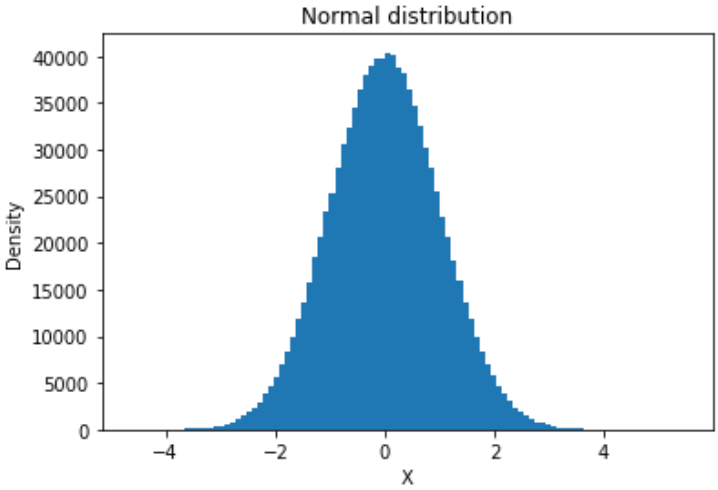
นายชินพัทธ์ นิธิพรศรี 6332009521

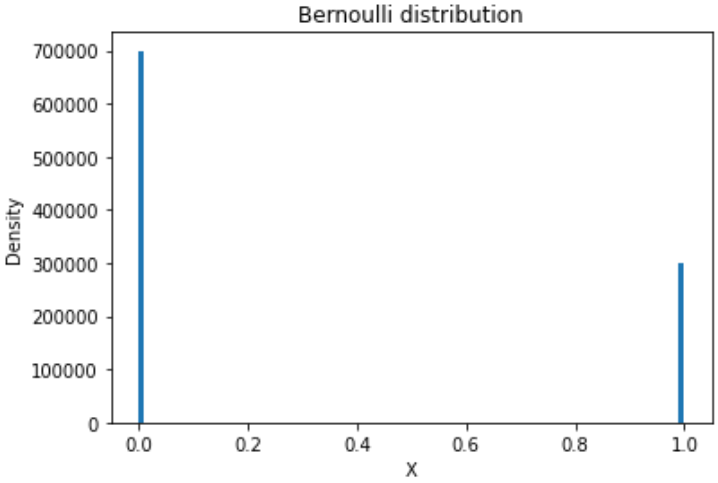
Assignment 9

TODO#1

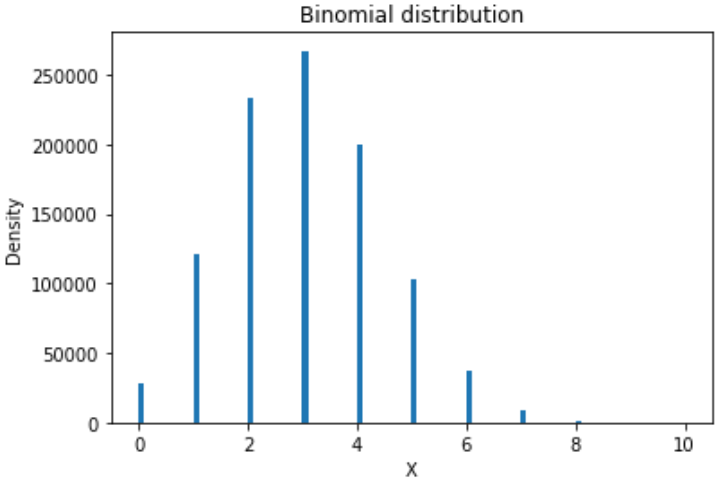
1) N(0, 1)



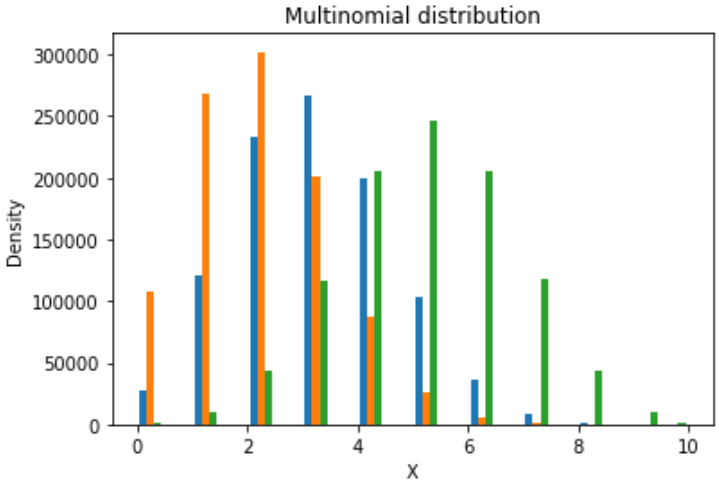
2) Bernoulli(0.3)



