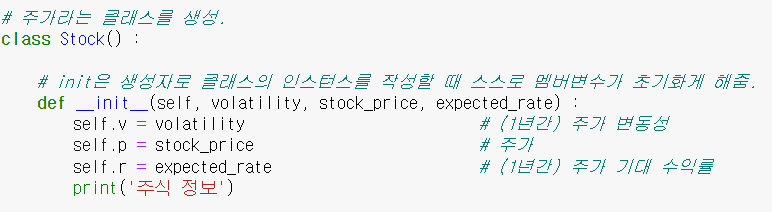
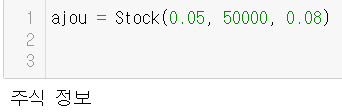
(1) 멤버변수(attribute)는 (1년간) 기대수익률, (1년간) 변동성, 현재주가를 반드시 포함

- 기대수익률 = expected\_rate, 변동성 = volatility, 현재 주가 = stock\_price로 정했다. 잘 작동 되는지 체크하려고 마지막에 print(‘주식 정보’) 기입.





(2) T 년 후의 주가를 리턴하는 멤버함수 구현 (T>0)

Z는 표준정규분포 N(0,1)을 따른다.

- 이 함수는 T, seed, size 의 인자들을 반드시 input parameter 로 포함해야하며 필요시 input parameter를 추가로 더 받을 수 있다.

- 여기서 seed 는 표준정규분포 z를 샘플링할 때 사용되는 시드값이며, size 는 리턴되는 주가의 크기(size or shape)에 해당 (만약 size = 10 으로 주어지면, T 년 후의 주가가 10개가 리턴되어야 하며, 이는 각각 다른 정규분포샘플에 의해서 생성되어야 함, 즉, 다른 값 10개가 리턴되어야 함)

🡺 표준정규분포 및 seed , e, sqrt 등등의 숫자와 관련된 클래스 및 하무들이 저장되어있는 라이브러리 ‘numpy’를 소환해야한다. (Import numpy as np)

🡺표준 정규분포에서 샘플들이 랜덤으로 나올 수 있도록 random이라는 라이브러리 또한 가져와야한다. (import random

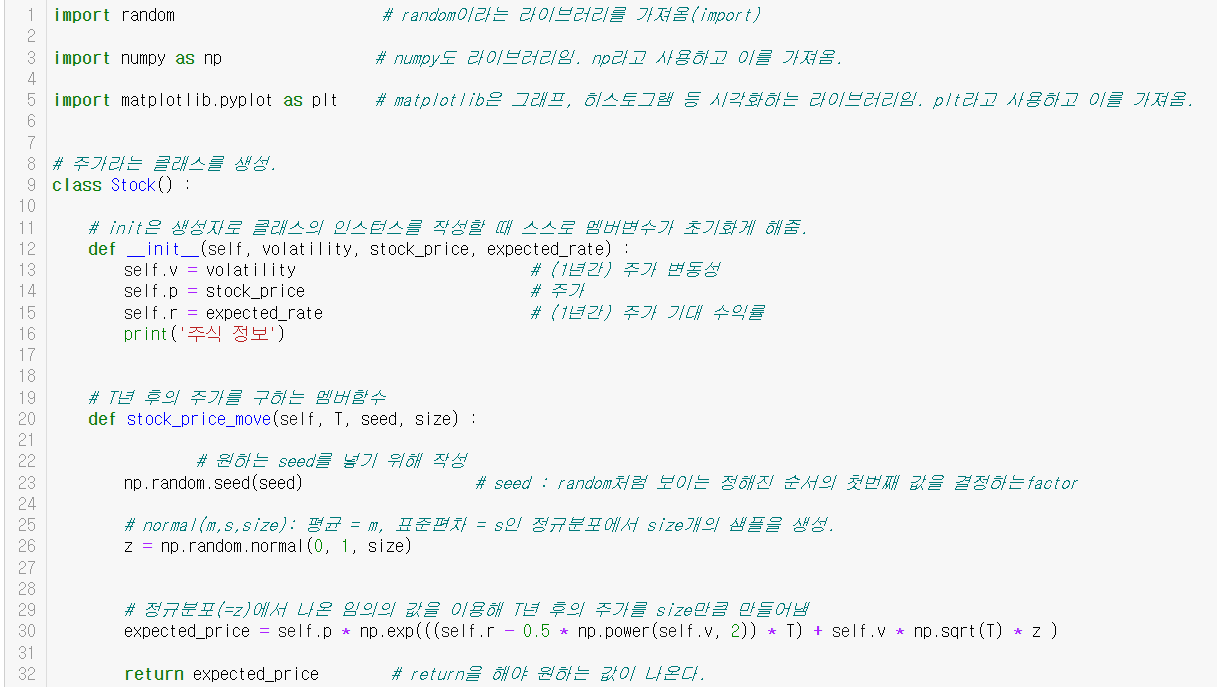
🡺 멤버함수의 이름은 stock\_price\_move이며 멤버함수의 경우 함수인자를 작성하기 이전에 무조건 self를 작성하고 함수인자를 작성해야한다.

