1 빅데이터 기반 AI 응용 솔루션 개발자 전문과정

1.1 교과목명 : 통계

- 평가일 : 22.09.08
- 성명 : 강유한
- 점수 :

Q1. df에서 mathematics 점수의 평균값, 중앙값, 최빈값, 분산, 표준편차, 범위, IQR을 구하세

In [322]:

```
import numpy as np
import pandas as pd
df = pd.read_csv('ch2_scores_em.csv',
                 index_col='student number')
df.head()
```

student number	english	mathematics
1	42	65
2	69	80
3	56	63
4	41	63
5	57	76

In [323]:

```
# 평균값
print(f'평균값: {df.mathematics.mean()}')
# 중앙값
print(f'중앙값: {np.median(df.mathematics)}','Wn')
# 최빈값
print(df.mathematics.mode(), 'Wn')
# 분산
mean = np.mean(df.mathematics)
deviation = df.mathematics - mean # deviation : 편차
print(f'분산 : {np.mean(deviation ** 2)}')
# 표준편차
print(f'표준편차 : {np.std(df.mathematics, ddof=0)}')
# 범위
print(f'범위 : {np.max(df.mathematics) - np.min(df.mathematics)}')
# IQR
df.mathematics_Q1 = np.percentile(df.mathematics, 25)
df.mathematics_Q3 = np.percentile(df.mathematics, 75)
df.mathematics_IQR = df.mathematics_Q3 - df.mathematics_Q1
print(f'IQR : {df.mathematics_IQR}')
```

```
평균값: 78.88
중앙값: 80.0

0    77
1    82
2    84
Name: mathematics, dtype: int64

분산 : 69.38559999999995
표준편차 : 8.329801918413184
범위 : 37
IQR : 8.0
```

Q2. df.english를 표준화한 후 배열로 변환하여 처음 5개 원소를 출력하세요.

In [324]:

```
score = df.english
type(score)
```

```
pandas.core.series.Series
```

In [325]:

```
# 표준화
z = (score - np.mean(score)) / np.std(score)
# 배열로 변환 후 처음 5개 원소 출력
np.array(z)[:5]
```

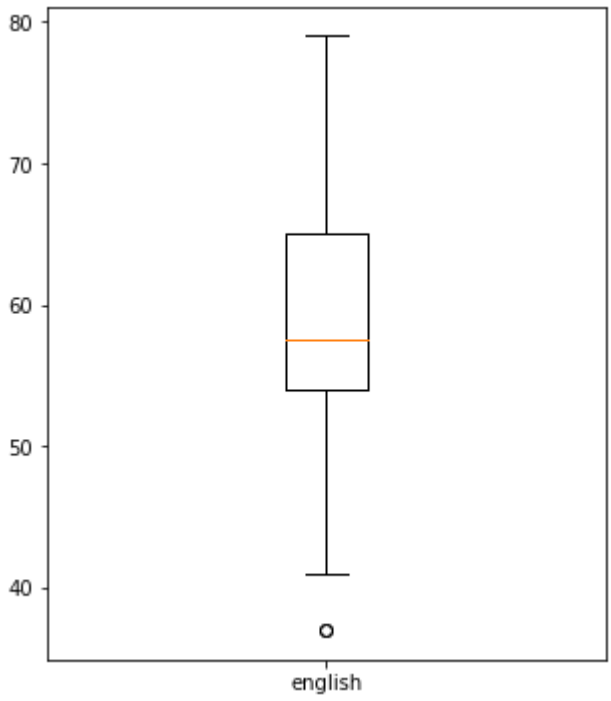
```
array([-1.688,  1.095, -0.245, -1.792, -0.142])
```

Q3. score에 대하여 다음사항을 수행하세요.

- 상자그림으로 시각화하여 이상치 여부를 탐색
- 이상치 값 및 인덱스 출력
- 이상치 삭제
- 상자그림으로 시각화하여 이상치 제거 여부 재확인.

In [326]:

```
import matplotlib.pyplot as plt

fig = plt.figure(figsize=(5,6))
ax = fig.add_subplot(111)
ax.boxplot(score, labels=['english'])

plt.show() # 아랫쪽에 이상치 확인

# 이상치 값 및 인덱스 출력
score_Q1 = np.percentile(score, 25)
score_Q3 = np.percentile(score, 75)
score_IQR = score_Q3 - score_Q1
print(score_IQR) # IQR : 11
print(score_Q1 - (1.5 * score_IQR)) # 37.5 이하는 이상치

# 이상치 값 및 인덱스 출력
df[df['english'] < 37.5]
```




```
11.0
37.5
```

student number	english	mathematics

student number	english	mathematics
20	37	70
35	37	57

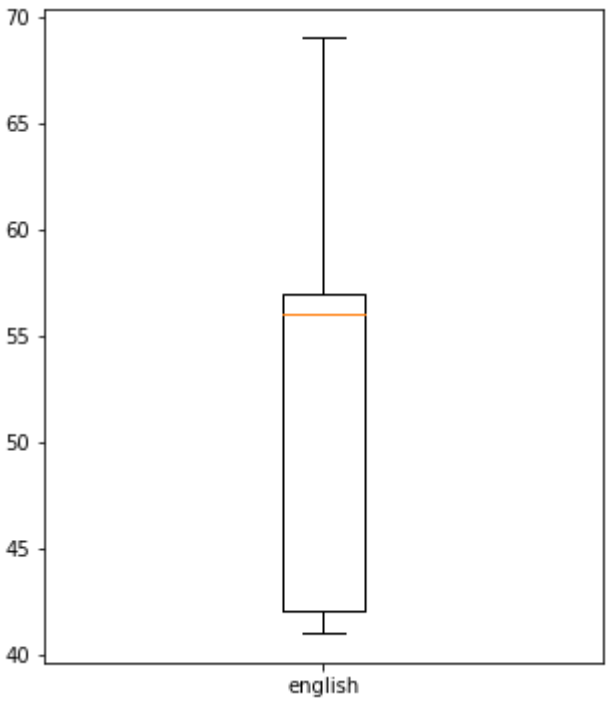
In [327]:

```
# 이상치 제거
df2 = df.copy()
df3 = df2.drop(index=[19,34]).head()
df3
```

student number	english	mathematics
1	42	65
2	69	80
3	56	63
4	41	63
5	57	76

In [328]:

```
score2 = df3.english

# 표준화
z2 = (score2 - np.mean(score2)) / np.std(score2)
# 배열로 변환 후 처음 5개 원소 출력
np.array(z2)[:5]

fig = plt.figure(figsize=(5,6))
ax = fig.add_subplot(111)
ax.boxplot(score2, labels=['english'])

plt.show() # 아랫쪽에 이상치 확인

# 이상치 값 및 인덱스 출력
score2_Q1 = np.percentile(score2, 25)
score2_Q3 = np.percentile(score2, 75)
score2_IQR = score_Q3 - score_Q1
print(score2_IQR) # IQR : 11
print(score2_Q1 - (1.5 * score2_IQR)) # 37.5 이하는 이상치

# 이상치 제거 확인
```




```
11.0
25.5
```