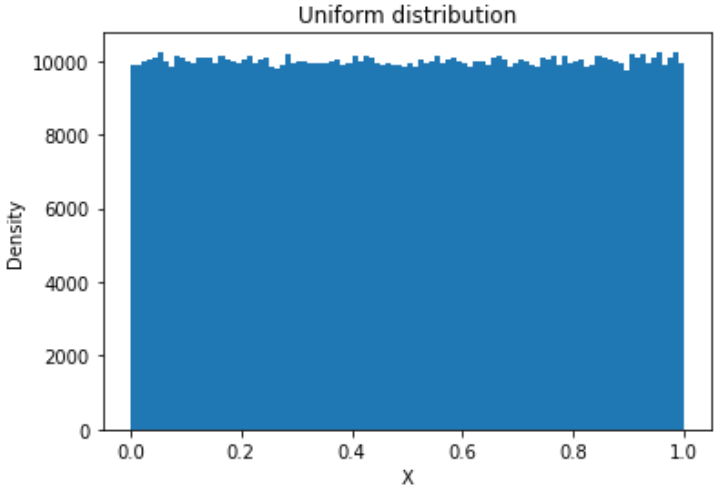
3) B(10, 0.3)



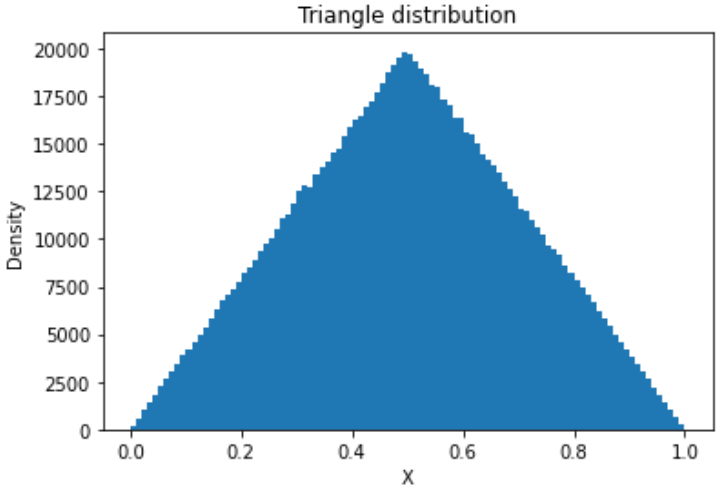
4) Multinomial(n = 10, p = [0.3, 0.2, 0.5])



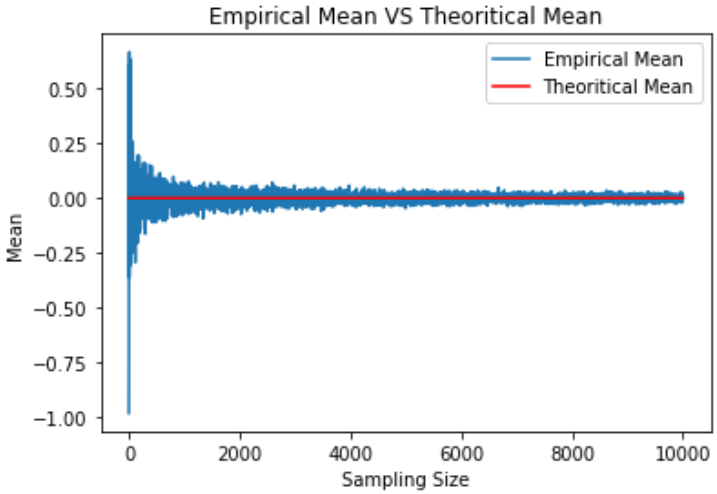
5) U(0, 1)



6) T(0, 1)

