Assignment: (Return) Predictability 1

The stock follows the GBM as where μ=0.05, σ=0.2. And, the risk free rate is 3% (annualized continuously compounded yield). S\_0(today’s price) is 100. We consider T=2. Let t denote the first time that the stock price hits 110. **Find the value of the derivative** which pays the average price over [0,t] at t, the first moment that the stock price hits 110. If the stock price does not hit 110 till T=2, you receive the average price over [0,2] at T=2.

Source code (.ipynb 파일도 함께 제출했음)

import numpy as np

from IPython.core.interactiveshell import InteractiveShell

InteractiveShell.ast\_node\_interactivity = "all" # 한 실행칸에 프린트 여러개 해도 다 출력시키도록 하는 코드.

import matplotlib.pyplot as plt

def Solution(mu, sigma, r, s0, T, B, num\_dt, num\_sim):

# Generating stock paths

dt = T/num\_dt

Initial\_log\_stock\_price = np.log(s0) \* np.ones((1, num\_sim))

binomial = -1 + 2 \* np.random.binomial(1, 0.5, (num\_dt, num\_sim))

# normal = np.random.randn(num\_dt, num\_sim)

dw = binomial \* np.sqrt(dt)

# dw = normal \* np.sqrt(dt)

d\_logS = (r-0.5\*(sigma\*\*2)) \* dt \* np.ones((num\_dt, num\_sim)) + dw \* sigma

tmp = np.concatenate((Initial\_log\_stock\_price, d\_logS), axis=0)

log\_stock\_price = np.cumsum(tmp, axis = 0)

stock\_price = np.exp(log\_stock\_price)

# Plotting Stock paths - time cumsuming..

# plt.title("Stock paths")

# plt.plot(stock\_price)

# plt.show()

# search the hitting point

hit\_idxs = []

for j in range(len(stock\_price[0])):

print(j)

for i in range(len(stock\_price)):

if(stock\_price[i][j] >= B or i == len(stock\_price)-1):

hit\_idxs.append(i)

break

# pricing

pv\_payoffs = []

for j in range(len(stock\_price[0])):

t = hit\_idxs[j]

pv\_payoff = stock\_price[:t+1,j].mean() \* np.exp(-r \* t \* dt)

pv\_payoffs.append(pv\_payoff)

return np.array(pv\_payoffs).mean()