Q4. 아래 scores_df에 대해서 아래사항을 수행하세요

- scores_df.english와 scores_df.mathematics에 대한 공분산을 소수점 2째자리까지 출력
- scores_df.english와 scores_df.mathematics에 대한 상관계수를 소수점 2째자리까지 출력
- 두개 변수의 상관관계와 회귀직선을 시각화(회귀직선 포함 및 미포함 비교하여 1행 2열
- 두개 변수의 상관관계를 히트맵으로 시각화(칼러바 포함)

In [329]:

```
import numpy as np
import pandas as pd
df = pd.read_csv('ch2_scores_em.csv',
                 index_col='student number')
en_scores = np.array(df['english'])[:10]
ma_scores = np.array(df['mathematics'])[:10]

scores_df = pd.DataFrame({'english':en_scores,
                          'mathematics':ma_scores},
                         index=pd.Index(['A', 'B', 'C', 'D', 'E',
                                         'F', 'G', 'H', 'I', 'J'],
                                        name='student'))
scores_df.head()
```

student	english	mathematics
A	42	65
B	69	80
C	56	63
D	41	63
E	57	76

In [330]:

```
%precision 2
scores_df = scores_df.copy()
scores_df['english_deviation'] =W
    scores_df['english'] - scores_df['english'].mean()
scores_df['mathematics_deviation'] =W
    scores_df['mathematics'] - scores_df['mathematics'].mean()
scores_df['product of deviations'] =W
    scores_df['english_deviation'] * scores_df['mathematics_deviation']
scores_df
```

student	english	mathematics	english_deviation	mathematics_deviation	product of deviations
A	42	65	-13.0	-6.4	83.2
B	69	80	14.0	8.6	120.4
C	56	63	1.0	-8.4	-8.4
D	41	63	-14.0	-8.4	117.6
E	57	76	2.0	4.6	9.2
F	48	60	-7.0	-11.4	79.8
G	65	81	10.0	9.6	96.0
H	49	66	-6.0	-5.4	32.4
I	65	78	10.0	6.6	66.0
J	58	82	3.0	10.6	31.8

In [331]:

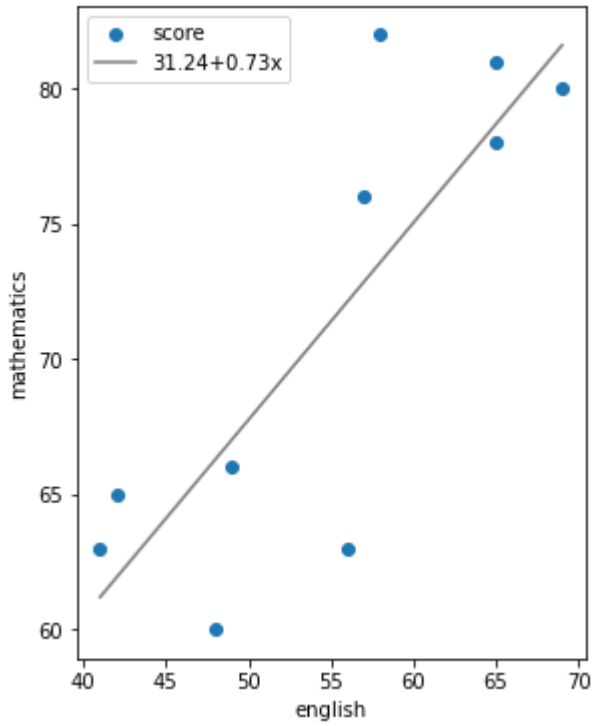
```
# 공분산
print(scores_df['product of deviations'].mean())
# 상관계수
np.cov(en_scores, ma_scores, ddof=0)[0,1]/(np.std(en_scores) * np.std(ma_scores))
```

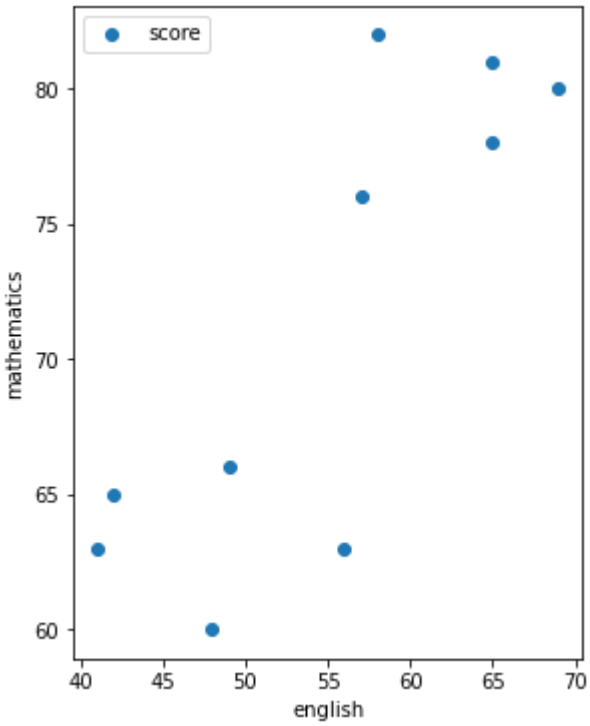
```
62.8
```

```
0.82
```

In [332]:

```
# 두개 변수의 상관관계와 회귀직선을 시각화(회귀직선 포함 및 미포함 비교하여 1행 2열로 출

# 회귀직선+상관관계
poly_fit = np.polyfit(en_scores, ma_scores, 1)
poly_1d = np.poly1d(poly_fit)
xs = np.linspace(en_scores.min(), en_scores.max())
ys = poly_1d(xs)

fig = plt.figure(figsize=(10, 6))
ax = fig.add_subplot(121)
ax.scatter(en_scores, ma_scores, label='score')
ax.plot(xs, ys, color='gray',
        label=f'{poly_fit[1]:.2f}+{poly_fit[0]:.2f}x')
ax.set_xlabel('english')
ax.set_ylabel('mathematics')
ax.legend(loc='upper left')

# 상관관계만
fig2 = plt.figure(figsize=(10, 6))
ax2 = fig.add_subplot(122)
ax2.scatter(en_scores, ma_scores, label='score')
ax2.set_xlabel('english')
ax2.set_ylabel('mathematics')
ax2.legend(loc='upper left')
plt.show()
```

