Py 이론적 확률분포 1

I. 포아송분포

II. 이항분포

1. 확률함수

- 단위구간 내에서 어떤 사건이 평균 μ 회 발생하고, 확률변수 X를 사건의 발생횟수라고 할 때
  X ~ Poisson (μ)
- 확률질량함수는 다음과 같음

$$P_r(x=k)=\frac{\mu^k}{k!}e^{-\mu},\quad 0,1,2,\cdots$$

- 포아송분포의 모수는 평균(μ)
- 포아송분포의 모양은 평균이 작을 때는
  좌우비대칭이나
  평균이 증가함에 따라 평균을 중심으로
  좌우대칭의 모양으로 변함.
  즉, 정규분포에 가까워지고, 분산은 커짐

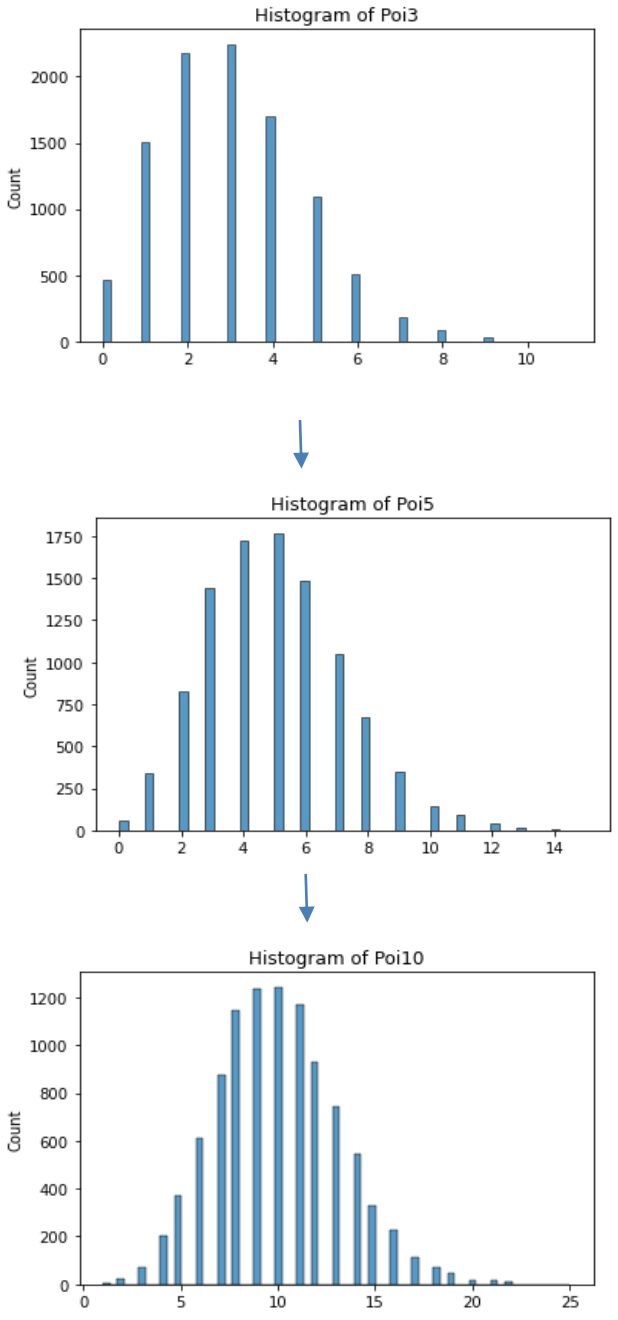
b1-ch4-5-r.py

```
import numpy as np
#import scipy.stats as stats
import seaborn as sns # seaborn package를 이용
import matplotlib.pyplot as plt # matplotlib.pyplot package를 이용
from numpy import random
# set the random seed:
np.random.seed(12345)
r=10000
poi1 = random.poisson(3, size=r)
sns.histplot(data=poi1, x=None).set(title='Histogram of Poi3')
plt.show()
poi2 = random.poisson(5.0, size=r)
sns.histplot(data=poi2, x=None).set(title='Histogram of Poi5')
plt.show()
poi3 = random.poisson(10.0, size=r)
sns.histplot(data=poi3, x=None).set(title='Histogram of Poi10')
plt.show()
poi4 = random.poisson(16, size=r)
sns.histplot(data=poi4, x=None).set(title='Histogram of Poi16')
plt.show()
```

