빅데이터 분석
-통계분석4-

강의자료 출처:

한밭대학교
임경태

CHAPTER

05

이산형 확률변수

CONTENTS

5.1.1 1차원 이산형 확률변수의 정의

- 확률변수 X가 취할 수 있는 값의 집합 $\{x_1, x_2, \cdots\}$
- X가 x_k라는 값을 취하는 확률

$$P(X = x_k) = p_k \quad (k = 1, 2, \cdots)$$

- 확률질량함수(확률함수)

$$f(x) = P(X = x)$$

5.1.1 1차원 이산형 확률변수의 정의

- 불공정한 주사위의 확률분포
  - 확률변수가 취할 수 있는 값의 집합 x_set

In [2] :

```
x_set = np.array([1, 2, 3, 4, 5, 6])
```

  - x_set에 대응하는 확률

[표 5-1] 불공정한 주사위의 확률분포

눈	1	2	3	4	5	6
확률	$\frac{1}{21}$	$\frac{2}{21}$	$\frac{3}{21}$	$\frac{4}{21}$	$\frac{5}{21}$	$\frac{6}{21}$

  - 불공정한 주사위의 확률변수

$$f(x) = \begin{cases} \frac{x}{21} & (x \in \{1, 2, 3, 4, 5, 6\}) \\ 0 & (otherwise) \end{cases}$$

$$p_1 = P(X = 1) = \frac{1}{21}$$

$$p_2 = P(X = 2) = \frac{2}{21}$$

$$\vdots$$

5.1.1 1차원 이산형 확률변수의 정의

- 파이썬으로 구현

$$\frac{1}{21} = 0.048,\ \frac{2}{21} = 0.095,\ \frac{3}{21} = 0.143,\ \cdots$$

In [3] :

```
def f(x):
    if x in x_set:
        return x / 21
    else:
        return 0
```