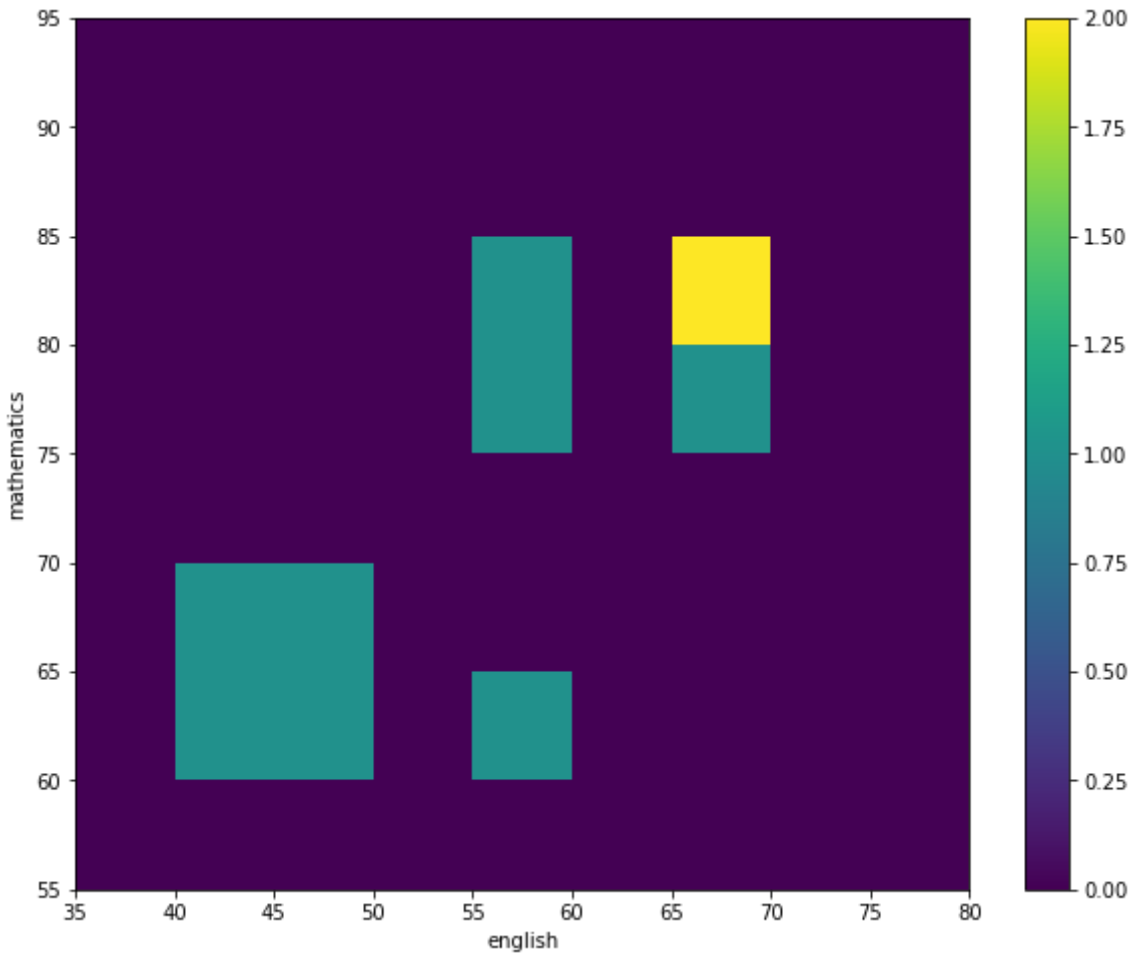





```
<Figure size 720x432 with 0 Axes>
```

In [333]:

```
# 두개 변수의 상관관계를 히트맵으로 시각화(칼러바 포함)
fig = plt.figure(figsize=(10,8))
ax = fig.add_subplot(111)

c = ax.hist2d(en_scores, ma_scores, bins=[9,8], range=[(35,80), (55,95)])
ax.set_xlabel('english')
ax.set_ylabel('mathematics')
ax.set_xticks(c[1])
ax.set_yticks(c[2])

fig.colorbar(c[3], ax=ax)
plt.show()
```



Q5. 아래 scores는 전교생의 시험점수이다. 무작위추출로 표본 크기가 20인 표본을 추출하0
10000번 수행해서 그 결과를 히스토그램으로 그려 표본평균이 어떻게 분포되는지 시각화를

In [334]:

```
df = pd.read_csv('ch4_scores400.csv')
scores = np.array(df['score'])
scores[:10]
```

```
array([76, 55, 80, 80, 74, 61, 81, 76, 23, 80], dtype=int64)
```

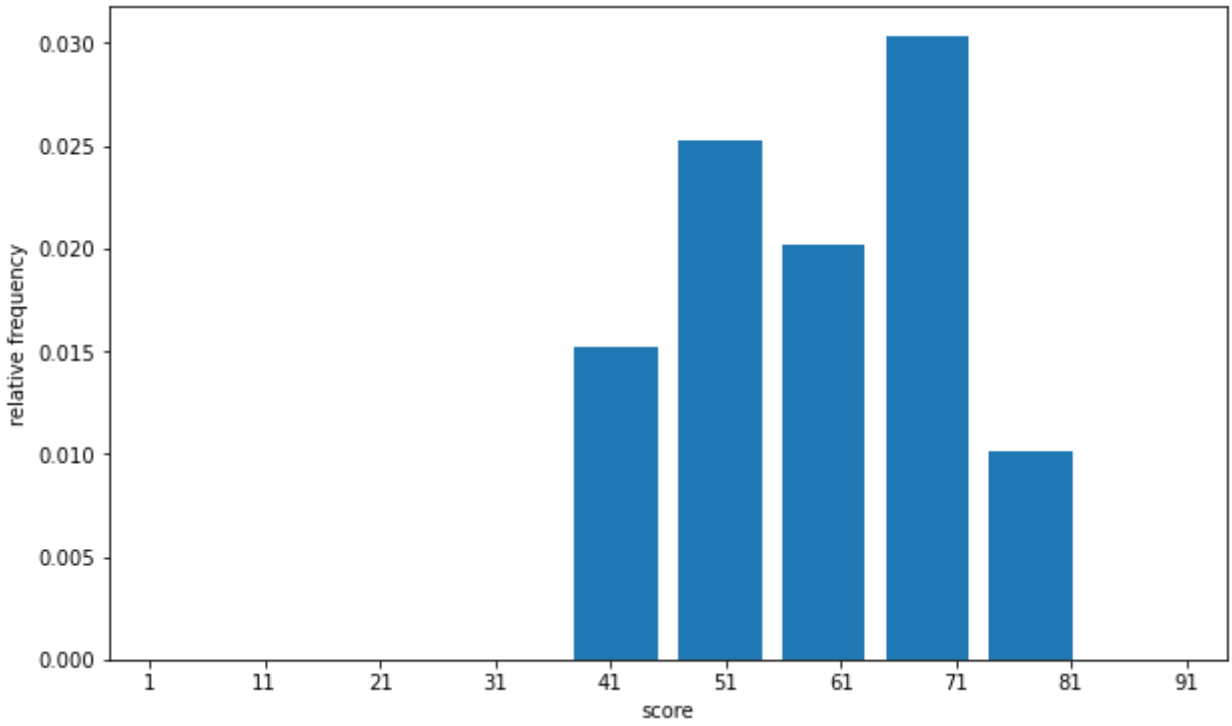
In [335]:

```
np.random.seed(0)
for i in range(10000) :
    sample = np.random.choice(scores, 20)
    print(sample.mean()) # 1만번 수행
```

```
70.4
72.45
63.7
66.05
71.7
74.15
70.7
71.9
71.25
67.4
67.7
69.15
69.5
71.35
69.4
71.5
70.65
69.95
68.8
69.15
71.25
67.85
73.1
```

In [336]:

```
# 히스토그램으로 표본평균 분포상태 시각화
fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111)
ax.hist(sample, bins=10, range=(1,100), density=True, rwidth=0.8)

ax.set_xticks(np.linspace(1, 100, 10))
ax.set_xticklabels(np.arange(1,100, 10))
ax.set_xlabel('score')
ax.set_ylabel('relative frequency')
plt.show()
```



Q6. Bern(0.5)을 따르는 확률변수 X에 대하여 기댓값과 분산을 계산하세요.

