lim_{x \rightarrow a} f(x) + \lim_{x \rightarrow a} g(x)$
- $\lim_{x \rightarrow a} (f(x) - g(x)) = \lim_{x \rightarrow a} f(x) - \lim_{x \rightarrow a} g(x)$
- $\lim_{x \rightarrow a} f(x)g(x) = \lim_{x \rightarrow a} f(x) \lim_{x \rightarrow a} g(x)$

만약 추가로 어떤 구간  $J \ni a$ 에서 항상  $g(x) \neq 0$ 이라면,

- $\lim_{x \rightarrow a} \frac{f(x)}{g(x)} = \frac{\lim_{x \rightarrow a} f(x)}{\lim_{x \rightarrow a} g(x)}$

함수 극한의 성질

$\lim_{x \rightarrow a} f(x) = a$ ,  $\lim_{x \rightarrow a} g(x) = \beta$ 에 수렴할 때,

(1)  $\lim_{x \rightarrow a} \{f(x) + g(x)\} = a + \beta$     (2)  $\lim_{x \rightarrow a} \{f(x) - g(x)\} = a - \beta$

(3)  $\lim_{x \rightarrow a} kf(x) = ka$  ( $k$ 는 상수)    (4)  $\lim_{x \rightarrow a} \{f(x)g(x)\} = a\beta$

(5)  $\lim_{x \rightarrow a} \frac{f(x)}{g(x)} = \frac{a}{\beta}$     (단,  $g(x) \neq 0$ ,  $\beta \neq 0$ )

# 지수함수

54

지수가 무한대로 갈 때 밑이 1보다 크면 무한대이고 밑이 1보다 작으면 -무한대로 갈때 무한대가 됨

- $a > 1$ 의 경우
  - $\lim_{x \rightarrow \infty} a^x = \infty$
  - $\lim_{x \rightarrow -\infty} a^x = 0$
- $0 < a < 1$ 의 경우
  - $\lim_{x \rightarrow \infty} a^x = 0$
  - $\lim_{x \rightarrow -\infty} a^x = \infty$

# limit 함수 : 극한

55

## 극한을 처리하는 함수limit

```
from sympy import limit, sin, Symbol, oo
from sympy.abc import x

# limit 함수 파라미터 : 함수, 변수, limit 값, dir은 극한값으로 접근방향
print(limit(sin(x)/x, x, 0))
# 우 극한값으로 접근, x>0 크면서 접근
print(limit(1/x, x, 0, dir="+"))
#좌 극한값으로 접근, x<0 작으면서 접근
print(limit(1/x, x, 0, dir="-"))

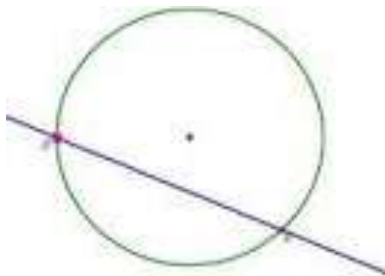
print(limit(1/x, x, oo))
```

```
1
oo
-oo
-
```

# 할선

56

할선(割線, 가름선)은 원 또는 곡선과 두 개 이상의 점에서 만나 그 원이나 곡선을 자르는 직선



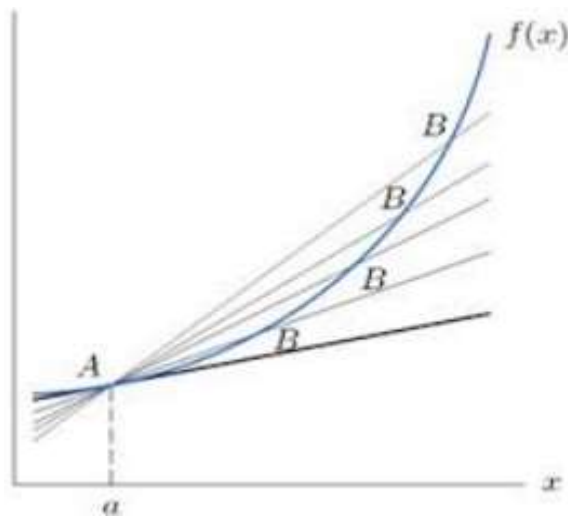


# 접선

57

접선(接線)은 곡선  $L$ 의 두 점  $A$ 와  $B$ 로 정의되는 할선  $AB$ 에서 점  $B$ 가 곡선을 따라 점  $A$ 에 한없이 가까워 질때, 이 새로운 선을 곡선  $L$ 의  $A$ 에서 만나는 접선이라 한다

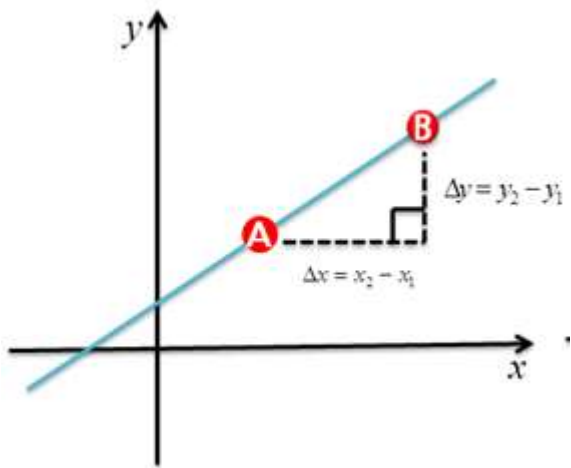
A 점과 만나는  
점을 접점이라고  
함



# 기울기

58

기울기(slope)는 직선이 기울어진 정도를 나타내는 수



$$\text{기울기 } a = \frac{\Delta y (\text{y의 변화량})}{\Delta x (\text{x의 변화량})} = \frac{y_2 - y_1}{x_2 - x_1} = \tan \theta$$

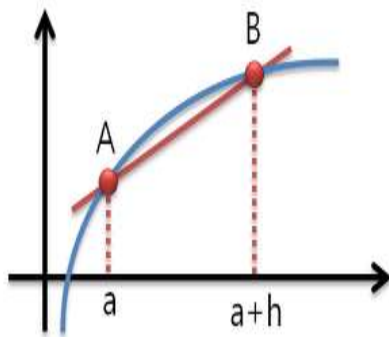
# 평균변화율 : 할선

59

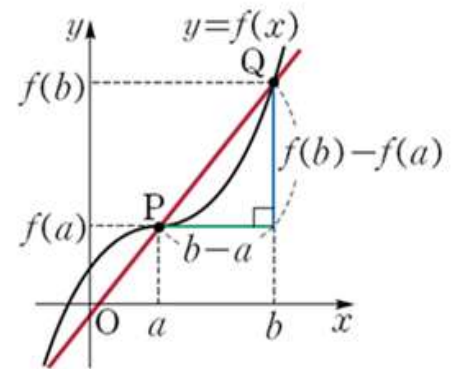
$y=f(x)$  에서  $A(a, f(a)), B(b, f(b))$  두 정점이 있을 때 두 점간의 변화를 계산하면 곡선 위의 두 점의 기울기를 의미

$$\frac{f(b)-f(a)}{b-a} \quad \text{일때} \quad b=a+h \quad \text{라고 두면}$$

$$\frac{f(a+h)-f(a)}{(a+h)-a} = \frac{f(a+h)-f(a)}{h}$$



두 점 A와 B의 기울기를 의미함  
=평균변화율



# 미분기호

60

## 미분 기호 이해하기

$\Delta$ 는 인접한 두 변수의 차이

$d$ 는 두 변수의 극미한 차이

$\partial$ 는 다변수함수에 대해 하나의 변수에 대한 변화량

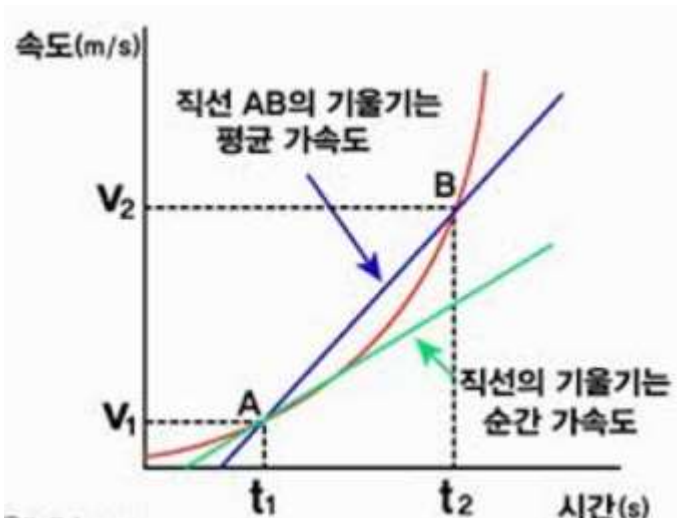
$d(\text{bar})$ 는 통계적인 변수의 변화량

$\delta$ 는 공변의 의미

# 순간변화율 / 미분계수

61

평균변화율의 극한을 취하여 함수  $f(x)$ 의 **특정 지점  $x$** 에서 변화량  $\Delta x$ 가 0으로 수렴할 때의 변화율을 순간변화율 즉 미분계수



$$\begin{aligned} \text{[순간 변화율]} &= \lim_{\Delta x \rightarrow 0} \frac{\Delta y}{\Delta x} = \lim_{x \rightarrow x_1} \frac{f(x) - f(x_1)}{x - x_1} \\ &= \lim_{\Delta x \rightarrow 0} \frac{f(a + \Delta x) - f(a)}{\Delta x} = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h} \end{aligned}$$

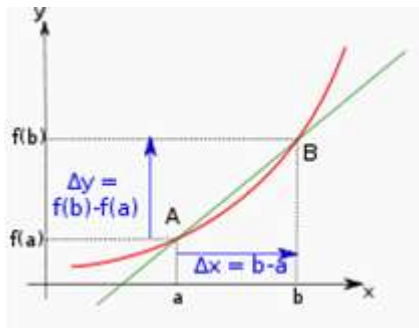
$$\lim_{h \rightarrow 0} \frac{f(a + h) - f(a)}{h} = f'(a) \quad \text{or} \quad \lim_{x \rightarrow a} \frac{f(x) - f(a)}{x - a} = f'(a)$$

# 순간변화율(미분계수) 흐름

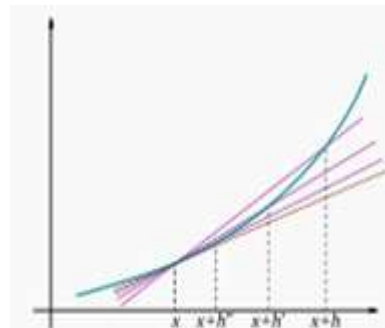
62

## 접점의 기울기를 구하기 위한 흐름

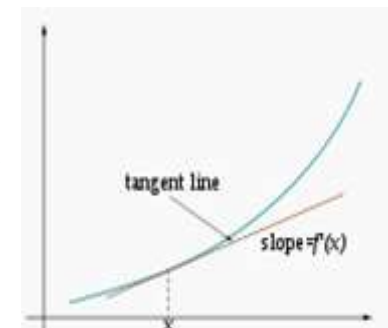
평균변화율 구하기



미분



순가변화율 구하기



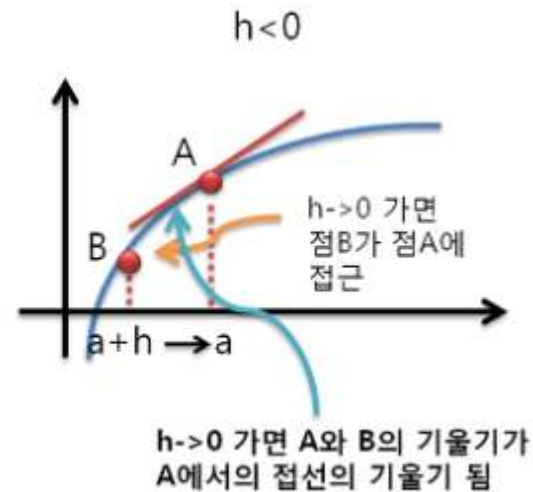
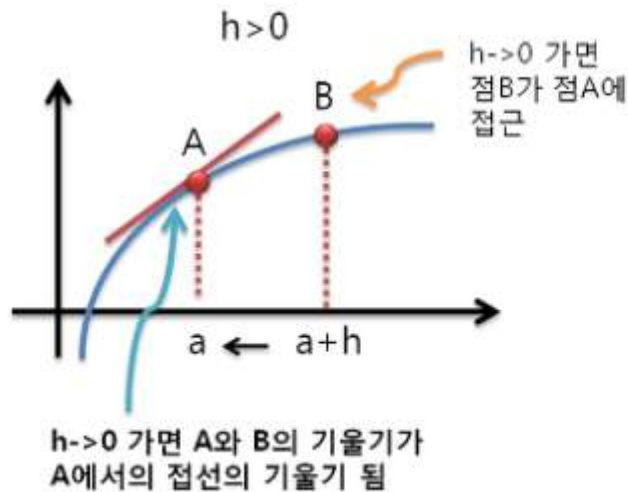
$$\frac{f(a+h) - f(a)}{(a+h) - a}$$

$$\lim_{x \rightarrow a} \frac{f(x) - f(a)}{x - a} = f'(a)$$

# 순간변화율(미분계수)의 그래프

63

B점에서 A점으로 이동하면 한점에서 만나는 접선이  
이 됨



# 미분가능한 함수 : differentiable function

64

미분은 함수  $f(x)$ 의 순간변화율을 계산하는 과정이다.

$f(x)$ 에서 미분계수  $f'(a)$ 가 존재할 때 **미분 가능**이라고 한다.

정의역의 모든 점에서 미분가능한 함수이다.

즉, 모든 곳에서 수직이 아닌 접선을 그릴 수 있다는 것이다



# 미분가능

65

함수  $f$ 가 점  $x_0$ 에서 미분가능하다는 건 그 점에서의 미분, 즉 평균변화율의 극한이 존재하는 것으로 정의된다

---

1. 함수  $f(x)$ 가  $x=a$ 에서 연속이고  $\lim_{x \rightarrow a} f'(x)$  값이 존재하면

함수  $f(x)$ 가  $x=a$ 에서 미분가능하다

2.  $f(x)$ 가  $x=a$ 에서 미분가능하면  $\lim_{x \rightarrow a} f'(x)$ 의 값이 존재한다.

3.  $f(x)$ 가  $x=a$ 에서 미분가능하면  $\lim_{x \rightarrow a} f'(x) = f'(a)$  이다.

$$f'(x_0) = \lim_{x \rightarrow x_0} \frac{f(x) - f(x_0)}{x - x_0}$$

# 미분계수와 도함수 차이

66

미분계수는 특정한 점  $(a, f(a))$ 에서 접선의 기울기를 의미하고 도함수는 임의의 점  $(x, f(x))$ 에서의 접선의 기울기를 의미

$$\begin{aligned} x=1 \text{에서 미분계수는 } f'(1) &= 6 \\ f(x) \text{의 도함수는 } f'(x) &= 3x+3 \end{aligned}$$

도함수에 특정값이 들어가면 미분계수를 구할 수 있음

# 도함수

67

미분(微分)은 함수의 순간변화율을 구하는 계산 과정  
이고 함수를 미분하여 생성되는 함수를 도함수  
즉 미분계수의 일반화

기호
$f'(x)$
$\frac{dy}{dx}, \frac{df(x)}{dx}, \frac{d}{dx}f(x)$

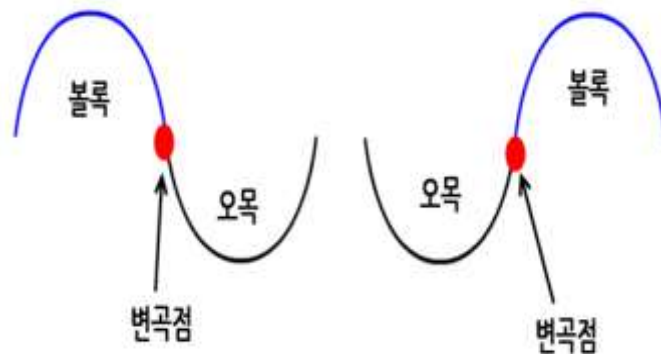
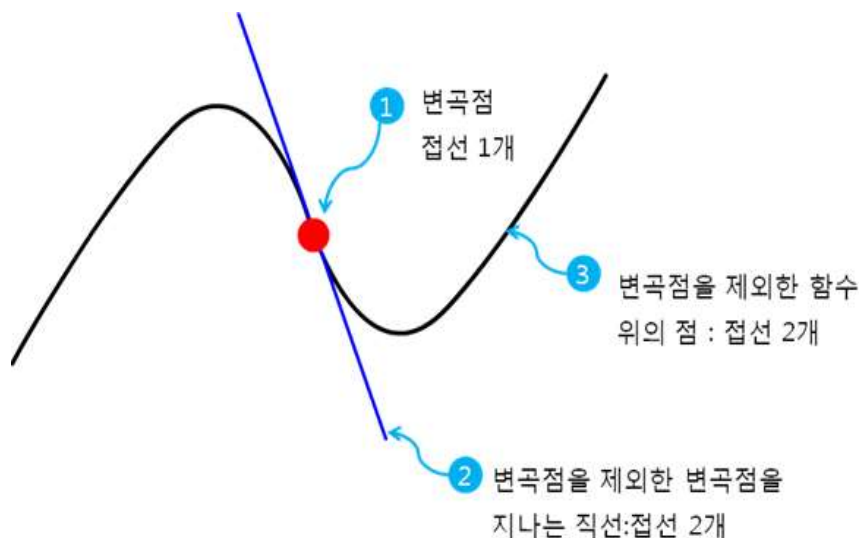
$$\frac{\Delta f(x)}{\Delta x} = \frac{f(x + \Delta x) - f(x)}{\Delta x} \quad \longrightarrow \quad f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

변화율에 대해 limit(극한)  
을 지정하면

# 변곡점

68

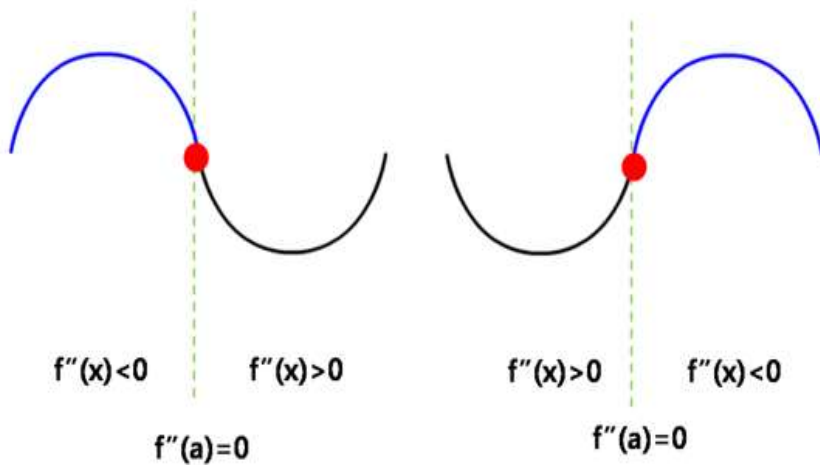
변곡점(point of inflection)은 위로 볼록인 곡선 상태에서 오목인 곡선 상태 또는 위로 오목인 곡선 상태에서 볼록인 곡선상태로 변하는 점



# 변곡점의 판정

69

$f''(a) = 0$ 되고  $x=a$ 의 좌우에서  $f''(x)$ 의 부호가 바뀌면 점  $(a, f(a))$ 는  $y=f(x)$ 의 변곡점



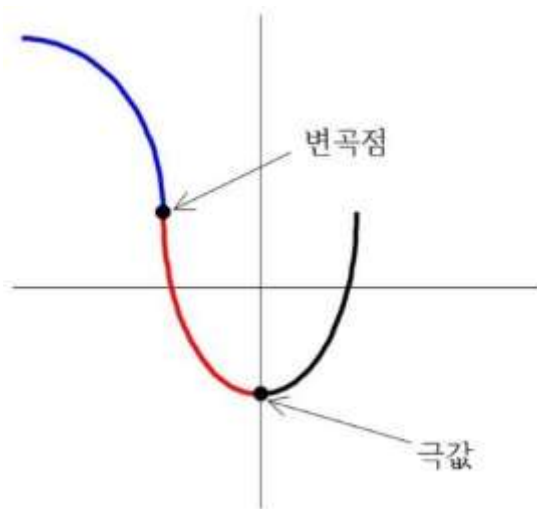
삼차함수가 변곡점에 대칭

- ▶ 극대점과 극소점은 변곡점에 대칭
- ▶ 극대점과 극소점의 중점은 변곡점
- ▶ 극대점과 극소점을 지나는 직선 위에 변곡점이 존재

# 극값

70

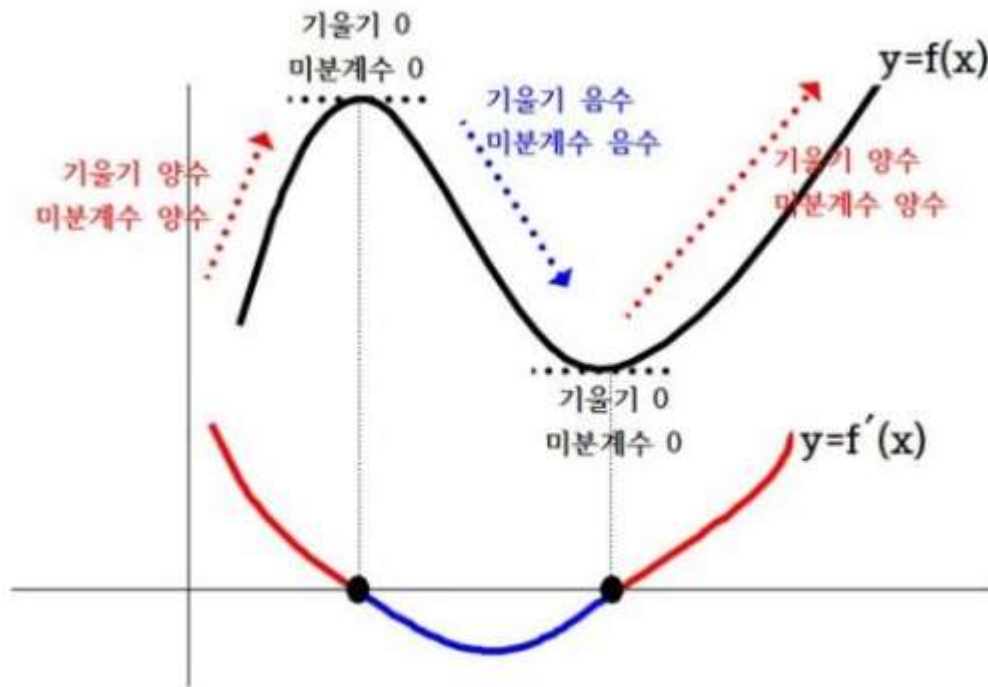
기울기가 양수에서 음수 또는 음수에서 양수로  
전환되는 점 즉 미적분에서 접선기울기가 0인 상  
태



# 그래프 이해하기

71

기울기가 양수이면 위로 상승하는 그래프이고 기울기가 음수이면 아래로 하강하는 그래프이다

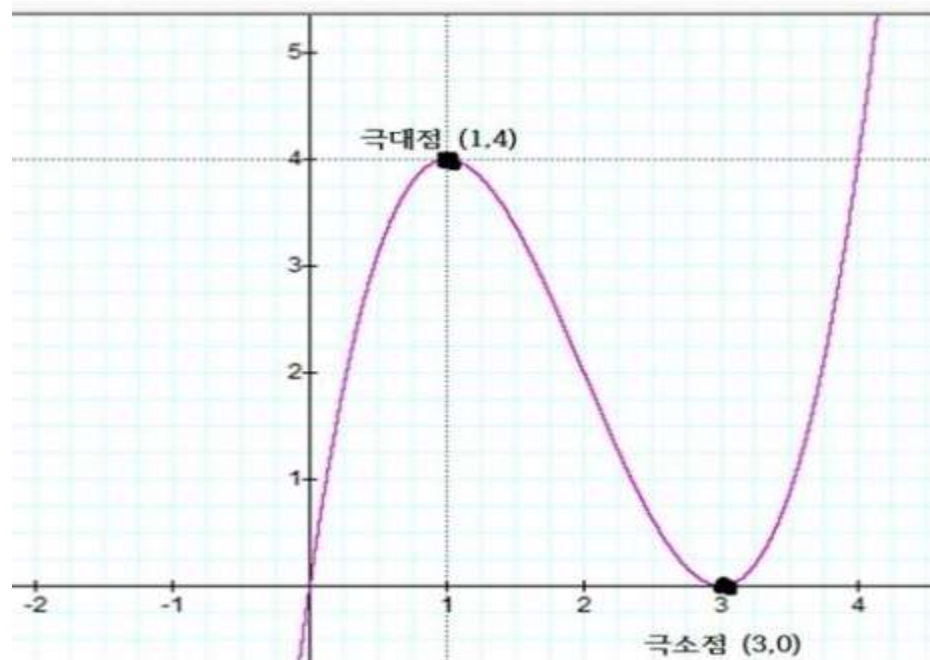


# 극대값, 극소값

72

증가 상태에서 감소상태로 전환하는 극값을 극대값, 감소상태에서 증가상태로 전환하는 극값을 극소값

$$y = x^3 - 6x^2 + 9x$$





# 이계 도함수

73

이계도함수란 도함수의 도함수라는 의미로서, 본 함수를 2번 미분하여 얻은 도함수. 함수  $f(x)$ 의 도함수  $f'(x)$ 가 미분이 가능하면  $f'(x)$ 의 도함수를 이계도함수  $f''(x)$

▶  $f'(x)$ 가 미분가능할때  $f'(x)$ 의 도함수

: 주 사용 용도는 변곡점을 구하거나 오목/볼록을 판단할때 사용됨

$$f''(x) = \lim_{h \rightarrow 0} \frac{f'(x+h) - f'(x)}{h} \quad \leftarrow \text{두번 미분한 함수를 말함}$$

$$y'' = f''(x) = \frac{d^2 y}{dx^2} = \frac{d}{dx} f'(x) \quad \leftarrow \text{이계도 함수를 표현하는 방법}$$

# 미분의 기본 공식 : 1

74

상수의 미분, 실수배 미분, 함수의 미분에 대한 일반 공식

$f(x) = c$  ( 단,  $c$ 는 상수 ) 이면

$$f'(x) = 0$$

$f(x) = x^n$  ( 단,  $n$ 은 자연수 ) 이면

$$f'(x) = nx^{n-1}$$

$\{cf(x)\}' = cf'(x)$  ( 단,  $k$ 는 상수 )

# 상수미분

75

상수의 미분은 실제 0이 됨

$f(x)=c$ ( $c$ 는 상수)

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} = \lim_{\Delta x \rightarrow 0} \frac{c - c}{\Delta x} = 0$$

# 상수미분 : sympy

76

## 상수의 미분은 실제 0이 됨

```
from __future__ import division
from sympy import *

print(diff(2, x))
```

0

```
help(diff)
```

Help on function diff in module sympy.core.function:

```
diff(f, *symbols, **kwargs)
    Differentiate f with respect to symbols.
```

This is just a wrapper to unify `.diff()` and the Derivative class; its interface is similar to that of `integrate()`. You can use the same shortcuts for multiple variables as with Derivative. For example, `diff(f(x), x, x, x)` and `diff(f(x), x, 3)` both return the third derivative of `f(x)`.

