**##Estimating π Using Monte Carlo Sampling-Use geometric probability by simulating random points in a square.**

import random

import matplotlib.pyplot as plt

# Number of points to simulate

N = 10000

inside\_circle = 0

x\_in, y\_in = [], []

x\_out, y\_out = [], []

for \_ in range(N):

x, y = random.random(), random.random()

if x\*\*2 + y\*\*2 <= 1:

inside\_circle += 1

x\_in.append(x)

y\_in.append(y)

else:

x\_out.append(x)

y\_out.append(y)

pi\_estimate = 4 \* inside\_circle / N

print(f"Estimated Pi: {pi\_estimate:.5f}")

# Visualization

plt.figure(figsize=(6,6))

plt.scatter(x\_in, y\_in, color='blue', s=1, label='Inside Circle')

plt.scatter(x\_out, y\_out, color='red', s=1, label='Outside Circle')

plt.legend()

plt.title(f"Monte Carlo Pi Estimate\nEstimated Pi ≈ {pi\_estimate:.5f}")

plt.xlabel("x")

plt.ylabel("y")

plt.axis("equal")

plt.show()

**#####Sampling to Approximate the Mean of a Normal Distribution-**Generate repeated samples from a normal distribution and observe the mean convergence.

import numpy as np

import matplotlib.pyplot as plt

# Parameters

mu, sigma = 0, 1

num\_samples = 1000

sample\_size = 30

means = []

for \_ in range(num\_samples):

sample = np.random.normal(mu, sigma, sample\_size)

means.append(np.mean(sample))

# Plotting the distribution of sample means

plt.hist(means, bins=30, color='skyblue', edgecolor='black', density=True)

plt.title("Distribution of Sample Means (CLT in action)")

plt.xlabel("Sample Mean")

plt.ylabel("Density")

plt.axvline(np.mean(means), color='red', linestyle='dashed', label=f"Mean ≈ {np.mean(means):.2f}")

plt.legend()

plt.show()

#####**Repeated Sampling to Estimate Probability-**Estimate the probability of getting a sum > 10 when rolling two dice.

def roll\_dice():

return random.randint(1, 6) + random.randint(1, 6)

num\_trials = 100000

success = 0

for \_ in range(num\_trials):

if roll\_dice() > 10:

success += 1

estimated\_prob = success / num\_trials

print(f"Estimated Probability of sum > 10 when rolling two dice: {estimated\_prob:.5f}")