In [337]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

%precision 3
%matplotlib inline

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])
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])
def check_prob(X):
    x_set, f = X
    prob = np.array([f(x_k) for x_k in x_set])
    assert np.all(prob >= 0), 'minus probability'
    prob_sum = np.round(np.sum(prob), 6)
    assert prob_sum == 1, f'sum of probability{prob_sum}'
    print(f'expected value {E(X):.4}')
    print(f'variance{(V(X)):.4}')
def plot_prob(X):
    x_set, f = X
    prob = np.array([f(x_k) for x_k in x_set])

    fig = plt.figure(figsize=(10,6))
    ax = fig.add_subplot(111)
    ax.bar(x_set, prob, label='prob')
    ax.vlines(E(X), 0, 1, label='mean')
    ax.set_xticks(np.append(x_set, E(X)))
    ax.set_ylim(0, prob.max()*1.2)
    ax.legend()

    plt.show()
```

In [338]:

```
def Bern(p) :
    x_set = np.array([0,1])
    def f(x):
        if x in x_set:
            return p ** x * (1-p) ** (1-x)
        else :
            return 0
    return x_set, f
```

In [339]:

```
# Bern(0.5)  >> p = 0.5
p = 0.5
X = Bern(p)
# 기댓값, 분산
check_prob(X)
```

```
expected value 0.5
variance0.25
```

Q7. Bin(10,0.5)을 따르는 확률변수 X에 대하여 기댓값과 분산을 계산하세요.

In [340]:

```
from scipy.special import comb

def Bin(n, p):
    x_set = np.arange(n+1)
    def f(x):
        if x in x_set:
            return comb(n, x) * p**x * (1-p)**(n-x)
        else:
            return 0
    return x_set, f
```

In [341]:

```
n = 10
p = 0.5
X = Bin(n, p)
# 기댓값, 분산
check_prob(X)
```

```
expected value 5.0
variance2.5
```

Q8. Poi(2)을 따른 확률변수 X에 대하여 기댓값과 분산을 계산하세요.

In [342]:

```
# 확률변수 X가 따르는 포아송분포의 람다값은 기대값, 분산과 같다.
from scipy.special import factorial

def Poi(lam):
    x_set = np.arange(20) # 임의로 20 넣음. 원래 범위는 양수 전체
    def f(x):
        if x in x_set:
            return np.power(lam, x) / factorial(x) * np.exp(-lam)
        else:
            return 0
    return x_set, f
```

In [343]:

```
lam = 2
X = Poi(lam)

check_prob(X)
```

```
expected value 2.0
variance2.0
```

Q9. 평균이 10, 표준편차가 3인 정규분포의 확률밀도함수를 그래프로 표현하세요.

In [344]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats, integrate
from scipy.optimize import minimize_scalar

%precision 3
%matplotlib inline

linestyles = ['-', '--', ':']

def E(X, g=lambda x: x):
    x_set, f = X
    def integrand(x):
        return g(x) * f(x)
    return integrate.quad(integrand, -np.inf, np.inf)[0]

def V(X, g=lambda x: x):
    x_range, f = X
    mean = E(X,g)
    def integrand(x):
        return (g(x) - mean) ** 2 * f(x)
    return integrate.quad(integrand, -np.inf, np.inf)[0]

def check_prob(X):
    x_range, f = X
    f_min = minimize_scalar(f).fun
    assert f_min >= 0, 'density function is minus value'
    prob_sum = np.round(integrate.quad(f, -np.inf, np.inf)[0], 6)
    assert prob_sum == 1, f'sum of probability is {prob_sum}'
    print(f'expected value{E(X):.3f}')
    print(f'variance{V(X):.3f}')

def plot_prob(X, x_min, x_max):
    x_range, f = X
    def F(x):
        return integrate.quad(f, -np.inf, x)[0]

    xs = np.linspace(x_min, x_max, 100)

    fig = plt.figure(figsize=(10,6))
```

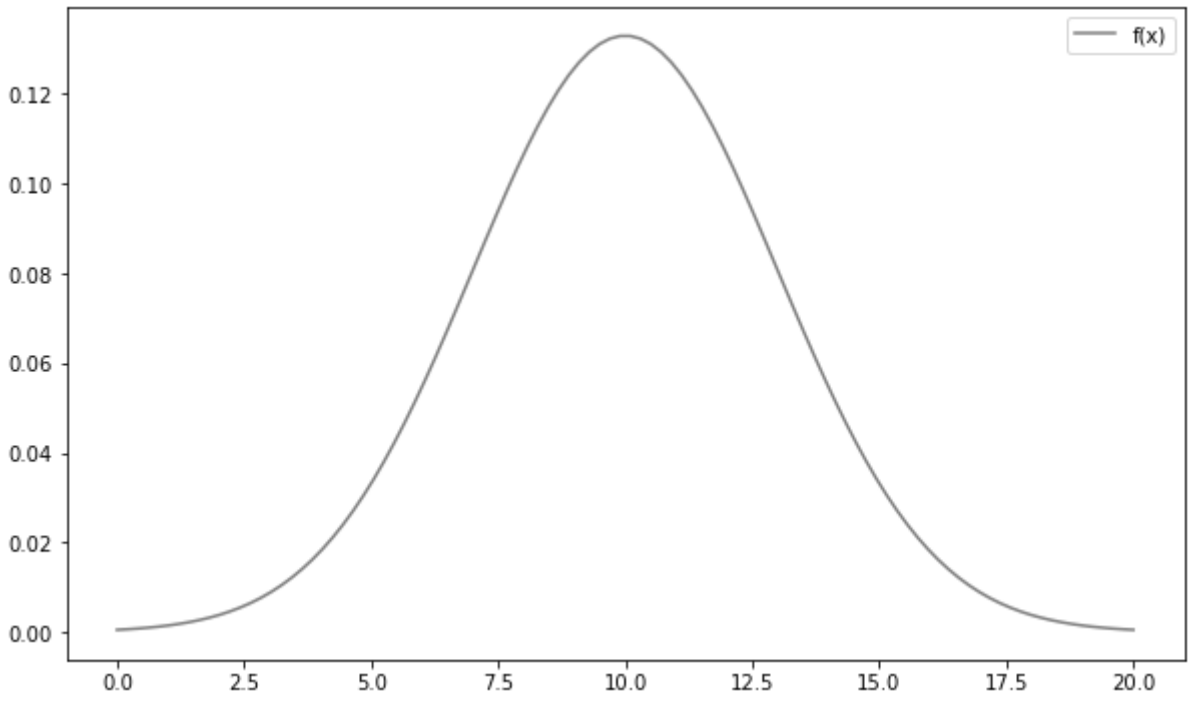
```
    ax = fig.add_subplot(111)
    ax.plot(xs, [f(x) for x in xs], label='f(x)', color='gray')
    # ax.plot(xs, [F(x) for x in xs], label='F(x)', ls='--', color='gray')

    ax.legend()
    plt.show()
```