In [4] :

```
X = [x_set, f]
```

5.1.1 1차원 이산형 확률변수의 정의

$$\frac{1}{21} = 0.048,\ \frac{2}{21} = 0.095,\ \frac{3}{21} = 0.143,\ \cdots$$

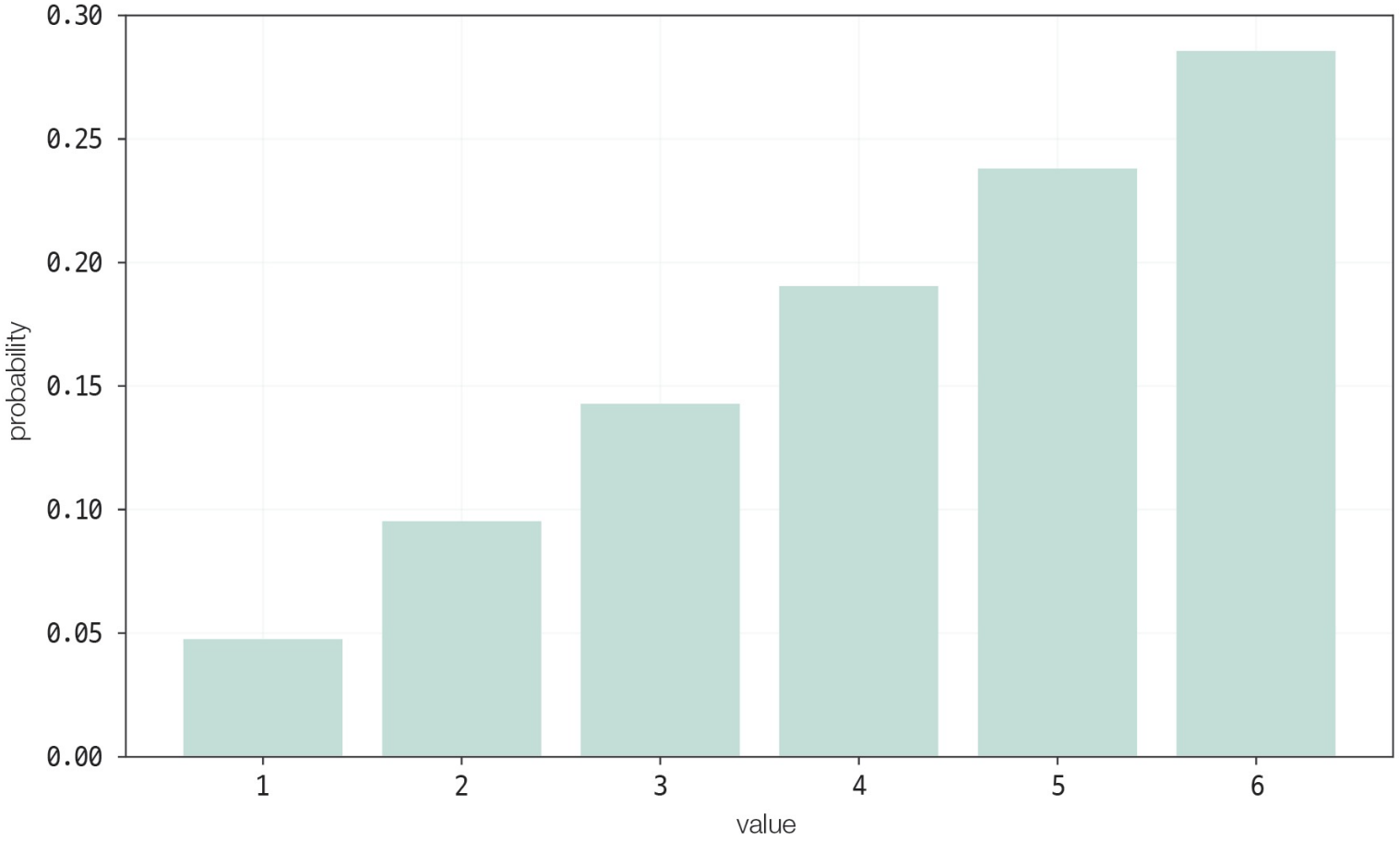
In [5] :

```
# 확률 p_k를 구한다
prob = np.array([f(x_k) for x_k in x_set])
# x_k와 p_k의 대응을 사전식으로 표시
dict(zip(x_set, prob))
```

Out [5] :

```
{1: 0.048, 2: 0.095, 3: 0.143, 4: 0.190, 5: 0.238, 6: 0.286}
```

5.1.1 1차원 이산형 확률변수의 정의



[그림 5-1] 확률분포

5.1.1 1차원 이산형 확률변수의 정의

- 확률의 성질

$$f(x_k) \geq 0$$

$$\sum_k f(x_k) = 1$$

In [7] :

```
np.all(prob >= 0)
```

Out [7] :

```
True
```

- np.all은 모든 요소가 참일 때만 참을 반환

- 확률의 총합은 1

In [8] :

```
np.sum(prob)
```

$$\frac{1}{21}+\frac{2}{21}+\frac{3}{21}+\frac{4}{21}+\frac{5}{21}+\frac{6}{21}=1$$

Out [8] :

```
1.000
```

5.1.1 1차원 이산형 확률변수의 정의

- 누적분포함수(분포함수) $F(x)$
  - X가 x 이하가 될 때의 확률을 반환하는 함수

$$F(x) = P(X \leq x) = \sum_{x_k \leq x} f(x_k)$$

In [9] :

```
def F(x):
    return np.sum([f(x_k) for x_k in x_set if x_k <= x])
```

- 눈이 3 이하가 되는 확률

In [10] :

```
F(3)
```

$$F(3) = P(X \leq 3) = \sum_{x_k \leq 3} f(x_k)$$

Out [10] :

```
0.286
```

$$\frac{1}{21} + \frac{2}{21} + \frac{3}{21} = 0.048 + 0.095 + 0.143 = 0.286$$

5.1.1 1차원 이산형 확률변수의 정의

- 확률변수의 변환
  - 확률변수 X에 2를 곱하고 3을 더한 $2X+3$도 확률변수
  - $2X+3$을 확률변수 Y라고 하면

- Y의 확률분포

In [11] :

```
y_set = np.array([2 * x_k + 3 for x_k in x_set])
prob = np.array([f(x_k) for x_k in x_set])
dict(zip(y_set, prob))
```