You can pass `evaluate=False` to get an unevaluated Derivative class. Note that if there are 0 symbols (such as `diff(f(x), x, 0)`), then the result will be the function (the zeroth derivative), even if `evaluate=False`.

# 실수배 미분

77

c라는 실수배를 가진 함수의 미분에서 실수배는 미분에 영향을 미치지 않는다

$$y = cf(x)$$

$$y' = \lim_{\Delta x \rightarrow 0} \frac{cf(x + \Delta x) - cf(x)}{\Delta x} = c \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} = cf'(x)$$

# 실수배 미분 : sympy

78

c라는 실수배를 가진 함수의 미분에서 실수배는 미분에 영향을 미치지 않는다

$y = cf(x)$

$$y' = \lim_{\Delta x \rightarrow 0} \frac{cf(x + \Delta x) - cf(x)}{\Delta x} = c \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} = cf'(x)$$

```
from __future__ import division
from sympy import *

x = symbols('x')
print(diff(2*x**2, x))
```

4\*x

# 함수의 미분

79

$n$  이 자연수일 경우  $x^n$ 의 미분은  $nx^{n-1}$

인수분해를 이용하여  $f(x) = x^n$  의 도함수를 구하면,

$$\begin{aligned} f'(x) &= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} \\ &= \lim_{\Delta x \rightarrow 0} \frac{(x + \Delta x)^n - x^n}{\Delta x} \\ &= \lim_{\Delta x \rightarrow 0} \frac{(x + \Delta x - x)(x^{n-1} + x^{n-2}(x + \Delta x) + x^{n-3}(x + \Delta x)^2 + \dots + (x + \Delta x)^{n-1})}{\Delta x} \\ &= \lim_{\Delta x \rightarrow 0} (x^{n-1} + x^{n-2}(x + \Delta x) + x^{n-3}(x + \Delta x)^2 + \dots + (x + \Delta x)^{n-1}) \text{ 이때 } \Delta x \text{ 는 } 0 \text{ 에 수렴하므로} \\ &= x^{n-1} + x^{n-1} + \dots + x^{n-1} = nx^{n-1} \quad \text{따라서,} \\ f'(x) &= nx^{n-1} \end{aligned}$$

# 함수의 미분 : sympy

80

$n$  이 자연수일 경우  $x^n$ 의 미분은  $nx^{n-1}$

```
from __future__ import division
from sympy import *

x = symbols('x')
n = symbols('n', integer=True)
n=10
print(diff(x**n,x))

10*x**9
```



# 미분의 기본 공식 : 2

81

합과 차의 미분, 곱의 미분

$$\{f(x) \pm g(x)\}' = f'(x) \pm g'(x)$$

$$\{f(x)g(x)\}' = f'(x)g(x) + f(x)g'(x)$$

# 합과차의 미분 : sympy

82

두 함수 합에 대한 미분은 각 함수의 미분에 합과 같다

$$\begin{aligned}(f(x) + g(x))' &= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) + g(x + \Delta x) - (f(x) + g(x))}{\Delta x} \\&= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x) + g(x + \Delta x) - g(x)}{\Delta x} \\&= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} + \lim_{\Delta x \rightarrow 0} \frac{g(x + \Delta x) - g(x)}{\Delta x} \\&= f'(x) + g'(x)\end{aligned}$$

```
from __future__ import division
from sympy import *
x, y, z, t = symbols('x y z t')
k, m, n = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)

n = 2
f = x**n
g = x**n
h = f + g

print(diff(f,x))
print(diff(h,x))

2*x
4*x
```

# 곱의 미분

83

두 함수 곱에 대한 미분은  $f(x)$  함수 미분과  $g(x)$  함수의 곱 +  $f(x)$  함수와  $g(x)$  함수 미분의 곱과의 합과 같다

$$\begin{aligned}(f(x)g(x))' &= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x)g(x + \Delta x) - (f(x)g(x))}{\Delta x} \\&= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x)g(x + \Delta x) - f(x)g(x + \Delta x) + f(x)g(x + \Delta x) - (f(x)g(x))}{\Delta x} \\&= \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} \lim_{\Delta x \rightarrow 0} g(x + \Delta x) + \lim_{\Delta x \rightarrow 0} \frac{g(x + \Delta x) - g(x)}{\Delta x} \lim_{\Delta x \rightarrow 0} f(x) \\&= f'(x)g(x) + f(x)g'(x)\end{aligned}$$

# 곱의 미분 : sympy

84

두 함수 곱에 대한 미분은  $f(x)$  함수 미분과  $g(x)$  함수의 곱 +  $f(x)$  함수와  $g(x)$  함수 미분의 곱과의 합과 같다

$$\begin{aligned}(f(x)g(x))' &= \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x)g(x+\Delta x) - (f(x)g(x))}{\Delta x} \\&= \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x)g(x+\Delta x) - f(x)g(x+\Delta x) + f(x)g(x+\Delta x) - (f(x)g(x))}{\Delta x} \\&= \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - f(x)}{\Delta x} \lim_{\Delta x \rightarrow 0} g(x+\Delta x) + \lim_{\Delta x \rightarrow 0} \frac{g(x+\Delta x) - g(x)}{\Delta x} \lim_{\Delta x \rightarrow 0} f(x) \\&= f'(x)g(x) + f(x)g'(x)\end{aligned}$$

```
: from __future__ import division
from sympy import *
x, y, z, t = symbols('x y z t')
k, m, n = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)

n = 2
f = x**n
g = x**n
h = f * g

print(diff(f,x))
print(diff(g,x))
print(diff(h,x))
print(diff(f,x)*g+ f*diff(g,x))

2*x
2*x
4*x**3
4*x**3
```

# 단순 함수의 미분 1 : sympy

85

## 단순 함수의 미분

$$\frac{d}{dx}c = 0$$

$$\frac{d}{dx}x = 1$$

$$\frac{d}{dx}|x| = \frac{x}{|x|} = \operatorname{sgn} x, \quad x \neq 0$$

```
: from __future__ import division
from sympy import *
x, y, z, t = symbols('x y z t')
k, m, n = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)
```

```
n = 2
f = 10
g = x
```

```
h = abs(x)
print(h)
```

```
print(diff(f,x))
print(diff(g,x))
print(diff(h,x))
```

```
Abs(x)
```

```
0
```

```
1
```

```
(re(x)*Derivative(re(x), x) + im(x)*Derivative(im(x), x))/Abs(x)
```

# 단순 함수의 미분 2 : sympy

86

## 단순 함수의 미분

$$\frac{d}{dx} x^c = cx^{c-1}$$
$$\frac{d}{dx} \sqrt{x} = \frac{1}{2\sqrt{x}}$$
$$\frac{d}{dx} \left( \frac{1}{x} \right) = -\frac{1}{x^2}$$

```
from __future__ import division
from sympy import *
x, y, z, t = symbols('x y z t')
k, m, n = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)

n = 2
f = 1/x
g = sqrt(x)
h = x**n

print(diff(f,x))
print(diff(g,x))
print(diff(h,x))

-1/x**2
1/(2*sqrt(x))
2*x
```

# 지수 / 로그 함수의 미분

87

## 지수 / 로그 함수의 미분

$$\frac{d}{dx} a^{f(x)} = a^{f(x)} f'(x) \ln a, \quad a > 0$$

$$\frac{d}{dx} c^x = c^x \ln c, \quad c > 0$$

$$\frac{d}{dx} e^x = e^x$$

$$\frac{d}{dx} \log_c x = \frac{1}{x \ln c}, \quad c > 0, c \neq 1$$

$$\frac{d}{dx} \ln x = \frac{1}{x}$$

# 삼각함수의 미분

88

## 삼각함수의 미분

$$\frac{d}{dx} \sin x = \cos x$$

$$\frac{d}{dx} \cos x = -\sin x$$

$$\frac{d}{dx} \tan x = \frac{1}{\cos^2 x} = \sec^2 x$$

$$\frac{d}{dx} \csc x = -\frac{1}{\tan x \sin x} = -\cot x \csc x$$

$$\frac{d}{dx} \sec x = \tan x \sec x$$

$$\frac{d}{dx} \cot x = -\frac{1}{\sin^2 x} = -\csc^2 x$$

$$\frac{d}{dx} \sin^{-1} x = \frac{1}{\sqrt{1-x^2}}$$

$$\frac{d}{dx} \cos^{-1} x = \frac{-1}{\sqrt{1-x^2}}$$

$$\frac{d}{dx} \tan^{-1} x = \frac{1}{1+x^2}$$

$$\frac{d}{dx} \csc^{-1} x = \frac{-1}{|x|\sqrt{x^2-1}}$$

$$\frac{d}{dx} \sec^{-1} x = \frac{1}{|x|\sqrt{x^2-1}}$$

$$\frac{d}{dx} \cot^{-1} x = \frac{-1}{1+x^2}$$



# diff 함수 : 미분

89

## diff 함수로 미분을 풀이

```
from sympy import *  
x, y, z = symbols("x y z")  
f, g, h = symbols('f g h', cls=Function)  
f = expand((x + 1)**20)
```

```
g = diff(f, x)  
print(g)  
print(factor(g))
```

```
20*x**19 + 380*x**18 + 3420*x**17 + 19380*x**16 + 77520*x**15 + 232560*x**14 + 542640*x**13 + 1007760*x**12 + 1511640*x**11 + 1  
847560*x**10 + 1847560*x**9 + 1511640*x**8 + 1007760*x**7 + 542640*x**6 + 232560*x**5 + 77520*x**4 + 19380*x**3 + 3420*x**2 + 3  
80*x + 20  
20*(x + 1)**19
```

# 기울기(gradient)

90

**기울기**(gradient 그라디언트)란 벡터 미적분학에서 스칼라장의 최대의 증가율을 나타내는 벡터장을 뜻한다.

기울기를 나타내는 벡터장을 화살표로 표시할 때 화살표의 방향은 증가율이 최대가 되는 방향이며, 화살표의 크기는 증가율이 최대일 때의 증가율의 크기를 나타낸다.

# 기울기(gradient)의 의미

91

어느 방안의 공간 온도 분포가 스칼라장  $\phi$ 로 주어졌다고 가정한다. 이 때, 방안의 어느 한 점  $(x,y,z)$ 에서의 온도는  $\phi(x,y,z)$ 로 표시할 수 있다. (온도는 시간에 의해 변화하지 않는다고 가정) 이 경우에 어느 한 지점에서의 기울기는 온도가 가장 빨리 증가하는 방향과 그 증가율을 나타낸다.

이번에는 산이나 언덕을 가정해보자. 어떤 지점  $(x,y)$ 에서의 높이를  $H(x,y)$ 로 표현하는 경우, 기울기는 가장 (위를 바라보는)경사가 가파른 방향과 그 경사의 크기를 나타낸다.

기울기를 이용해 다른 방향의 증가율을 구하려면 기울기와 그 방향의 단위 벡터의 내적을 취하면 된다.

기울기는 무회전성 벡터계이다. 즉, 기울기 벡터계에 대해 선적분을 구하면 결과값은 경로와 상관없이 시작점과 끝점에 따라서만 변화함을 뜻한다.

# 수학적 정의

92

## 수학적 정의

스칼라 함수  $f(x)$ 의 기울기는  $\nabla f$ 로 표현한다.  $\nabla$  기호는 벡터 미분 연산자로 나블라(nabla) 혹은 델(del)연산자라고 부른다.

기울기는  $f$ 의 각 성분의 편미분으로 구성된 열벡터로 정의하며 다음과 같이 표시한다.

$$\nabla f = \left( \frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)$$

$$\nabla f = \left( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

예를 들어 함수

$f(x, y, z) = 2x + 3y^2 - \sin(z)$ 의 기울기는

$$\nabla f = \left( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right) = (2, 6y, -\cos(z)) \text{이다.}$$

# 적분 : integral

93

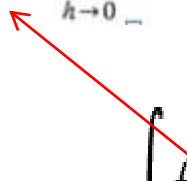
적분이란 부정적분이나 정적분의 값을 구하는 것이고 주어진 함수를 적분한다고 표현하면 부정적분을 구하는 것이고 구간을 주어 주어진 함수를 적분한다고 하면 정적분이라고 함

# 적분 기호의 의미

94

적분은 미분된 함수에 대한 값들을 합하라는 뜻

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

  $\int f(x) dx$  에서  $\int$  은 적분하여라  $dx$  는  $x$  에 대하여  $\int$  과  $dx$  사이에는 적분할 대상

예시  $\int \left( \frac{d}{dx} x^2 \right) dx = \int 2x dx = x^2 + C$

# 부정적분

95

부정적분(不定積分, indefinite integral)은 (모든) 역도함수를 구하는 연산이다. 함수  $f(x)$ 가  $F(x)$ 의 미분이면,  $F(x)$ 는  $f(x)$ 의 역도함수이다.

The diagram illustrates the components of the indefinite integral formula  $\int f(x)dx = F(x) + C$ . It includes the following labels and arrows:

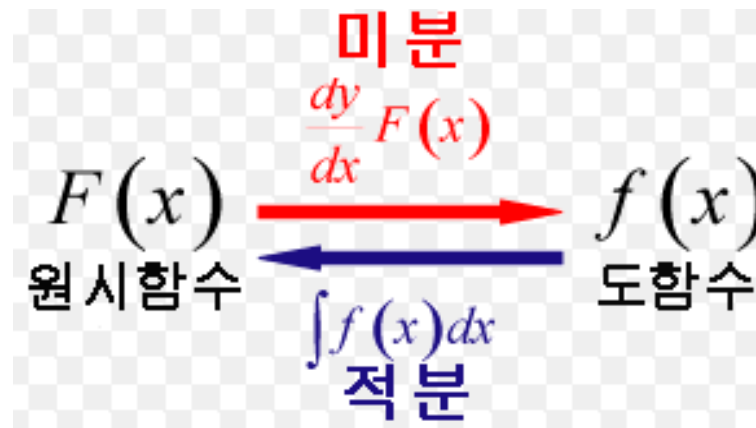
- 적분변수** (Integration variable): A yellow dot with an arrow pointing to  $x$  in  $f(x)$ .
- x에 대한 적분(다른 문자는 상수로 취급)** (Integration with respect to x (other letters are treated as constants)): A green dot with an arrow pointing to the  $\int$  symbol.
- 피적분 함수** (Integrand function): A blue dot with an arrow pointing to  $f(x)$ .
- 적분상수** (Integration constant): A purple dot with an arrow pointing to  $C$ .
- 적분 기호:  $\int$  인테그랄(integral) - Sum의 첫글자 S를 따서 만듦** (Integration symbol:  $\int$  integral - Sum's first letter S, hence the name): A red dot with an arrow pointing to the  $\int$  symbol.

$$\int f(x)dx = F(x) + C$$

# 부정적분과 미분의 관계

96

미분의 역연산으로서 정의되는 것을 부정적분이라 함





# 정적분: definite integral

97

리만 적분에서 다루는 고전적인 정의에 따르면 실수의 척도를 사용하는 축도 공간에 나타낼 수 있는 연속인 함수  $f(x)$ 에 대하여 그 함수의 정의역의 부분 집합을 이루는 구간  $[a, b]$ 에 대응하는 치역으로 이루어진 곡선의 리만 합의 극한을 구하는 것이다. 이를 정적분(定積分, definite integral)이라 한다.

# 정적분: 식의 이해

98

x를 a부터 b까지 변화시키면서  $f(x)$ 에  $dx$ 를 곱한 것을 전부 합쳐라

$$\begin{aligned}\int_a^b f(x)dx &= f(a)dx + f(a+dx)dx + f(a+2dx)dx + \dots + f(b)dx \\ &= \lim_{\Delta x \rightarrow 0} \{ f(a)\Delta x + f(a+\Delta x)\Delta x + \dots + f(b)\Delta x \} \\ &= \lim_{\Delta x \rightarrow 0} \sum_{x=a}^b f(x)\Delta x\end{aligned}$$

# 일반 적분 공식 : sympy

99

x에 대한 적분은  $1/n+1 * x^{n+1}$ 로 처리

```
from sympy import integrate, log, exp, oo
x = symbols('x')
print(integrate(x, x))

x**2/2
```

# 상수와 함수: sympy

100

## 일반 적분 공식

$$\int a f(x) dx = a \int f(x) dx \quad (a \text{ constant}) \quad \int k dx = kx + C$$

```
from sympy import integrate, log, exp, oo

a = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)
x = symbols('x')
f=x**2
print(integrate(f,x))
print(integrate(2*f, x))

x**3/3
2*x**3/3
```

# n차 함수: sympy

101

## 일반 적분 공식

$$\int x^n dx = \frac{x^{n+1}}{n+1} + C \quad (\text{for } n \neq -1)$$

```
from sympy import integrate, log, exp, oo

a = symbols('a', integer=True)
f, g, h = symbols('f g h', cls=Function)
x, n = symbols('x n')
f = x**n

print(integrate(f, x))

Piecewise((log(x), Eq(n, -1)), (x**(n + 1)/(n + 1), True))
```

# 함수의 곱: sympy

102

## 일반 적분 공식

$$\int f(x)g(x) dx = f(x) \int g(x) dx - \int \left[ f'(x) \left( \int g(x) dx \right) \right] dx$$

```
from sympy import integrate, log, exp, oo

a = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)
x = symbols('x')
f=x**2
g=x
h = f*g

print(integrate(h, x))
print(f*integrate(g,x)-integrate(diff(f,x)*integrate(g,x)))

x**4/4
x**4/4
```

# 도함수와 함수 곱: sympy

103

## 일반 적분 공식

$$\int f'(x)f(x) dx = \frac{1}{2}[f(x)]^2 + C$$

```
from sympy import integrate, log, exp, oo

a = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)
x = symbols('x')
f=x**2

print(diff(f,x))
print(diff(f,x)*f)
print(integrate(diff(f,x)*f, x))

2*x
2*x**3
x**4/2
```

# 일반 적분 공식

104

## 일반 적분 공식

$$\int [f(x) + g(x)] dx = \int f(x) dx + \int g(x) dx$$

```
from sympy import integrate, log, exp, oo

a = symbols('a', integer=True)
f, g, h = symbols('f g h', cls=Function)
x, n = symbols('x n')
f = x**2
g = x

print(integrate(f+g, x))

print(integrate(f, x) + integrate(g, x))

x**3/3 + x**2/2
x**3/3 + x**2/2
```



# 일반 적분 공식

105

## 일반 적분 공식

$$\int [f(x)]^n f'(x) dx = \frac{[f(x)]^{n+1}}{n+1} + C \quad (\text{for } n \neq -1)$$

```
from sympy import integrate, log, exp, oo
```

```
a = symbols('a', integer=True)
```

```
f, g, h = symbols('f g h', cls=Function)
```

```
x, n = symbols('x n')
```

```
f=x**2
```

```
g= diff(f)
```

```
print(g)
```

```
print(integrate(f**n,x))
```

```
print(integrate(f**n * g, x))
```

```
2*x
```

```
Piecewise((None, Eq(n, -1/2)), (x*(x**2)**n/(2*n + 1), True))
```

```
2*Piecewise((log(x), Eq(n, -1)), (x**2*(x**2)**n/(2*n + 2), True))
```

# 치환적분 공식

106

## 일반 적분 공식

$$\int f(ax+b)dx = \frac{1}{a}F(ax+b) + C$$

$$\int \frac{1}{(2x+1)^2} dx = -\frac{1}{2} \bullet \frac{1}{(2x+1)} + c$$

$$\int \frac{f'(x)}{f(x)} dx = \ln |f(x)| + C$$

$$\int \frac{1}{x^2-1} dx = \int \frac{1}{(x-1)(x+1)} dx = \int \frac{1}{2} \left( \frac{1}{x-1} - \frac{1}{x+1} \right) dx = \frac{1}{2} \ln|x-1| - \frac{1}{2} \ln|x+1| + c$$

# 분수 적분 공식

107

## 가분수를 대분수로 바꾸고 적분

$$\int \frac{x^2 + 1}{x} dx = \int \left( x + \frac{1}{x} \right) dx = \frac{1}{2} x^2 + \ln x + c$$



앞의 가분수를 대분수로 바꿈

$$\int x^n dx = \frac{x^{n+1}}{n+1} + C \quad \text{if } n \neq -1$$

$$\int x^{-1} dx = \ln |x| + C$$

$$\int \frac{1}{1+x^2} dx = \arctan x + C$$

$$\int \frac{1}{x} dx = \ln |x| + c$$

$$\int \frac{1}{x^n} dx = \int x^{-n} dx = \frac{1}{-n+1} x^{-n+1} + c = \frac{1}{-n+1} \bullet \frac{1}{x^{n-1}} + c (n \neq 1)$$