In [345]:

```
# 확률밀도함수 pdf
def N(mu, sigma):
    x_range = [-np.inf, np.inf]
    def f(x):
        return 1 / np.sqrt(2 * np.pi * sigma**2) * np.exp(-(x-mu)**2 / (2 * sigma**2))
    return x_range, f
```

In [346]:

```
# 평균 10, 표준편차 3
mu, sigma = 10, 3
X = N(mu, sigma)

plot_prob(X, 0, 20)
```



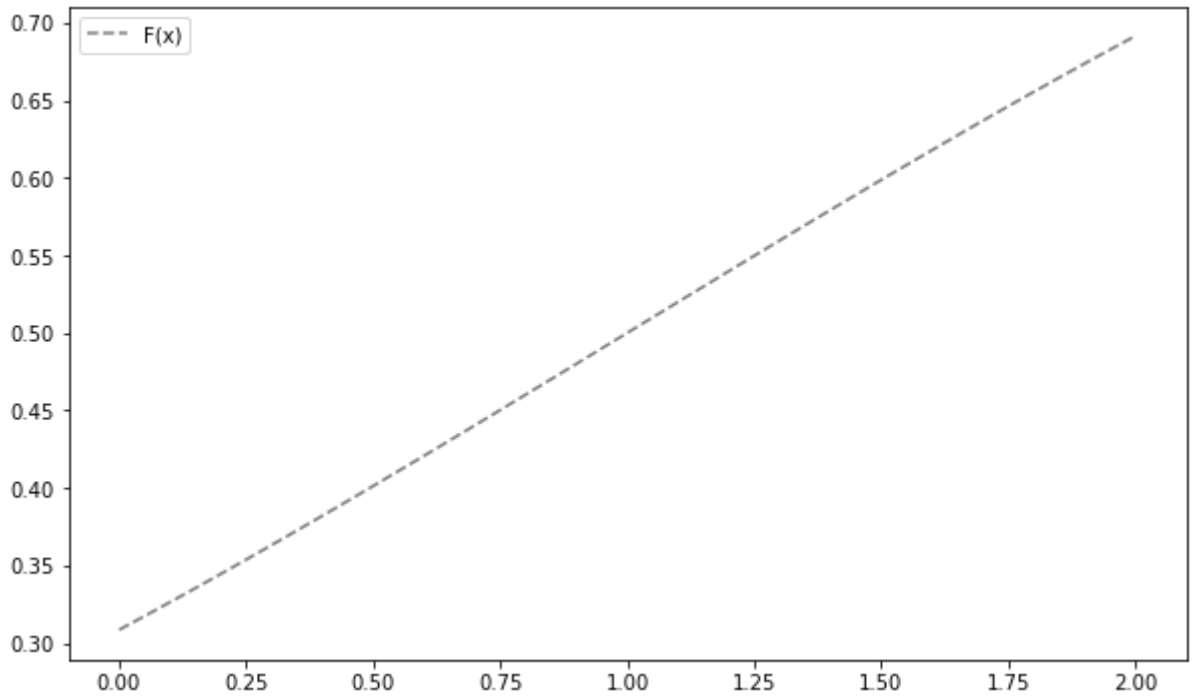
Q10. 평균이 1, 표준편차가 2인 정규분포의 누적분포함수를 그래프로 표현하세요.

In [347]:

```
def plot_prob(X, x_min, x_max):
    x_range, f = X
    def F(x):
        return integrate.quad(f, -np.inf, x)[0]

    xs = np.linspace(x_min, x_max, 100)

    fig = plt.figure(figsize=(10,6))
    ax = fig.add_subplot(111)
    # ax.plot(xs, [f(x) for x in xs], label='f(x)', color='gray')
    ax.plot(xs, [F(x) for x in xs], label='F(x)', ls='--', color='gray')

    ax.legend()
    plt.show()
```

In [348]:

```
# 평균 1, 표준편차 2
mu, sigma = 1, 2
X = N(mu, sigma)

plot_prob(X, 0, 2)
```



Q11. "5_2_fm.csv"을 df1으로 불러와서 다음사항을 수행하세요.

- df1을 df2 이름으로 복사한 후 df2의 species의 A, B를 C,D로 변경하세요.

- df의 length를 species가 C인 것은 2배로 d인 것은 3배로 변경하여 df1과 df2를 행방향.
- df를 species 칼럼을 기준으로 그룹별 평균과 표준편차를 산출

In [349]:

```
import pandas as pd
import numpy as np

df1 = pd.read_csv("5_2_fm.csv")
df1
```

	species	length
0	A	2
1	A	3
2	A	4
3	B	6
4	B	8
5	B	10

In [350]:

```
df2 = df1.copy()
```

In [351]:

```
import warnings
warnings.filterwarnings('ignore')
# C, D로 변경
df2.species[:3] = 'C'
df2.species[3:] = 'D'
df2
```

	species	length
0	C	2
1	C	3
2	C	4
3	D	6
4	D	8
5	D	10

- df의 length를 species가 C인 것은 2배로 d인 것은 3배로 변경하여 df1과 df2를 행방향.

- df를 species 칼럼을 기준으로 그룹별 평균과 표준편차를 산출

In [352]:

```
df3 = df2.copy()
df3.length[:3] = df3.length[:3]*2
df3.length[3:] = df3.length[3:]*3
df3
```

	species	length
0	C	4
1	C	6
2	C	8
3	D	18
4	D	24
5	D	30

In [353]:

```
# df = pd.concat(df1, df3)
```

Q12. "./dataset/5_2_shoes.csv" 을 데이터프레임으로 불러와서 아래작업을 수행하세요.