Out [11] :

```
{5: 0.048, 7: 0.095, 9: 0.143, 11: 0.190, 13: 0.238, 15: 0.286}
```

5.1.2 1차원 이산형 확률변수의 지표

- 기댓값 = 확률변수의 평균
  - 확률변수를 몇 번이나(무제한) 시행하여 얻어진 실현값의 평균
  - 무제한 시행할 수 없으므로 확률변수가 취할 수 있는 값과 확률의 곱의 총합

$$E(X) = \sum_k x_k f(x_k)$$

  - 불공정한 주사위의 기댓값

In [12] :

```
np.sum([x_k * f(x_k) for x_k in x_set])
```

$$1 \times 0.048 + 2 \times 0.095 + 3 \times 0.143 + 4 \times 0.190 + 5 \times 0.238 + 6 \times 0.286 = 4.333$$

5.1.2 1차원 이산형 확률변수의 지표

- 기댓값 = 확률변수의 평균
  - 주사위를 100만(10^6)번 굴린 실현값의 평균

In [13] :

```
sample = np.random.choice(x_set, int(1e6), p=prob)
np.mean(sample)
```

Out [13] :

```
4.333
```

  - 확률변수 X를 $2X+3$ 으로 변환한 Y의 기댓값

$$E(Y) = E(2X+3) = \sum_k (2x_k + 3) f(x_k)$$

5.1.2 참고 : 데이터 샘플링

이미 있는 데이터 집합에서 일부를 무작위로 선택하는 것을 샘플링(sampling)이라고 한다. 샘플링에는 `choice` 명령을 사용한다. `choice` 명령은 다음과 같은 인수를 가질 수 있다.

```
numpy.random.choice(a, size=None, replace=True, p=None)
```

- a : 배열이면 원래의 데이터, 정수이면 arange(a) 명령으로 데이터 생성
- size : 정수. 샘플 숫자
- replace : 불리언. True이면 한번 선택한 데이터를 다시 선택 가능
- p : 배열. 각 데이터가 선택될 수 있는 확률

In [11]:

```
np.random.choice(5, 5, replace=False)  # shuffle 명령과 같다.
```

Out:

```
array([1, 4, 0, 3, 2])
```

In [12]:

```
np.random.choice(5, 3, replace=False)  # 3개만 선택
```

Out:

```
array([2, 1, 3])
```

In [13]:

```
np.random.choice(5, 10)  # 반복해서 10개 선택
```

Out:

```
array([0, 4, 1, 4, 1, 2, 2, 0, 1, 1])
```

In [14]:

```
np.random.choice(5, 10, p=[0.1, 0, 0.3, 0.6, 0])  # 선택 확률을 다르게 해서 10개 선택
```

Out:

```
array([0, 3, 3, 2, 2, 3, 3, 2, 0, 3])
```

5.1.2 1차원 이산형 확률변수의 지표

- 기댓값 = 확률변수의 평균

이산형 확률변수의 기댓값

$$E(g(X)) = \sum_k g(x_k) f(x_k)$$

- 수식을 기댓값의 함수로 구현
- 인수 g가 확률변수에 대한 변환의 함수

In [14] :

```
def E(X, g=lambda x: x):
    x_set, f = X
    return np.sum([g(x_k) * f(x_k) for x_k in x_set])
```

- g에 아무것도 지정하지 않으면 확률변수 X의 기댓값이 구해짐

5.1.2 1차원 이산형 확률변수의 지표

- 기댓값 = 확률변수의 평균
  - 확률변수 $Y = 2X + 3$의 기댓값

$$(2\times1+3)\times0.048+(2\times2+3)\times0.095+\cdots(2\times6+3)\times0.286$$
$$=11.667$$

In [16] :

```
E(X, g=lambda x: 2*x + 3)
```

Out [16] :

```
11.667
```

참고 : 람다(lambda) 함수(익명 함수)

- 값을 반환하는 단순한 한 문장으로 이루어진 함수
- 코드를 적게 쓰고 더 간결해짐

```
def short_function(x):
    return x*2

equiv_anon = lambda x: x*2
```

```
def apply_to_list(some_list, f):
    return [(f(x) for x in some_list]

ints = [4, 0, 1, 5, 6]
apply_to_list(ints, lambda x: x*2)
```