# 무리함수 적분공식

108

## 무리함수 적분 공식

$$\begin{aligned}\int \frac{1}{\sqrt{1-x^2}} dx &= \arcsin x + C \\ \int \frac{-1}{\sqrt{1-x^2}} dx &= \arccos x + C \\ \int \frac{1}{|x|\sqrt{x^2-1}} dx &= \operatorname{arcsec} x + C\end{aligned}$$

# 로그/지수함수 적분공식

109

## 로그/지수함수 적분공식

---

$$\int \ln x \, dx = x \ln x - x + C$$
$$\int \log_a x \, dx = x \log_a x - \frac{x}{\ln a} + C$$

$$\int e^x \, dx = e^x + C$$
$$\int a^x \, dx = \frac{a^x}{\ln a} + C$$

# 삼각함수 적분공식

110

## 삼각함수 적분공식

$$\int \cos x \, dx = \sin x + C$$

$$\int \sin x \, dx = -\cos x + C$$

$$\int \tan x \, dx = -\ln |\cos x| + C$$

$$\int \csc x \, dx = \ln |\csc x - \cot x| + C$$

$$\int \sec x \, dx = \ln |\sec x + \tan x| + C$$

$$\int \cot x \, dx = \ln |\sin x| + C$$

$$\int \sec^2 x \, dx = \tan x + C$$

$$\int \csc^2 x \, dx = -\cot x + C$$

$$\int \sin^2 mx \, dx = \frac{1}{2m} (mx - \sin mx \cos mx) + C$$

$$\int \cos^2 mx \, dx = \frac{1}{2m} (mx + \sin mx \cos mx) + C$$

$$\int \sin^n x \, dx = -\frac{\sin^{n-1} x \cos x}{n} + \frac{n-1}{n} \int \sin^{n-2} x \, dx + C$$

$$\int \cos^n x \, dx = \frac{\cos^{n-1} x \sin x}{n} + \frac{n-1}{n} \int \cos^{n-2} x \, dx + C$$

$$\int \sec^n x \, dx = \frac{\sec^{n-2} x \tan x}{n-1} + \frac{n-2}{n-1} \int \sec^{n-2} x \, dx + C$$

$$\int \csc^n x \, dx = \frac{\csc^{n-2} x \cot x}{-(n-1)} + \frac{n-2}{n-1} \int \csc^{n-2} x \, dx + C$$

# 경우의 수

Moon Yong Joon

# 경우의 수 (number of cases)

112

경우는 수는 사건에서 일어날 수 있는 경우의 가  
짓수(원소의 개수)를 말하며,  $n * (n-1)!$ 로 계산  
됨

#팩토리얼

```
def fact(x) :  
    if x == 1 :  
        return 1  
    if x == 0 :  
        return 1  
  
    return x * fact(x-1)  
  
print(fact(5))  
print(5*4*3*2*1)
```

120

120

$$5! = 1 \times 2 \times 3 \times 4 \times 5 = 120$$

$$0! = 1 \quad 0! = 0^0 = 0^{n-n} = \frac{0^n}{0^n} = 1$$



# 합의 법칙

113

합의 법칙은 두 사건 A,B가 동시에 일어나지 않을 때 각각의 사건이 일어날 경우의 수를 서로 더해서 구하는 경우

사건 A가 일어나는 경우의 수 :  $m$

사건 B가 일어나는 경우의 수 :  $n$

사건 A 또는 B가 일어날 경우의 수 =  $m+n$

두 사건 A, B에 대하여 합집합  $A \cup B$ 의 원소의 개수는

$$n(A \cup B) = n(A) + n(B) - n(A \cap B)$$

# 곱의 법칙

114

곱의 법칙은 두 사건 A,B가 동시에(연달아서) 일어날 때 각각의 사건이 일어날 경우의 수를 서로 곱해서 구하는 경우

사건 A 가 일어나는 경우의 수 :  $m$

사건 B가 일어나는 경우의 수 :  $n$

사건 A와 사건 b가 동시에 일어날 경우의 수 :  $m*n$

# 합/곱의 법칙 구별

115

동시에 일어나는 사건을 경우 별개의 두 사건이 모두 발생하는 것(곱의 법칙)이고, 별개의 두 사건이 있는 경우 둘다 일어나지 않아도 상관없는 것(합의 법칙)임

독립사건 A 또는 B가 일어나는 경우의 수를 구할 때에는 합의 법칙이 이용된다.

독립사건 A와 B가 동시에 일어나는 경우의 수를 구할 때에는 곱의 법칙이 이용된다

# 곱의 법칙: 데카르트 곱

116

## 집합 A의 원소와 집합 B의 원소인 순서쌍을 만드는 집합

집합 A, B가 공집합이 아닐때 A, B의 데카르스 곱은 다음과 같이 정의됩니다.

$$A \times B = \{ (x,y) \mid x \in A \text{ 이고 } y \in B \}$$

즉,  $A \times B$ 는 A의 원소가 첫번째 원소, B의 원소가 두번째 원소인 순서쌍들의 집합입니다.

집합  $A = \{a, b\}$ ,  $B = \{1, 2\}$  일때  
 $A \times B = \{(a, 1), (a, 2), (b, 1), (b, 2)\}$   
즉  $2 \times 2 = 4$

```
import itertools as it

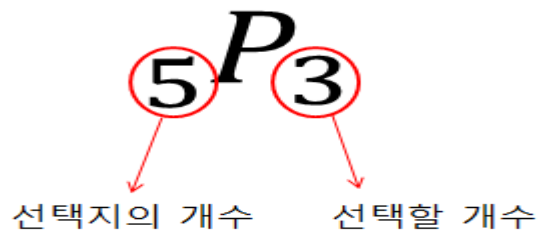
for i in it.product([1,2,3],[1,2,3]):
    print(i)
```

```
(1, 1)
(1, 2)
(1, 3)
(2, 1)
(2, 2)
(2, 3)
(3, 1)
(3, 2)
(3, 3)
```

# 순열

117

서로 다른  $n$ 개에서  $r$ 개를 택해 순서를 정해 일렬로 나열하는 것을 순열이라고 함



$${}_nP_r = n(n-1)(n-2) \cdots (n-r+1)$$

$$= \frac{n!}{(n-r)!}$$

- 서로 다른  $n$ 개
- 중복을 허락하지 않는  $r$  개를 선택
- 일렬로 나열하는 수

```
def fact(x) :  
    if x == 1 :  
        return 1  
    if x == 0 :  
        return 1  
    |  
    return x * fact(x-1)
```

```
def permute(x,r):  
    if x < r :  
        return None  
    a = fact(x)  
    b = fact(x-r)  
    return a/b
```

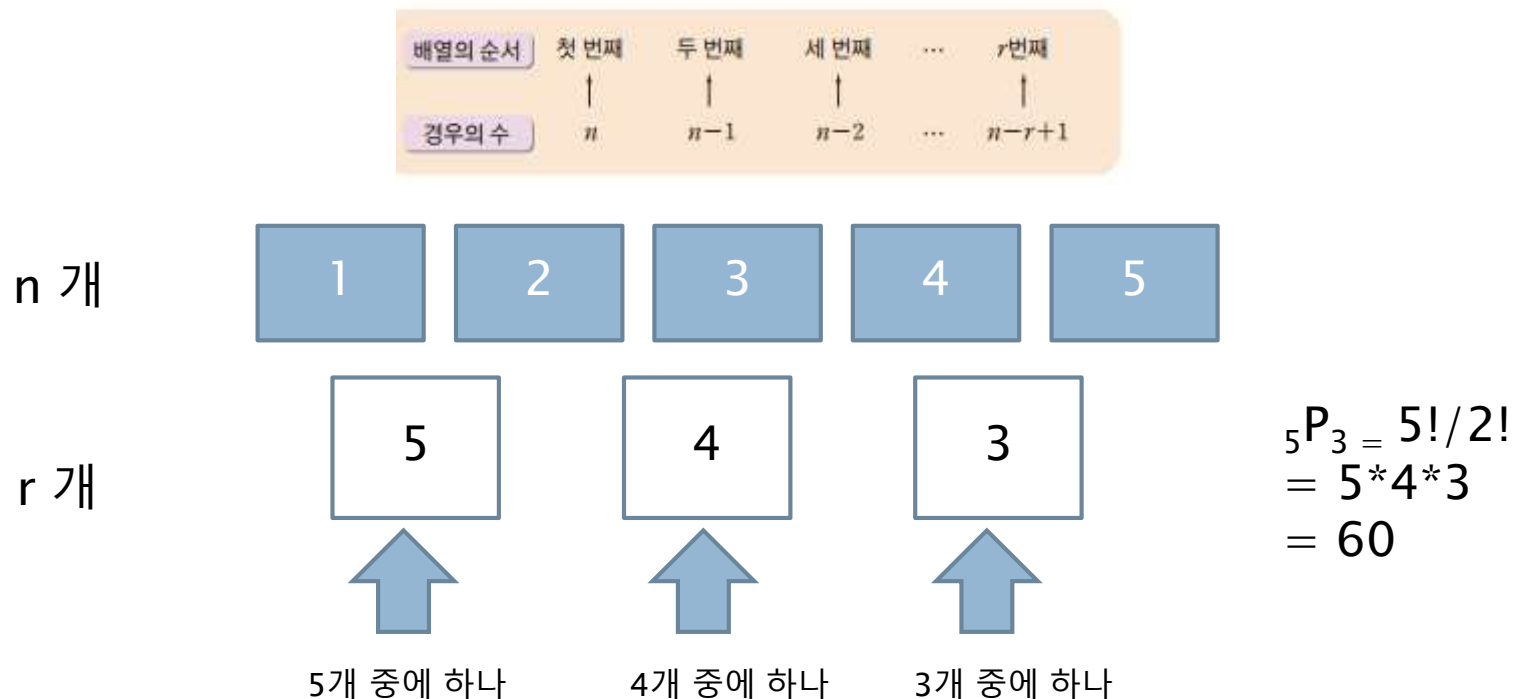
```
print(permute(5,3))  
print(fact(5))  
print(fact(5-3))
```

```
60.0  
120  
2
```

# 순열 예시

118

서로 다른 5개에서 3개를 택해 순서를 정해 일렬로 나열하면 총 60가지의 경우의 수가 생김



# Itertools:permutations

119

iterable에 대한 r에 대한 순서쌍을 구성하는 순열

```
: help(permutations)
```

Help on class permutations in module itertools:

```
class permutations(builtins.object)
| permutations(iterable[, r]) --> permutations object
|
| Return successive r-length permutations of elements in the iterable.
|
| permutations(range(3), 2) --> (0,1), (0,2), (1,0), (1,2), (2,0), (2,1)
```

```
from itertools import permutations
per = permutations(['빨', '주', '노', '초'], 2)

for i in per :
    print(i)
```

```
('빨', '주')
('빨', '노')
('빨', '초')
('주', '빨')
('주', '노')
('주', '초')
('노', '빨')
('노', '주')
('노', '초')
('초', '빨')
('초', '주')
('초', '노')
```

# 동일한 값이 있는 순열

120

$n$ 개에서 서로 같은 것이 각각  $p, q, \dots, r$ 개씩 있을 때, 이들을 모두 택하여 일렬로 나열하는 순열의 수

같은 것이 있는 순열의 수

$n$ 개에서 서로 같은 것이 각각  $p, q, \dots, r$ 개씩 있을 때, 이들을 모두 택하여 일렬로 나열하는 순열의 수는

$$\frac{n!}{p! \times q! \times \dots \times r!} \quad (\text{단, } p+q+\dots+r=n)$$



# 중복순열 : itertools.product

121

서로 다른  $n$ 개에서 중복을 허용해서  $r$ 개를 택해  
일렬로 나열하는 것을 중복순열이라고 함

배열의 순서	첫 번째	두 번째	세 번째	...	$r$ 번째
	↑	↑	↑		↑
경우의 수	$n$	$n$	$n$	...	$n$

$${}_n\Pi_r = n^r$$

```
from itertools import product  
per = product(['빨', '주'], repeat=2)
```

```
for i in per :  
    print(i)
```

```
('빨', '빨')  
( '빨', '주')  
( '주', '빨')  
( '주', '주')
```

# 조합

122

서로 다른  $n$ 개에서 순서를 생각하지 않고  $r$ 개를 택하는 것을 조합(순열에서 중복을 제거하면 조합이 됨)

$$\begin{aligned} {}_nC_r &= \frac{{}_nP_r}{r!} \\ &= \frac{n(n-1)(n-2)\cdots(n-r+1)}{r!} \\ &= \frac{n(n-1)(n-2)\cdots(n-r+1)(n-r)\cdots 3 \cdot 2 \cdot 1}{r!(n-r)\cdots 3 \cdot 2 \cdot 1} \\ &= \frac{n!}{r!(n-r)!} \end{aligned}$$

$$(1) {}_nC_r = {}_nC_{n-r} \quad (0 \leq r \leq n)$$

$$(2) {}_nC_0 = 1, {}_nC_n = 1, {}_nC_1 = n$$

$$(3) {}_nC_r = {}_{n-1}C_r + {}_{n-1}C_{r-1} \quad (1 \leq r < n)$$

# Itertools.combinations

123

${}_4C_2$  는  $4!/2!2! = 6$ 을 가지는 조합을 만듦

서로다른  $n$  개중  
 $r$ 개를 나열한다.

$${}_nC_r = \frac{{}_nP_r}{r!}$$

조합은 순서를 고려하지  
않기 때문에 중복되는  
것을 나눈다.

↓

$${}_nP_r = {}_nC_r \times r!$$

$r$ 를 나열한다.

서로다른  $n$ 개중  $r$ 를 뽑는다.

```
from itertools import combinations  
per = combinations(['빨', '주', '노', '초'], 2)
```

```
for i in per :  
    print(i)
```

```
('빨', '주')  
( '빨', '노')  
( '빨', '초')  
( '주', '노')  
( '주', '초')  
( '노', '초')
```

# 중복조합

124

조합과 마찬가지로  $n$ 개의 원소에서  $r$ 개를 순서에 상관없이 뽑는데, **중복을 허락할 때**의 가짓 수.

기호로는  $((n_r))$  또는  $nHr$  로 사용

$${}_n H_r = {}_{n+r-1} C_r$$

$${}_3 H_4 = \frac{6!}{4!2!} = 15$$

$${}_n H_r = \frac{(r+n-1)!}{r!(n-1)!} = {}^{r+(n-1)} C_r$$

```
from itertools import combinations_with_replacement

for i in combinations_with_replacement('1234',2) :
    print(i)
```

```
('1', '1')
('1', '2')
('1', '3')
('1', '4')
('2', '2')
('2', '3')
('2', '4')
('3', '3')
('3', '4')
('4', '4')
```

# 확률 용어

# 확률

126

확률현상은 불확실성에 대해 좌우되는 현상이다. 확률현상에 대해 현실에서 실험하는 것을 확률실험( $\epsilon$ )이라고 한다.

확률현상에서 얻어질 수 있는 모든 결과를 표본공간( $\Omega$ ) 이라고 한다.

또한 표본공간의 부분집합을 확률사건(Event)라고 한다. 한 확률실험에서 '어떤 사건이 일어나리라고 예측되는 정도를 나타내는 수치'를 그 사건이 일어날 확률이라고 한다.

# 확률의 3가지 공리

127

- 공리 1. 임의의 사건  $A$ 에 대하여  $1 > P(A) \geq 0$

- 공리 2.  $P(S) = 1$

(여기서,  $S$  는 표본 전체의 집합, 표본공간)

- 공리 3. 상호배타적인 사건  $A_1, A_2, A_3, \dots$ 에 대하여  $P(A_1 \cup A_2 \cup A_3 \cup \dots)$

$$= P(A_1) + P(A_2) + P(A_3) + \dots$$

# 확률의 종류

128

- 수학적 확률 - 누구에 의해서나 동일한 값으로 계산되는 확률
- 통계적 확률 - 동일조건/독립적으로 무한반복하였을 때 발생 확률 : 도수이론(frequency theory)
- 주관적 확률 - 관찰자 주관에 따라 다르게 표현되는 확률 : 주관적 견해(subjective view) 반영



# 수학적 확률

129

어떤 시행의 표본공간  $S$ 가  $n$ 개의 근원사건으로 이뤄져 있고 각 근원사건이 일어날 가능성이 모두 똑같이 기대될 때, 사건  $A$ 가  $r$ 개의 근원사건으로 이루어져 있으면 사건  $r$ 이 일어날 확률  $P(A)$ 를  $P(A) = r/n$ 으로 정의

$$P(A) = \frac{A\text{사건의 경우의 수}}{\text{전체경우의 수}}$$

# 경우의 수를 이용

130

모든 경우의 수에 사건 A가 일어나는 경우의 수로 확률을 표시

$$P(A) = \frac{\text{number of event } A \text{ occurs}}{\text{number of total event occurs}}$$
$$= \frac{n(A)}{n(U)}$$

```
sp = [1,2,3,4,5,6]
event1 = [1,3,5]
event2 = [2,4,6]

# 확률
print(" odd events ", len(event1)/len(sp))
print(" even events ", len(event2)/len(sp))

def prob(x) :
    if x % 2 == 0 :
        return len(event2)/len(sp)
    else :
        return len(event1)/len(sp)

print(list(map(prob, [1,2,3,4,5,6])))

odd events  0.5
even events 0.5
[0.5, 0.5, 0.5, 0.5, 0.5, 0.5]
```

# 상대도수

131

특정 실험을 많이 해서 이에 실제 발생한 빈도에 따라 확률을 계산하는 경우

계 급	계급 구간	중앙값	도 수	상대 도수
1	147.5 - 152.5	150	6	0.13
2	152.5 - 157.5	155	11	0.24
3	157.5 - 162.5	160	14	0.31
4	162.5 - 167.5	165	9	0.20
5	167.5 - 172.5	170	3	0.07
6	172.5 - 177.5	175	2	0.04
총 계			45	1.00

도수 분포표

키 누적도수분포표

키(cm)	도수(명)	누적도수	상대누적도수
135~140	4	4	0.08
140~145	9	13	0.26
145~150	12	25	0.50
150~155	14	39	0.78
155~160	6	45	0.90
160~165	4	49	0.98
165~170	1	50	1.00
계	50		

상대도수 기반의 확률

# 실험

132

실험(實驗)은 가설이나 이론이 실제로 들어맞는지를 확인하기 위해 다양한 조건 아래에서 여러 가지 측정을 실시하는 일이다. 지식을 얻기 위한 방법의 하나이다. 실험은 관찰(측정도 포함)과 함께 과학의 기본적인 방법의 하나이다. 다만 관찰이 대상 그 자체를 있는 그대로 살펴보는 일이라면, 실험은 어떤 조작을 가해 그에 따라 일어나는 변화를 조사하고 결론을 내는 일이다.

# 시행(trial)

133

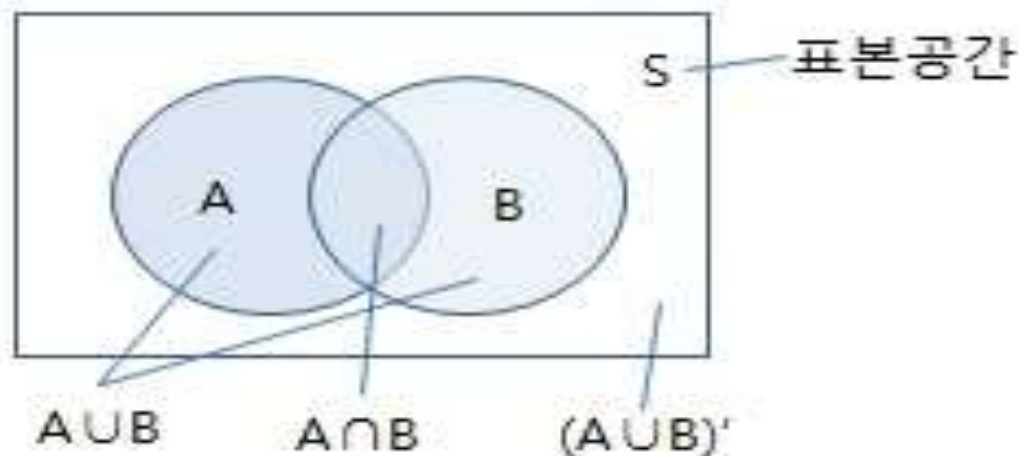
모든 결과가 우연에 의해서 지배가 되고 반복가능하며 모든 결과가 예측이 가능한 실험이나 관찰

확률실험(random experiment)즉 시행은 결과가 우연에 좌우되는 현상을 실현시키는 행위는 실험, 관찰, 조사 등으로써 몇 가지 특징을 가지는 것을 시행 또는 확률실험이라고 한다.

# 표본공간(sample space)

134

표본공간 또는 전체 표본공간은 종종  $S$ ,  $\Omega$  또는  $U$ 로 표기되며, 실험 또는 임의 시도의 모든 가능한 산출들의 집합이다.



# 사건

135

확률에서는 실험에서 발생할 수 있는 결과들이 기본 사건 (elementary event) 이라고 불리는데 이것의 모음은 그  
냥 사건이라고 일컫는다.

어떤 실험의 표본공간  $S$ 에 관한 사건(event)이란  $S$ 의 부  
분 집 합을 뜻함. 즉, 실험의 가능한 결과로 이루어진 집  
합을 사건 (event)이라 함

근원사건

사건 중에 표본공간의 전체집합에서  
원소가 1개인 부분집합

주어진 사건이 모두 배타적인 사건인 경우의 모든 확률이 합

$$P(E_1) + P(E_2) + \dots + P(E_n) = 1$$

# 주사위 시행에 대한 사건 예시

136

주사위를 여러 번 시행할 경우 이를 기반으로 특정 사건에 대한 예시

```
import numpy as np
from itertools import product

# sample 만들기
# x 주사위 시행횟수
def dice(x) :
    a = np.arange(1,7)

    b = product(a,repeat=x)
    c = [ x for x in b]
    return c

# event 추출
def event(iterable, ev) :
    d = []
    for x in iterable :
        if np.sum(x) == ev :
            d.append(x)
    return d

print(event(dice(2),7))
print(event(dice(3),7))
|
```

```
[(1, 6), (2, 5), (3, 4), (4, 3), (5, 2), (6, 1)]
[(1, 1, 5), (1, 2, 4), (1, 3, 3), (1, 4, 2), (1, 5, 1),
 (2, 4, 1, 2), (4, 2, 1), (5, 1, 1)]
```



# 주요 사건들

137

## 주요 사건들.

$$P(A) = \frac{\text{number of event } A \text{ occurs}}{\text{number of total event occurs}} \\ = \frac{n(A)}{n(U)}$$

사건  
event

시행의 결과로서 나타나는 것

$$0 \leq P(A) \leq 1$$

여사건  
complementary  
event

어떤 사건 A가 일어나지 않는 사건

$$P(A^c) = 1 - P(A)$$

전사건  
total event

일어날 수 있는 모든 경우의 사건

$$P(U) = 1$$

공사건  
empty event

일어날 수 있는 모든 경우의 사건

$$P(\phi) = 0$$

두 사건이 독립(獨立, independent)이라는 것은, 둘 중 하나의 사건이 일어날 확률이 다른 사건이 일어날 확률에 영향을 미치지 않는다는 것을 의미한다.

예를 들어서, 주사위를 두 번 던지는 경우 첫 번째에 1이 나오는 사건은 두 번째에 1이 나올 확률에 독립적이다.