b1-ch4-5-r.py

(앞에서 계속)

```
fig = plt.figure(figsize=(14,7))
# 두 개의 그래프를 한 페이지에 그림
fig, axs = plt.subplots(ncols=2)
fig, ays = plt.subplots(ncols=2)

sns.histplot(data=poi1, x=None, ax=axs[0]).set(title='Histogram of Poisson(mu=3.0 & mu=5.0)')
sns.histplot(data=poi2, x=None, ax=axs[1])
fig.subplots_adjust(wspace=0.5) # 우측그림의 좌우 간격을 조정
#plt.savefig('C:/BOOK/PyBasics/PyStat/code/poison-1.png')

sns.histplot(data=poi3, x=None, ax=ays[0]).set(title='Histogram of Poisson(mu=10.0 & mu=16.0)')
sns.histplot(data=poi4, x=None, ax=ays[1])
fig.subplots_adjust(wspace=0.5) # 우측그림의 좌우 간격을 조정
#plt.savefig('C:/BOOK/PyBasics/PyStat/code/poison-2.png')
```



b1-ch4-6.py

```
import scipy.stats as stats
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# values for x (all between 0 and 30):
x = np.linspace(0, 30, num=31)
# PMF for all these values:
fx_3 = stats.poisson.pmf(x, 3)
fx_5 = stats.poisson.pmf(x, 5)
fx_10 = stats.poisson.pmf(x, 10)
fx_15 = stats.poisson.pmf(x, 15)
# collect values in DataFrame:
result_3 = pd.DataFrame({'x': x, 'fx_3': fx_3})
result_5 = pd.DataFrame({'x': x, 'fx_5': fx_5})
result_10 = pd.DataFrame({'x': x, 'fx_10': fx_10})
result_15 = pd.DataFrame({'x': x, 'fx_15': fx_15})
#print(f'result_3: \n{result_3}\n')
#print(f'result_5: \n{result_5}\n')
#print(f'result_10: \n{result_10}\n')
#print(f'result_15: \n{result_15}\n')
"""
# plot:
plt.bar(x, fx_3, color='0.6', width = 0.1)
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.show()
```

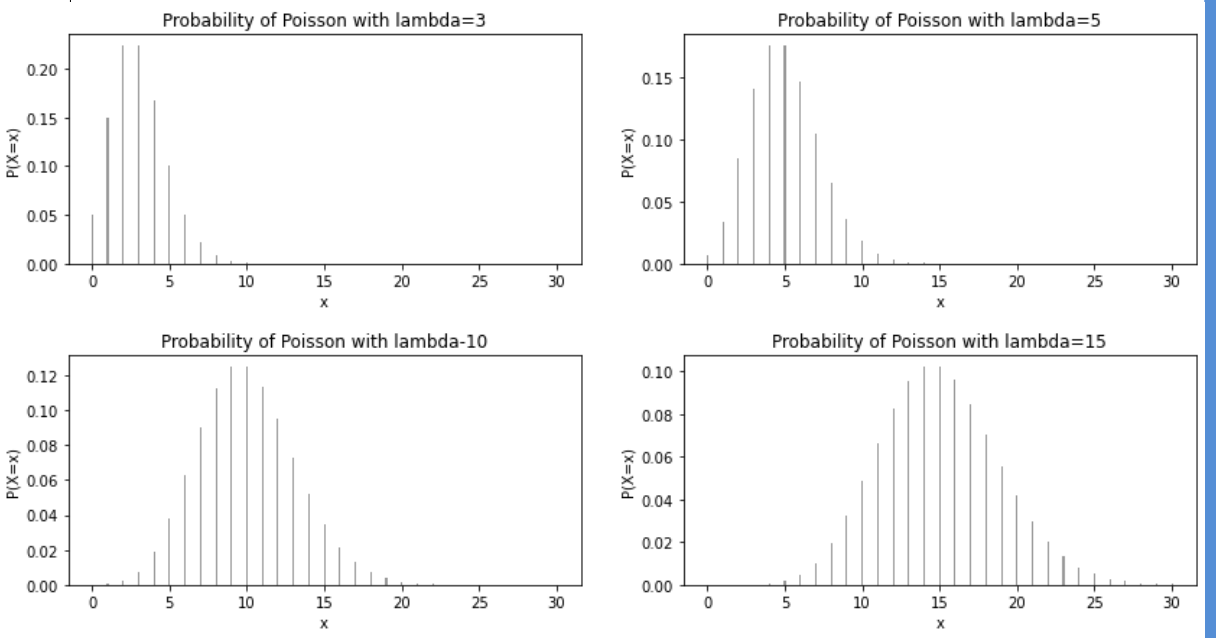
b1-ch4-6.py

```
(앞에서 계속)
Plt.bar(x, fx_5, color= ' 0.6 ' , width = 0.1)
Plt.티뮤디( ' x ' )
Plt.ylabel( ' P(X=x) ' )
Plt.show()
plt.bar(x, fx_10, color='0.6', width = 0.1)
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.show()
plt.bar(x, fx_15, color='0.6', width = 0.1)
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.show()
"""
# create graph:
fig = plt.figure(figsize=(14,7))
plt.subplot(221).set(title='Probability of Poisson with lambda=3')
_ = plt.bar(x, fx_3, color='0.6', width = 0.1)
#_ = plt.stem(x, fx_3, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
plt.subplot(222).set(title='Probability of Poisson with lambda=5')
_ = plt.bar(x, fx_5, color='0.6', width = 0.1)
#_ = plt.stem(x, fx_5, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
```