[x*2 for x in ints]

참고 : 람다(lambda) 함수(익명 함수)

- 리스트의 sort 메서드에 람다 함수를 넘겨 정렬 가능

```
In [1]: strings = ['hyeja', 'parkhyeja', 'youngtae', 'kimyoungtae', 'bbangtae']

In [3]: strings.sort(key=lambda x: len(set(list(x))))

In [4]: strings
Out[4]: ['hyeja', 'bbangtae', 'parkhyeja', 'youngtae', 'kimyoungtae']
```

5.1.2 1차원 이산형 확률변수의 지표

- 기댓값 = 확률변수의 평균

기댓값의 선형성

a, b를 실수, X를 확률변수로 했을 때

$$E(aX+b) = aE(X) + b$$

가 성립합니다.

- $\mathrm{E}(2X+3) \equiv 2\mathrm{E}(X)+3$

In [17] :

```
2 * E(X) + 3
```

Out [17] :

```
11.667
```

5.1.2 1차원 이산형 확률변수의 지표

- 분산

$$V(X) = \sum_k (x_k - \mu)^2 f(x_k)$$
$$= (1-4.333)^2 \times 0.048 + (2-4.333)^2 \times 0.095 + \cdots + (6-4.333)^2 \times 0.286$$
$$= 2.222$$

- 불공정한 주사위의 분산

In [18] :

```
mean = E(X)
np.sum([(x_k-mean)**2 * f(x_k) for x_k in x_set])
```

Out [18] :

```
2.222
```

- 확률변수 $Y = 2X + 3$의 분산

$$V(2X+3) = \sum_k ((2x_k + 3) - \mu)^2 f(x_k)$$

5.1.2 1차원 이산형 확률변수의 지표

- 분산
  - 이산형 확률변수의 분산식을 분산의 함수로 구현

이산형 확률변수의 분산

$$V(g(X)) = \sum_k (g(x_k)) - E(g(X)))^2 f(x_k)$$

  - 인수 g가 확률변수에 대한 변환의 함수

In [19] :

```
def V(X, g=lambda x: x):
    x_set, f = X
    mean = E(X, g)
    return np.sum([(g(x_k)-mean)**2 * f(x_k) for x_k in x_set])
```

In [20] :

```
V(X)
```

Out [20] :

```
2.222
```

  - 확률변수 $Y = 2X + 3$의 분산

In [21] :

```
V(X, lambda x: 2*x + 3)
```

Out [21] :

```
8.889
```

5.1.2 1차원 이산형 확률변수의 지표

- 분산

분산의 공식

a, b를 실수, X를 확률변수라고 하면

$$V(aX+b)=a^2V(X)$$

가 성립합니다.

- $V(2X+3)=2^2V(X)$

In [22] :

```
2**2 * V(X)
```

Out [22] :

```
8.889
```

5.2.1 2차원 이산형 확률변수의 정의

1차원 확률분포 2개를 동시에 다룹니다$(X,\ Y)$

확률은 X와 Y가 각각 취할 수 있는 값의 조합에 관해서 정의

- 결합확률분포
  - 확률변수 X가 x_i, 확률변수 Y가 y_j를 취하는 확률

$$P(X = x_i,\ Y = y_j) = p_{ij} \qquad (i = 1, 2, \cdots ;\ j = 1, 2, \cdots)$$

  - 확률변수 (X, Y)의 움직임을 동시에 고려한 분포
  - 불공정한 주사위 A와 B
    - A와 B의 눈을 더한 것 X, A의 눈을 Y로 하는 2차원 확률분포

