11.16.3.3.1

EE24BTECH11064 - Harshil Rathan

Question:
A die is thrown, find the probability of following events :

i) A prime number will appear

Theoretical Solution:
The sample space S of a fair six-sided die is

$$S = 1, 2, 3, 4, 5, 6 \tag{0.1}$$

The prime numbers in the Sample space are

$$A = 2, 3, 5 \tag{0.2}$$

Thus, the number of favorable outcomes = 3

$$|S| = 6 \tag{0.3}$$

$$|A| = 3 \tag{0.4}$$

The probability of getting a prime number when a fair die is rolled

$$P(A) = \frac{|A|}{|S|} \tag{0.5}$$

on substituing 0.3, 0.4

$$P(A) = \frac{1}{2} \tag{0.6}$$

Introduction

The task involves simulating the roll of a single die using a C program, compiling it into a shared object (.so) file, and then using Python to call this function. The Python code processes the results and generates a probability distribution plot for outcomes $2, 3, 5$

C Code Description

The C program generates random outcomes for rolling a single die, where the possible outcomes range from 1 to 6. Outcomes are categorized as follows:
The program uses the `rand()` function to generate random numbers and increments the respective indices of the results array based on the generated outcome.

Python Code Description

The Python code is responsible for:

1) Loading the shared object file generated from the C program using the `ctypes` library.
2) Calling the C function to roll the die for a specified number of trials (e.g., $10,00,000$).
3) Retrieving the results from the C function as an array.
4) Calculating the probabilities for each outcome using the formula:

$$P(\text{outcome}) = \frac{\text{frequency of outcome}}{\text{total trials}}$$

5) Plotting the probability distribution using `matplotlib`.

Graphical Output

The Python code generates a bar chart where:

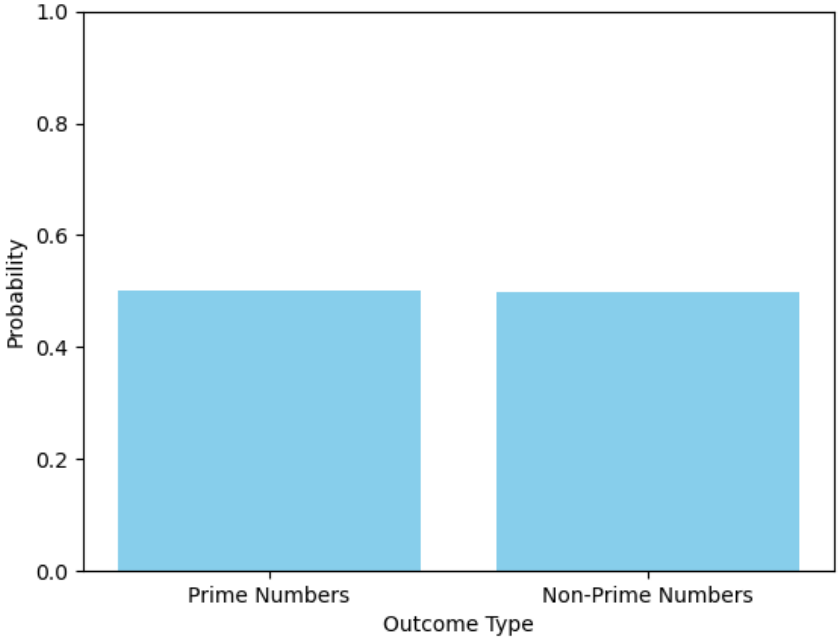
- The x-axis represents the outcomes: $2, 3, 5$.
- The y-axis represents the probabilities of each outcome, ranging from 0 to 1.
- Each bar height corresponds to the probability of the respective outcome.

Key Points

- Using the C program ensures efficient computation of outcomes for large numbers of trials.
- The shared object file facilitates seamless integration with Python, leveraging Python's powerful visualization capabilities.
- The probabilities provide a normalized representation of the frequency distribution, making it easier to interpret the results.

Conclusion

This task demonstrates the integration of C and Python for simulating and visualizing a probabilistic experiment. By combining the computational efficiency of C with the graphical capabilities of Python, we achieve an effective solution for analyzing and representing data. The code clearly shows that the probability of the given event is equal to **half**


1.0
0.8
0.6
0.4
0.2
0.0
Probability
Prime Numbers
Non-Prime Numbers
Outcome Type