b1-ch4-6.py

(앞에서 계속)

```
plt.subplot(223).set(title='Probability of Poisson with lambda-10')
_ = plt.bar(x, fx_10, color='0.6', width = 0.1)
#_ = plt.stem(x, fx_10, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
plt.subplot(224).set(title='Probability of Poisson with lambda=15')
_ = plt.bar(x, fx_15, color='0.6', width = 0.1)
#_ = plt.stem(x, fx_15, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
fig.subplots_adjust(hspace=0.4, wspace=0.2) # 하측그림의 상하 및 우측그림의 좌우 간격을 조정
plt.show()
#plt.savefig('D:/BOOK/PyBasics/PyStat/code/b1-ch4-6.png')
```



2. 확률분포표

b1-ch4-7.py

```
import pandas as pd
import numpy as np
from scipy.stats import poisson
r = 5
poisson_02 = np.empty(r)
poisson_04 = np.empty(r)
poisson_06 = np.empty(r)
poisson_08 = np.empty(r)
poisson_1 = np.empty(r)
poisson_2 = np.empty(r)
poisson_3 = np.empty(r)
poisson_4 = np.empty(r)
# repeat r times:
for j in  range(r):
  poisson_02[j] = poisson.cdf(k=j, mu=0.02)
  j=j+1
for j in  range(r):
  poisson_04[j] = poisson.cdf(k=j, mu=0.04)
  j=j+1
for j in  range(r):
  poisson_06[j] = poisson.cdf(k=j, mu=0.06)
  j=j+1
for j in  range(r):
  poisson_08[j] = poisson.cdf(k=j, mu=0.08)
  j=j+1
```

2. 확률분포표

b1-ch4-7.py

(앞에서 계속)

```
for j in  range(r):
  poisson_1[j] = poisson.cdf(k=j, mu=0.1)
  j=j+1
for j in  range(r):
  poisson_2[j] = poisson.cdf(k=j, mu=0.2)
  j=j+1
for j in  range(r):
  poisson_3[j] = poisson.cdf(k=j, mu=0.3)
  j=j+1
for j in  range(r):
  poisson_4[j] = poisson.cdf(k=j, mu=0.4)
  j=j+1
poisson_mu = pd.DataFrame({'0.02':poisson_02,'0.04':poisson_04, '0.06':poisson_06,'0.08':poisson_08, '0.1':poisson_1,'0.2':poisson_2,'0.3':poisson_4,'0.4':poisson_4})
round(poisson_mu, 3)
print("Probability Distribution Table of Poisson with 'mu=0.02, ..., 0.4' : ", f'\n{round(poisson_mu, 3)}\n')
```

c \ μ	0.02	0.04	0.06	0.08	0.10	0.20	0.30	0.40
0	0.980	0.961	0.942	0.923	0.905	0.819	0.741	0.670
1	1.000	0.999	0.998	0.997	0.995	0.982	0.963	0.938
2	1.000	1.000	1.000	1.000	1.000	0.999	0.996	0.992
3	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.999
4	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

```
Probability Distribution Table of Poisson with 'mu=0.02, ..., 0.4' :
   0.02   0.04   0.06   0.08    0.1    0.2    0.3    0.4
0  0.98  0.961  0.942  0.923  0.905  0.819  0.670  0.670
1  1.00  0.999  0.998  0.997  0.995  0.982  0.938  0.938
2  1.00  1.000  1.000  1.000  1.000  0.999  0.992  0.992
3  1.00  1.000  1.000  1.000  1.000  1.000  0.999  0.999
4  1.00  1.000  1.000  1.000  1.000  1.000  1.000  1.000
```