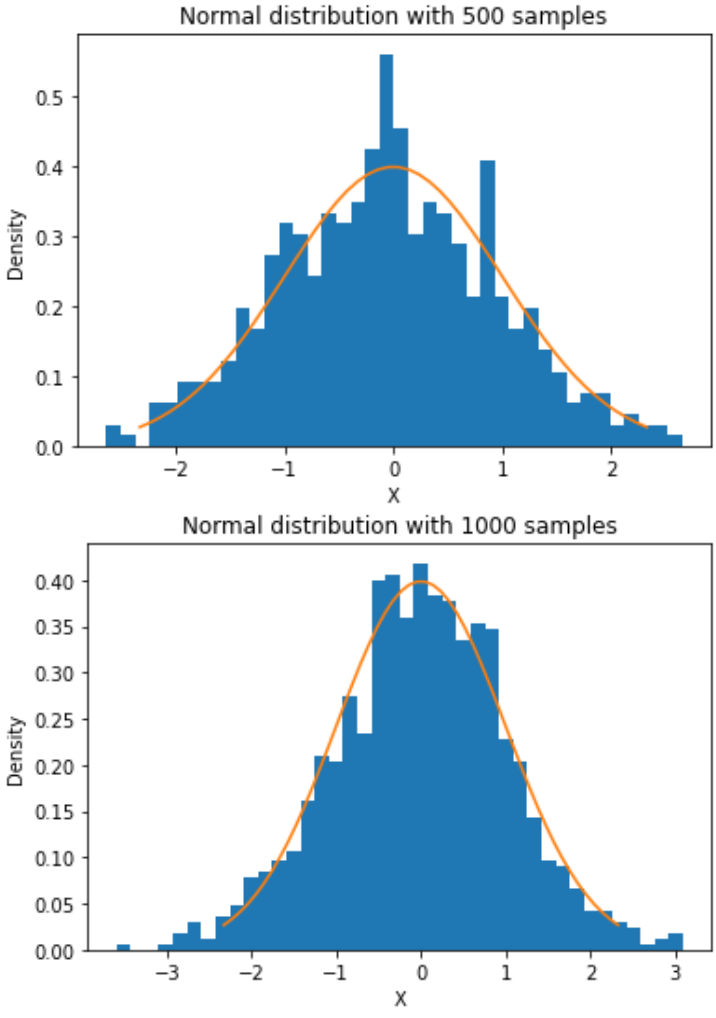


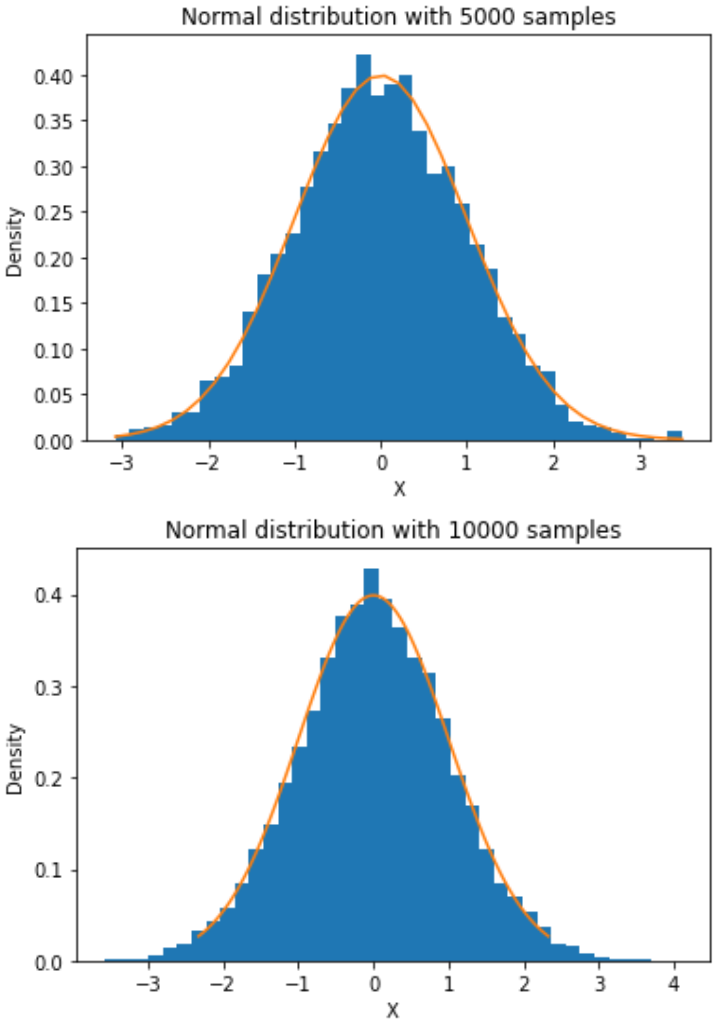
TODO#2



จากการ plot เปรียบเทียบระหว่าง Empirical Mean และ Theoretical Mean จะเห็นว่ายิ่ง Sampling size มีขนาดใหญ่ขึ้นก็จะยิ่งทำให้ Empirical Mean มีค่าลู่เข้า Theoretical Mean มากเท่านั้น

