**prj1 - 202211360 컴퓨터공학부 장서현**

확률과 통계 0435분반 - 이향원 교수님

(a)

확률변수 Xi는 원 C안에 떨어지면 1, 아니면 0이기 때문에 원 C안에 떨어지는 확률은 원의 면적과 비례합니다. 따라서, P(Xi = 1) = π/4, P(Xi = 0) = 1 - π/4 입니다.

(b)

E(Xi) = 1 \* P(Xi = 1) + 0 \* P(Xi = 0) = 1 \* (π/4) + 0 \* (1 - π/4) = π/4

(c)

Sn = X1 + ... + Xn, 그리고 Xi 또한 베르누이 분포입니다. 따라서, Var(Xi) = (π/4) \* (1 - π/4)입니다.

평균을 먼저 구하면, E(Sn) = E(X1 + ... + Xn) = E(X1) + ... + E(Xn) = n \* (π/4) = π/4 \* n 입니다. 이에 따라, E(Sn/n) = E(Sn) / n = (π/4) \* n / n = π/4 입니다.

분산은 Var(Sn) = Var(X1 + ... + Xn) = Var(X1) + ... + Var(Xn) = n \* Var(Xi) = n \* π/4 \* (1 - π/4) 이고, 따라서 Var(Sn/n) = Var(Sn) / n^2 = (π/4) \* (1 - π/4) / n 입니다.

n이 무한대로 커질수록 Sn/n의 분산은 0에 가까워집니다. 분산이 감소한다는 뜻은, 평균에 가까운 값이 많아진다는 것을 의미하고, 따라서 분산이 0에 가까워진다는 것은 기댓값, 즉 평균의 실제 정확도가 높아진다는 것을 의미합니다.

(d)

n이 충분히 크다면 기댓값의 정확도가 상당히 높으므로, 기댓값을 이용해서 π를 추정할 수 있습니다. E(Sn/n) = E(Sn) / n = (π/4) \* n / n = π/4 이기 때문에, n을 충분히 큰 값으로 설정하고, n번만큼 0과 1 사이의 x와 y를 무작위로 선택합니다. 만약, 해당 좌표가 원 C안의 값이면 1이라고 하고, 아니면 0으로 설정한 후 n번 수행한 값들을 모두 더해줍니다. 해당 합을 n으로 나누고, 4를 곱해주면 π를 추정할 수 있습니다.

(e)

import random

*def* estimate\_pi(*n*):

num\_points\_circle = 0

for i in range(n):

x = random.uniform(0, 1)

y = random.uniform(0, 1)

checkCircleC = (x - 1/2)\*\*2 + (y - 1/2)\*\*2

if checkCircleC <= 1/4:

num\_points\_circle += 1

return 4 \* num\_points\_circle / n

print(estimate\_pi(10000000))

(f)

네. 수렴합니다.

ns = [2, 3, 4, 5, 6, 7]

estimates1 = [estimate\_pi(10\*\*n) for n in ns]

estimates2 = [estimate\_pi(10\*\*n) for n in ns]

estimates3 = [estimate\_pi(10\*\*n) for n in ns]

estimates4 = [estimate\_pi(10\*\*n) for n in ns]

estimates5 = [estimate\_pi(10\*\*n) for n in ns]

plt.plot(ns, estimates1, 'r--', ns, estimates2, 'y--', ns, estimates3, 'g--', ns, estimates4, 'b--', ns, estimates5, 'k--')

plt.xlabel('10^x')

plt.ylabel('estimates of pi')

plt.axhline(estimates1[5], 0, 1, *color* = 'lightgray', *linestyle* = '--')

plt.show()

해당 코드를 통해 도출된 그래프는 다음과 같습니다.

차트이(가) 표시된 사진

자동 생성된 설명

회색의 수평선은 estimates1[5]인 값인데, 이는 3.14에 수렴해있음을 확인할 수 있습니다.