- 4행 3열을 복사 후 추가하여 8행 3열로 작성
- 피봇을 이용해서 교차분석표 작성(values='sales',aggfunc='sum', index= 'store', columr
- 독립성 검정을 수행(보너스 문제)

In [354]:

```
import pandas as pd
shoes = pd.read_csv("5_2_shoes.csv")
shoes1=shoes.copy()
shoes1
```

	store	color	sales
0	tokyo	blue	10
1	tokyo	red	15
2	osaka	blue	13
3	osaka	red	9

Q13. 'dataset/titanic3.csv'을 불러와서 pclass 와 sex 칼럼을 각각 인덱스, 칼럼으로 하고 val 적용하여 pivot_table을 만든 후 히트맵으로 시각화 및 인사이트를 기술하세요

In [355]:

```
titanic = pd.read_csv('titanic3.csv')
titanic.head(2)
```

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	b
0	1	1	Allen, Miss. Elisabeth Walton	female	29.00	0	0	24160	211.3375	B5	S	2
1	1	1	Allison, Master. Hudson Trevor	male	0.92	1	2	113781	151.5500	C22 C26	S	1

Q14. 평균 4, 표준편차 0.8인 정규분포에서 샘플사이즈 10인 표본 10000개의 표본평균을 바
요.(넘파이 zeros 함수 이용)

In [ ]:

```
```

Q15. Q14에서 구한 배열의 히스토그램을 시각화하세요.(확률밀도 포함)

Q16. 서로 독립인 X~N(1,2), Y~N(2,3)이 있을 때 확률변수 X + Y의 분포는 N(3,5)를 따른다

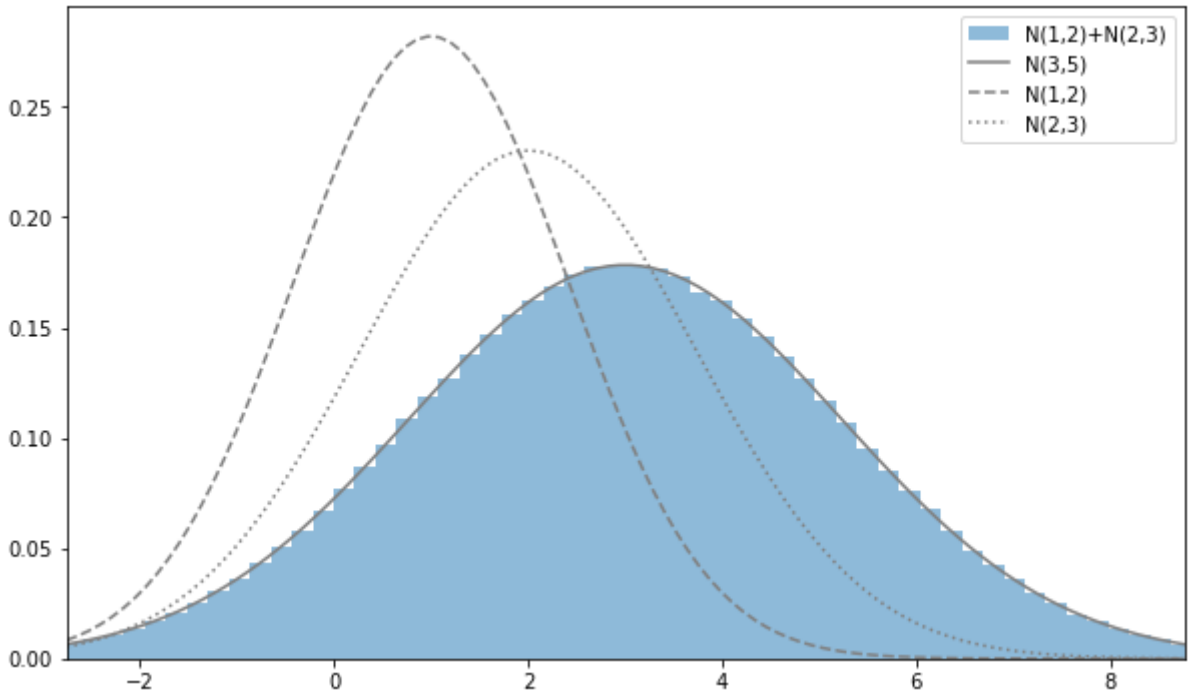
In [356]:

```
from scipy import stats

rv1 = stats.norm(1, np.sqrt(2))
rv2 = stats.norm(2, np.sqrt(3))

sample_size = int(1e6)
X_sample = rv1.rvs(sample_size)
Y_sample = rv2.rvs(sample_size)
sum_sample = X_sample + Y_sample

np.mean(sum_sample), np.var(sum_sample)
```

```
(3.005, 4.989)
```

In [357]:

```
fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111)

rv = stats.norm(3, np.sqrt(5))
xs = np.linspace(rv.isf(0.995), rv.isf(0.005), 100)

ax.hist(sum_sample, bins=100, density=True, alpha=0.5, label='N(1,2)+N(2,3)')
ax.plot(xs, rv.pdf(xs), label='N(3,5)', color='gray')
ax.plot(xs, rv1.pdf(xs), label='N(1,2)', ls='--', color='gray')
ax.plot(xs, rv2.pdf(xs), label='N(2,3)', ls=':', color='gray')

ax.legend()
ax.set_xlim(rv.isf(0.995), rv.isf(0.005))
plt.show()
```



Q17. 서로 독립인 X ~ Poi(3)과 Y ~ Poi(4)가 있을 때 확률변수 X + Y도 포아송 분포를 따른요.

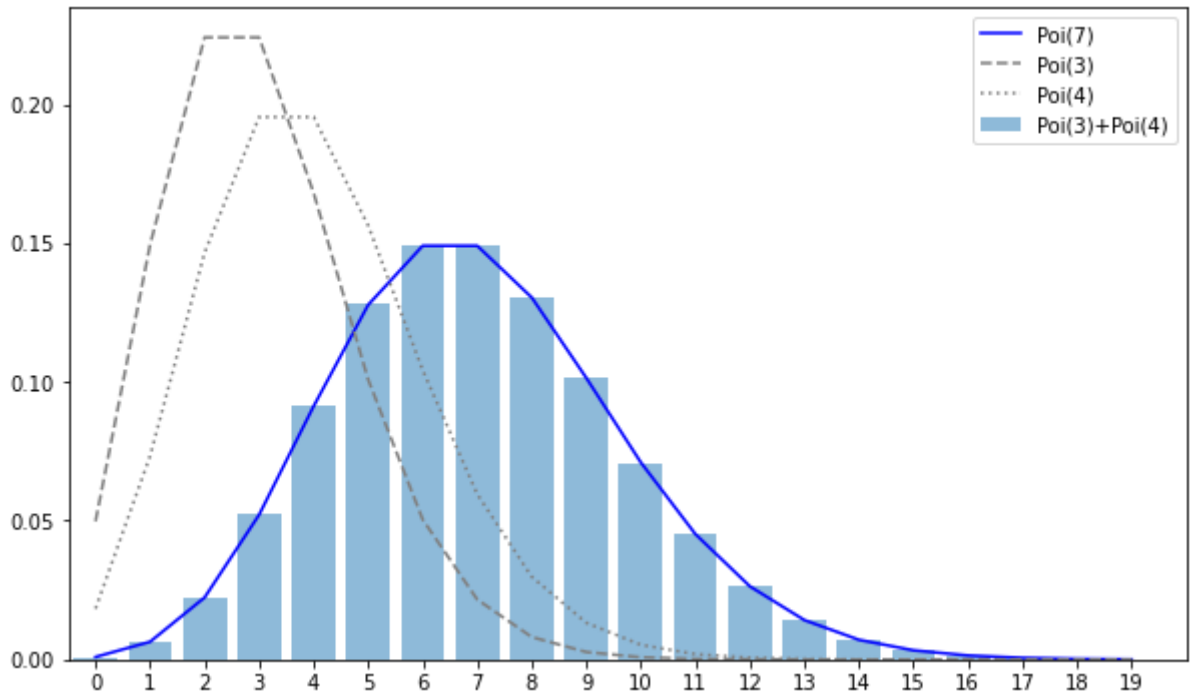
In [358]:

```
rv1 = stats.poisson(3)
rv2 = stats.poisson(4)

sample_size = int(1e6)
X_sample = rv1.rvs(sample_size)
Y_sample = rv2.rvs(sample_size)
sum_sample = X_sample + Y_sample

np.mean(sum_sample), np.var(sum_sample)
```

```
(6.999, 6.993)
```