  - 결합확률함수
    - $P(X = x, Y = y) = f_{xy}(x, y)$

$$f_{XY}(x,\ y) = \begin{cases} \dfrac{y(x-y)}{441} \\ 0 \end{cases}$$

5.2.1 2차원 이산형 확률변수의 정의

[표 5-2] 불공정한 주사위의 결합확률분포

X \ Y	1	2	3	4	5	6
2	$\frac{1}{441}$	0	0	0	0	0
3	$\frac{2}{441}$	$\frac{2}{441}$	0	0	0	0
4	$\frac{3}{441}$	$\frac{4}{441}$	$\frac{3}{441}$	0	0	0
5	$\frac{4}{441}$	$\frac{6}{441}$	$\frac{6}{441}$	$\frac{4}{441}$	0	0
6	$\frac{5}{441}$	$\frac{8}{441}$	$\frac{9}{441}$	$\frac{8}{441}$	$\frac{5}{441}$	0
7	$\frac{6}{441}$	$\frac{10}{441}$	$\frac{12}{441}$	$\frac{12}{441}$	$\frac{10}{441}$	$\frac{6}{441}$
8	0	$\frac{12}{441}$	$\frac{15}{441}$	$\frac{16}{441}$	$\frac{15}{441}$	$\frac{12}{441}$
9	0	0	$\frac{18}{441}$	$\frac{20}{441}$	$\frac{20}{441}$	$\frac{18}{441}$
10	0	0	0	$\frac{24}{441}$	$\frac{25}{441}$	$\frac{24}{441}$
11	0	0	0	0	$\frac{30}{441}$	$\frac{30}{441}$
12	0	0	0	0	0	$\frac{36}{441}$

$$\frac{4}{21} \times \frac{5}{21} = \frac{20}{441}$$

5.2.1 2차원 이산형 확률변수의 정의

- 확률의 성질

$$f_{XY}(x_i, y_j) \geq 0$$

$$\sum_i \sum_j f_{XY}(x_i, y_j) = 1$$

$$f_{XY}(x_2, y_1) + f_{XY}(x_3, y_1) + f_{XY}(x_3 + y_2) + \cdots + f_{XY}(x_{12} + y_6)$$
$$= \frac{1}{441} + \frac{2}{441} + \frac{2}{441} + \cdots + \frac{36}{441}$$
$$= 1$$

5.2.1 2차원 이산형 확률변수의 정의

- 확률의 성질
  - X와 Y가 취할 수 있는 값의 집합

In [23] :

```
x_set = np.arange(2, 13)
y_set = np.arange(1, 7)
```

  - 결합확률함수

In [24] :

```
def f_XY(x, y):
    if 1 <= y <=6 and 1 <= x - y <= 6:
        return y * (x-y) / 441
    else:
        return 0
```

In [25] :

```
XY = [x_set, y_set, f_XY]
```

5.2.1 2차원 이산형 확률변수의 정의

- 확률의 성질
  - 확률분포의 히트맵

In [26] :