Ⅱ. 이항분포

1. 확률함수

- 성공의 확률이 p이고 실패의 확률이 q(q = 1–p)인 베르누이 시행을 독립적으로 n번 반복하였을 때 나타나는 성공의 횟수를 확률변수 X 라고 할 때

  X~B(n,p)

- 확률질량함수는 다음과 같음

$$P_r(X = k) = \binom{n}{k} P^k q^{n-k}, \quad k = 0, 1, 2, \cdots, n$$

- 이항분포의 모수는 각 시행에서의 성공확률 p와 시행횟수 n
- 이항분포는 n과 p의 크기에 따라 모양이 결정되는데 p가 0.5이면 평균($\mu = np = \frac{n}{2}$)을 중심으로 좌우 대칭인 분포를 가지며, p=0.5이고 n이 커질 때 좌우대칭의 정규분포와 유사함

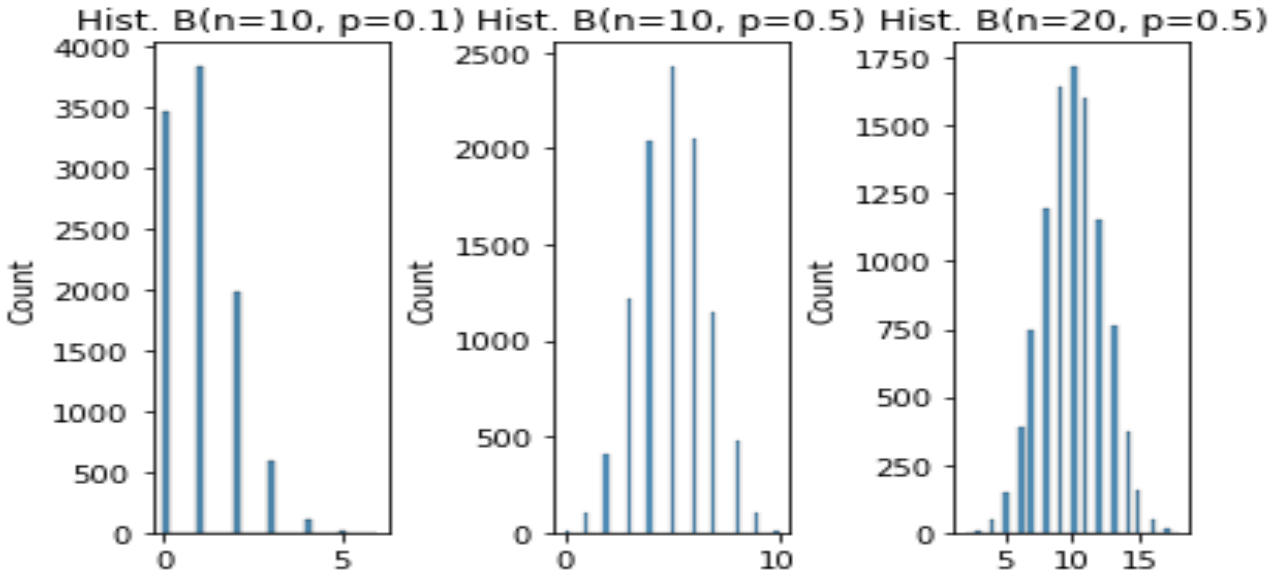
b1-ch4-2.py

```
import numpy as np
import seaborn as sns # seaborn package를 이용
import matplotlib.pyplot as plt # matplotlib.pyplot package를 이용
from numpy import random
# set the random seed:
np.random.seed(12345)
r=10000
binom1 = random.binomial(n=10, p=0.1, size=r)
sns.histplot(data=binom1, x=None).set(title='Histogram of Binom1')
plt.show()
binom2 = random.binomial(n=10, p=0.5, size=r)
sns.histplot(data=binom2, x=None).set(title='Histogram of Binom2')
plt.show()
binom3 = random.binomial(n=20, p=0.5, size=r)
sns.histplot(data=binom3, x=None).set(title='Histogram of Binom3')
plt.show()
```