In [359]:

```
fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111)

rv = stats.poisson(7)
xs = np.arange(20)
hist, _ = np.histogram(sum_sample, bins=20, range=(0,20), normed=True)

ax.bar(xs, hist, alpha=0.5, label='Poi(3)+Poi(4)')
ax.plot(xs, rv.pmf(xs), label='Poi(7)', color='blue')
ax.plot(xs, rv1.pmf(xs), label='Poi(3)', ls='--', color='gray')
ax.plot(xs, rv2.pmf(xs), label='Poi(4)', ls=':', color='gray')

ax.legend()
ax.set_xlim(-0.5, 20)
ax.set_xticks(np.arange(20))
plt.show()
```



Q18. 베르누이 분포의 합은 이항분포가 되는 성질을 시각화하여 출력하세요

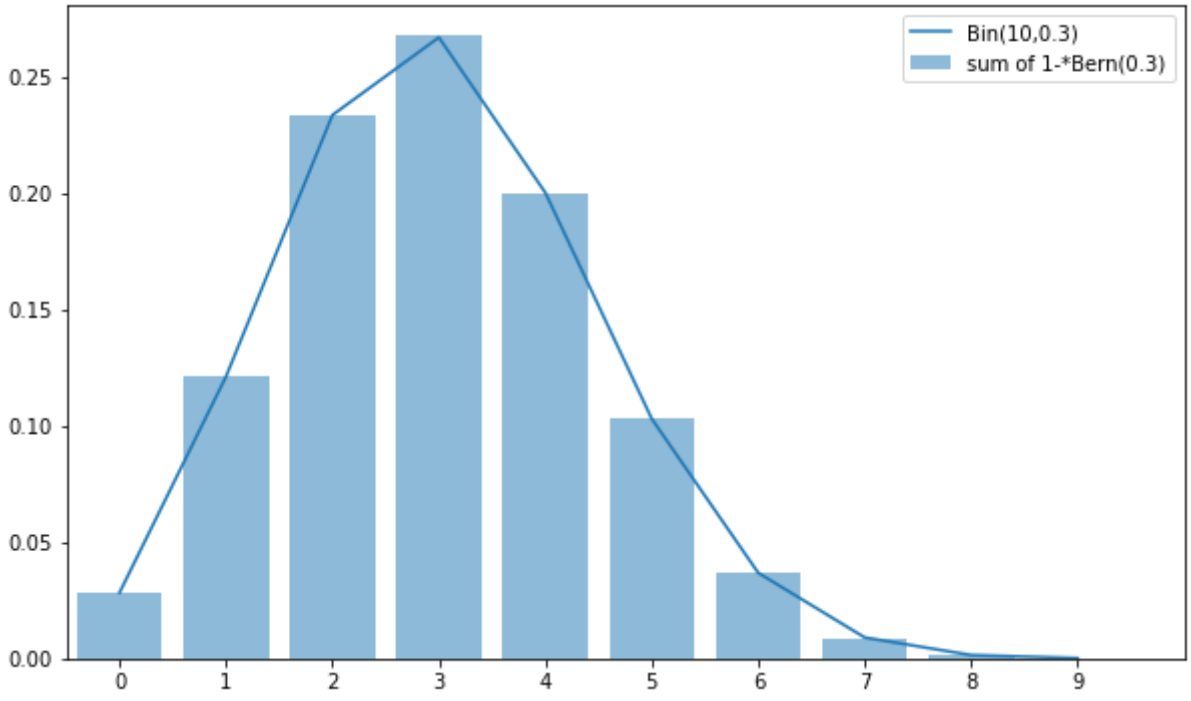
In [360]:

```
p = 0.3
rv = stats.bernoulli(p)

sample_size = int(1e6)
Xs_sample = rv.rvs((10, sample_size))
sum_sample = np.sum(Xs_sample, axis=0)

np.mean(sum_sample), np.var(sum_sample)
```

```
(2.997, 2.097)
```

In [361]:

```
fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111)

rv = stats.binom(10,p)
xs = np.arange(10)
hist, _ = np.histogram(sum_sample, bins=10, range=(0,10), normed=True)
ax.bar(xs, hist, alpha=0.5, label='sum of 1-*Bern(0.3)')
ax.plot(xs, rv.pmf(xs), label='Bin(10,0.3)')
ax.legend()
ax.set_xlim(-0.5, 10)
ax.set_xticks(np.arange(10))
plt.show()
```



Q19. 포아송 분포의 표본분포는 근사적으로 정규분포를 따른다는 것을 시각화하고 그 핵심 설명하세요.

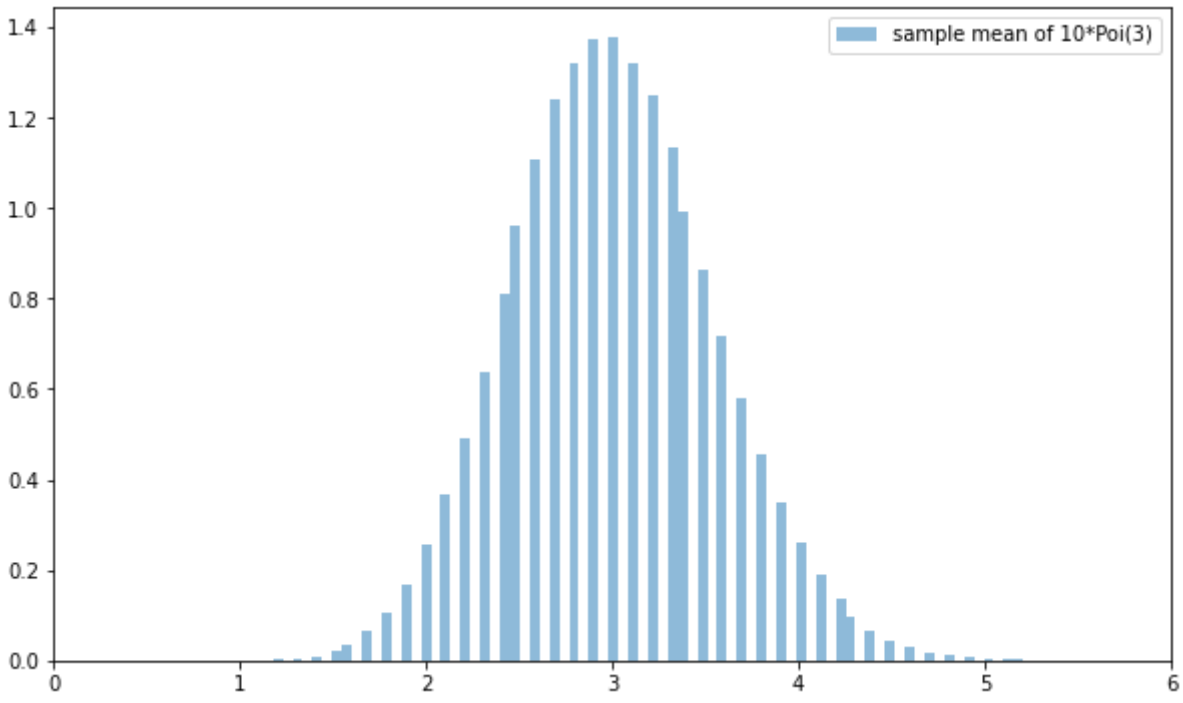
In [362]:

```
l = 3
rv = stats.poisson(l)

n = 10
sample_size = int(1e6)
Xs_sample = rv.rvs((n, sample_size))
sample_mean = np.mean(Xs_sample, axis=0)

np.mean(sample_mean), np.var(sample_mean)
```

```
(3.000, 0.300)
```