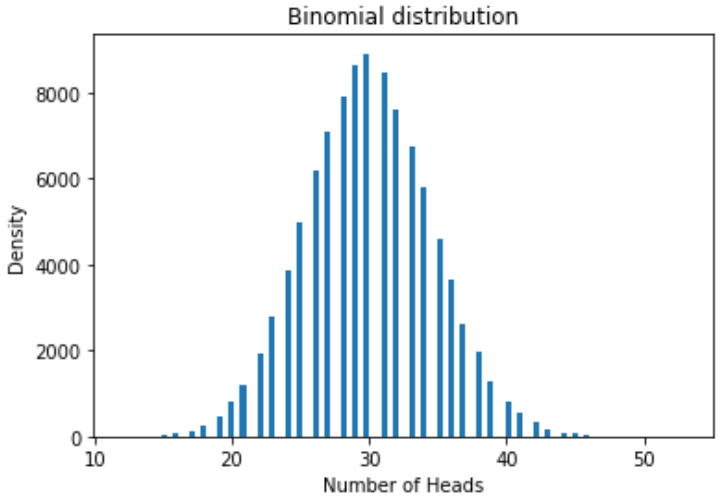
TODO#3 [หมายเหตุ : ในข้อนี้จะทำการตั้ง density = True เพื่อการเปรียบเทียบที่ชัดเจน]





จากการ plot histogram ของ sample size ขนาด 500, 1000, 5000, 10000 ตามลำดับ เทียบกับ true PDF พบว่า histogram มีความใกล้เคียงกับ true PDF มากขึ้น เมื่อ sample size มีขนาดมากขึ้น

TODO#4



เมื่อ plot จำนวนครั้งของการออกหัวจากการโยนเหรียญ 100 ครั้งของเหรียญ 100000 เหรียญ (จำนวนมาก) จะเห็นว่า histrogram ที่ได้ออกมามีหน้าตาคล้ายกับ normal distribution

TODO#5

ต้องการหา $P(X > 40) = 1 - P(X \leq 40)$

โดยทฤษฎี CLT จะได้ว่าจำนวนครั้งของการออกหัวจากการโยนเหรียญ 100 ครั้งของเหรียญ 100000 เหรียญ (จำนวนมาก) มีลักษณะคล้าย normal distribution นั่นคือเราสามารถคำนวณ $P(X \leq 40)$ จาก $CDF(z)$ เมื่อ $z = \frac{40-mean}{std}$

ซึ่งเมื่อทำการคำนวนผ่าน python ตามโค้ดดังนี้

```
n = 100
p = 0.3

z = (40 - binom.mean(n, p, loc=0))/binom.std(n, p, loc=0)
probability = 1-norm.cdf(z)
print(probability)
```

```
0.014548165870626129
```

จึงได้ว่า probability ที่ได้จากการคำนวณผ่าน CLT มีค่าประมาณ 0.0145

TODO#6

ทำการคำนวณหา actual probability โดยการนับจำนวนเหรียญที่ทอยแล้วออกหัวมากกว่า 40 ครั้ง แล้วนำมาหารจำนวนเหรียญทั้งหมดโดยทำผ่านโค้ดดังนี้

```
# TODO#6
count = 0
for c in s:
  if c >= 41:
    count += 1
print(count/len(s))
```

```
0.01235
```

ซึ่งได้ว่า actual probability มีค่า 0.01235

ซึ่งจะเห็นว่า probability ที่ได้จากการคำนวณผ่าน CLT มีค่าสูงกว่า actual probability ประมาณ 0.002 เท่านั้น โดยความแตกต่างนี้อาจเกิดจากการสุ่มทอยลูกเต๋าที่ไม่สามารถกำหนดให้เป็น normal distribution อย่างสมบูรณ์แบบได้

TODO#7

จาก $Z = X + Y$ จะได้ว่า $p_Z(z) = p_X(x) * p_Y(y)$ ตาม code ข้างล่างนี้

```
tx = np.arange(-3, 3, delta)
X = 0.1*np.heaviside(tx+2, 1) + 0.3*np.heaviside(tx, 1) - 0.4*np.heaviside(tx-2, 1)

ty = np.arange(2, 6, delta)
Y = (np.heaviside(ty-3, 1) - np.heaviside(ty-5, 1))/(5-3)

# have to find p(z) = p(x+y) = p(x)*p(y)
tz = np.arange(-1, 9-delta, delta)
Z = np.convolve(X, Y)*delta
```