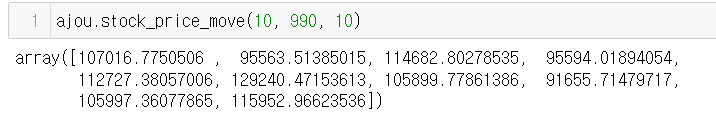
🡺z(표준정규분포)를 작성하기 이전에 표준정규분포의 시드값을 고정 시켜야지 원하는 시드를 넣었을 때 시드만 알고 있다면 고정된 값을 알 수 있다. 그래서 seed를 작성하는 코드를 쓰면 np.seed(seed)라는 코드를 작성하면 된다.

🡺 seed의 값을 정했으므로 이제 표준정규분포 z를 구하는 코드를 작성하면 된다. 표준정규분포의 코드는 normal(m,s,size): 평균 = m, 표준편차 = s , 표본 개수 = size 로 여기서 m과 s는 문제에서 각각 0과 1로 정해졌다.(표준정규분포이니까)

🡺T년후 주가의 공식을 작성하여 주가(expected\_price를 계산하면 된다.

🡺마지막으로 expected\_price가 나오도록 return 하면 된다.





🡺결과 값으로 10개의 10년후 주가표본이 각각 다르게 나옴을 알 수 있다. (seed=990) 🡺Seed를 z보다 먼저 작성해야 seed에 따른 정해진 표본이 나온다. z보다 늦게 작성 시 표본 고정이 안됨.

(3) T 년 후 주가 샘플 n 개에 대해 표본평균과 표본분산을 리턴하는 멤버함수 구현

- 이 함수 역시 (2)와 마찬가지로 T, seed, n 의 인자를 input paramter 로 포함

- 샘플 개수 n 이 커졌을 때, 표본평균과 표본분산이 각각 모평균과 모분산으로 수렴하는지 확인해보기

🡺 먼저 멤버함수로 sample\_mean\_variance(self, T, seed, n) 표본평균과 표본분산을 구하는 함수.

🡺 T년후의 표본주가를 이용해 표본평균과 표본분산을 구하기 때문에 위 멤버함수인 stock\_price\_move(T,seed,size=n)을 호출했다.