# 독립사건

139

독립사건( Independent Event)은 서로 다른 사건이 일어날 확률에 영향을 주지 않은 사건들

독립사건의 조건

조건부 확률은 사건 B가 주어졌을 때 A가 일어날 확률이지만 두 사건이 독립이라면

$$P(A|B) = P(A|B^c) = P(A)$$

$$P(B|A) = P(B|A^c) = P(B)$$

$$P(A \cap B) = P(A)P(B)$$

$$P(A|B) = P(A) \Leftrightarrow \frac{P(A \cap B)}{P(B)} = P(A)$$

분자영역 변환



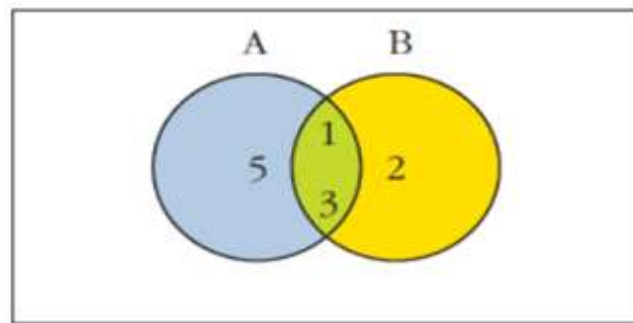
$$P(A \cap B) = P(A) \times P(B)$$

# 종속사건

140

종속사건(Dependent Events)은 사건 A가 일어났을 경우에 사건 B가 일어날 확률을 말함

$$P(A \cap B) = P(A|B) \times P(B)$$
$$= \frac{P(A \cap B)}{P(B)} \times P(B)$$



$$P(A|B) \neq P(A^c|B) \quad \longrightarrow \quad \frac{P(A \cap B)}{P(B)}; \text{ 같지 않다 } \frac{P(A^c \cap B)}{P(B)}$$

A와 B의 교집합은 1, 3  
A여집합과 B의 교집합은 2

# 배반사건

141

배반사건(Exclusive Events)는 동시에 일어날 수 없는 두 사건, 두 사건의 교집합이 공집합이 되는 사건

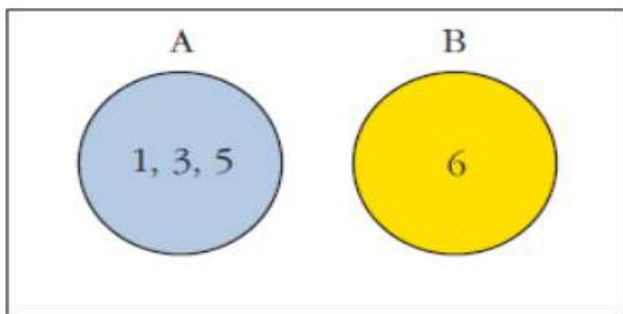
배반사건은

$$A \cap B = \phi \Rightarrow P(A \cap B) = 0$$

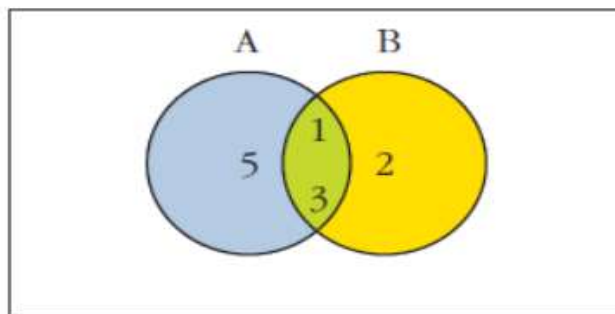
$$P(A \cup B) = P(A) + P(B)$$

성립할 때 사건 A와 B는 배반사건이라고 합니다.

배반인 경우



배반이 아닌 경우



# 확률의 기본 법칙 정리

Moon Yong Joon

# 독립시행의 확률

143

동일한 시행이 여러 번 반복되면 사건이 서로 독립이므로 각 시행이 다른 시행에 영향을 미치지  
않음

- ① 매회 시행에서 사건 A가 일어날 확률  $p$ 로 일정
- ②  $n$ 회 독립시행  
사건A가  $r$ 번 일어날 확률?

$$P(X=r) = {}_n C_r p^r q^{n-r} \quad (q=1-p)$$

$$p(x) = {}_n C_x p^x (1-p)^{n-x}$$

성공확률      실패확률

시행횟수      성공횟수

# 독립시행 예시

144

주사위를 3번 던질 때 6의 눈이 2번 나올 확률

1회	2회	3회
O	O	X
O	X	O
X	O	O

$$\left. \begin{aligned} \frac{1}{6} \times \frac{1}{6} \times \frac{5}{6} &= \left(\frac{1}{6}\right)^2 \cdot \frac{5}{6} \\ \frac{1}{6} \times \frac{5}{6} \times \frac{1}{6} &= \left(\frac{1}{6}\right)^2 \cdot \frac{5}{6} \\ \frac{5}{6} \times \frac{1}{6} \times \frac{1}{6} &= \left(\frac{1}{6}\right)^2 \cdot \frac{5}{6} \end{aligned} \right\} {}_3C_2 \times \left(\frac{1}{6}\right)^2 \cdot \frac{5}{6}$$

표를 이용해서 이해를 하는것이 효과적임

6의 눈이 3번 중에 2번 나오는 경우의 수

1회, 2회, 3회는 모두 독립사건이기 때문에 확률을 구할때 모두 곱해서 확률을 구함



# 독립시행 예시

145

주사위를 6번 던질 때 1 또는 2의 눈이 2번 나올 확률을 구하여라.

	1	2	3	4	5	6
1 or 2	o	o	x	x	x	x
1 or 2	o	o	x	x	x	x
1 or 2	o	o	x	x	x	x
1 or 2	o	o	x	x	x	x
1 or 2	o	o	x	x	x	x
1 or 2	o	o	x	x	x	x

$$1 \text{ or } 2 \text{ 확률} = 2/6 = 1/3$$

$$p(x) = {}^nC_x p^x (1-p)^{n-x}$$

Diagram labels with arrows pointing to the formula components:

- 성공확률 (Success Probability) points to  $p$
- 실패확률 (Failure Probability) points to  $(1-p)$
- 시행횟수 (Number of Trials) points to  $n$
- 성공횟수 (Number of Successes) points to  $x$

$$\begin{aligned} &= {}^6C_2 (1/3)^2 (1-1/3)^{6-2} \\ &= 6!/2!4! (1/9)(4/81) \\ &= 15 * (16/729) \\ &= 80/243 \end{aligned}$$

# 확률의 덧셈정리 조건

146

- 두 사건 중 적어도 하나의 사건이 일어날 확률과 관련

두 사건  $A, B$ 에 대하여

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

- 좁은 의미의 덧셈법칙은 상호배반일 경우만 가능

두 사건  $A, B$ 가 서로 배반사건이면, 즉  $A \cap B = \emptyset$ 이면

$$P(A \cup B) = P(A) + P(B)$$

- 배반이 아닌 경우 중복 계산되는 부분을 제거해야 한다.

# 곱셈법칙 조건 : 결합확률

147

- 두 사건이 함께 일어날 확률과 관련

- 두 사건 A, B에 대해

$$P(A \cap B) = P(A|B) \times P(B)$$

- 좁은 의미의 곱셈법칙은 상호독립일 경우만 가능

- A와 B가 독립사건일 때

$$P(A \cap B) = P(A) \times P(B)$$

$$P(B|A) = P(B) \quad (\text{단, } P(A) > 0)$$

$$P(A|B) = P(A) \quad (\text{단, } P(B) > 0)$$

- 독립이 아닌 (종속의) 경우 하나의 주변확률과 다른 하나의 조건부확률을 곱해야 한다

# 조건부 확률

148

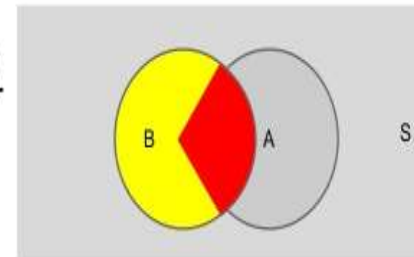
조건부 확률 (conditional probability)이란 사건 B가 일어났다고 가정했을 때에 사건 A가 일어날 확률을 가리켜,

**$P(A \mid B)$**  로 표시하고,

이를 사건 B를 조건으로 하는 A의 조건부 확률이라 한다.

$$p(A|B) = \frac{(A \text{ 와 } B \text{ 가 발생할 확률})}{(B \text{ 가 발생할 확률})}$$

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$



# 조건부 확률공식 : 벤다이어그램

149

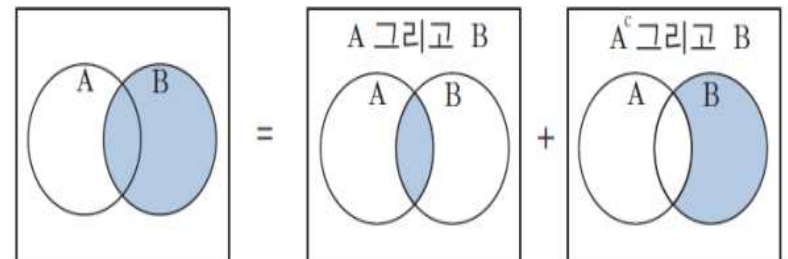
$P(A|B)$  : 조건부확률

- 사건 B 가 주어진 조건 하에서 사건 A 가 일어날 확률
- 사건 B 에만 포커스를 맞춤
- 사건 B의 확률에 대한 사건 A와 B 결합사건확률의 상대적 크기

$$P(A|B) = \frac{P(A \text{ 그리고 } B)}{P(B)} = \frac{P(A \text{ 그리고 } B)}{P(A \text{ 그리고 } B) + P(A^c \text{ 그리고 } B)}$$

$$P(A|B) \neq P(B|A)$$

$P(B)$  구하기: 벤다이어그램



# 조건부 확률공식 유도

150

## 조건부 확률에 대한 식 유도 $P(A|B) \neq P(B|A)$

이벤트 A가 일어난 상황에서 이벤트 B가 일어날 확률. Conditional Probability of B given A.  $P(B|A)$

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

위식을 통해 다음식을 유도할 수 있다.

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B)$$

만일 A, B 두 사건이 독립이라면 다음이 성립한다.

$$P(A \cap B) = P(A)P(B)$$

$$P(A|B) = P(A)$$

$$P(B|A) = P(B)$$

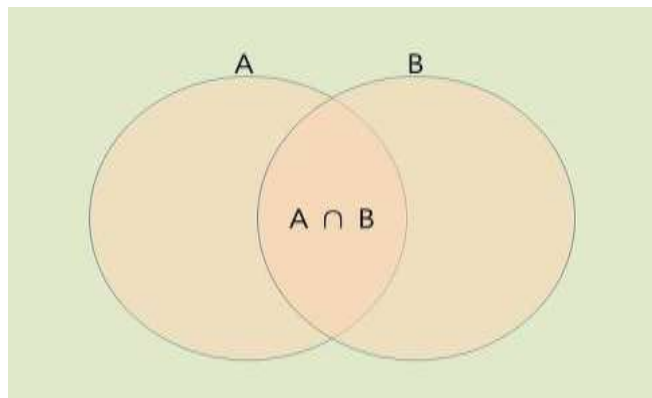
# 조건부 확률공식 : 예시

151

주사위가 소수이면서 홀수가 나올 확률은

A:홀수 = {1,3,5}, B:소수={2,3,5}

A와 B의 교집합 = {3,5}



$$p(A|B) = \frac{p(A \cap B)}{p(B)} = \frac{1/3}{1/2} = \frac{2}{3}$$

$$p(A) = \text{홀수일 확률} \left(\frac{1}{2}\right)$$

$$p(B) = \text{소수일 확률} \left(\frac{1}{2}\right)$$

$$p(A \cap B) = \text{홀수이면서 소수일 확률} \left(\frac{1}{3}\right)$$

$$P(B | A) = \text{'4 이하의 눈이 나왔을 때 그 눈이 짝수 일 확률'} = 2/4$$

$$P(A | B) = \text{'짝수의 눈이 나왔을 때 그 눈이 4이하일 확률'} = 2/3$$

# 조건부 확률 예시 1

152

y를 뽑을 사건 A  $p(A) = 13/20$ , 불량률 뽑을 사건 B  $p(B) = 8/20$ , y를 뽑았는데 불량인 사건  $p(A \& B) = 2/20$

y를 뽑혔을 때 그것이 썩은 것일 확률은

	x	y	sum
양호	1	11	12
불량	6	2	8
	7	13	20

전체 경우의 수  $n(S)$  라고 두면

$$\frac{n(A \cap B)}{n(A)} = \frac{\frac{n(A \cap B)}{n(S)}}{\frac{n(A)}{n(S)}} = \frac{P(A \cap B)}{P(A)}$$

$$P(B|A) = \frac{P(A \cap B)}{P(A)} = \frac{2/20}{13/20} = \frac{2}{13}, P(A) \neq 0$$



# 조건부 확률 예시 2

153

y를 뽑을 사건 A  $p(A) = 13/20$ , 불량을 뽑을 사건 B  $p(B) = 8/20$ , y를 뽑았는데 불량인 사건  $p(A \& B) = 2/20$

불량을 뽑을 사건이 발생할 때 y를 뽑을 사건

	x	y	sum
양호	1	11	12
불량	6	2	8
	7	13	20

전체 경우의 수  $n(S)$  라고 두면

$$\frac{n(A \cap B)}{n(A)} = \frac{\frac{n(A \cap B)}{n(S)}}{\frac{n(A)}{n(S)}} = \frac{P(A \cap B)}{P(A)}$$

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{2/20}{8/20} = \frac{2}{8}, P(B) \neq 0$$

# 파이썬 예시

154

주사위 짝수일 때(event1) 특정 값(event2,4)이 나오는  $P(A \& B)$ 를 조건부확률로 구하기

위식을 통해 다음식을 유도할 수 있다.

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B)$$

```
sp = [1,2,3,4,5,6]
event1 = set([2,4,6])
event2 = set([4])
# 확률 event1 & event2
print(event1 & event2)
pe12 = len(event1 & event2)/len(sp)
# 확률 event 1
pe1 = len(event1)/len(sp)
print(pe1)
# 확률 event 2
pe2 = len(event2)/len(sp)
print(pe2)
print(pe1 * pe2)
#p( pe1 & pe2)= p(pe1|pe2) * p(pe2)
print(pe12/pe2 * pe2)
```

```
{4}
0.5
0.16666666666666666
0.08333333333333333
0.16666666666666666
```

# 베이지 정리

155

사건 A가 있고 사건 B가 있을 때 사건 B가 일어난 것을 전제로 한 사건 A의 조건부 확률을 구하고 싶다. 그런데 지금 알고 있는 것은 사건 A가 일어난 것을 전제로 한 사건 B의 조건부 확률, A의 확률, B의 확률뿐이다.

# 베이지스 사용 이유

156

기존 : 모집단을 변하지 않은 대상으로 봄

- 모집단에 대해 규정시킨 확률분포 또는 모수를 출발점으로 삼음

베이지스 : 모집단을 미리 확정짓지 않음

- 모수를 확률변수 처럼 취급
- 매 표본 마다 나오는 데이터를 출발점으로 삼음
  - . 정보가 증가됨에 따라, 확률이 수정/정제됨

# 베이즈 정리 용어

157

## 사전확률(prior probability)

→ 이미 알고 있는 사건으로부터 나온 확률로 위의 베이즈 정리에서  $P(A_1), \dots, P(A_n)$ 을 의미

## 우도(likelihood probability)

→ 이미 알고 있는 사건이 발생했다는 조건하에 다른 사건이 발생할 확률  $P(B|A_1), \dots, P(B|A_n)$

## 사후확률(posterior probability)

→ 사전확률과 우도를 통해 알게되는 조건부 확률  $P(A_k|B)$

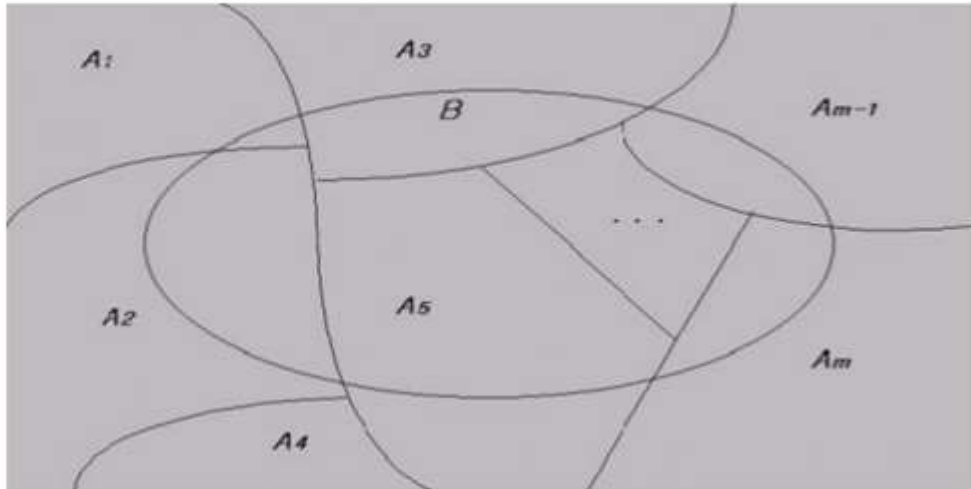
# 베이지 정리 산식 1

158

① 표본공간  $S$ 는 서로소인  $A_1, A_2, A_3, \dots, A_n$ 의 합집합

$\{A_1, A_2, \dots, A_m\} = \text{전체 } S \text{의 분할}$

$A_1, \dots, A_n$ 은 서로소  
(배반)



# 베이즈 정리 산식 2

159

$$P(A|B) = P(B|A) P(A) / P(B)$$

- .  $P(A|B)$  : A의 조건부확률 또는 B라는 특정 값에 의한 사후 확률 (Posteriori)
  - \* 사건 B가 일어났다는 것을 알고, 그것이 원인 A로부터 일어난 것이라고 생각되는 조건부확률
- .  $P(B|A)$  : A가 주어졌을 때의 조건부확률 (Likelihood, 우도)
- .  $P(A)$  : A의 사전확률 (Priori)
- .  $P(B)$  : B의 사전확률 (Evidence)

# 베이즈 정리 산식 3

160

사후확률은 사전확률과 우도를 곱하고 사전확률로 나눈 것도 같음

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$P(B) > 0$  일 때,

$$P(A_1 | B) = \frac{P(A_1 \text{ 그리고 } B)}{P(B)} = \frac{P(A_1)P(B | A_1)}{P(A_1)P(B | A_1) + \cdots + P(A_m)P(B | A_m)}$$



# 베이지 정리 산식 4 : 전체

161

② B는 S위에서 정의된 사건,  $P(A) \neq 0$

사후확률

$$\rightarrow P(A_k|B)$$

$$= \frac{P(A_k \cap B)}{P(B)}$$



$$\textcircled{1} P(B) = P(A_1 \cap B) + P(A_2 \cap B) + P(A_3 \cap B)$$

$$\textcircled{2} \text{ 곱셈정리 } P(A \cap B) = P(A) \times P(B|A)$$

$$= \frac{P(A_k \cap B)}{P(A_1 \cap B) + P(A_2 \cap B) + \cdots + P(A_n \cap B)}$$

$$= \frac{P(B|A_k)P(A_k)}{P(B|A_1)P(A_1) + P(B|A_2)P(A_2) + \cdots + P(B|A_n)P(A_n)}$$

(단 k가  $k=1, 2, 3, \cdots, n$ )

사전확률



우도



# 확률 변수, 확률 분포, 확률 함수 기초

# 확률 변수

163

확률 변수(確率變數, random variable)는 확률 실험)시행에서 나타낼 수 있는 모든 결과에 대해 수치를 대응시키는 것이며, 확률적인 과정에 따라 값이 결정되는 변수

# 이산확률변수

164

이산확률변수(discrete probability variable) 유한개의 값, 또는 셀 수 있는 개수의 값으로 구성되어 있는 확률변수

## 동전을 두번 던질 경우

X	0	1	2
P	1/4	1/2	1/4

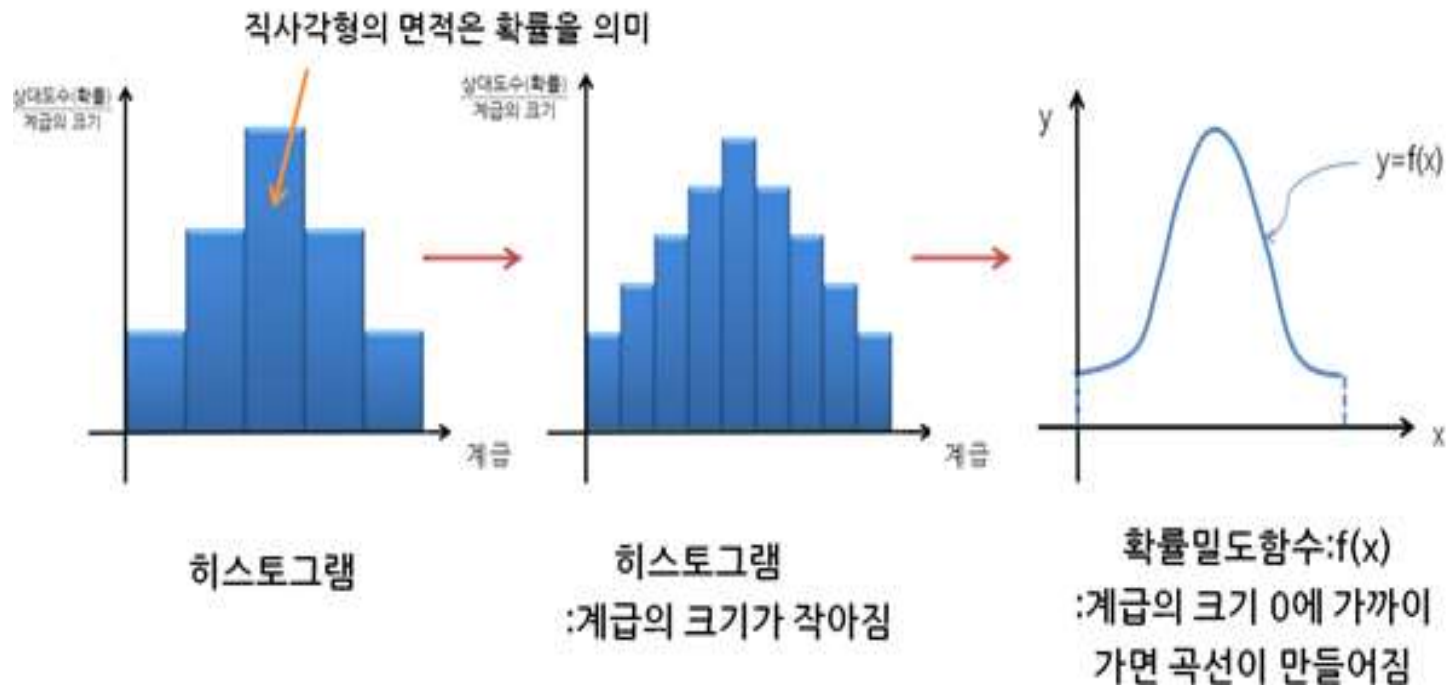
확률분포를 확률질량함수로 표현하면

$$P(X=x) = \begin{cases} \frac{1}{4} & (x=0,2) \\ \frac{1}{2} & (x=1) \end{cases}$$

# 연속확률변수

165

연속적인 범위의 값을 지니는 확률변수, 연속량으로 표기되어 가능한 변수 값의 셀 수 없는 변수



# 확률분포

166

확률분포(確率分布, probability distribution)는 어떤 확률변수가 취할 수 있는 모든 값들과 그 값을 취할 확률의 대응관계(상대도수, 함수)로 표시를 하는 것

주사위 하나를 던질 경우

표본점	1	2	3	4	5	6
X	1	2	3	4	5	6
P	1/6	1/6	1/6	1/6	1/6	1/6

# 이산확률분포

167

이산 확률분포는 이산 확률변수가 가지는 확률분포를 의미한다. 여기에서 확률변수가 이산 확률변수라는 말은 확률변수가 가질 수 있는 값의 개수가 가산 개 있다는 의미

- 이산균등분포
- 푸아송 분포
- 베르누이 분포
- 기하 분포
- 초기하 분포
- 이항 분포
- 음의 이항 분포
- 다항 분포

- 확률변수  $X$ 의 각 값에 대한 확률  $P(X=x)$ 는  
$$0 \leq P(X=x) \leq 1$$

- 모든 확률값의 합은 1

$$\sum p(x) = 1$$

# 이산확률분포의 성질

168

특정 확률변수의 확률은  $0 \leq p(X=x_i) \leq 1$  값  
이고 모든 확률변수의 합은 1이다.