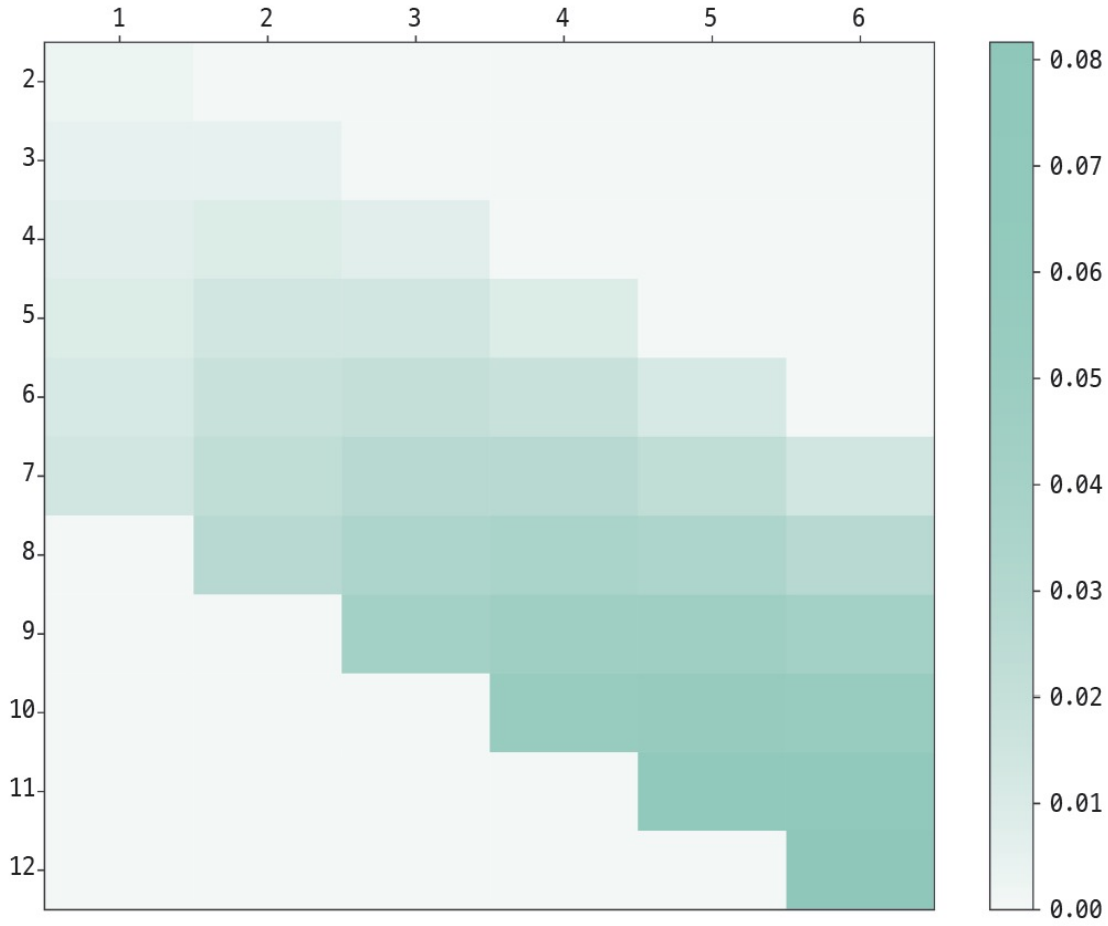
```
prob = np.array([[f_XY(x_i, y_j) for y_j in y_set]
                 for x_i in x_set])

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111)

c = ax.pcolor(prob)
ax.set_xticks(np.arange(prob.shape[1]) + 0.5, minor=False)
ax.set_yticks(np.arange(prob.shape[0]) + 0.5, minor=False)
ax.set_xticklabels(np.arange(1, 7), minor=False)
ax.set_yticklabels(np.arange(2, 13), minor=False)
# y축을 내림차순의 숫자가 되게 하여, 위 아래를 역전시킨다
ax.invert_yaxis()
# x축 눈금을 그래프 위쪽에 표시
ax.xaxis.tick_top()
fig.colorbar(c, ax=ax)
plt.show()
```

5.2.1 2차원 이산형 확률변수의 정의

- 확률의 성질
  - 확률분포의 히트맵



[그림 5-2] 2차원 확률분포의 히트맵

5.2.1 2차원 이산형 확률변수의 정의

- 확률의 성질

$$f_{XY}(x_i, y_j) \geq 0$$

$$\sum_i \sum_j f_{XY}(x_i, y_j) = 1$$

$$f_{XY}(x_2, y_1) + f_{XY}(x_3, y_1) + f_{XY}(x_3 + y_2) + \cdots + f_{XY}(x_{12} + y_6)$$
$$= \frac{1}{441} + \frac{2}{441} + \frac{2}{441} + \cdots + \frac{36}{441}$$
$$= 1$$

In [27] :

```
np.all(prob >= 0)
```

Out [27] :

```
True
```

In [28] :

```
np.sum(prob)
```

Out [28] :

```
1.000
```

5.2.1 2차원 이산형 확률변수의 정의

개별 확률변수에만 흥미

- 주변확률분포
  - 확률변수 (X, Y)는 결합확률분포에 의해 동시에 정의되지만, 확률변수 X의 확률함수 $f_X(x)$를 알고 싶을 때
  - f_{XY}에서 Y가 취할 수 있는 값 모두를 대입한 다음 모두 더한

$$f_X(x) = \sum_k f_{XY}(x, y_k)$$

결합확률함수 f_{XY}에서 확률변수 Y의 영향을 제거

5.2.1 2차원 이산형 확률변수의 정의

- 주변확률분포

In [29] :

```
def f_X(x):
    return np.sum([f_XY(x, y_k) for y_k in y_set])
```