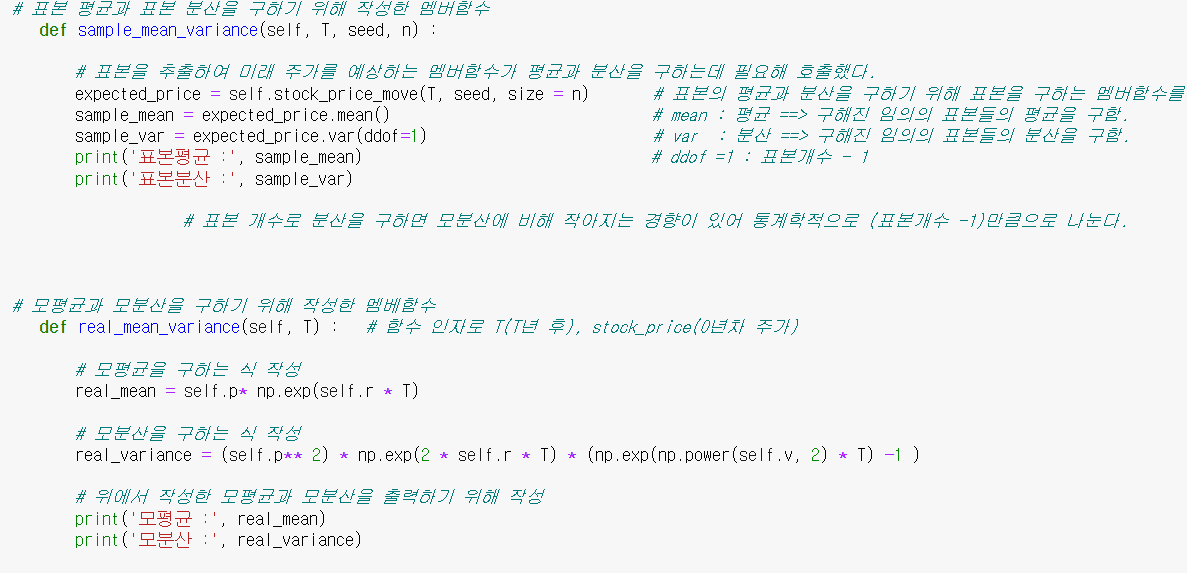
🡺 np에 존재하는 mean(평균), var(분산) 기능을 이용했다.

🡺 주의할 점은 표본 개수로 분산을 구하면 모분산에 비해 작아지는 경향이 있어 통계학적으로 (표본개수 -1)만큼으로 나눈다. 그래서 자유도(ddof=1)을 작성해야한다.

🡺 두번째 멈버함수로 real\_mean\_variance(self, ,T) 이론상 평균과 분산을 구하는 함수.

🡺 이론상의 평균과 분산은 표본을 사용하지 않고 공식이 존재한다.

🡺 공식 작성





🡺 size 즉, 표본이 커질수록 표본평균과 표본분산이 모평균과 모분산에 가까워짐을 알 수 있다.

(4) 그 외에 각자 멤버함수와 멤버변수를 추가로 삽입해서 각각 총 5개 이상이 되도록 구현하세요.

- 멤버변수는 기대수익률, 변동성, 현재주가 포함해서 5개 이상

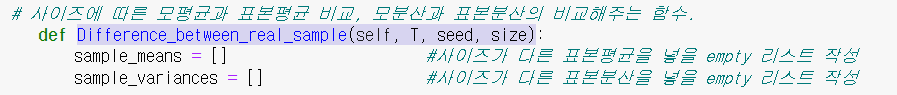
- 멤버함수는 (2)와 (3)을 포함해 5개 이상

1st.

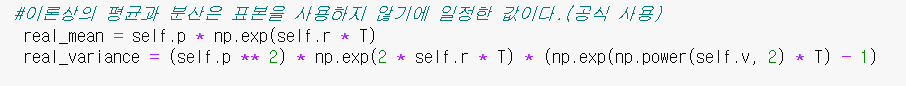
🡺 또다른 멤버함수로 Difference\_between\_real\_sample(self, T, seed, size)를 만들었다.

🡺 size에 따른 표본평균과 모평균 비교, 그리고 size에 따른 표본분산과 모분산의 비교를 그래프를 통해 확인 할 수 있는 함수이다.

🡺 size에 따른 표본평균과 표본분산을 리스트로 담을 수 있도록 empty list를 만들어줬다.



🡺 먼저 모평균과 모분산은 표본을 사용하지 않기에 일정하므로 공식을 작성했다.



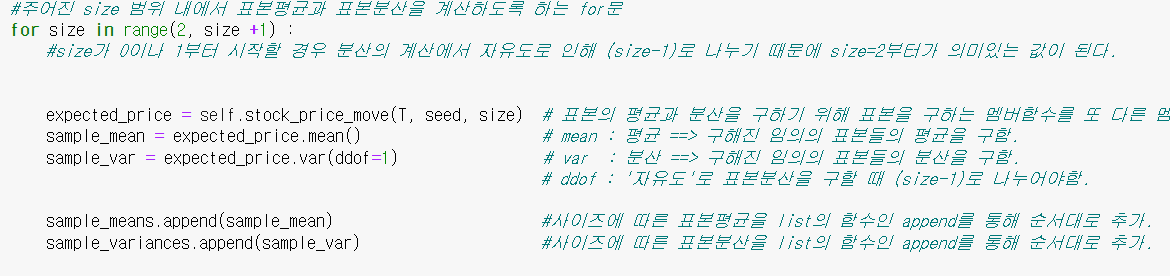
🡺 표본평균과 표본분산을 주어진size내에서 모두 계산하여 값을 구하도록 for문을 이용했다.