동전 2개를 던질 경우 앞면의  
나올 확률 변수와 분포

X	0	1	2
P	1/4	1/2	1/4

$$(1) 0 \leq P(X = x_i) \leq 1$$

$$(2) \sum_{i=1}^n P(X = x_i) = 1$$

$$(3) P(X = x_i \text{ or } X = x_j) = P(X = x_i) + P(X = x_j) \text{ (단, } i \neq j)$$

$$(4) P(a \leq X \leq b) = \sum_a^b P(X=x) + P(X=a) + P(X=a+1) + \dots + P(X=b)$$



# 연속확률분포

169

연속 확률분포는 확률 밀도 함수를 이용해 분포를 표현할 수 있는 경우를 의미한다. 연속 확률분포를 가지는 확률변수는 연속 확률변수라고 부른다.

- 정규 분포
- 연속균등분포
- 카이제곱 분포
- 감마 분포

# 연속균등분포

170

연속균등분포(continuous uniform distribution)은 연속 확률 분포로, 분포가 특정 범위 내에서 균등하게 나타나 있을 경우를 가리킨다. 이 분포는 두 개의 매개변수  $a, b$ 를 받으며, 이때  $[a, b]$  범위에서 균등한 확률을 가진다.

기호로  $\mathcal{U}(a, b)$ .

$\mathcal{U}(0, 1)$ 인 경우를 표준연속균등분포

# 정규분포

171

정규분포(正規分布, normal distribution) 또는  
가우스 분포(Gauß分布, Gaussian distribution)  
는 연속 확률 분포의 하나 정규분포

$N(\mu, \sigma^2)$ 로 표기:

매개 변수 평균  $\mu$ 과 표준편차  $\sigma$ 에 대해 모양이 결정!

평균이 0이고 표준편차가 1인 정규분포  $N(0, 1)$ 을 표준정규분포

# Die: 이산확률변수 생성

172

## 유한확률변수를 생성하는 함수

```
from sympy.stats import Die, density
D6 = Die('D6', 6) # Six sided Die
print(D6, type(D6))
print(density(D6).dict)
```

```
(D6, <class 'sympy.stats.rv.RandomSymbol'>)
{1: 1/6, 2: 1/6, 3: 1/6, 4: 1/6, 5: 1/6, 6: 1/6}
```

```
help(Die)
```

Help on function Die in module sympy.stats.frv\_types:

Die(name, sides=6)

Create a Finite Random Variable representing a fair die.

Returns a RandomSymbol.

# given : 특정 사건을 배정

173

특정 확률변수의 집합에서 표현식에 맞는 부분 집합을 만드는 함수

```
from sympy.stats import given, density, Die

X = Die('X', 6)
print(density(X).dict)

Y = given(X, X > 3)
print(Y, type(Y))
print(density(Y).dict)
```

```
{1: 1/6, 2: 1/6, 3: 1/6, 4: 1/6, 5: 1/6, 6: 1/6}
(X, <class 'sympy.stats.rv.RandomSymbol'>)
{4: 1/3, 5: 1/3, 6: 1/3}
```

# 이산균등분포

174

이산 균등 분포 함수인 `random.randint` : Discrete uniform distribution

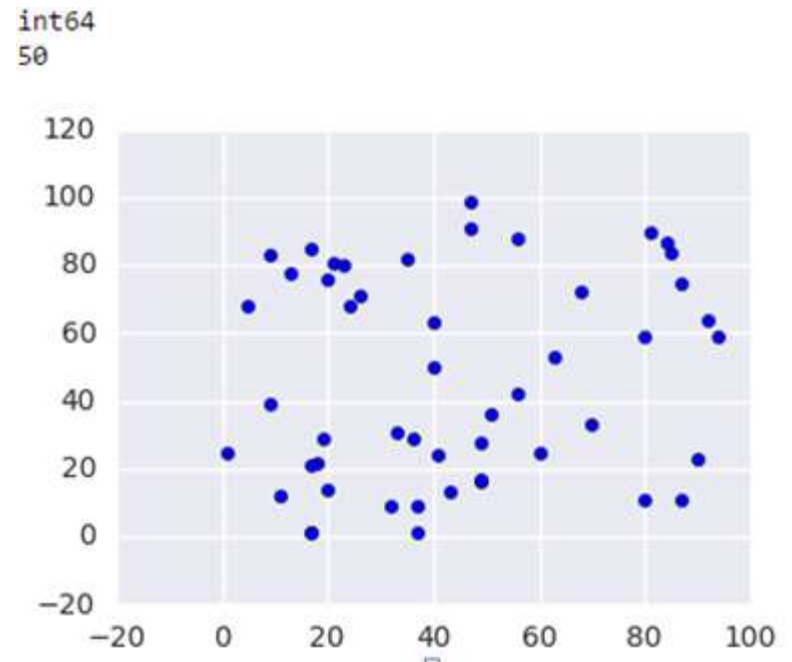
`randint(low, high=None, size=None, dtype='l')`

```
import matplotlib.pyplot as plt
import numpy as np
x = np.random.randint(0,100,(50,2))
plt.figure(figsize=(4,3))
print(x.dtype)

print(len(x[:,0]))

plt.scatter(x[:,0],x[:,1])

plt.show()
```



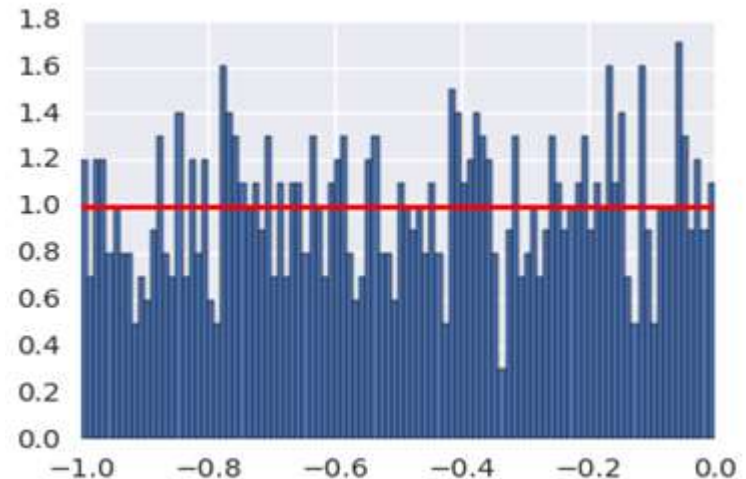
# 연속균등분포 예시

175

Uniform 메소드를 이용해서 연속균등분포를 표시

`uniform(low=-1, high=0, size=None)`

```
import matplotlib.pyplot as plt
import numpy as np
plt.figure(figsize=(4,3))
s = np.random.uniform(-1,0,1000)
count, bins, ignored = plt.hist(s, 100, normed=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
plt.show()
```



# 정규분포 예시

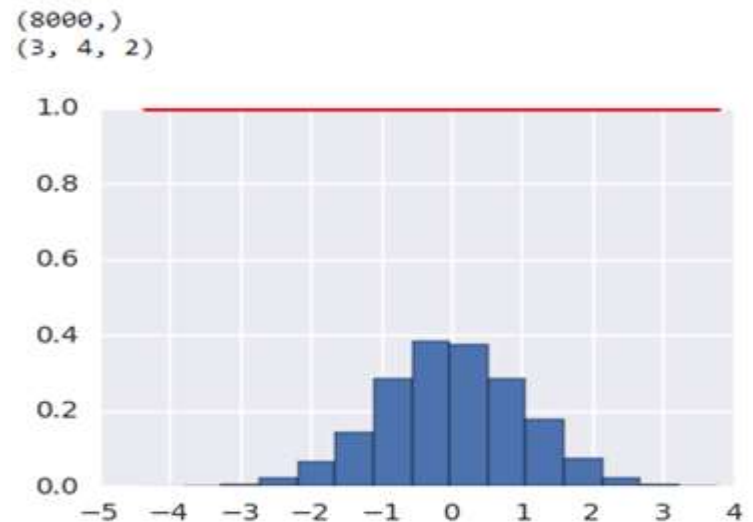
176

**정규분포 : random.standard\_normal**

평균은 0 이고 표준편차가 1 인 분포를 정규분포  
`standard_normal(size=None)`

```
s1 = np.random.standard_normal(8000)
print(s1.shape)
plt.figure(figsize=(4,3))
s2 = np.random.standard_normal(size=(3, 4, 2))
print(s2.shape)

count, bins, ignored = plt.hist(s1, 15, normed=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
plt.show()
```

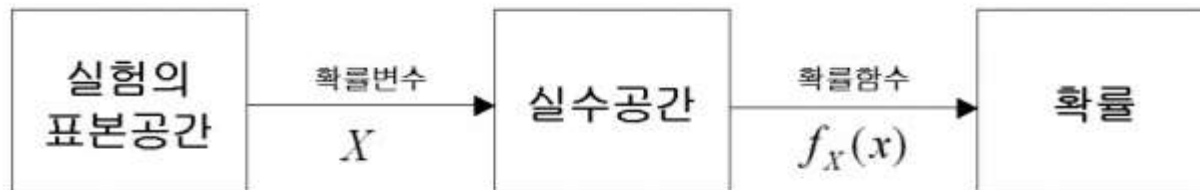




# 확률함수

177

확률변수에 대하여 정의된 실수를 0과 1 사이의 실수(확률)에 대응시키는 함수



# 확률 질량함수

178

확률질량함수(probability mass function)은 이산확률변수에서 특정한 값에 대한 확률을 나타내는 함수

$x$	$x_1$	$x_2$	$x_3$	$x_4$	$\dots$	$x_n$
$P(X=x_i)$	$p_1$	$p_2$	$p_3$	$p_4$	$\dots$	$p_n$

확률질량함수가  $P(X=x_i)=p_i$  ( $i=1,2,\dots,n$ ) 일 때,

$$\textcircled{1} \quad 0 \leq P(X=x_i) \leq 1$$

$$\textcircled{2} \quad \sum_{i=1}^n P(X=x_i) = p_1 + p_2 + \dots + p_n = 1$$

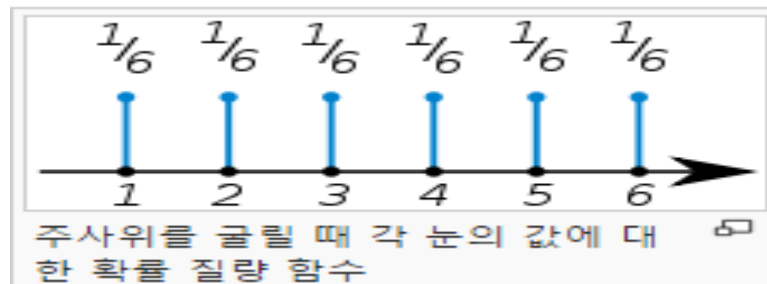
$$\textcircled{3} \quad P(x_i \leq X \leq x_j) = p_i + p_{i+1} + \dots + p_j \quad (i \leq j)$$

# 확률질량함수 예시

179

주사위를 한 번 굴릴 때의 값을 나타내는 확률 변수가  $X$ 일 때, 이 확률 변수에 대응되는 확률 질량 함수는  $f(x)=1/6$ 이다.

$x$	1	2	3	4	5	6
$P(X=x_i)$	$1/6$	$1/6$	$1/6$	$1/6$	$1/6$	$1/6$



# 확률 밀도 함수

180

확률 밀도 함수(probability density function 약자 PDF ) $f(x)$ 는 일정한 값들의 범위를 포괄하는 연속확률변수의 확률을 찾는 데 사용하고 연속데이터에서의 확률을 구하기 위해서 확률 밀도 함수의 면적을 구함

확률 밀도 함수  $f(x)$ 와 구간  $[a,b]$ 에 대해 확률변수  $X$ 가 구간에 포함될 확률  $P(a \leq X \leq b)$ 는

$$\int_a^b f(x)dx$$

확률 밀도 함수  $f(x)$ 는 다음의 두 조건을 만족해야 한다.

1. 모든 실수값  $x$ 에 대해  $f(x) \geq 0$

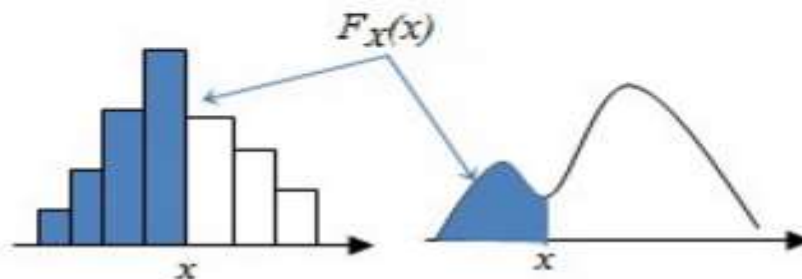
2. 
$$\int_{-\infty}^{\infty} f(x)dx = 1$$

# 누적분포함수

181

누적분포함수(CDF)는 분포와 관련된 누적 확률입니다. 특히, 누적분포함수는 사용자가 지정한 값까지에 대한 확률밀도함수 아래의 영역입니다. 특정 값 미만, 특정 값 이상 또는 두 값 사이에 반응값이 있을 확률을 구하는 경우에 누적분포함수를 사용합니다.

$$F_X(x) = P\{X \leq x\} = \begin{cases} \sum_{\xi=-\infty}^x p(\xi) & (\text{이산}) \\ \int_{-\infty}^x f(\xi) d\xi & (\text{연속}) \end{cases}$$



# 확률 질량함수: binomial 예시

182

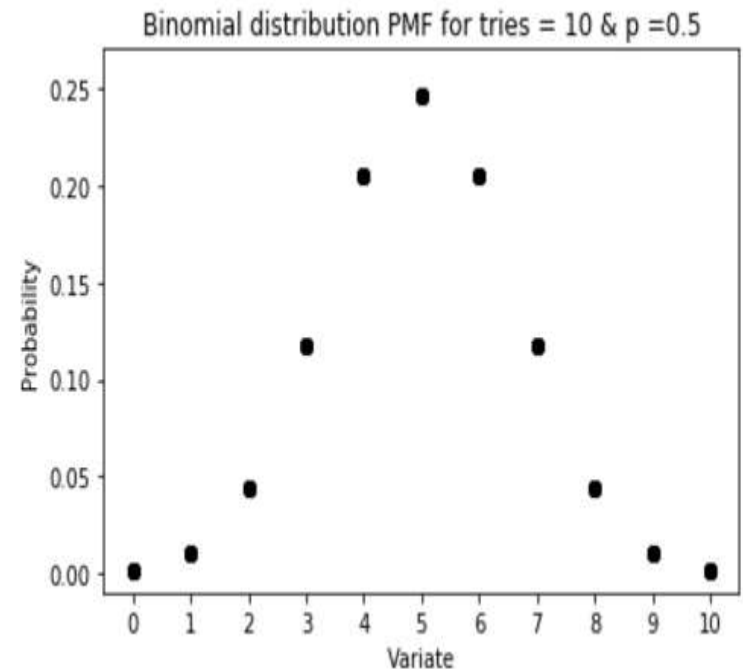
총 10개의 원소에 대한 binomial에 대한 PMF  
함수 처리

```
from scipy import stats
x = range(11) # 0 to 10
y = stats.binom.pmf(tries, 10, 0.5)

plt.plot(x,y,"o", color="black")

# Format x-axis and y-axis.
plt.axis([-1*(max(x)-min(x))*0.05, max(x)*1.05, -0.01, max(y)*1.10])
plt.xticks(x)
plt.title("Binomial distribution PMF for tries = {0} & p = {1}".format(10,0.5))
plt.xlabel("Variate")
plt.ylabel("Probability")

plt.show()
```



# 확률 밀도 함수: normal 예시

183

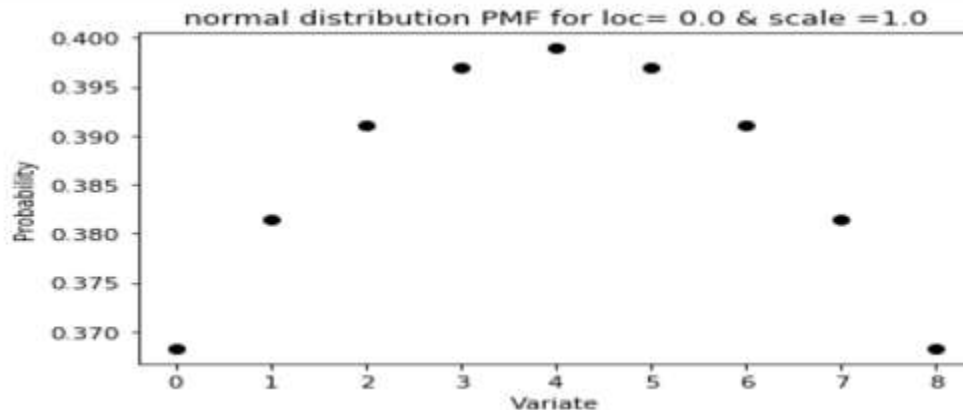
총 9개의 원소에 대한 normal에 대한 PDF 함수 처리

```
from scipy import stats
# PDF of Gaussian of mean = 0.0 and std. deviation = 1.0 at 0.
x = [-0.4, -0.3, -0.2, -0.1, 0.0, 0.1, 0.2, 0.3, 0.4]
s1 = stats.norm.pdf(x, loc=0.0, scale=1.0)
plt.plot(s1, "o", color="black")

# Format x-axis and y-axis.

plt.title("normal distribution PMF for loc= {0} & scale = {1}".format(0.0, 1.0))
plt.xlabel("Variate")
plt.ylabel("Probability")

plt.show()
```



# density : 이산형 데이터

184

이산 데이터에 대해 확률분포에 대한 데이터를 산출해서 dict으로 값을 나타냄

```
from sympy.stats import DiscreteUniform, density
from sympy import symbols

X = DiscreteUniform('X', symbols('a b c'))
print(density(X).dict)

Y = DiscreteUniform('Y', list(range(5)))
print(density(Y).dict)
```

```
{c: 1/3, b: 1/3, a: 1/3}
{0: 1/5, 1: 1/5, 2: 1/5, 3: 1/5, 4: 1/5}
```



# density : 연속형 데이터

185

연속 데이터는 lambdas 즉 함수로 나타냄

```
from sympy.stats import density, Die, Normal
from sympy import symbols
x = Symbol('x')
X = Normal(x, 0, 1)

print(density(X))
print(density(X)(x))
```

```
NormalDistribution(0, 1)
sqrt(2)*exp(-x**2/2)/(2*sqrt(pi))
```

# 누적분포함수 예시 : 균등분포

186

균등분포에서 loc, scale을 지정해서 각 확률값의 누적분포를 구하기

```
from scipy.stats import uniform
|
print(uniform.cdf([0, 1, 2, 3, 4, 5], loc = 1, scale = 4))
print(uniform.cdf([0, 1, 2, 3, 4, 5], loc = 0, scale = 5))
```

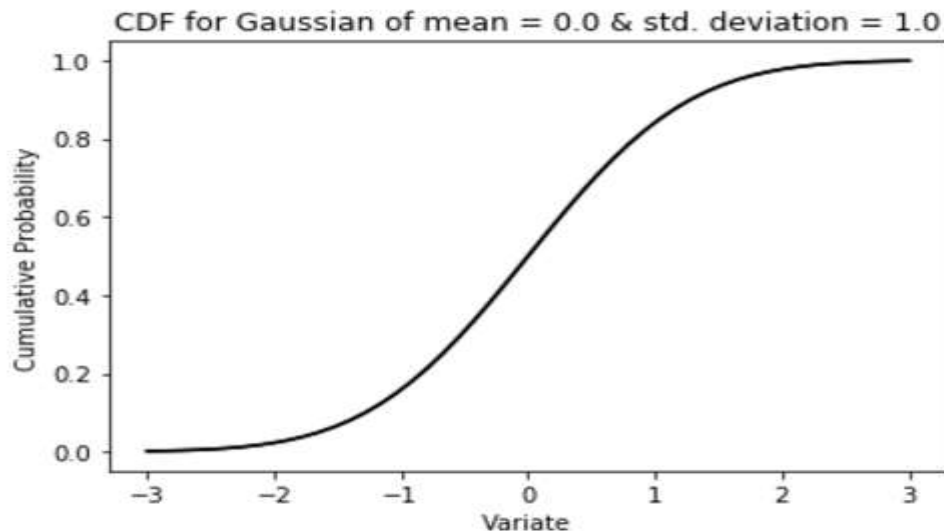
```
[ 0.    0.    0.25  0.5   0.75  1. ]
[ 0.    0.2  0.4  0.6  0.8  1. ]
```

# 누적분포함수 예시 : 정규분포

187

```
import numpy as np
mean=0.0
std=1.0
# 50 numbers between  $-3\sigma$  and  $3\sigma$ 
x = np.linspace(-3*std, 3*std, 50)
# CDF at these values
y = stats.norm.cdf(x, loc=mean, scale=std)

plt.plot(x,y, color="black")
plt.xlabel("Variate")
plt.ylabel("Cumulative Probability")
plt.title("CDF for Gaussian of mean = {0} & std. deviation = {1}".format(
    mean, std))
plt.show()
```



# NUMPY.RANDOM ARRAY 생성하기

# rand : uniform distribution

189

rand(균등분포)에 따라 ndarray 를 생성 모양이 없을 경우는 scalar 값을 생성

```
help(np.random.rand)
```

Help on built-in function rand:

```
rand(...)  
    rand(d0, d1, ..., dn)
```

Random values in a given shape.

Create an array of the given shape and populate it with random samples from a uniform distribution over ``[0, 1)``.

Parameters

-----

d0, d1, ..., dn : int, optional  
 The dimensions of the returned array, should all be positive.  
 If no argument is given a single Python float is returned.

```
import numpy as np
```

```
a = np.random.rand(3,2)  
print a
```

```
b = np.random.rand(3,3,3)  
print b
```

```
[[ 0.0752175  0.79891669]  
 [ 0.1987389  0.28034686]  
 [ 0.11460388  0.78417325]]  
[[[ 0.99693818  0.41354745  0.36525727]  
  [ 0.46231114  0.10932102  0.42984677]  
  [ 0.10382347  0.8395964   0.27662694]]]
```

```
[[ 0.26679694  0.18482395  0.34896755]  
 [ 0.41792243  0.41757001  0.33796646]  
 [ 0.52556837  0.41251447  0.60324924]]]
```

```
[[ 0.93210158  0.26566491  0.39140693]  
 [ 0.56683329  0.27211027  0.4313506 ]  
 [ 0.48310325  0.48292389  0.43183832]]]
```

# randn : "standard normal"distribution

190

randn(정규분포)에 따라 ndarray 를 생성 모양이 없을 경우는 scalar 값을 생성

```
help(np.random.randn)
```

Help on built-in function randn:

```
randn(...)  
    randn(d0, d1, ..., dn)
```

Return a sample (or samples) from the "standard normal" distribution.

If positive, int\_like or int-convertible arguments are provided, `randn` generates an array of shape ``(d0, d1, ..., dn)``, filled with random floats sampled from a univariate "normal" (Gaussian) distribution of mean 0 and variance 1 (if any of the :math:`d\_i` are floats, they are first converted to integers by truncation). A single float randomly sampled from the distribution is returned if no argument is provided.

This is a convenience function. If you want an interface that takes a tuple as the first argument, use `numpy.random.standard\_normal` instead.

Parameters

-----

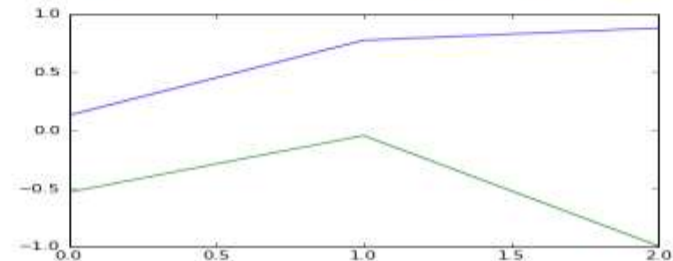
d0, d1, ..., dn : int, optional

The dimensions of the returned array, should be all positive.