จากนั้นทำการหาพื้นที่ใต้กราฟทั้งหมดของ $p_Z(z)$ ใน result

```
result_start = 0
result_start_index = (result_start+1)*1000
result_stop = 8
result_stop_index = (result_stop+1)*1000
result = round(np.sum(Z[result_start_index:result_stop_index+1])*delta, 3)
```

และทำการหาพื้นที่ใต้กราฟตั้งแต่ 3 ถึง 5 ของ $p_Z(z)$ ใน partial

```
partial_start = 3
partial_start_index = (partial_start+1)*1000
partial_stop = 5
partial_stop_index = (partial_stop+1)*1000
partial = round(np.sum(Z[partial_start_index:partial_stop_index+1])*delta, 3)
```

ซึ่งจะได้ว่า $P(3 < Z < 5) = \frac{partial}{result} = 0.5$

TODO#8

```
# TODO#8
X = sample_uniform(sample_size=100000, from_x=-1, to_x=1)

# 8.1
A1 = 10
Y1 = X + A1
print("correlation of  X  and  Y1 : ", np.corrcoef(X, Y1)[0][1])

# 8.2
A2 = sample_uniform(sample_size=100000, from_x=-1, to_x=1)
Y2 = X + A2
print("correlation of  X  and  Y2 : ", np.corrcoef(X, Y2)[0][1])

# 8.3
A3 = sample_uniform(sample_size=100000, from_x=-10, to_x=10)
Y3 = X + A3
print("correlation of  X  and  Y3 : ", np.corrcoef(X, Y3)[0][1])

# 8.4
A4 = sample_uniform(sample_size=100000, from_x=-100, to_x=100)
Y4 = X + A4
print("correlation of  X  and  Y4 : ", np.corrcoef(X, Y4)[0][1])
```

```
correlation of  X  and  Y1 :  0.9999999999999998
correlation of  X  and  Y2 :  0.7066700835728315
correlation of  X  and  Y3 :  0.10031037591820145
correlation of  X  and  Y4 :  0.006530791298734063
```

TODO#9

1) จากข้อ 8 จะเห็นว่า X และ Y มี correlation ที่ลดน้อยลงเรื่อย ๆ เมื่อ A มีการกระจายตัวที่มากขึ้น
2) จาก code ข้างล่าง แสดงให้เห็นว่า ถึงแม้ ช่วงของ A จะไม่ได้อยู่ตำแหน่งเดิม แต่มีการกระจายตัวแบบก็จะทำให้ได้ correlation เท่าเดิม

```
# TODO#9
A5 = A3+10000
Y5 = X + A5
print("correlation of  X  and  Y3 : ", np.corrcoef(X, Y3)[0][1])
print("correlation of  X  and  Y5 : ", np.corrcoef(X, Y5)[0][1])
```

```
correlation of  X  and  Y3 :  0.10031037591820145
correlation of  X  and  Y5 :  0.1003103759182018
```

TODO#10

1) เราต้องการให้ $E[P(Fail|t)] = E\left[\frac{0.97}{2250}(t-15)^2 + 0.001\right]$ มีค่าน้อยที่สุด

โดย $E\left[\frac{0.97}{2250}(t-15)^2 + 0.001\right] = \frac{0.97}{2250}E[(t-15)^2] + 0.001$

$$= \frac{0.97}{2250}\int_{-\infty}^{\infty}(t-15)^2 p(t)dt + 0.001$$

$$= \frac{0.97}{2250}\int_{\mu-1}^{\mu+1}\frac{(t-15)^2}{(\mu+1)-(\mu-1)}dt + 0.001$$

$$= \frac{0.97}{2250x2}\left(\frac{(\mu-14)^3-(\mu-16)^3}{3}\right) + 0.001$$

ทำการ differentiate เพื่อหาค่าต่ำสุดของ $E[P(Fail|t)]$ จะได้

$$0 = 3(\mu-14)^2 - 3(\mu-16)^2$$
$$0 = \big((\mu-14)-(\mu-16)\big)\big((\mu-14)+(\mu-16)\big)$$
$$0 = 2\mu - 30$$
$$\mu = 15$$

