AI 1104: Programming for AI

Assignment 1

Due on: 14 April 2024, 11:59 PM IST.

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Read each question carefully, and answer all parts. Write well-documented codes with contextual variable names. Submit a zip file containing the codes (.py or .ipynb) and plots (.png) of all the questions.
Name the zip file as `AI1104_assignment_1_<your roll number>.zip`.

1. (Uniform Distribution $\longleftrightarrow$ Geometric Distribution)

   A random variable U taking values in the interval $[0, 1]$ is said to be distributed uniformly if its cumulative distribution function (CDF) F_U has the form

   $$F_U(u) = \begin{cases} 0, & u < 0, \\ u, & 0 \leq u \leq 1, \\ 1, & u > 1. \end{cases} \tag{1}$$

   Likewise, a discrete random variable X taking values in $\{1, 2, \ldots\}$ is said to have Geometric distribution with parameter $p \in (0, 1]$ if its CDF F_X has the form

   $$F_X(x) = \begin{cases} 0, & x < 1, \\ 1 - (1-p)^{\lfloor x \rfloor}, & x \geq 1. \end{cases} \tag{2}$$

   where $\lfloor x \rfloor$ in (2) denotes the floor of x.

   Write a Python code to execute the following.

   (a) Use the `random.random()` method in Python to generate a sample $u \in [0, 1]$.
   Write a function named `uniform_to_geometric` that takes $u \in [0, 1]$ as input and returns

   $$X = \min\{x \in \mathbb{R} : F_X(x) \geq u\}, \tag{3}$$

   where F_X is as defined in (2) with $p = 0.3$.
   Using the above function, generate $100,000$ independent samples of X. Store the samples in an array named `geometric_samples_1`.
   Use `np.random.default_rng().geometric` to generate $100,000$ independent samples from a Geometric distribution with parameter $p = 0.3$. Store the obtained samples in an array named `geometric_samples_2`.
   Plot the empirical CDFs of the samples in `geometric_samples_1` and `geometric_samples_2`. You may use the `scipy` module to generate the empirical CDFs. Label the x and y axes appropriately, and provide a legend to the plot.

   (b) Use `np.random.default_rng().geometric` to generate a sample $x \in \mathbb{R}$ from Geometric distribution with parameter $p = 0.3$.
   Write a function named `geometric_to_uniform` that takes as input $x \in \mathbb{R}$ and returns $U = F_X(x)$, where F_X is as defined in (2) with $p = 0.3$.
   Using the above function, generate $100,000$ independent samples of U. Store these samples in an array named `uniform_samples_1`.
   Use `random.random()` to generate $100,000$ independent samples from the uniform distribution on $[0, 1]$. Store the samples in an array named `uniform_samples_2`.
   Plot the empirical CDFs of the samples in `uniform_samples_1` and `uniform_samples_2`.

2. (SNAKES AND LADDERS)

In this question, you will implement a single-player version of the SNAKES AND LADDERS game.

Consider a $n \times n$ square board, with a total of n^2 squares, in which the enumeration of squares on every odd-numbered row is from left to right, while that on every even-numbered row is from right to left. Figure 1 depicts the enumeration of squares on a prototypical 4×4 square board.

16	15	14	13
9	10	11	12
8	7	6	5
1	2	3	4

Figure 1: A prototypical 4×4 board with enumeration of squares.

Starting from square 1, the goal is to reach square n^2 through a sequence of fair dice throws. At each time step, the player moves ahead as many squares on the board as the outcome of the dice throw at that time step. During the game, if the player encounters the foot of a ladder at any square, the player simply climbs the ladder and reaches the square corresponding to the ladder head. In contrast, if the player encounters the head of a snake at any square, then the player simply slides down the snake and reaches the square corresponding to snake's tail. The game resumes from the new square.

If the player is in any square numbered i, where $n^2 - 6 \leq i \leq n^2 - 1$, and square i does not contain a snake's head, then the game finishes only if the dice outcome is $j = n^2 - i$. Else, if the dice outcome is any $k < j$ and it is feasible to move k squares further, the player does so. Else, the player stays at square i until it is feasible to move further or finish the game.

Write a Python code to execute the following.

(a) Ask the user to input a value of $n \geq 3$.

(b) Ask the user to enter the number of ladders desired. Subsequently, ask the user to input a collection of square numbers for the ladder feet and heads using commas and semicolons. For instance, if the user inputs

$$3, 6; 5, 10; 9, 14$$

as the ladder coordinates, then your code should interpret the above input as follows:

$$\begin{aligned} \text{Ladder } 1 &\longrightarrow \text{ foot: square } 3, \text{ head: square } 6, \\ \text{Ladder } 2 &\longrightarrow \text{ foot: square } 5, \text{ head: square } 10, \\ \text{Ladder } 3 &\longrightarrow \text{ foot: square } 9, \text{ head: square } 14. \end{aligned}$$

You may assume that the user is aware of the rule for enumeration of squares, and also aware that the first coordinate entered will be interpreted as the ladder's foot.

(c) Following the rule in (b) above, ask the user to input the number of snakes desired and the coordinates of its tails and heads. The first coordinate after each semicolon should be used as the tail position.

(d) Use the `random` module of Python to simulate the throwing of a fair dice. Run $100,000$ independent trials of the above game, and for each trial, record the number of dice throws that are required to complete the game in that trial. Print the mean and standard deviation of the recorded values.