In [30] :

```
def f_Y(y):
    return np.sum([f_XY(x_k, y) for x_k in x_set])
```

In [31] :

```
X = [x_set, f_X]
Y = [y_set, f_Y]
```

5.2.1 2차원 이산형 확률변수의 정의

In [32] :

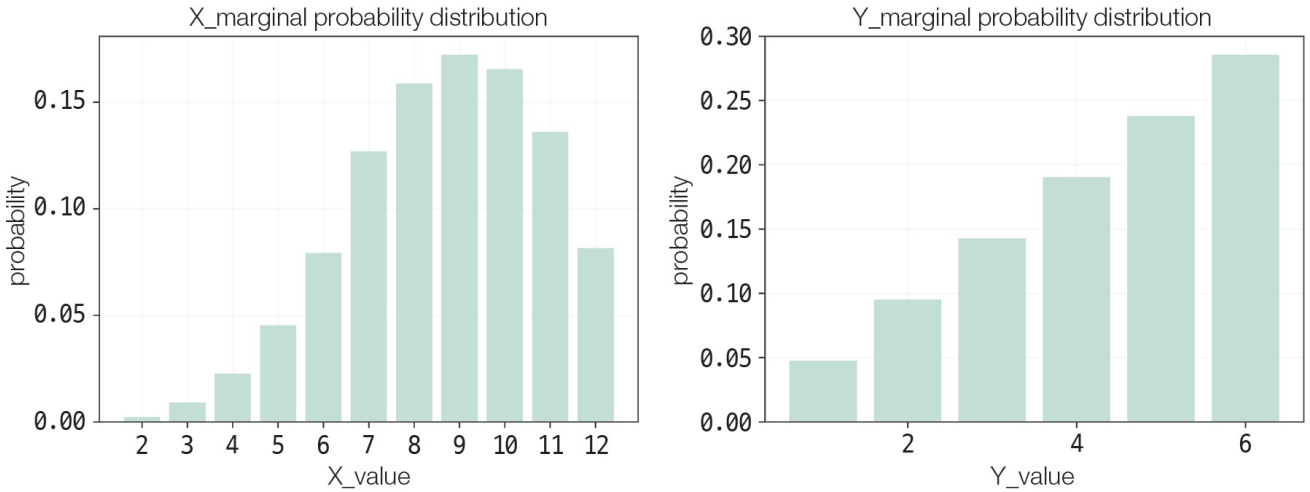
```
prob_x = np.array([f_X(x_k) for x_k in x_set])
prob_y = np.array([f_Y(y_k) for y_k in y_set])

fig = plt.figure(figsize=(12, 4))
ax1 = fig.add_subplot(121)
ax2 = fig.add_subplot(122)

ax1.bar(x_set, prob_x)
ax1.set_title('X_marginal probability distribution')
ax1.set_xlabel('X_value')
ax1.set_ylabel('probability')
ax1.set_xticks(x_set)

ax2.bar(y_set, prob_y)
ax2.set_title('Y_marginal probability distribution')
ax2.set_xlabel('Y_value')
ax2.set_ylabel('probability')

plt.show()
```



[그림 5-3] 주변분포

5.2.2 2차원 이산형 확률변수의 지표

- 기댓값

$$\mu_X = E(X) = \sum_i \sum_j x_i f_{XY}(x_i, y_j)$$

- 파이썬으로 구현

In [33] :

```
np.sum([x_i * f_XY(x_i, y_j) for x_i in x_set for y_j in y_set])
```

Out [33] :

```
8.667
```

$$E(g(X, Y) = \sum_i \sum_j g(x_i, y_j) f_{XY}(x_i, y_j)$$

5.2.2 2차원 이산형 확률변수의 지표

- 기댓값
  - 기댓값의 함수로 구현

$$2\times\frac{1}{441}+3\times\left(\frac{2}{441}+\frac{2}{441}\right)+\cdots+36\times\frac{36}{441}$$
$$=8.667$$

In [34] :

```
def E(XY, g):
    x_set, y_set, f_XY = XY
    return np.sum([g(x_i, y_j) * f_XY(x_i, y_j)
                   for x_i in x_set for y_j in y_set])
```