🡺 여기서 size의 구간은 0부터가 아닌 2부터인데 그 이유는 표본분산을 계산하려면 (size-1)로 나눠줘야 하는데 size=0과 1일 경우 0이나 -1로 나누어 계산되지 않기 때문에 size=2부터 시작해야한다.

🡺 for문에서 range(a,b)는 a부터 b-1까지이므로 함수인자로 지정한 size까지 하려면 range(2,size+1)로 작성해야한다.

🡺stock\_price\_move 멤버함수를 다시 호출해 expected\_price를 정의해줬으며 numpy의 mean()과 var(ddof=1)을 사용해 표본평균과 표본분산을 구해준다.

🡺 사이즈에 따른 표본평균과 표본분산을 리스트 함수 append를 이용해 순차적으로 추가해주도록 한다.



🡺 이후 모평균과 size에 따른 표본평균을 그래프로 나타내는 코드를 작성.

🡺 그래프를 나타내기 위해 matplotlib라는 라이브러리를 또다시 불러와야한다.(numpy처럼)

🡺 plt.plot(np.arange(2, size+1), sample\_means, 'b.', label='Sample Mean')

np.arange(2, size+1)은 x축 값의 범위를 말하는데, 2부터 size까지의 값을 생성하여 x축 값으로 사용한다.

sample\_means는 y축 값으로 사이즈에 따른 표본평균의 값들을 나타낸다. 즉, x축 값size에 대응하는 표본 평균의 리스트이다.

‘b.’은 해당 값을 blue 파란색으로 점으로 나타낸다는 뜻이고 ‘label’은 해당 그래프에 대한 레이블이다.

🡺 plt.plot([2, size], [real\_mean, real\_mean], 'g-', label='Real Mean')

[2,size]는 x축의 값의 범위를 말하며 2부터 size까지의 값을 생성한다.

[real\_mean, real\_mean]은 모평균은 size가 바뀌어도 항상 일정한 값을 가지기 때문에 y 축 값의 범위로 모두 real\_mean을 작성했다.

‘g-’은 해당 값을 green초록색으로 점으로 나타낸다는 뜻이고 ‘label’은 해당 그래프에 대한 레이블이다.

🡺 그래프 격자는 plt.grid(), lable을 작성했다면 항상 plt.legend를 작성해야지 그래프의 레이블이 보인다. 이외 타이틀과 x라벨 y라벨 작성.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

🡺이후 모분산과 size에 따른 표본분산을 그래프로 나타내는 코드를 작성.

🡺 모평균과 표본평균을 구하 코드와 same.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

<전체>

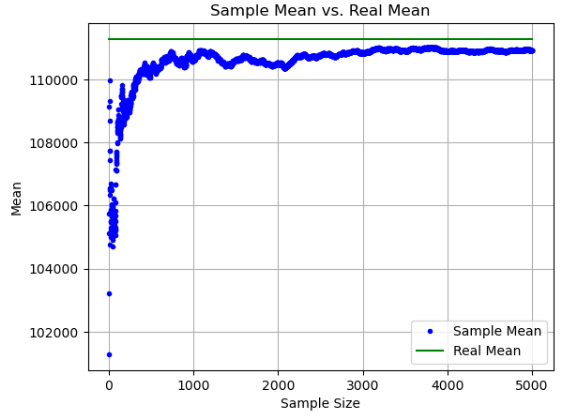
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명



텍스트, 라인, 그래프, 도표이(가) 표시된 사진

자동 생성된 설명

🡺 사이즈가 커질수록 표본평균과 표본분산이 모평균과 모분산에 각각 수렴하는 걸 볼 수 있다.