ดังนั้นจึงสรุปได้ว่า ควรตั้งค่าแอร์ที่ 15 องศา

2) นำค่า μ ที่ได้จากข้อ 1 มาแทนใน

$$E[P(Fail|t)] = \frac{0.97}{2250x2}\left(\frac{(\mu-14)^3-(\mu-16)^3}{3}\right) + 0.001$$

$$= \frac{0.97}{2250x2}\left(\frac{(15-14)^3-(15-16)^3}{3}\right) + 0.001$$

$$= 0.0011437037$$

ดังนั้น probability of failure at the temperature used in part 1 คือ 0.0011437037

3) ให้ p แทนความน่าจะเป็นที่ disk ทั้งหมดจะพัง

และ n แทนจำนวนครั้งที่ส่ง request

จาก $$P(Fail > 1) < 0.01\%$$

$$1 - P(Fail \leq 1) < 0.01\%$$
$$1 - {}_0^nCp^0(1-p)^n - {}_1^nCp^1(1-p)^{n-1} < 0.01\%$$
$$1 - \frac{0.01}{100} < (1-p)^n + np(1-p)^{n-1}$$
$$0.9999 < (1-p)^n + np(1-p)^{n-1}$$

พิจารณา ใช้ 1 disk : จะได้ $n = 10000, p = 0.0011437037$ แทนค่าลงในอสมการ

พบว่า $0.9999 < 0.00013344101517483 68$ ซึ่งไม่เป็นจริง

พิจารณา ใช้ 2 disks : จะได้ $n = 10000, p = 0.0011437037^2$

แทนค่าลงในอสมการพบว่า $0.9999 < 0.9999151999167454$ ซึ่งเป็นจริง

ดังนั้นจึงควรใช้อย่างน้อย 2 disks

TODO#11

1) มีทั้งหมด 3 คู่ ได้แก่ a-b, b-c, b-d

เนื่องจาก covariance matrix ระหว่าง a-b คือ $\begin{bmatrix} 10x10^{-3} & 0 \\ 0 & 3x10^{-3} \end{bmatrix}$ ซึ่ง diagonal

นั่นคือ a กับ b independent กัน

เนื่องจาก covariance matrix ระหว่าง b-c คือ $\begin{bmatrix} 3x10^{-3} & 0 \\ 0 & 12x10^{-3} \end{bmatrix}$ ซึ่ง diagonal

นั่นคือ b กับ c independent กัน

เนื่องจาก covariance matrix ระหว่าง b-d คือ $\begin{bmatrix} 3x10^{-3} & 0 \\ 0 & 15x10^{-3} \end{bmatrix}$ ซึ่ง diagonal

นั่นคือ b กับ d independent กัน

2)

```
# TODO#11.2
def expected_return(n):
  sampling_size = 10000
  expected_value = [0, 0, 0, 0]
  for i in range(sampling_size): # simulate 10000 returns
    return_value = day0
    rits = multivariate_normal.rvs(mean=miu, cov=cov_matrix, size = n)
    rits_product = np.prod(rits, axis=0)
    return_value *= rits_product
    expected_value += (return_value-day0)/sampling_size
  return expected_value

result_30days = expected_return(30)
print("Expected return of coin a after 30 days : ", result_30days[0])
print("Expected return of coin b after 30 days : ", result_30days[1])
print("Expected return of coin c after 30 days : ", result_30days[2])
print("Expected return of coin d after 30 days : ", result_30days[3])
print()
result_180days = expected_return(180)
print("Expected return of coin a after 180 days : ", result_180days[0])
print("Expected return of coin b after 180 days : ", result_180days[1])
print("Expected return of coin c after 180 days : ", result_180days[2])
print("Expected return of coin d after 180 days : ", result_180days[3])
```

```
Expected return of coin a after 30 days :  0.9181916176900436
Expected return of coin b after 30 days :  0.6321683613423271
Expected return of coin c after 30 days :  1.293898138865285
Expected return of coin d after 30 days :  1.3555383458431087

Expected return of coin a after 180 days :  6.561157161892945
Expected return of coin b after 180 days :  4.464782919693632
Expected return of coin c after 180 days :  10.818915439930684
Expected return of coin d after 180 days :  9.401930978218006
```

3) ทำการหา probability ของแต่ละเหรียญตาม code ด้านล่าง

```
# TODO#11.3
def profit_probability(n):
  sampling_size = 10000
  count_profit = np.array([0, 0, 0, 0])
  for i in range(sampling_size): # simulate 10000 returns
    rits = multivariate_normal.rvs(mean=miu, cov=cov_matrix, size = n)
    rits_product = np.prod(rits, axis=0)
    for j in range(4):
      if rits_product[j] > 1:
        count_profit[j] += 1
  return count_profit/sampling_size

days = 100
profit_prob = profit_probability(days)
print("a :", profit_prob[0])
print("b :", profit_prob[1])
print("c :", profit_prob[2])
print("d :", profit_prob[3])
```

```
a : 0.4179
b : 0.5447
c : 0.4348
d : 0.3838
```