5.2.2 2차원 이산형 확률변수의 지표

- 기댓값
  - X와 Y의 기댓값

In [35] :

```
mean_X = E(XY, lambda x, y: x)
mean_X
```

Out [35] :

```
8.667
```

In [36] :

```
mean_Y = E(XY, lambda x, y: y)
mean_Y
```

Out [36] :

```
4.333
```

5.2.2 2차원 이산형 확률변수의 지표

기댓값의 선형성

a, b를 실수, X, Y를 확률변수로 했을 때

$$E(aX+bY)=aE(X)+bE(Y)$$

가 성립합니다.

In [37] :

```
a, b = 2, 3
```

In [38] :

```
E(XY, lambda x, y: a*x + b*y)
```

Out [38] :

```
30.333
```

In [39] :

```
a * mean_X + b * mean_Y
```

$$2\times 8.667+3\times 4.333=30.333$$

Out [39] :

```
30.333
```

5.2.2 2차원 이산형 확률변수의 지표

- 분산
  - X의 분산은 X에 관한 편차 제곱의 기댓값

$$\sigma_X^2 = V(X) = \sum_i \sum_j (x_i - \mu_X)^2 f_{XY}(x_i, y_j)$$

  - 파이썬으로 구현

In [40] :

```
np.sum([(x_i-mean_X)**2 * f_XY(x_i, y_j)
        for x_i in x_set for y_j in y_set])
```

Out [40] :

```
4.444
```

5.2.2 2차원 이산형 확률변수의 지표

- 분산
  - X와 Y의 함수 g(X, Y)의 분산

$$V(g(X,\ Y)) = \sum_i \sum_j (g(x_i,\ y_j) - E(g(X,\ Y)))^2 f_{XY}(x_i,\ y_j)$$

  - 함수 구현

In [41] :

```
def V(XY, g):
    x_set, y_set, f_XY = XY
    mean = E(XY, g)
    return np.sum([(g(x_i, y_j)-mean)**2 * f_XY(x_i, y_j)
                   for x_i in x_set for y_j in y_set])
```

5.2.2 2차원 이산형 확률변수의 지표

- 분산
  - X와 Y의 분산

In [42] :

```
var_X = V(XY, g=lambda x, y: x)
var_X
```

Out [42] :

```
4.444
```

In [43] :

```
var_Y = V(XY, g=lambda x, y: y)
var_Y
```

Out [43] :

```
2.222
```

5.2.2 2차원 이산형 확률변수의 지표

- 공분산
  - 두 확률변수 X, Y 사이의 상관

$$\sigma_{XY} = Cov(X,\ Y) = \sum_i \sum_j (x_i - \mu_X)(y_j - \mu_Y) f_{XY}(x_i,\ y_j)$$

In [44] :

```
def Cov(XY):
    x_set, y_set, f_XY = XY
    mean_X = E(XY, lambda x, y: x)
    mean_Y = E(XY, lambda x, y: y)
    return np.sum([(x_i-mean_X) * (y_j-mean_Y) * f_XY(x_i, y_j)
                   for x_i in x_set for y_j in y_set])
```

In [45] :

```
cov_xy = Cov(XY)
cov_xy
```

Out [45] :

```
2.222
```

5.2.2 2차원 이산형 확률변수의 지표

분산과 공분산의 공식

a, b를 실수, X, Y를 확률변수로 했을 때

$$V(aX+bY)=a^2V(X)+b^2V(Y)+2abCov(X,\ Y)$$

가 성립합니다.

$$V(2X+3Y)=4V(X)+9V(Y)+12Cov(X,\ Y)$$

In [46] :

```
V(XY, lambda x, y: a*x + b*y)
```

Out [46] :

```
64.444
```

In [47] :

```
a**2 * var_X + b**2 * var_Y + 2*a*b * cov_xy
```

Out [47] :

```
64.444
```

5.2.2 2차원 이산형 확률변수의 지표

- 상관계수

$$\rho_{XY} = \rho(X,\ Y) = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}$$

In [48] :

```
cov_xy / np.sqrt(var_X * var_Y)
```

Out [48] :

```
0.707
```

Q&A