In [363]:

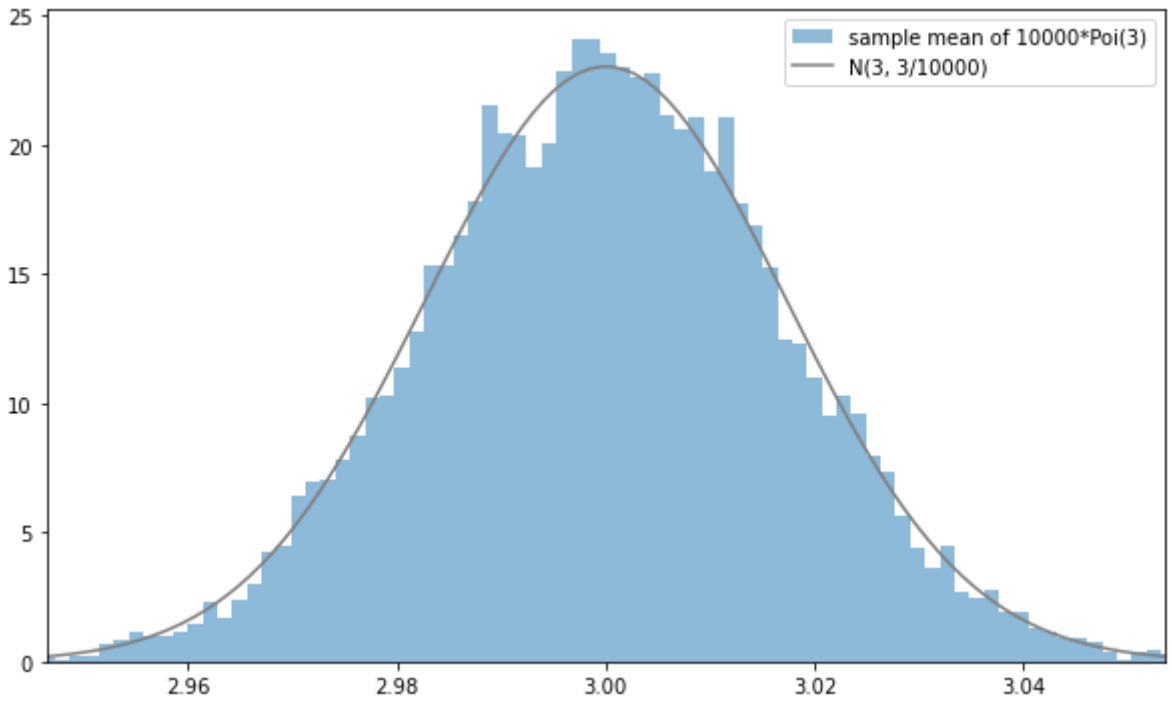
```
fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111)

ax.hist(sample_mean, bins=100, density=True, alpha=0.5, label='sample mean of 10*Poi(3)

ax.legend()
ax.set_xlim(0,6)
plt.show()
```



중심극한정리 : 확률변수X들이 각각 독립이고, 기댓값이 뮤, 분산이 시그마제곱인 확률분포
표본평균 X의 분포는 정규분포 N(뮤, (시그마제곱/n))에 가까워진다.

In [364]:

```
l = 3
rv = stats.poisson(l)
n = 10000
sample_size = 10000
Xs_sample = rv.rvs((n, sample_size))
sample_mean = np.mean(Xs_sample, axis=0)

rv_true = stats.norm(l, np.sqrt(l/n))
xs = np.linspace(rv_true.isf(0.999), rv_true.isf(0.001), 100)

fig = plt.figure(figsize=(10,6))
ax = fig.add_subplot(111)

ax.hist(sample_mean, bins=100, density=True, alpha=0.5, label='sample mean of 10000*Poi
ax.plot(xs, rv_true.pdf(xs), label='N(3, 3/10000)', color='gray')
ax.legend()
ax.set_xlim(rv_true.isf(0.999), rv_true.isf(0.001))
plt.show()
```



Q20. 아래 df 데이터셋에서 "무게의 평균이 130kg이다."라는 귀무가설에 대한 유의성 검정을

In [365]:

```
np.random.seed(1111)
```

대립가설1 : 무게의 평균이 130kg이 아니다. (단측검정)

대립가설2 : 무게의 평균이 130kg이 아니다. (양측검정)

In [366]:

```
df = pd.read_csv('ch11_potato.csv')
print(df.head(),len(df))
```

```
      무게
0  122.02
1  131.73
2  130.60
3  131.82
4  132.05 14
```

In [367]:

```
sample = np.array(df['무게'])
s_mean = np.mean(sample)
```

In [368]:

```
# 모분산이 9임을 알고있고, 모집단은 정규분포를 따름을 전제
rv = stats.norm(130, np.sqrt(9/14))
```

In [369]:

```
# 검정통계량
z = (s_mean - 130) / np.sqrt(9/14)
z
```

```
-1.932
```

In [370]:

```
# 단측검정
rv = stats.norm()
rv.isf(0.95)
```

```
-1.645
```

In [371]:

```
rv.cdf(z)
```

```
0.027
```

단측검정에서는 검정통계량<임곗값, p값<유의수준(0.05) 이므로 귀무가설 기각

In [372]:

```
# 양측검정의 임곗값
rv = stats.norm()
rv.interval(0.95)
```

```
(-1.960, 1.960)
```

In [373]:

```
rv.cdf(z) * 2
```

```
0.053
```

양측검정에서는 검정통계량이 채택역 안에 들어옴 : 귀무가설 기각하지 않음

p값이 유의수준 0.05보다 크므로 귀무가설 기각하지 않음