If no argument is given a single Python float is returned.

```
import numpy as np  
import matplotlib.pyplot as plt  
  
a = np.random.randn(3,2)  
print a  
  
b = np.random.randn(3,3,3)  
print b  
  
plt.plot(a)  
plt.show()
```

```
[[ 0.12716211 -0.53405469]  
 [ 0.77326819 -0.04651046]  
 [ 0.87867714 -0.99659679]]  
[[[ 0.30387366  0.73511578  0.36523878]  
  [-0.50895664 -0.02040772 -1.15439572]  
  [-0.33166337  0.50971304 -0.53499204]]  
 [[-0.06169323  0.58266167  0.73162031]  
  [ 0.94361527  0.34827003  0.2303303 ]  
  [ 0.91941712  0.17552203 -0.77952844]]  
 [[ 0.15293104  0.03218082 -0.39159998]  
  [ 1.16038804  0.00441932 -1.49884859]  
  [-1.42187087  0.51848507  0.91256891]]]
```



# randint

191

`randint(low, high=None, size=None)`는 최저값, 최고값-1, 총 길이 인자를 넣어 `ndarray`로 리턴  
Size에 tuple로 선언시 다차원 생성

```
help(np.random.randint)
```

Help on built-in function randint:

```
randint(...)
    randint(low, high=None, size=None, dtype='i')

    Return random integers from 'low' (inclusive) to 'high' (exclusive).

    Return random integers from the "discrete uniform" distribution of
    the specified dtype in the "half-open" interval [low, high). If
    high is None (the default), then results are from [0, low).

    Parameters
    -----
    low : int
        Lowest (signed) integer to be drawn from the distribution (unless
        ``high=None``, in which case this parameter is the *highest* such
        integer).
    high : int, optional
        If provided, one above the largest (signed) integer to be drawn
        from the distribution (see above for behavior if ``high=None``).
    size : int or tuple of ints, optional
        Output shape. If the given shape is, e.g., ``(m, n, k)``, then
        ``m * n * k`` samples are drawn. Default is None, in which case a
        single value is returned.
    dtype : dtype, optional
        Desired dtype of the result. All dtypes are determined by their
        name, i.e., 'int64', 'int', etc, so byteorder is not available
        and a specific precision may have different C types depending
        on the platform. The default value is 'np.int'.

    .. versionadded:: 1.11.0
```

```
import numpy as np
```

```
outcome = np.random.randint(1, 7, size=10)
print outcome
print type(outcome)
print len(outcome)

print np.random.randint(2, size=10)

print np.random.randint(1, size=10)
print np.random.randint(5, size=(2, 4))
```

```
[1 2 1 3 6 1 3 3 4 4]
<type 'numpy.ndarray'>
10
[0 0 0 0 0 0 1 1 0 1]
[0 0 0 0 0 0 0 0 0 0]
[[2 4 0 0]
 [0 4 4 3]]
```

# random\_sample

192

random\_sample(size=None)에 size가 없을 경우는 하나의 값만 생성하고 size를 주면 ndarray를 생성

```
help(np.random.random_sample)
```

Help on built-in function random\_sample:

```
random_sample(...)  
    random_sample(size=None)
```

Return random floats in the half-open interval [0.0, 1.0).

Results are from the "continuous uniform" distribution over the stated interval. To sample :math:`\text{Unif}[a, b)`,  $b > a$  multiply the output of `random\_sample` by `(b-a)` and add `a`::

```
(b - a) * random_sample() + a
```

Parameters

-----

size : int or tuple of ints, optional

Output shape. If the given shape is, e.g., `(m, n, k)`, then `m \* n \* k` samples are drawn. Default is None, in which case a single value is returned.

```
: import numpy as np
```

```
print np.random.random_sample(5)
```

```
x = np.random.random_sample((3, 4))
```

```
print(x)
```

```
a = -3.4
```

```
b = 5.9
```

```
A = (b - a) * np.random.random_sample((3, 4)) + a
```

```
print(A)
```

```
[ 0.83365888  0.91664721  0.84164093  0.98135521  0.67782277]  
[[ 0.08354712  0.61649359  0.99886791  0.30827659]  
 [ 0.9902709   0.70137324  0.51211216  0.37672715]  
 [ 0.12145167  0.57292599  0.72421099  0.16748678]]  
[[ 2.80786455  1.42660285 -2.91233791  0.34301562]  
 [ 0.94101945 -1.71592464 -0.19670713  2.00826254]  
 [-2.06173124  3.71974593  3.67553201  0.67845224]]
```



# ranf

193

ranf(size=None)에 size가 없을 경우는 하나의 값만 생성하고 size를 주면 ndarray를 생성

```
import numpy as np

print np.random.ranf(5)
print np.random.random_sample(5)
print np.random.ranf((2,2))
```

```
[ 0.13547879  0.09421555  0.38304681  0.59292648  0.45954881]
[ 0.31153134  0.23721227  0.25197815  0.96749517  0.64254198]
[[ 0.27879309  0.76770788]
 [ 0.58450171  0.77284016]]
```

# random

194

random(size=None)에 size가 없을 경우는 하나의 값만 생성하고 size를 주면 ndarray를 생성

```
import numpy as np

print np.random.random(5)
print np.random.random_sample(5)
print np.random.random((2,2))
```

```
[ 0.95588669  0.43502756  0.75036209  0.42992576  0.65638792]
[ 0.52576854  0.44877826  0.34428077  0.50822697  0.85515137]
[[ 0.97766932  0.53618779]
 [ 0.97520169  0.95049509]]
```

# RandomState : 생성

195

size를 argument로 취하는데 기본값은 None이다. 만약 size가 None이라면, 하나의 값이 생성되고 반환된다. 만약 size가 정수라면, 1-D 행렬이 랜덤변수들로 채워져 반환된다.

```
import numpy as np

rng=np.random.RandomState(10)
z=np.asarray(rng.uniform(size=(2,5)))
print(z)
z1=np.asarray(rng.standard_normal(size=(2,5)))
print(z1)
```

```
[[ 0.77132064  0.02075195  0.63364823  0.74880388  0.49850701]
 [ 0.22479665  0.19806286  0.76053071  0.16911084  0.08833981]]
[[ 0.26551159  0.10854853  0.00429143 -0.17460021  0.43302619]
 [ 1.20303737 -0.96506567  1.02827408  0.22863013  0.44513761]]
```

# RandomState : seed/get\_state

196

Seed는 반복 가능한 것을 처리할 때 사용하면  
get-state()로 처리하면 현재 상태가 출력

```
import numpy as np

np.random.seed(1234)
print np.random.uniform(0, 10, 5)

r = np.random.RandomState(1234)
print r.uniform(0, 10, 5)
print r.get_state()
```

```
[ 1.9151945  6.22108771  4.37727739  7.85358584  7.79975808]
[ 1.9151945  6.22108771  4.37727739  7.85358584  7.79975808]
('MT19937', array([2260313690,  348938374,  3392255680, 2909033704, 140638832,
    1016917445, 4051655600,  976942074, 1628339371,  932989997,
    417988570,  3106230116, 3847402493, 2846838083, 1854065059,
    2365406610,  631390710, 3006558680, 1855109059, 230064328,
    758538135, 1999313224, 2345696623, 4174662269, 280561112,
    1706268812, 4182435209, 1014638053,  610687375, 2331525695,
    3432349290, 1302213857, 2461808965, 1211193860, 3120004290,
    159403718,  785407708, 1103582039, 2181742160, 4003474818,
    3333684546, 2164025542, 3329631014, 3331897623,  44841503,
    2124190575, 4103716897, 1985760015, 3231349092, 2579223365,
    2045506447, 1684183393, 4105280043,  264622240, 3457386480,
    3640204395, 3308050770, 1877785652, 1671495555,  489816736,
    3645271096, 1053221445, 4106060276, 3422181599, 2615443735,
    1804244509, 3539838858, 3452443235, 2399113344, 1565187405,
    2474733055, 1149428025, 3675651662, 3578586687, 2141819878,
    3912530560, 3882533229,  502129057, 2262588256, 243511723,
    3669928491, 2909147132, 2533094556, 1693913013, 1031455591,
```

# seed

197

seed는 반복적인 random을 동일한 범주에서 처리하기 위한 방식으로 random변수를 고정시킴

```
help(np.random.seed)
```

Help on built-in function seed:

```
seed(...)
```

```
    seed(seed=None)
```

Seed the generator.

This method is called when `RandomState` is initialized. It can be called again to re-seed the generator. For details, see `RandomState`.

Parameters

-----

seed : int or array\_like, optional

Seed for `RandomState`.

Must be convertible to 32 bit unsigned integers.

```
import numpy as np
```

```
for i in range(5):
```

```
    arr = np.arange(5) # [0, 1, 2, 3, 4]
```

```
    np.random.seed(1) # Reset random state
```

```
    np.random.shuffle(arr) # Shuffle!
```

```
    print arr
```

```
[2 1 4 0 3]
```

```
[2 1 4 0 3]
```

```
[2 1 4 0 3]
```

```
[2 1 4 0 3]
```

```
[2 1 4 0 3]
```

# Binomial : 이항분포

198

n은 trial , p는 구간 [0,1]에 성공 P는 확률  
이항 분포에서 작성한 것임

$$P(N) = \binom{n}{N} p^N (1-p)^{n-N},$$

```
# number of trials, probability of each trial
n, p = 10, .5
# result of flipping a coin 10 times, tested 1000 times.
s = np.random.binomial(n, p, 100)
print s
a = sum(np.random.binomial(9, 0.1, 20000) == 0)/20000.
# answer = 0.38885, or 38%.
print a
```

```
[ 7  2  3  3  8  5  4  6  5  6  5  6  5  5  4  3  8  4  4 10  6  7  3  2  4
  6  4  5  7  6  7  3  5  4  4  1  3  4  6  4  7  7  5  6  4  6  7  4  4  5
  6  4  4  5  9  5  5  5  4  3  5  4  7  8  4  3  5  6  4  5  2  4  5  5  3
  4  6  7  3  7  6  5  6  2  4  6  7  5  1  6  7  7  4  1  5  6  5  7  8  7]
0.38575
```

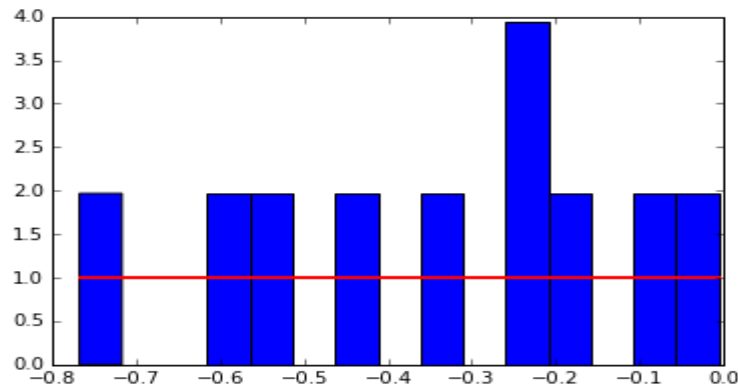
# Uniform

199

[low, high)는 low를 포함하지만 high를 포함하지 않는 정규분포를 표시

```
: s = np.random.uniform(-1,0,10)
print s
print np.all(s >= -1)
print np.all(s < 0)
import matplotlib.pyplot as plt
count, bins, ignored = plt.hist(s, 15, normed=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
plt.show()
```

[-0.24566282 -0.57820591 -0.21149247 -0.34151746 -0.52850514 -0.76802671  
-0.06483795 -0.42628098 -0.00543043 -0.19371869]  
True  
True



# standard\_normal

200

a standard Normal distribution (mean=0, stdev=1) 표시

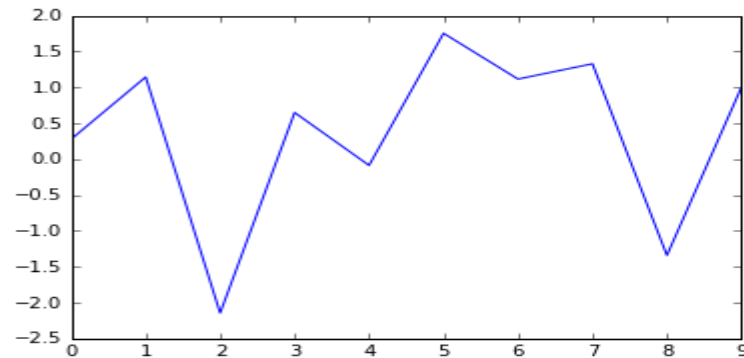
```
s = np.random.standard_normal(10)
print s
print s.shape

import matplotlib.pyplot as plt

plt.plot(s)
plt.show()

s1 = np.random.standard_normal(size=(3, 4, 2))
print s1.shape
```

[ 0.27796344 1.1438807 -2.14531 0.64878788 -0.08756643 1.75379778  
 1.11513234 1.32812215 -1.34186116 0.99861337]  
(10,)



(3, 4, 2)



# permutation

201

## 순열의 값인 원소로 선택된 배열의 원소를 섞기

```
help(np.random.permutation)
```

Help on built-in function permutation:

```
permutation(...)  
permutation(x)
```

Randomly permute a sequence, or return a permuted range.

If `x` is a multi-dimensional array, it is only shuffled along its first index.

Parameters

-----

x : int or array\_like

If `x` is an integer, randomly permute ``np.arange(x)``.

If `x` is an array, make a copy and shuffle the elements randomly.

```
import numpy as np  
  
arr = np.random.permutation(10)  
print(arr)  
  
arr1 = np.random.permutation([1, 4, 9, 12, 15])  
print(arr1)  
  
arr2 = np.arange(9).reshape((3, 3))  
print(arr2)  
arr3 = np.random.permutation(arr2)  
print(arr3)  
  
[3 1 5 7 4 8 9 2 6 0]  
[ 9 15  4  1 12]  
[[0 1 2]  
 [3 4 5]  
 [6 7 8]]  
[[3 4 5]  
 [0 1 2]  
 [6 7 8]]
```

# choice : replace 속성

202

`choice(a, size=None, replace=True, p=None)`  
a값을 array, size는 모형, replace=False는 사이즈 변경 불가, p는 나오는 원소에 대한 확률을 정의

```
import numpy as np

a = np.random.choice(5, 3, replace=False)
print a
# a[4] = 10 IndexError: index 4 is out of bounds for axis 0 with size 3

b = np.random.permutation(np.arange(5))[:3]
print b
#b[4] = 10 IndexError: index 4 is out of bounds for axis 0 with size 3
```

```
[0 1 4]
[3 0 2]
```

# shuffle

203

## 선택된 배열의 원소를 섞기

```
help(np.random.shuffle)
```

Help on built-in function shuffle:

```
shuffle(...)  
  shuffle(x)
```

Modify a sequence in-place by shuffling its contents.

Parameters

-----

x : array\_like

The array or list to be shuffled.

```
import numpy as np  
  
arr = np.arange(10)  
print(arr)  
np.random.shuffle(arr)  
print(arr)  
  
arr2 = np.arange(9).reshape((3, 3))  
print(arr2)  
np.random.shuffle(arr2)  
print(arr2)  
  
[0 1 2 3 4 5 6 7 8 9]  
[0 3 7 5 6 1 9 2 4 8]  
[[0 1 2]  
 [3 4 5]  
 [6 7 8]]  
[[6 7 8]  
 [0 1 2]  
 [3 4 5]]
```

# 확률 변수의 기대값(평균)

# 확률 변수의 기대값

205

확률 변수의 기댓값(期待값, expected value)은 각 사건이 벌어졌을 때의 이득과 그 사건이 벌어질 확률을 곱한 것을 전체 사건에 대해 합한 값이다.

이것은 어떤 확률적 사건에 대한 평균의 의미로 생각할 수 있다.

# 이산확률변수의 기대값

206

각 사건이 발생할 때의 결과와 그 사건이 벌어질 확률을 곱한 것들을 전체 사건에 대해 합한 값입니다. 어떤 확률적 사건에 대한 평균의 의미로 생각할 수 있습니다. 확률변수(=랜덤변수)가 나타내는 확률분포에서 중심 위치를 나타내는 척도이다.

여기서  $x_i$ 는 가능한 모든 사건,  $p(x_i)$ 는  $x_i$  사건이 일어날 확률을 의미한다.

예를 들어, 이산 확률 변수일 경우에는 다음과 같다.

$$E(X) = \sum_i p_i x_i$$

# 연속확률 변수의 기대값

207

연속확률 변수의 기대값은 확률 밀도함수의 적분

$$E(X) = m = \int_a^b xf(x)dx$$

# 기대값의 연산법칙

208

## 기대값의 연산법칙

[상수의 기댓값은 상수 자체이다.]

$$E(a) = a$$

[확률변수  $X$ 에 일정한 배수  $a$ 를 곱한 확률변수의 기댓값은 본래의 확률변수  $X$ 의 기댓값에  $a$ 를 곱한 값과 같다.]

$$E(aX) = a \times E(X)$$

[확률변수  $X$ 와  $Y$ 의 합의 기댓값은 각각의 기댓값의 합과 같다.]

$$E(X + Y) = E(X) + E(Y)$$

[위의 두 법칙으로부터 다음 식이 성립한다.]

$$E(aX + bY) = a \times E(X) + b \times E(Y)$$



# 이산확률변수의 기대값 예시

209

## 기대값의 예시

$\bar{X}$	1	1.5	2	2.5	3	3.5	4
$P(\bar{X})$	$\frac{1}{16}$	$\frac{2}{16}$	$\frac{3}{16}$	$\frac{4}{16}$	$\frac{3}{16}$	$\frac{2}{16}$	$\frac{1}{16}$

기대값

$$\begin{aligned} E(\bar{X}) &= \sum \bar{X} \cdot P(\bar{X}) \\ &= 1 \cdot \frac{1}{16} + 1.5 \cdot \frac{2}{16} + 2 \cdot \frac{3}{16} + 2.5 \cdot \frac{4}{16} + 3 \cdot \frac{3}{16} + 3.5 \cdot \frac{2}{16} + 4 \cdot \frac{1}{16} \\ &= \frac{5}{2} \end{aligned}$$

# 기대값 $E(5X+3)$ 예시

210

## 기대값 $E(5X+3)$ 에 대한 처리

[상수의 기댓값은 상수 자체이다.]

$$E(a) = a$$

[확률변수  $X$ 에 일정한 배수  $a$ 를 곱한 확률변수의 기댓값은 본래의 확률변수  $X$ 의 기댓값에  $a$ 를 곱한 값과 같다.]

$$E(aX) = a \times E(X)$$

$X$	1	2	4	계
$P(X = x)$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{6}$	1

$$\therefore E(X) = 1 \times \frac{1}{3} + 2 \times \frac{1}{2} + 4 \times \frac{1}{6} = 2$$

$$\therefore E(5X + 3) = 5E(X) + 3 = 13$$

# $Y = 10X^2$ 기대값

211

주사위를 던져 나오는 수의 제곱에 10배를 상금으로 준다고 할 때 상금의 기대값은 ?

$x$	1	2	3	4	5	6	기대값
$f(x)$	1/6	1/6	1/6	1/6	1/6	1/6	
$xf(x)$	1/6	$2 \cdot 1/6$	$3 \cdot 1/6$	$4 \cdot 1/6$	$5 \cdot 1/6$	$6 \cdot 1/6$	3.5
$y$	10	40	90	160	250	360	
$f(y)$	1/6	1/6	1/6	1/6	1/6	1/6	
$yf(x)$	$10 \cdot 1/6$	$40 \cdot 1/6$	$90 \cdot 1/6$	$160 \cdot 1/6$	$250 \cdot 1/6$	$360 \cdot 1/6$	156.66666

# 하나의 주사위 : 기대값, 분산

212

하나의 주사위를 던질 경우 기대값, 분산, 표준편차 예시

```
import numpy as np

s = np.array([1,2,3,4,5,6])
p = np.array([1/6,1/6,1/6, 1/6,1/6,1/6])

e = np.sum(s*p)
print(" expectation ", e)
x_e = s - e
print(x_e)
v = np.sum(x_e**2 *p)
print(np.var(s-e))
print(" variance ", v)
print(np.std(s-e))
print(" standard deviation ", np.sqrt(v))

expectation 3.5
[-2.5 -1.5 -0.5  0.5  1.5  2.5]
2.916666666667
variance 2.916666666667
1.70782512766
standard deviation 1.70782512766
```

# 확률변수의 분산/표준편차

# 확률변수의 분산

214

분산(Variance)은 각 확률변수들이 기대값(=확률사건 평균)으로부터 얼마나 떨어져서 분산되어 있는지 가늠할 수 있는 하나의 척도, 즉 하나의 값을 말한다. 사실 분산은 표준편차를 구하기 위한 과정이라고 보면 된다. 분산은 소문자 시그마의 제곱으로 표현  $\sigma^2$  한다

# 분산의 연산법칙 1

215

## 연산법칙

[두 확률변수  $X$ 와  $Y$ 의 합의 분산은 각각의 분산에 둘 사이의 공분산을 두 배 하여 더한 것과 같다.]

$$\text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y) + 2 \times \text{Cov}(X, Y)$$

$$= \sigma_X^2 + \sigma_Y^2 + 2 \times \sigma_{XY}$$

[위의 두 관계를 이용하면 다음의 관계가 성립한다.]

$$\text{Var}(aX + bY) = a^2 \times \text{Var}(X) + b^2 \times \text{Var}(Y) + 2 \times a \times b \times \text{Cov}(X, Y)$$

# 분산의 연산법칙 2

216

## 연산법칙

아래에서  $X, Y$  각각은 확률변수를, 그리고  $a, b$ 는 상수(정수 혹은 소수 혹은 분수)를 나타낸다고 가정하겠습니다.

[상수  $a$ 의 분산은 0이다.]

$$\text{Var}(a) = 0$$

[확률변수  $X$ 에 일정한 배수  $a$ 를 곱한 확률변수의 분산은 본래의 확률변수  $X$ 의 분산에  $a^2$ 을 곱한 것과 같다.]

$$\text{Var}(aX) = a^2 \times \text{Var}(X)$$



# 확률변수의 분산식

217

확률변수  $X$ 와 평균의 차의 제곱에 대한 평균을 분산으로 나타냄

편차의 제곱

$$(X_i - \mu)^2$$



편차의 제곱\* 사건확률

$$(X_i - \mu)^2 P(X_i)$$



확률변수  $X$ 와 평균의 차의 제곱에 대한 평균

$$E[(X - \mu)^2] = \sum (X - \mu)^2 P(X)$$

# 확률변수의 분산식 변형

218

$X^2$ 의 평균에서  $X$ 의 평균을 제곱한 것을 빼면  
간단한 분산식이 됨

$$m = E(X) = \sum_{i=1}^n x_i \cdot p_i$$

$$\begin{aligned}\sigma^2 = V(X) &= \sum_{i=1}^n (x_i - m)^2 \cdot p_i \\ &= \sum_{i=1}^n x_i^2 \cdot p_i - m^2\end{aligned}$$

$$\begin{aligned}\text{Var}(X) &= E[(X - E[X])^2] \\ &= E[X^2 - 2XE[X] + (E[X])^2] \\ &= E[X^2] - 2E[X]E[X] + (E[X])^2 \\ &= E[X^2] - (E[X])^2 \\ &= \sum_{i=1}^n x_i^2 \cdot p_i - m^2\end{aligned}$$

# 확률변수의 분산식

219

분산은 소문자 시그마의 제곱으로 표현  $\sigma^2$  한다

기대값

$$\begin{aligned} m = E(X) &= \sum X \cdot P(X) \\ &= 1 \cdot \frac{1}{4} + 2 \cdot \frac{1}{4} + 3 \cdot \frac{1}{4} + 4 \cdot \frac{1}{4} \\ &= \frac{5}{2} \end{aligned}$$

분산

$$\begin{aligned} \sigma^2 = V(X) &= \sum X^2 P(X) - m^2 \\ &= \left\{ 1^2 \cdot \frac{1}{4} + 2^2 \cdot \frac{1}{4} + 3^2 \cdot \frac{1}{4} + 4^2 \cdot \frac{1}{4} \right\} - \left\{ \frac{5}{2} \right\}^2 \\ &= \frac{5}{4} \end{aligned}$$

# 연속확률변수의 분산식

220

연속확률변수의 분산식은  $X^2$ 의 적분에서 평균의 제곱을 빼면 나옴

$$E(X) = m = \int_{\alpha}^{\beta} xf(x)dx$$

$$\begin{aligned} V(X) &= E((X-m)^2) = \int_{\alpha}^{\beta} (x-m)^2 f(x)dx \\ &= \int_{\alpha}^{\beta} x^2 f(x)dx - m^2 = E(X^2) - \{E(X)\}^2 \end{aligned}$$

# 확률변수의 표준편차

221

표준편차(Standard deviation)란, 확률 사건의 평균인 기대값으로부터 일반적인 평균 차이로 인식하면 된다. 표준편차는 적을 수록 기대값에 가까이 위치하고 있다는 의미다.