ซึ่งพบว่า coin b ความน่าจะเป็นที่จะได้กำไรมากกว่า coin อื่น ๆ

ทำการหา variance ของแต่ละเหรียญตาม code ด้านล่าง

```
# TODO#11.3 (cont.)
def variance(n):
  sampling_size = 10000
  expected_value = [0, 0, 0, 0]
  sqrt_expected_value = [0, 0, 0, 0]
  for i in range(sampling_size): # simulate 10000 returns
    return_value = day0
    rits = multivariate_normal.rvs(mean=miu, cov=cov_matrix, size = n)
    rits_product = np.prod(rits, axis=0)
    return_value *= rits_product
    expected_value += (return_value-day0)/sampling_size
    sqrt_expected_value += (return_value-day0)**2/sampling_size
  return sqrt_expected_value-expected_value**2

var = variance(days)
print("a :", var[0])
print("b :", var[1])
print("c :", var[2])
print("d :", var[3])
```

```
a : 307.4776729433312
b : 48.97578082427312
c : 475.63944622465857
d : 751.5395229509239
```

ซึ่งพบว่า coin b มี variance น้อยกว่า coin อื่น ๆ อย่างเห็นได้ชัด

4) เนื่องจาก μ ของ $r_{i,t}$ ของแต่ละ coin มีค่าประมาณ 1 และมีการกระจายแบบ normal distribution ทำให้เมื่อคำนวณ rate ทั้งหมดออกมาแล้วทำให้มีความน่าจะเป็นที่ rate ของทั้ง n วันมีค่ามากกว่า 1 (กำไร) และ ความน่าจะเป็นที่ rate ของทั้ง n วันมีค่าน้อยกว่า 1 (ขาดทุน) มีค่าเท่า ๆ กัน นั่นคือ แต่ละเหรียญจะมีความน่าจะเป็นที่จะได้กำไรประมาณ 50%
5) ทำการคำนวณค่า Expected[return] , Variance[return] ,Probability of having profit ได้ตามภาพข้างล่าง

```
STRATEGY # 1
T = 30 :
        Expected[return] : 0.989
        Variance[return] : 41.045
        Probability of having profit : 0.46
T = 180 :
        Expected[return] : 7.905
        Variance[return] : 1374.388
        Probability of having profit : 0.396

STRATEGY # 2
T = 30 :
        Expected[return] : 0.591
        Variance[return] : 10.556
        Probability of having profit : 0.518
T = 180 :
        Expected[return] : 4.389
        Variance[return] : 146.559
        Probability of having profit : 0.546

STRATEGY # 3
T = 30 :
        Expected[return] : 1.382
        Variance[return] : 54.608
        Probability of having profit : 0.469
T = 180 :
        Expected[return] : 11.36
        Variance[return] : 3469.786
        Probability of having profit : 0.408
```

```
STRATEGY # 4
T = 30 :
        Expected[return] : 1.275
        Variance[return] : 71.382
        Probability of having profit : 0.442
T = 180 :
        Expected[return] : 10.762
        Variance[return] : 8016.166
        Probability of having profit : 0.342

STRATEGY # 5
T = 30 :
        Expected[return] : 0.79
        Variance[return] : 25.801
        Probability of having profit : 0.489
T = 180 :
        Expected[return] : 6.147
        Variance[return] : 760.473
        Probability of having profit : 0.471
```

```
STRATEGY # 6
T = 30 :
        Expected[return] : 1.185
        Variance[return] : 47.827
        Probability of having profit : 0.464
T = 180 :
        Expected[return] : 9.632
        Variance[return] : 2422.087
        Probability of having profit : 0.402

STRATEGY # 7
T = 30 :
        Expected[return] : 1.132
        Variance[return] : 56.214
        Probability of having profit : 0.451
T = 180 :
        Expected[return] : 9.333
        Variance[return] : 4695.277
        Probability of having profit : 0.369
```

