Contents

1 Exercise 3.25

From a box containing 4 dimes and 2 nickels, 3 coins are selected at random without replacement. Find the probability distribution for the total T of the 3 coins. Express the probability distribution graphically as a probability histogram.

To make this interesting, I want to take a traditional statistical approach using combinations and probabilities to gather the probability distribution as well as use a Monte Carlo simulation.

1.1 Part 1: Calculating the probability distribution

I calculated the set by hand:

$S = DDD, DDN, DNN$

I wanted to confirm this mathematically (spoiler alert: I was not).

1.1.1 Attempt 1

I can have 3 at a time, and for me $D_1 = D_2 = D_n$. There are $\frac{6!}{(6-3}!$ permutations but only if D_1 is counted differently than D_2.

Using $\frac{n!}{n_1!n_2!n_k!} = \frac{6!}{2!4!} = 15$ which obviously doesn't make sense. Setting $n = 3$ yields a result less than 1.

1.1.2 Attempt 2

I need combinations, not permutations.

I should take the sum of the combination of choosing d dimes where d goes from 0 to n.

$\binom{3}{1}\binom{3}{2} + \binom{3}{2}\binom{3}{1} + \binom{3}{3}\binom{3}{2}$

$9 + 9 + 3$

This makes sense for probability, but not for counting.

1.1.3 Attempt 3

At this point, I decided to calculate the probability distribution by hand without trying to confirm my set mathematically. This is what I got:

$$P(0.20) = \frac{\binom{4}{1}\binom{2}{2}}{\binom{6}{3}} = \frac{1}{5}$$

$$P(0.25) = \frac{\binom{4}{2}\binom{2}{1}}{\binom{6}{3}} = \frac{3}{5}$$

$$P(0.30) = \frac{\binom{4}{3}\binom{2}{0}}{\binom{6}{3}} = \frac{1}{5}$$

1.2 Part 2: Monte Carlo Simulation

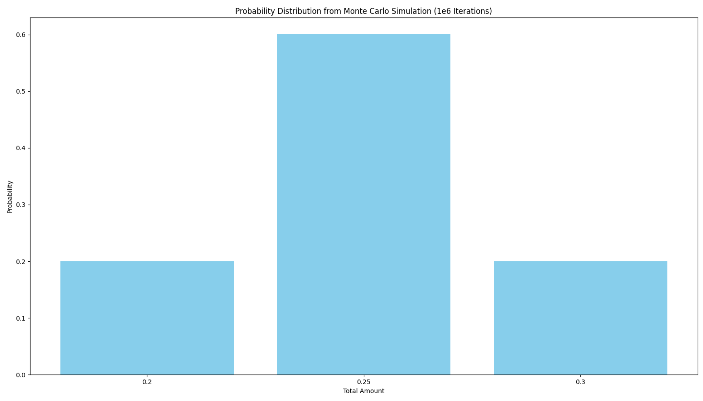
Instead of using a traditionally statistical approach, I will not attempt to approximate the probability distribution using a Monte Carlo simulation.

```
import matplotlib.pyplot as plt
from random import random as rand

COINS_DRAWN = 3
ITERS = 1_000_000

coin_sum_count = {}

def draw_coins():
    # 4 dimes, 2 nickels
    coins = [0.1, 0.1, 0.1, 0.1, 0.05, 0.05]
    drawn_coins = []
    for draw in range(COINS_DRAWN):
        # Generate index
        idx = int(rand() * len(coins))
        drawn_coins.append(coins[idx])
        coins.pop(idx)
    return f"{sum(drawn_coins):0.2}"

for iter in range(ITERS):
    result = draw_coins()
    coin_sum_count[result] = coin_sum_count.get(result, 0) + 1

coin_sum_count = dict(sorted(coin_sum_count.items()))
coin_sum_probabilities = {key: value/ITERS for key, value in coin_sum_count.items()}
print(coin_sum_probabilities)
```

```
plt.bar(list(coin_sum_probabilities.keys()), coin_sum_probabilities.values(), color="sk
plt.title("Probability Distribution from Monte Carlo Simulation (1e6 Iterations)")
plt.xlabel("Total Amount")
plt.ylabel("Probability")
plt.show()
```

1.3 Part 3: Results

The Monte Carlo simulation performed just as expected.


Probability Distribution from Monte Carlo Simulation (1e6 Iterations)
Probability
Total Amount