b1-ch4-2.py

(앞에서 계속)

```
#fig = plt.figure(figsize=(10,7))
#figure, axs = plt.subplots(1,3) # 세 개의 그래프를 한 페이지에 그림
fig, axs = plt.subplots(ncols=3)


sns.histplot(data=binom1, x=None, ax=axs[0]).set(title='Hist. B(n=10, p=0.1)')
sns.histplot(data=binom2, x=None, ax=axs[1]).set(title='Hist. B(n=10, p=0.5)')
sns.histplot(data=binom3, x=None, ax=axs[2]).set(title='Hist. B(n=20, p=0.5)')


fig.subplots_adjust(wspace=0.7) # 우측그림의 좌우 간격을 조정
#plt.savefig('C:/BOOK/PyBasics/PyStat/code/binom.png')
```



b1-ch4-3.py

```
import scipy.stats as stats
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# values for x (all between 0 and 25):
x = np.linspace(0, 25, num=26)
# PMF for all these values:
fx_01 = stats.binom.pmf(x, 25, 0.1)
fx_02 = stats.binom.pmf(x, 25, 0.2)
fx_05 = stats.binom.pmf(x, 25, 0.5)
fx_07 = stats.binom.pmf(x, 25, 0.7)
# collect values in DataFrame:
result_01 = pd.DataFrame({'x': x, 'fx_01': fx_01})
result_02 = pd.DataFrame({'x': x, 'fx_02': fx_02})
result_05 = pd.DataFrame({'x': x, 'fx_05': fx_05})
result_07 = pd.DataFrame({'x': x, 'fx_07': fx_07})
"""
# plot:
plt.bar(x, fx_01, color='0.6', width = 0.1)
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.show()
plt.bar(x, fx_02, color='0.6', width = 0.1)
plt.xlabel('x')
plt.ylabel('P(X=x)')
plt.show()
```

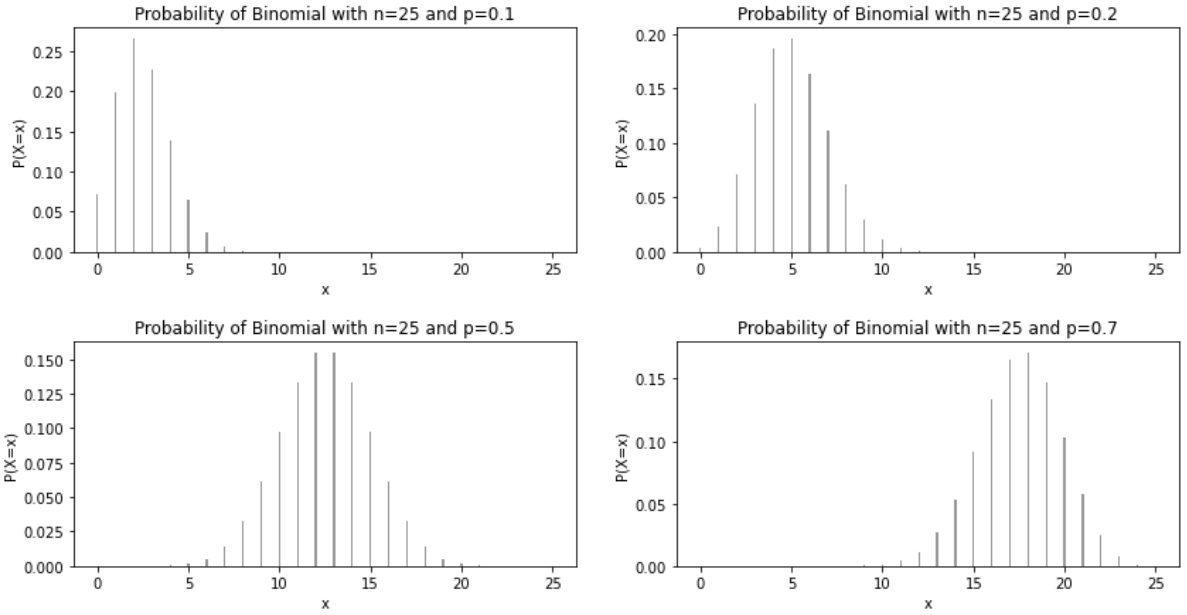
b1-ch4-3.py

```
(앞에서 계속)
Plt.bar(x, fx_05, color= ' 0.6 ' , width = 0.1)
Plt.xlabel( ' x ' )
Plt.ylabel( ' P(X=x) ' )
Plt.show()


Plt.bar(x, fx_07, color= ' 0.6 ' , width = 0.1)
Plt.xlabel( ' x ' )
Plt.ylabel( ' P(X=x) ' )
Plt.show()
 " " "

# create graph:
Fig = plt.figure(figsize=(14,7))
plt.subplot(221).set(title='Probability of Binomial with n=25 and p=0.1')
_ = plt.bar(x, fx_01, color='0.6', width = 0.1)
#_ = plt.stem(x, fx_01, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
plt.subplot(222).set(title='Probability of Binomial with n=25 and p=0.2')
_ = plt.bar(x, fx_02, color='0.6', width = 0.1)
#3_ = plt.stem(x, fx_02, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
```