T = 30

Strategy	Buy a	Buy b	Buy c	Buy d	Expected[return]	Variance[return]	Probability of having profit
1	100%	0%	0%	0%	0.989	41.045	0.46
2	0%	100%	0%	0%	0.591	10.556	0.518
3	0%	0%	100%	0%	1.382	54.608	0.469
4	0%	0%	0%	100%	1.275	71.382	0.442
5	50%	50%	0%	0%	0.79	25.801	0.489
6	50%	0%	50%	0%	1.185	47.827	0.464
7	50%	0%	0%	50%	1.132	56.214	0.451

T = 180

Strategy	Buy a	Buy b	Buy c	Buy d	Expected[return]	Variance[return]	Probability of having profit
1	100%	0%	0%	0%	7.905	1374.388	0.396
2	0%	100%	0%	0%	4.389	146.559	0.546
3	0%	0%	100%	0%	11.36	3469.786	0.408
4	0%	0%	0%	100%	10.762	8016.166	0.342
5	50%	50%	0%	0%	6.147	760.473	0.471
6	50%	0%	50%	0%	9.632	2422.087	0.402
7	50%	0%	0%	50%	9.333	4695.277	0.369

6) พิจารณาจากทั้ง T = 30 และ 180 จะเห็นว่า strategy 3 ให้ expected value ที่มีค่าสูงที่สุดทั้ง 2 ตาราง จึงสรุปว่า strategy 3 ให้ return สูงที่สุด

7) พิจารณาจากทั้ง T = 30 และ 180 จะเห็นว่า strategy 2 มี Probability of having profit สูงที่สุดทั้ง 2 ตาราง นอกจากนี้ยังมีค่า Variance[return] ที่ต่ำอย่างเห็นได้ชัด ทำให้ค่า return ที่คาดการณ์ไว้นั้นจะไม่คลาดเคลื่อนมาก ด้วยทั้งสองอย่างที่กล่าวมาจึงทำให้สรุปได้ว่า strategy 2 ปลอดภัยที่สุด

8)

```python
# TODO#11.8
def cov(n):
  sampling_size = 10000
  r = [[], [], [], []]
  for i in range(sampling_size):
    rits = multivariate_normal.rvs(mean=miu, cov=cov_matrix, size = n)
    rits_product = np.prod(rits, axis=0)
    for j in range(4):
      r[j] += [rits_product[j]]
  return r

coin_name = ['a', 'b', 'c', 'd']
coin30 = cov(30)
coin180 = cov(180)

print("T = 30")
for i in range(4):
  for j in range(i+1, 4):
    print("Covariance of", coin_name[i], "and", coin_name[j],
          ":", round(np.cov(coin30[i], coin30[j])[0][1], 6))
print("\nT = 180")
for i in range(4):
  for j in range(i+1, 4):
    print("Covariance of", coin_name[i], "and", coin_name[j],
          ":", round(np.cov(coin180[i], coin180[j])[0][1], 6))
```

```
T = 30
Covariance of a and b : -0.000978
Covariance of a and c : 0.153294
Covariance of a and d : 0.208118
Covariance of b and c : -0.004009
Covariance of b and d : -0.001292
Covariance of c and d : 0.075911

T = 180
Covariance of a and b : -0.052952
Covariance of a and c : 4.453592
Covariance of a and d : 7.317359
Covariance of b and c : 0.082997
Covariance of b and d : 0.023325
Covariance of c and d : 2.127712
```

จากการคำนวณค่า covariance ของ coin คู่ต่าง ๆ พบว่า มีเพียง $r_a - r_c$ และ $r_a - r_d$ ที่มีค่า covariance สูงอย่างเห็นได้ชัด ซึ่งแสดงให้เห็นว่าปริมาณการเปลี่ยนแปลงของ $r_a - r_c$ และ $r_a - r_d$ มีการเปลี่ยนแปลงตามกันค่อนข้างมากเมื่อเทียบกับคู่อื่น ๆ ทำให้ถึงแม้ variance ที่ T = 30 จะไม่ได้มีค่าสูงมาก แต่เมื่อ T = 180 จะเห็นว่า Variance[return] มีค่าสูงขึ้นอย่างก้าวกระโดด

9) จากคำถามข้างต้น ทำให้ได้ว่า strategy ที่ดีควรที่จะต้องมีค่า Expected[return] และ Probability of having profit ที่สูง และ Variance[return] ที่น้อย โดยการเลือกให้ค่า variance ที่น้อยจะต้องเลือกสิ่งที่มีค่า correlation ต่ำ เพราะหากมีค่าที่สูงจะทำให้มีความสัมพันธ์กันมากและค่า variance จะสูงส่งผลให้เกิดความคลาดเคลื่อนสูง