2nd.

표본을 이용한 시간T에 따른 주가변동과 모평균을 이용한 시간 T에 따른 주가 변동 비교함수를 했다.

주가의 변동을 그래프로 그리기 이전에 해가 지날수록 변동된 표본주가 평균을 구하는 멤버 함수를 코드했다.

🡺 for문을 이용해 T(년)가 0부터 T까지 각각의 표본평균을 모두 구하도록 만들었다.

🡺 T년동안의 예측 주가를 모두 저장하기 위해 멤버변수 expected\_stock\_price\_history=[] empty상태로 지정했다.

🡺 표본의 평균을 구하기 위해 표본주가를 구하는 멤버함수를 호출시켰다.

🡺 np에 존재하는 mean(평균)기능을 이용해 표본평균들을 구했다.

🡺 마지막으로 구해진 표본평균들을 (예측주가) history에 저장하기 위해 append를 이용해 순차적으로 추가해주도록 했다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

T에 따른 실제 주가를 구하기 위해서도 for문을 이용했다.

🡺 T년동안의 예측 주가를 모두 저장하기 위해 멤버변수 real\_stock\_price\_history=[] empty상태로 지정했다.

🡺 range를 이용 T년까지이므로 T+1을 작성.

🡺 이론상의 평균 공식을 작성하고 마찬가지로 구해진 모든 모평균을 history에 저장하기 위해 append를 사용해 순차적으로 추가해줬다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

마지막 단계로 expected\_stock\_price\_history=[], real\_stock\_price\_history=[]를 합쳐 그래프로 그리는 멤버함수를 작성했다.

plt.plot(self.expected\_stock\_price\_history, 'b-', label = 'expected\_stock\_price')

🡺 예측 주가를 파란색 선으로 나타내고 label도 지정했다.

plt.plot(self.expected\_stock\_price\_history, 'r.')

🡺 빨간색 점까지 추가해줬다.

plt.plot(self.real\_stock\_price\_history, 'g-',label = 'real\_stock\_price')

🡺 마찬가지로 실제 주가를 초록색 선으로 나타내고 label도 지정했다.

plt.plot(self.real\_stock\_price\_history, 'r.')

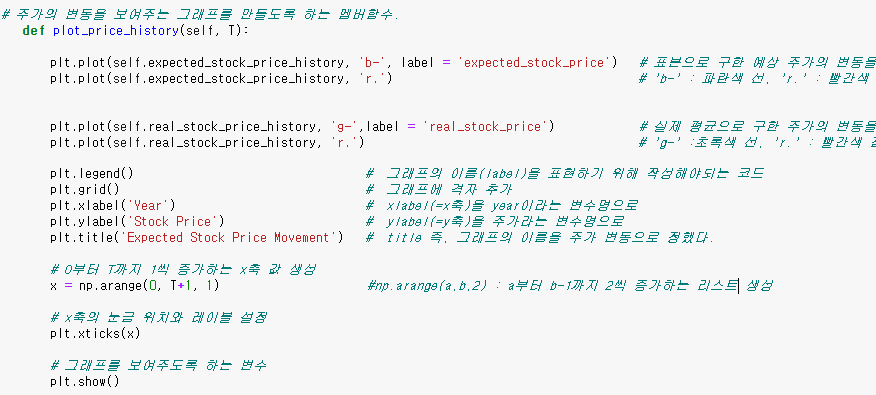
🡺 빨간색 점까지 추가해줬다.

레이블을 사용하려면 무조건 plt.legend()를 작성해준다.

더나아가 그래프의 x축 명, y축 명, 그래프 제목, 격자도 해주었다.

🡺 x축의 값이 년 수로(T) 1씩 증가하도록 만들고 싶어 임의의 x를 np.arange를 이용해 1씩 증가하고 0부터 T년까지 나타내도록 지정한 다음

plt.xticks()를 이용해 x축의 눈금 위치와 레이블을 설정하게 했다.



텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명

텍스트, 라인, 그래프, 도표이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명

텍스트, 라인, 스크린샷, 그래프이(가) 표시된 사진

자동 생성된 설명

🡺 표본이 커질수록 가격이 거의 같아진다.