b1-ch4-3.py

(앞에서 계속)

```
plt.subplot(223).set(title='Probability of Binomial with n=25 and p=0.5')
_ = plt.bar(x, fx_05, color='0.6', width = 0.1)
#3_ = plt.stem(x, fx_05, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
plt.subplot(224).set(title='Probability of Binomial with n=25 and p=0.7')
_ = plt.bar(x, fx_07, color='0.6', width = 0.1)
#_ = plt.stem(x, fx_07, use_line_collection = True)
_ = plt.xlabel('x')
_ = plt.ylabel('P(X=x)')
fig.subplots_adjust(hspace=0.4, wspace=0.2) # 하측그림의 상하 및 우측그림의 좌우 간격을 조정
#plt.savefig('C:/BOOK/PyBasics/PyStat/code/binom_pmf.png')
```



2. 확률분포표

b1-ch4-4.py

```
import pandas as pd
import numpy as np
#import scipy.stats as stats
from scipy.stats import binom
r = 5
binom_01 = np.empty(r)
binom_02 = np.empty(r)
binom_03 = np.empty(r)
binom_04 = np.empty(r)
binom_05 = np.empty(r)
binom_06 = np.empty(r)
binom_07 = np.empty(r)
binom_08 = np.empty(r)
binom_09 = np.empty(r)
# repeat r times:
for j in  range(r):
  binom_01[j] = binom.cdf(k=j, n=5, p=0.1)
  j=j+1
for j in  range(r):
  binom_02[j] = binom.cdf(k=j, n=5, p=0.2)
  j=j+1
for j in  range(r):
  binom_03[j] = binom.cdf(k=j, n=5, p=0.3)
  j=j+1
```

b1-ch4-4.py

(앞에서 계속)

```
For j in  range®:
  binom_04[j] = binom.cdf(k=j, n=5, p=0.4)
  j=j+1
for j in  range(r):
  binom_05[j] = binom.cdf(k=j, n=5, p=0.5)
  j=j+1
for j in  range(r):
  binom_05[j] = binom.cdf(k=j, n=5, p=0.6)
  j=j+1
for j in  range(r):
  binom_07[j] = binom.cdf(k=j, n=5, p=0.7)
  j=j+1
for j in  range(r):
  binom_08[j] = binom.cdf(k=j, n=5, p=0.8)
  j=j+1
for j in  range(r):
  binom_09[j] = binom.cdf(k=j, n=5, p=0.9)
  j=j+1
for j in  range(r):
  binom_095[j] = binom.cdf(k=j, n=5, p=0.95)
  j=j+1
binom_n5 = pd.DataFrame({'0.10':binom_01,'0.20':binom_02,'0.30':binom_03,'0.40':binom_04,'0.50':binom_05,'0.60':binom_07,'0.70':binom_07,'0.80':binom_08,'0
.90':binom_09,'0.95':binom_095 })
round(binom_n5,3)
print("Probability Distribution Table of Binomial with 'n=5 and p=0.10, ..., 0.95' : ", f'\n{round(binom_n5,3)}\n')
```

n=5

a	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	0.95
0	.590	.328	.168	.078	.031	.010	.002	.000	.000	.000
1	.919	.737	.528	.337	.187	.087	.031	.007	.000	.000
2	.991	.942	.837	.683	.500	.317	.163	.058	.009	.001
3	1.00	.993	.969	.913	.813	.683	.472	.263	.081	.023
4	1.00	1.00	.998	.990	.969	.922	.832	.672	.410	.226

Probability Distribution Table of Binomial with 'n=5 and p=0.10, ..., 0.95' :

	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	0.95
0	0.590	0.328	0.168	0.078	0.010	0.002	0.002	0.000	0.000	0.000
1	0.919	0.737	0.528	0.337	0.087	0.031	0.031	0.007	0.000	0.000
2	0.991	0.942	0.837	0.683	0.317	0.163	0.163	0.058	0.009	0.001
3	1.000	0.993	0.969	0.913	0.663	0.472	0.472	0.263	0.081	0.023
4	1.000	1.000	0.998	0.990	0.922	0.832	0.832	0.672	0.410	0.226