표준편차는 소문자 시그마(sigma)  $\sigma$  문자로 표현

# 확률 변수의 표준편차식

222

표준편차는 소문자 시그마(sigma)  $\sigma$  문자로 표현

④ 이제 분산을 제곱근으로 하여 구한 값이 표준편차(standard deviation)이다.

$$\sigma = \sqrt{E[(X - \mu)^2]}$$

# 연속확률 변수의 표준편차식

223

분산의 제공근

$$E(X) = m = \int_{\alpha}^{\beta} xf(x)dx$$

$$\begin{aligned} V(X) &= E((X-m)^2) = \int_{\alpha}^{\beta} (x-m)^2 f(x)dx \\ &= \int_{\alpha}^{\beta} x^2 f(x)dx - m^2 = E(X^2) - \{E(X)\}^2 \end{aligned}$$

$$\sigma(X) = \sqrt{V(X)} = \sqrt{\int_{\alpha}^{\beta} (x-m)^2 f(x)dx}$$

# 확률변수의 공분산

224

공분산은 두 확률변수의 상관관계를 나타내는 것이다. 만약 두 확률 변수중 하나의 변수 값이 상승하는 경향을 보일 때, 다른 확률변수 값도 상승하는 상관관계에 있다면, 공분산의 값은 양수가 된다. 반대로 두 개의 확률 변수중 하나의 변수 값이 상승하는 경향을 보일 때, 다른 확률변수 값이 하강하는 경향을 보인다면 공분산의 값은 음수가 된다.

공분산이 0에 가까우면 상관관계가 적다는 것을 의미한다.  $X_i$ 와  $X_j$ 의 공분산  $\sigma_{ij} = 0$  이면  $X_i$ 와  $X_j$ 는 독립이다.



# 확률변수의 공분산 식

225

공분산은 하나의 사건에 대하여 발생하는 두 확률변수의 각 기대값과의 차이를 곱한 것들의 합이라고 보면 된다.

$$\sigma_{AB}^2 = \sum (X_A - \mu_A)(X_B - \mu_B)P(X)$$

# 공분산의 연산법칙

226

## 연산법칙

아래에서  $X, Y, Z$  각각은 확률변수를, 그리고  $a, b, c$ 는 상수(정수 혹은 소수 혹은 분수)를 나타낸다고 가정하겠습니다.

[상수  $a$ 와 확률변수  $X$ 의 공분산은 0이다.]

$$\text{Cov}(a, X) = 0$$

[확률변수  $X$ 에 일정한 배수  $a$ 를 곱한 값과 확률변수  $Y$ 의 공분산은 두 확률변수  $X$ 와  $Y$ 의 공분산에  $a$ 를 곱한 것과 같다.]

$$\text{Cov}(aX, Y) = a \times \text{Cov}(X, Y)$$

[위의 관계를 이용하면 다음의 관계가 성립한다.]

$$\text{Cov}(aX, bY) = a \times b \times \text{Cov}(X, Y)$$

[두 확률변수  $X$ 와  $Y$ 의 합과 또 다른 확률변수  $Z$ 의 공분산은  $X$ 와  $Z$ 의 공분산과  $Y$ 와  $Z$ 의 공분산의 합과 같다.]

$$\text{Cov}(X + Y, Z) = \text{Cov}(X, Z) + \text{Cov}(Y, Z)$$

[위의 두 관계를 이용하면 다음의 관계가 성립한다.]

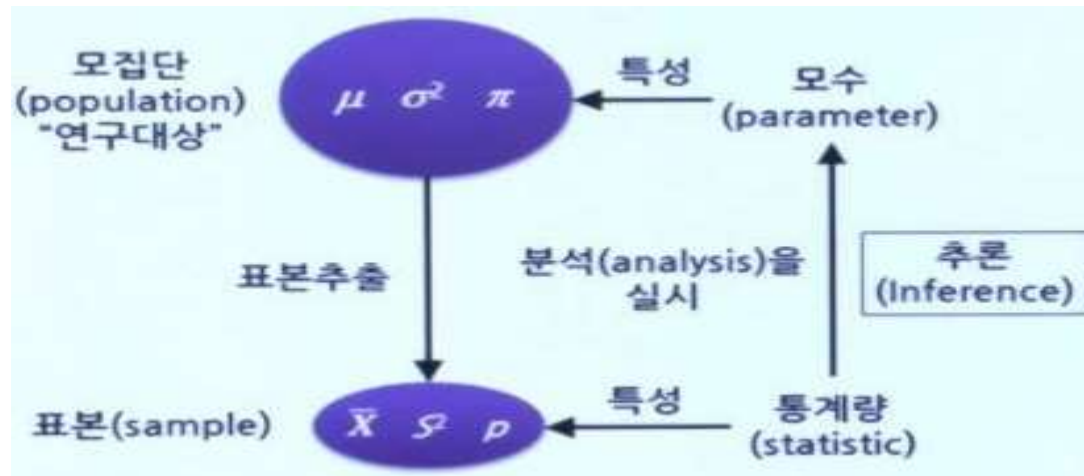
$$\text{Cov}(aX + bY, cZ) = a \times c \times \text{Cov}(X, Z) + b \times c \times \text{Cov}(Y, Z)$$

# 통계

# 통계

228

통계학(統計學, statistics)은 수량적 비교를 기초로 하여, 많은 사실을 통계적으로 관찰하고 처리하는 방법을 연구하는 학문이다.



# 통계학의 종류

229

기술통계학(descriptive statistics)는 모집단 전체 혹은 표본으로부터 얻은 데이터에 대한 숫자 요약(평균, 분산 등)이나 그래프 요약을 통해 데이터가 가진 정보를 정리하는 이론이나 방법론

추론통계학(inferential statistics)는 표본으로부터 얻은 정보를 이용해서 모집단의 특성(모수:parameter)을 추론(추정, 검정)하거나 변수들간의 적절한 함수관계(modeling)의 진위 여부를 판단하는 일련의 과정에 관한 이론과 방법론

# 확률과 통계 비교

230

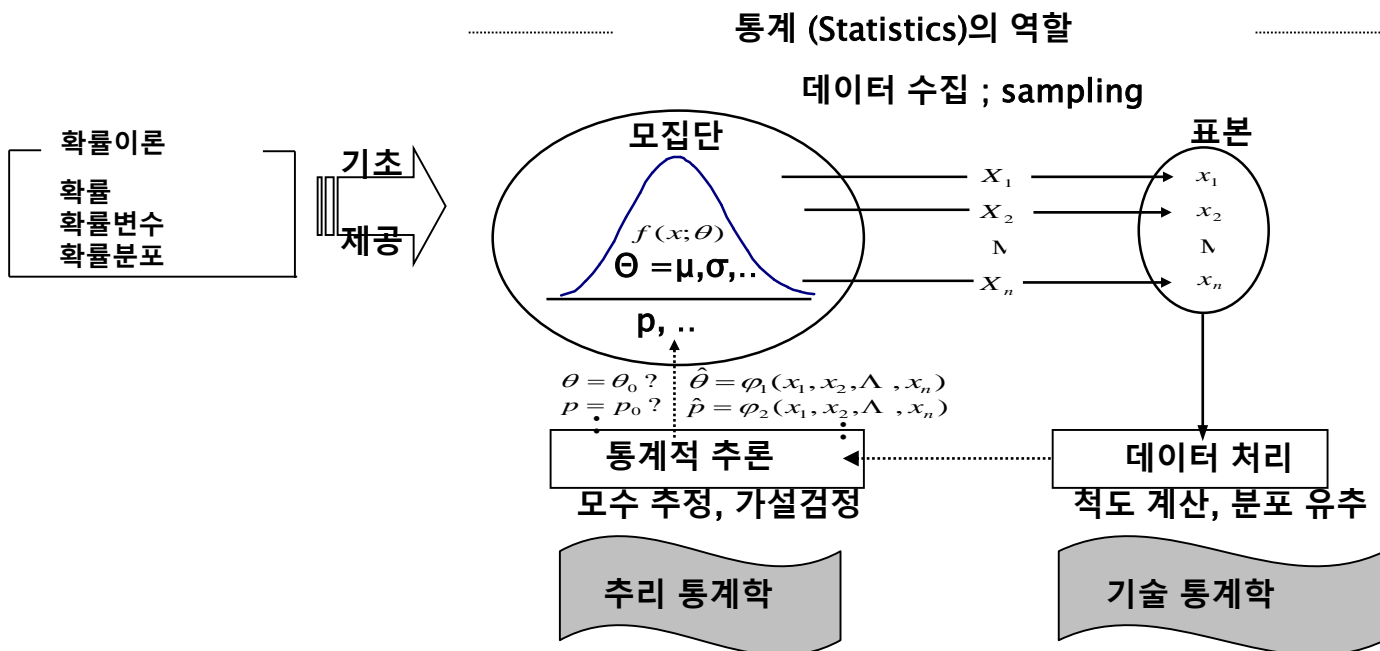
확률적 관점 : 이미 알고 있는 모집단에서 어떤 사건이 일어날 확률에 관심이 있다.

통계적 관점 : 표본에서 얻은 정보를 이용하여 미지의 모집단을 미루어 짐작하는 추론에 관심이 있다.

# 통계 분석

231

연구 목적이 설정되면 그에 맞는 통계적 가설이나 모형을 설정하고, 관련 데이터 수집하여 정리하고 분석하여, 가설 혹은 모형의 유의성을 검정하는게 통계 분석



# 전수조사

232

전수 조사(全數調査)는 대상이 되는 통계 집단의 단위를 하나하나 전부 조사하는 관찰 방법으로, 전부 조사(全部調査)라고도 부른다.

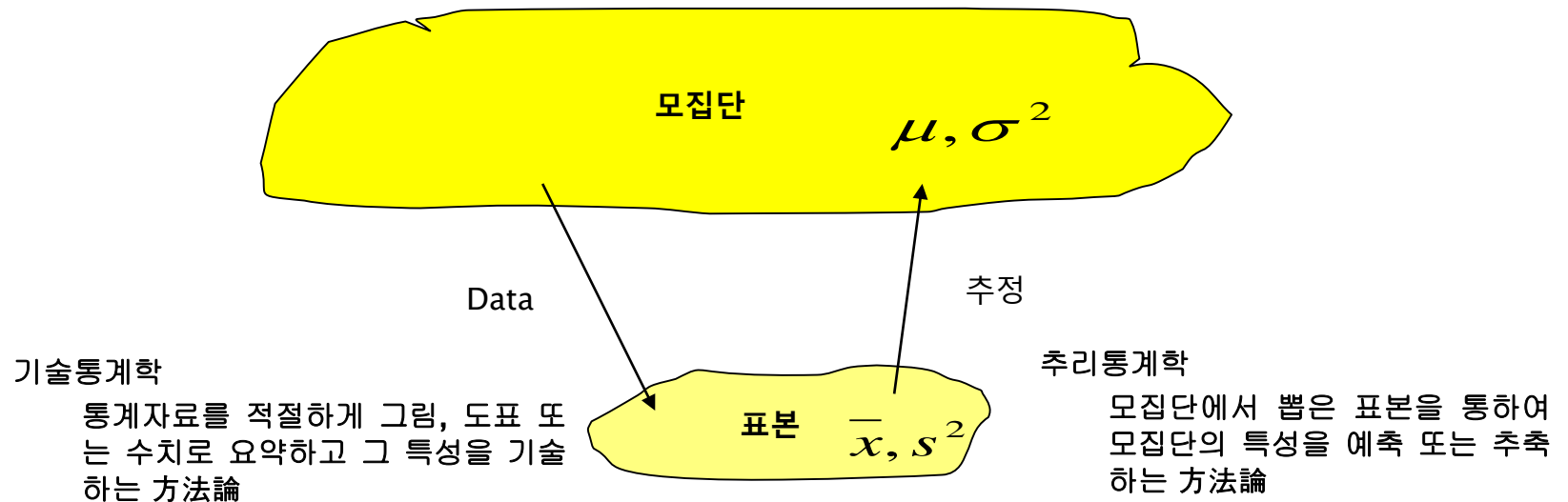


# 모집단과 표본의 관계

233

모집단에서 표본을 뽑고 이를 통해 모집단의 특성을 예측

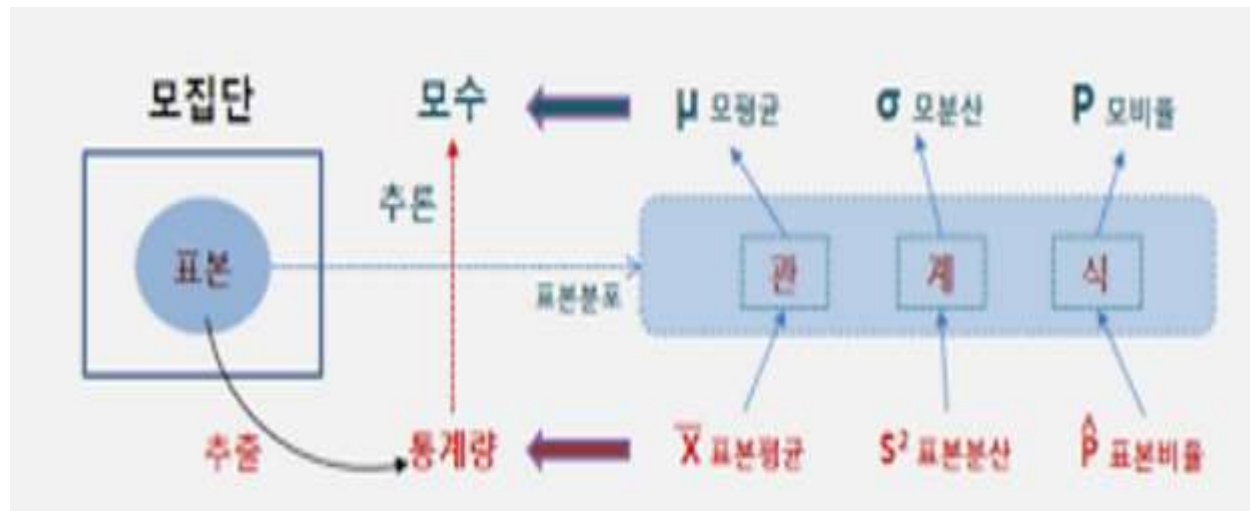
모집단과 표본의 관계



# 평균 / 표준편차 / 분산 기호

234

## 평균 표준편차 분산 기호



	평균(mean)	표준편차	분산(Variance)
모집단	$\mu$ (뮤) 또는 m	$\sigma$ (시그마)	$\sigma^2$
표본	$\bar{X}$	s	$s^2$

# 통계 데이터의 정리

# 모집단

236

모집단(母集團, population or universe)이란 정보를 얻고자 하는 관심 대상의 전체집합을 말한다.

모집단은 우리가 무엇을 알고 하느냐에 따라 다르게 정의되기 때문에 모집단을 명확하게 정의하는 것은 매우 중요하다.

# 모수

237

모수(parameter) : 모집단의 특성을 나타내는 양적인 값으로 고유한 상수로 나타낸다.

모수와 통계량의 표기와 명칭				
	모집단		표본	
	표기	명칭	표기	명칭
갯수	$N$	모집단 원소수	$n$	표본수
평균	$\mu$	모평균	$\bar{X}$	표본평균
분산	$\sigma^2$	모분산	$S^2$	표본분산
표준편차	$\sigma$	모표준편차	$S$	표본 표준편차

# 표본

238

표본(sample)은 모집단(population)의 부분집합이다. 전형적으로, 모집단은 매우 크며, 모집단의 모든 값에 대해 전수조사(census)나 전부 조사(complete enumeration)을 하는 것은 실용적이지 않거나 불가능하다.

# 표집 :sampling

239

표본은 다를 수 있을만한 크기의 부분 집합을 나타낸다. 표본이 얻어지면 모집단으로부터 얻은 표본에 대해 추론(inference) 또는 외삽법(extrapolation)을 하기 위하여 통계적 계산이 수행된다. 표본으로부터 이러한 정보를 얻기 위한 과정(process)을 표집(sampling)이라고 한다.

# 통계량

240

통계량(Statistic)은 표본으로부터 계산되는 표본의 특어값, 관측가능한 확률변수들의 실수값 함수로 통계량은 확률변수

관심정보	모수(parameter)	통계량(statistic)
분포의 중심위치 (central location)	모평균 ( $\mu$ )	표본평균 ( $\bar{x}$ )
분포의 산포 또는 흩어짐 (dispersion)	모분산 ( $\sigma^2$ ) 모표준편차 ( $\sigma$ )	표본분산 ( $s^2$ ) 표본표준편차 ( $s$ )
비율 (proportion)	모비율 ( $p$ )	표본비율 ( $\bar{p}$ )



# 표본분포

241

표본분포(Sampling Distribution)는 한 모집단에서 같은 크기로 뽑을 수 있는 모든 표본에서 통계량을 계산할 때 이 통계량이 이루는 확률분포

# 수리적 통계 변수

242

이산형 변수 (discrete variable) : 남 / 녀, 자동차 종류 등과 같이 연속성이 없는 변수

연속형 변수 (continuous variable) : 키, 몸무게 등 실수와 같이 쪼개면 쪼갤수록 무한히 쪼개지는 연속된 변수

# 데이터 분석적 통계 변수

243

측정형 변수(metric) : 셀 수 있거나 측정 가능한 특성을 측정한 변수

분류형 / 순서형 변수(non-metric) : 개체를 분류하기 위해 측정된 변수

명목형(nominal) : 범주의 크기 순서가 없는 경우

순서형(ordinal) : 범주의 크기 순서가 있는 경우

# 가설검증

244

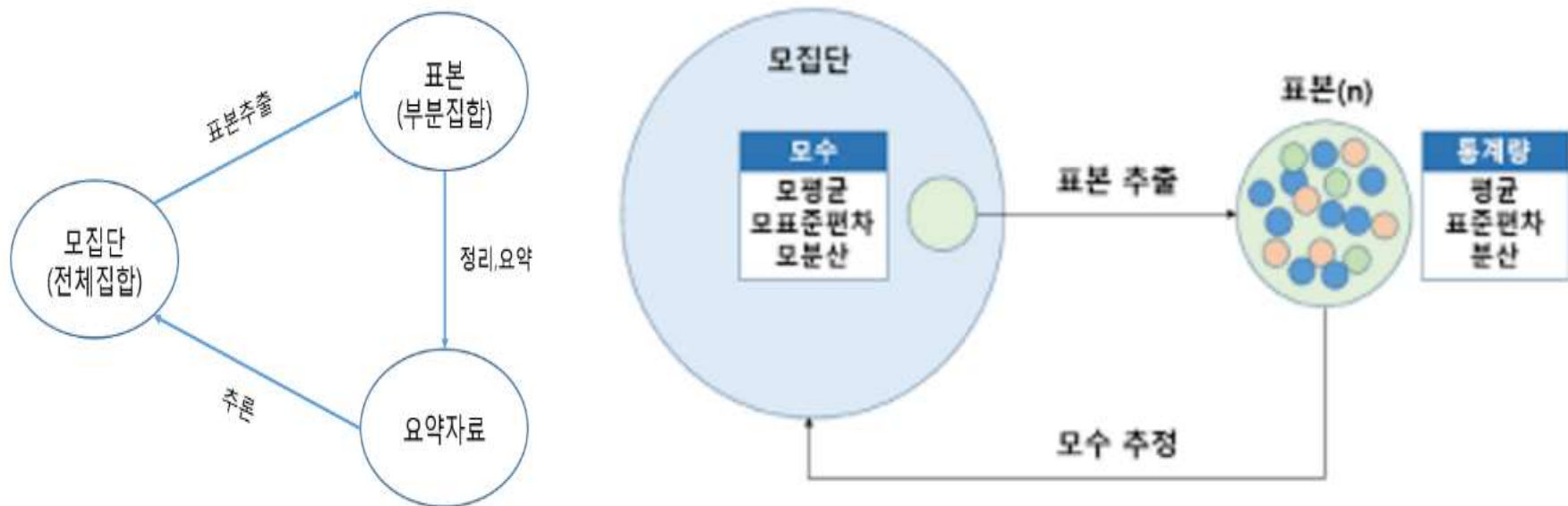
가설의 검정(test of hypotheses) : 모수의 값의 범위를 규정하는 두개의 가설을 세우고, 이들 중 어느 것이 참인지를 표본의 결과로 부터 판단하는 것

1. 자연, 사회 현상을 관찰하여 이론을 세운다.(가설)
2. 실험, 조사를 통해 실제 현상을 관찰한다. (표본 추출)
3. 실제 관찰 결과가 이론에서 예측되는 것과 부합하는지 판단한다.(가설 검정)
4. 부합되면 1번의 이론을 사실로 판단하고, 아니면 탐구를 반복한다.

# 모수의 추정

245

추정(estimation) : 미지의 모수의 값이 얼마인지, 또는 어떤 범위내에 있는지 표본결과로부터 추측하는 것



# 신뢰수준

246

## 신뢰수준

어떤 모수가 신뢰구간에 속할 확률  
추정 구간에 그 모집단 특성값의 참값이 존재할 확률  
구간 추정량이 모집단의 모수를 포함하게 될 확률  
범위: 0% ~ 100%가 가능  
보통 95%를 사용함

# 신뢰구간

247

## 신뢰구간

신뢰수준의 확률로 모평균을 포함하는 구간  
모수가 어느 범위 안에 있는지를 확률적으로 보여주는 방법

$$P(U < \theta < V) = 0.95$$

→ 신뢰구간:  $U \sim V$  → 신뢰수준: 95%

248

# 데이터의 정리



# 도수분포표 (frequency table)

249

도수분포표 (frequency table) : 빈도수 분포표라고도 하는데, 연속형 변수의 경우 히스토그램 처럼 구간을 나눠서 그래프가 아니라 테이블로 표현한 거나, 이산형 변수의 경우 각 변수의 값을 테이블로 표현한 것을 도수 분포표 라고 한다

# 도수분포표 : 그리는 단계

250

1. 관측치 중 최대값과 최소값을 찾는다.
2. 최대값과 최소값의 차이, 즉 범위를 구한다.
3. 몇 개의 구간으로 나눌 것인지 결정한다  
(대략 6개-14개).
4. 구간이 중복되지 않도록 범위를 정한다.
5. 각 구간에 속하는 관측치의 수를 세어  
도수를 구한다.

# 도수분포표의 평균

251

각 계급값과 도수를 곱해서 총합을 구하고 도수의 총합으로 나누면 평균이 나옴

점수	명
30	3
50	3
70	4
총합	10

$$\text{평균} = \frac{\text{변량의 총합}}{\text{변량의 갯수}} = \frac{\{(\text{계급값}) \times (\text{도수})\} \text{의 총합}}{\text{도수의 총합}}$$

# 히스토그램

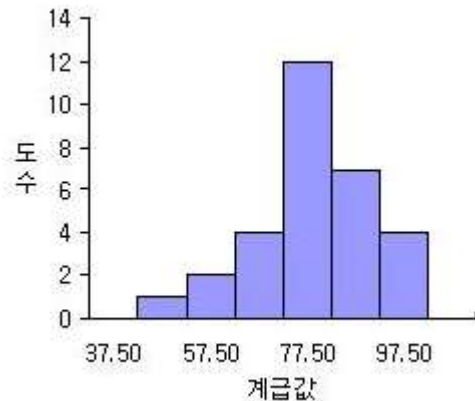
252

히스토그램 : 연속형 변수는 히스토그램을 이용하여 표현이 가능하다. 히스토그램이란, 연속된 변수의 X축을 일정 구간으로 나눠서, (5씩) 그 구간에 들어가는 데이터를 표현하는 방법으로, 키를 예를 들면 160~170, 170~180에 각각 몇 명이 있는지 그래프로 나타내면 히스토그램이라고 한다.

# 히스토그램 형태의 특징

253

## 도수분포표에 대한 히스토그램 그래프



		형태의 특징
일반형		도수의 중심부근이 가장 많고 좌우대칭형으로 산과 같은 형태의 분포
이빠진형		기둥이 하나(또는 몇 개) 간격으로 들쭉날쭉한 분포
우(좌)측 편침형		좌우 어느 한쪽으로 기슭이 보이고 불균형한 형태의 분포
절벽형		분포의 한쪽(또는 양쪽)이 끊어진 형태의 분포
고원형		중심부근의 몇 개 구간의 도수에 차이가 없고, 산의 정상이 평평한 분포
쌍봉우리형		중심부근의 도수가 적고 산 정상이 2개 나타나는 분포
낙도형		일부 데이터가 외떨어진 곳에 있는 분포

# 중심위치의 척도

# 산출평균: arithmetic mean

255

주어진 값들을 더해서 총 개수로 나누는 값

$$\mu = \frac{1}{n}(x_1 + x_2 + \dots + x_n)$$

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{1}{n}(x_1 + \dots + x_n)$$

# 기하평균: geometric mean

256

복수개의 수치의 곱을 수치의 개수로 제곱근을 취해서 산출되는 평균값

- 인구성장률, 투자이률 등 성장률 평균 산출시 사용

$$\text{Geometric mean} = \sqrt[n]{x_1 \times x_2 \times \cdots \times x_n}$$

$$G = \left( \prod_{i=1}^n a_i \right)^{1/n} = \sqrt[n]{a_1 a_2 \cdots a_n}$$



# 조화평균

257

수학에서 조화 평균(調和平均)은 주어진 수들의 역수의 산술 평균의 역수, 평균적인 변화율을 구할 때에 주로 사용된다.

실수  $a_1, \dots, a_n$ 이 주어졌을 때, 조화 평균  $H$ 는

$$H = \frac{n}{\frac{1}{a_1} + \frac{1}{a_2} + \dots + \frac{1}{a_n}}$$

# 중앙값: median

258

중앙에 위치하는 값은 어떤 주어진 값들을 크기의 순서대로 정렬했을 때 가장 중앙에 위치하는 값

n개의 값이 주어질 때

- $n$ 이 홀수이면  $(n + 1)/2$  번째 값
- $n$ 이 짝수이면  $n/2$  번째 값과  $n/2 + 1$  번째 값의 평균

# 최빈값: mode

259

가장 많이 관측되는 수는 주어진 값 중에서 가장 자주 나오는 값

{1,2,3,4,4,4,5} 주어질 때

- 최빈값은 4이다

# 평균 / 중간값 / 최빈값 : scipy

260

## Normal/uniform에 대한 평균 / 중간값 구하기

```
from scipy import stats

print(stats.norm.mean())
print(stats.norm.std())
print(stats.norm.var())
print(stats.norm.median())
```

```
# 평균과 분산
print(stats.norm.stats())
```

```
0.0
1.0
1.0
0.0
(array(0.0), array(1.0))
```

```
from scipy import stats

print(stats.uniform.mean())
print(stats.uniform.std())
print(stats.uniform.var())
print(stats.uniform.median())
```

```
# 평균과 분산
print(stats.uniform.stats())
```

```
0.5
0.288675134595
0.0833333333333
0.5
(array(0.5), array(0.08333333333333333))
```

# 평균 / 중간값 / 최빈값 : scipy

261

## Normal/uniform에 대한 평균 / 중간값 구하기

```
: import scipy as sp
   from scipy import stats
   s = sp.randn(100)

   n, min_max, mean, var, skew, kurt = stats.describe(s)
   print("Number of elements: {0:d}".format(n))
   print("Minimum: {0:8.6f} Maximum: {1:8.6f}".format(min_max[0], min_max[1]))
   print("Mean: {0:8.6f}".format(mean))
   print("Variance: {0:8.6f}".format(var))
   print("Skew : {0:8.6f}".format(skew))
   print("Kurtosis: {0:8.6f}".format(kurt))
```

```
Number of elements: 100
Minimum: -3.423643 Maximum: 2.099471
Mean: -0.077272
Variance: 1.137397
Skew : -0.352209
Kurtosis: -0.184145
```

# 평균 / 중간값 / 최빈값 : statistics

262

## 평균 / 중간값 / 최빈값 구하기

```
import statistics as sta
l = [1,2,3,3,4]
print(sta.mean(l))
print(sta.median(l))
print(sta.median_grouped(l))
print(sta.median_high(l))
print(sta.median_low(l))
print(sta.mode(l))
```

```
2.6
3
2.75
3
3
3
```

# 평균 / 중간값 / 최빈값 : numpy

263

평균 / 중간값 / 최빈값 구하기, scipy.stats 내의 mode 연산으로 최빈값을 찾아야 함

```
import numpy as np
from scipy.stats import mode
x = np.array([-2.1, -1, 1, 1, 4.3])
print(np.mean(x))
print(np.median(x))
print(mode(x))
```

0.64

1.0

ModeResult(mode=array([ 1.]), count=array([2]))

# 산포 / 산포의 척도



# 산포

265

산포는 데이터가 흩어진 정도를 나타내는 것  
분산,  
표준편차,  
범위,  
사분위수범위  
...

# 산포의 척도

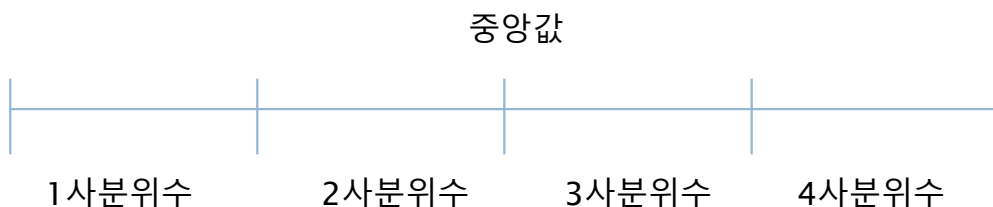
266

산포의 척도(Measure of dispersion) : 데이터가 흩어진 정도를 수치로서 측정하는 것으로 분산, 표준편차, 범위, 사분위범위 등이 많이 사용됨

# 사분위수

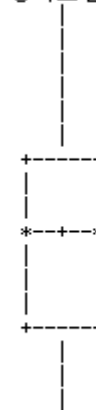
267

데이터 표본을 4개의 동일한 부분으로 나눈 값



사분위수	설명	비고
제1사분위수 (Q1)	누적 백분율이 25%에 해당하는 값	25번째 백분위수
제2사분위수 (Q2)	누적 백분율이 50%에 해당하는 값	50번째 백분위수, 중앙값과 같음
제3사분위수 (Q3)	누적 백분율이 75%에 해당하는 값	75번째 백분위수

상자그림



<- 최댓값

<- Q3

<- Q2=중앙값

<- Q1

<- 최솟값

그래프

# 범위

268

## 범위(range)

범위 (Range)	정의	데이터의 최대값에서 최소값을 뺀 차이. 계산은 간편하나 극단점이 있을 경우 올바른 산포의 측도가 되지 못함
	산식	범위=최대값-최소값

# 변동계수

269

변동계수(Coefficient of Variation)은 분포의 퍼짐의 정도를 비교하게 해준다

자료 단위가 다르고, 표준편차와 평균이 상당히 다른 두 분포가 있을 때 그 두분포의 산포를 타당하게 비교하기 위해 사용

$$v = \frac{s}{\bar{x}}, \quad CV = \frac{\text{표준편차}}{\text{평균}} \times 100$$

# 왜도

270

왜도(歪度; skewness)란, 데이터의 분포형태가 기울어진 정도를 의미한다.

분포의 형태는 좌우대칭이면 왜도는 0이 되고, 왜도가 +의 값을 가지면 오른쪽으로 긴 꼬리를 가지는 형태, -의 값을 가지면 왼쪽으로 긴 꼬리를 가지는 형태를 보인다.

첨도(尖度;kurtosis)란 분포가 평균치 주변에 몰려 있는 형태인지 멀리 퍼져 있는 형태인지 그 뾰족한 정도를 의미한다.

표준정규분포의 첨도계수는 0이며, 0보다 크면 표준정규분포에 비해 더 뾰족하게 몰려 있는 형태를 가지고, 0보다 작으면 보다 넓게 퍼져 있는 형태의 데이터라고 할 수 있다.

# 두개 이상의 연속 변수 정리 – 분산 / 표준편차



# 분산

273

평균을 중심으로 자료의 흩어진 정도가 어느 정도인지를 측정하는 것이며 평균에서 얼마나 벗어났는지를 나타내는 것이다.

모분산과 표준분산이 있다.

# 모분산 : population variance

274

분산은 편차(값 - 평균)을 제곱해서 n(전체 개수)로 나누어서 값을 구함. 전체집단이라 가정할 경우 계산

모분산(Population Variance)

모집단의 크기 :  $N$

모평균 :  $\mu$

모분산 :  $\sigma^2$

$$\begin{aligned} VARP &= \frac{(\text{데이터}_1 - \text{평균})^2 + (\text{데이터}_2 - \text{평균})^2 + \dots + (\text{데이터}_n - \text{평균})^2}{n} \\ &= \frac{\sum_{i=1}^n (\text{데이터}_i - \text{평균})^2}{n} \end{aligned}$$

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (X_i - \mu)^2$$

# 표본분산 : variance

275

분산은 편차(값 - 평균)을 제곱해서  $n-1$ (전체 개수)로 나누어서 값을 구함. 전체 중에 일부인 표본공간이라 가정할 경우 계산

표본분산(sample variance)

표본의 크기 :  $n$

표본평균 :  $\bar{X}$

표본분산 :  $s^2$

$$VAR = \frac{(\text{데이터}_1 - \text{평균})^2 + (\text{데이터}_2 - \text{평균})^2 + \dots + (\text{데이터}_n - \text{평균})^2}{n-1}$$

$$= \frac{\sum_{i=1}^n (\text{데이터}_i - \text{평균})^2}{n-1}$$

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2$$

# pvariance

276

## 평균에서 얼마나 떨어져 있는지 구하기

```
help(sta.pvariance)
```

Help on function pvariance in module statistics:

```
pvariance(data, mu=None)
```

Return the population variance of ``data``.

data should be an iterable of Real-valued numbers, with at least one value. The optional argument mu, if given, should be the mean of the data. If it is missing or None, the mean is automatically calculated.

Use this function to calculate the variance from the entire population. To estimate the variance from a sample, the ``variance`` function is usually a better choice.

Examples:

```
>>> data = [0.0, 0.25, 0.25, 1.25, 1.5, 1.75, 2.75, 3.25]
>>> pvariance(data)
1.25
```

# variance

277

## 평균에서 얼마나 떨어져 있는지 구하기

```
help(sta.variance)
```

Help on function variance in module statistics:

```
variance(data, xbar=None)
```

Return the sample variance of data.

data should be an iterable of Real-valued numbers, with at least two values. The optional argument xbar, if given, should be the mean of the data. If it is missing or None, the mean is automatically calculated.

Use this function when your data is a sample from a population. To calculate the variance from the entire population, see ``pvariance``.

Examples:

```
>>> data = [2.75, 1.75, 1.25, 0.25, 0.5, 1.25, 3.5]
>>> variance(data)
1.3720238095238095
```

# variance / pvariance

278

## 평균에서 얼마나 떨어져 있는지 구하기

```
import statistics as sta
l = [1,2,3,3,4]
mu = sta.mean(l)
print(mu)
ls = []
for i in l :
    ls.append(i - mu)

print(ls)
lp = map(lambda x: x**2,ls)
print(lp)
lsum = sum(lp)
print("pvariance ",lsum/len(l))
print("variance ", lsum/(len(l)-1))
print(sta.variance(l))
print(sta.pvariance(l))
```

```
2.6
[-1.6, -0.6000000000000001, 0.3999999999999999, 0.3999999999999999, 1.4]
[2.5600000000000005, 0.3600000000000001, 0.15999999999999992, 0.15999999999999992, 1.9599999999999997]
('pvariance ', 1.0400000000000003)
('variance ', 1.3000000000000003)
1.3
1.04
```

# variance

279

Numpy.var는 sta.pvariance와 동일한 처리

```
import numpy as np
import statistics as sta
x = np.array([-2.1, -1, 1,1, 4.3])
print(np.mean(x))
print(np.median(x))
print(mode(x))

x_m = np.mean(x)
x_a = x - x_m

x_p = np.power(x_a,2)
print(" variance x ")
print(np.var(x))
print(sta.pvariance(x))
print(sta.variance(x))

0.64
1.0
ModeResult(mode=array([ 1.]), count=array([2]))
variance x
4.7704
4.7704
5.963
```

# 편차

280

통계학에서 편차(deviation)는 관측값과 평균의 차이를 말한다. 편차점수라고도 한다.

어떤 변인  $y$ 에서 특정 사례의 편차  $d$ 를 다음과 같이 나타낼 수 있다.

편차는 양수일수도 있고 음수일 수 있으며, 이는 평균보다 크거나 작음을 나타낸다. 값의 크기는 관측값이 평균으로부터 얼마나 떨어져 있는가를 나타낸다. 편차는 오류 또는 잔차라고 할 수 있다. 모집단 평균에서의 편차는 오류이며, 표집 평균에서의 편차는 잔차이다..



# 편차의 특징

281

## 편차의 특징

1. 주어진 표본에서 편차를 모두 더하면 항상 0이 된다.

$$\Sigma(y - \bar{y}) = 0$$

2. 편차 D의 표준편차는 변인 Y의 표준편차와 같다.

$$s_d = s_y$$

$$\therefore s_d = \sqrt{\frac{\Sigma(d - \bar{d})^2}{n - 1}} = \sqrt{\frac{\Sigma(y - \bar{y})^2}{n - 1}} = s_y$$

# Standard deviation

282

variance에 대한 표준편차

---

$$\text{모 표준편차} \quad \sigma = \sqrt{\frac{\sum_{i=1}^N (x_i - \mu)^2}{N}}$$

$$\text{표본 표준편차} \quad s = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}}$$

※ N: 모집단개수,  $\mu$ : 모평균, n: 표본개수,  $\bar{x}$ : 표본평균

# stdev/pstdev

283

## 표본 표준편차/모 표준편차

```
import statistics as sta

l = [1,2,3,3,4]
mu = sta.mean(l)
print(mu)
ls = []
for i in l :
    ls.append(i - mu)

print(ls)
lp = map(lambda x: x**2,ls)
print(lp)
lsum = sum(lp)
print("pvariance ",lsum/len(l))
print("variance ", lsum/(len(l)-1))
print(sta.variance(l))
print(sta.stdev(l) , sta.math.sqrt(sta.variance(l)) )
print(sta.pvariance(l))
print(sta.pstdev(l),sta.math.sqrt(sta.pvariance(l)))

2.6
[-1.6, -0.6000000000000001, 0.3999999999999999, 0.3999999999999999, 1.4]
[2.5600000000000005, 0.3600000000000001, 0.15999999999999992, 0.15999999999999992, 1.9599999999999997]
('pvariance ', 1.0400000000000003)
('variance ', 1.3000000000000003)
1.3
(1.140175425099138, 1.140175425099138)
1.04
(1.019803902718557, 1.019803902718557)
```

# std

284

variance에 대한 표준편차이며  
statistics.pstdev와 동일한 결과

```
import numpy as np
import statistics as sta
x = np.array([-2.1, -1, 1,1, 4.3])
print(np.mean(x))
print(np.median(x))
print(mode(x))

x_m = np.mean(x)
x_a = x - x_m

x_p = np.power(x_a,2)
print(" variance x ")
print(np.var(x))
print(sta.pvariance(x))
print(sta.variance(x))

print(np.std(x))
print(sta.pstdev(x))
print(sta.stdev(x))

0.64
1.0
ModeResult(mode=array([ 1.]), count=array([2]))
 variance x
4.7704
4.7704
5.963
2.18412453857
2.18412453857
2.44192546979
```

# scipy 분산과 표준편차

285

## scipy normal/uniform 분포에 대한 분산과 표준편차

```
import scipy as sp
s = sp.randn(100) # Hundred random numbers from a standard Gaussian
print(len(s))

print("Mean : {0:8.6f}".format(s.mean()))
print("Minimum : {0:8.6f}".format(s.min()))
print("Maximum : {0:8.6f}".format(s.max()))
print("Variance : {0:8.6f}".format(s.var()))
print("Std. deviation : {0:8.6f}".format(s.std()))
print("Median : {0:8.6f}".format(sp.median(s)))
```

```
100
Mean : 0.145966
Minimum : -2.228590
Maximum : 2.637412
Variance : 0.990195
Std. deviation : 0.995086
Median : 0.098489
```

# scipy.stats 모듈 분산과 표준편차

286

scipy normal/uniform 분포에 대한 분산과 표준편차

```
from scipy import stats

print(stats.uniform.mean())
print(stats.uniform.std())
print(stats.uniform.var())

print(stats.norm.mean())
print(stats.norm.std())
print(stats.norm.var())

# 평균과 분산
print(stats.uniform.stats())
print(stats.norm.stats())
```

```
0.5
0.288675134595
0.0833333333333333
0.0
1.0
1.0
(array(0.5), array(0.08333333333333333))
(array(0.0), array(1.0))
```

# 두개 이상의 연속 변수 정리 – 상관관계

# 상관분석

288

- 상관관계는 서열 척도, 등간 척도, 비율 척도로 측정된 변수들간의 관련성 정도를 알아보기 위한 것
- 하나의 변수가 다른 변수와의 어느 정도 밀접한 관련성을 갖고 변화하는 가를 알아보기 위해 사용
- 두 변수간의 관련성을 구할 경우 단순상관관계를 실시하며, 부분 또는 편 상관관계는 어떤 변수를 통제된 상태에서 두 변수의 상관관계를 구하는 것



# 변화요인

289

변인의 분류	특성	보기
비율변인	절대영점을 갖고 있다 측정치는 비율 또는 퍼센트로 비교될 수 있다.	거리, 시간, 무게 등
등간변인	동간적이다. 측정시간의 거리가 비교될 수 있다.	연(year), 온도, IQ 등
서열변인	순위의 정보만 있다.	백분위수, 랭킹, 순위(학업성적) 등
명목변인	서로 다른 속성의 정보만 있다.	성별, 국적, 눈의 색깔, 출신학교 등

# 공분산(Covariance)

290

두 변량(확률변수 등) 사이에 상관성/의존성/유사성의 방향 및 정도에 대한 척도

공분산  
(covariance)

- 두 변량이 각각의 평균으로부터 변화하는 방향 및 양에 대한 기대값
- 공분산이 +면 한 변수 값이 증가할 때 다른 변수 값도 증가
- 두 변수의 단위에 의존하여 다른 데이터와 비교 시 불편  
→ 상관계수 사용 (표준화된 공분산)

$$\text{Population Covariance} = \text{Cov}(X, Y) = \sigma_{XY} = E[(X - \mu_X)(Y - \mu_Y)]$$

$$\text{Sample Covariance} = S_{XY} = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

# 공분산 산식

291

두변수의 평균의 차를 곱이 평균을 구하는 것이  
공분산 : 이산형과 연속형 산식

이산형

$$\text{Cov}(X,Y) = \sigma_{XY} = E[(X - \mu_X)(Y - \mu_Y)] = \sum_x \sum_y (x - \mu_X)(y - \mu_Y) p(x,y)$$

연속형

$$\text{Cov}(X,Y) = \sigma_{XY} = E[(X - \mu_X)(Y - \mu_Y)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \mu_X)(y - \mu_Y) f(x,y) dx dy$$

```
import numpy as np
x = np.array([10,2,3])
y = np.array([4,50,6])

cxy = np.cov(x,y,ddof=0)[0][1]
print(cxy)
sss = (x - avergA) * (y - avergB)
cosss = sssS/3
print(cosss)

-47.3333333333
-47.3333333333
```

# 공분산 성질

292

공분산은 교환법칙이 가능하고 동일변량이 공분산은 분산이 됨

- 교환법칙 :  $Cov(X, Y) = Cov(Y, X)$

- 동일 변량에 대한 공분산은 분산이 됨 :  $Cov(X, X) = Var(X)$

- 기타성질

$$Cov(aX + b, cY + d) = acCov(X, Y)$$

$$Cov(X, Y) = E(XY) - E(X)E(Y)$$

# 상관계수

293

상관계수  
(correlation  
coefficient)

- 상관계수는 표준화된 공분산
- 공분산은 각 변량의 단위에 의존하게 되어 변동 크기량이 모호하므로, 공분산을 각 변량의 표준편차로 나누어 표준화
- 양의 값이면 두 변수가 같은 방향으로 움직임 (음이면 반대)
- '0'이면 선형관계가 없음

$$\text{Population Correlation coefficient} = \rho = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y}$$

$$\text{Sample Correlation coefficient} = \gamma = \frac{s_{XY}}{s_X s_Y} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}}$$

# 상관계수 공식 구조

294

두 변수의 공분산을 두 변수 표준편차의 곱으로 나눔

$$r = \frac{\frac{\sum (x - \bar{x})(y - \bar{y})}{n - 1}}{\sqrt{\frac{\sum (x - \bar{x})^2}{n - 1} \times \frac{\sum (y - \bar{y})^2}{n - 1}}}$$

공분산

루트 빼면,  
변수 x의 분산

루트 빼면,  
변수 y의 분산

# 상관관계 정도

295

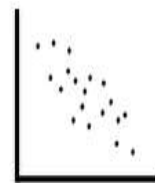
$-1 \leq r \leq 1$  사이의 값으로 상관관계 정도를 나타냄

상관계수		상관관계 정도
음(-)	양(+)	
-1	1	매우 강함
-0.9	0.9	
-0.8	0.8	강함
-0.7	0.7	
-0.6	0.6	상관관계가 있음
-0.5	0.5	
-0.4	0.4	
-0.3	0.3	약함
-0.2	0.2	
-0.1	0.1	매우 약함
0	0	



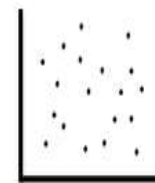
$r = -1$

음의 상관관계가  
강하다.



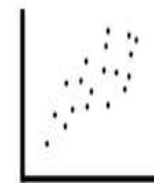
$-1 < r < 0$

음의 상관관계가  
있기는 하다.



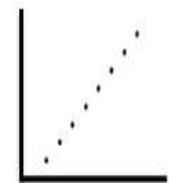
$r = 0$

상관관계가 없다.



$0 < r < 1$

양의 상관관계가  
있기는 하다.



$r = +1$

양의 상관관계가  
강하다.

# 상관계수 : numpy

296

두 변량간의 상관 관계를 나타내는 계수

python은 기본이  $n$ 으로 처리되므로  $(n-1)$ 로 처리하려면  $ddof = 1$  을 인수로 넣어야  $n-1$ 로 처리된다.

```
# 아이스크림 판매량
a = np.array([29,28,34,31,25,29,32,31,24,33,25,31,26,30])

# 아이스크림 판매량
b = np.array([77,62,93,84,59,64,80,75,58,91,51,73,65,84])

averg = np.ones(14)
avergA = averg * np.mean(a)
avergB = averg * np.mean(b)
sss = (a-avergA) * (b-avergB)
#각 변수들이 값을 합산하기
sssS = sum(sss)

ppp = np.std(a) * np.std(b)
cor = sssS/(14 * ppp)
print(cor)
print(np.corrcoef(a,b))

0.906922978051
[[ 1.          0.90692298]
 [ 0.90692298  1.         ]]
```