Monte Carlo

Zadání

Metoda Monte Carlo představuje rodinu metod a filosofický přístup k modelování jevů, který využívá vzorkování prostoru (například prostor čísel na herní kostce, které mohou padnout) pomocí pseudonáhodného generátoru čísel. Jelikož se jedná spíše o filosofii řešení problému, tak využití je téměř neomezené. Na hodinách jste viděli několik aplikaci (optimalizace portfolia aktiv, řešení Monty Hall problému, integrace funkce, aj.). Nalezněte nějaký zajímavý problém, který nebyl na hodině řešen, a získejte o jeho řešení informace pomoci metody Monte Carlo. Můžete využít kódy ze sešitu z hodin, ale kontext úlohy se musí lišit.

Řešení

Pro realizaci této úlohy jsem si vybral hru v kostky mezi dvěma hráči. Na závěr po simulaci většího počtu her se pokusím vypočítat pravděpodobnost výhry jednotlivých hráčů. Výherní skóre je 50. Hra má jednoduchá pravidla. Ke skóre se přičítá hodnota, která padne na šestihranné kostce. Výjimkou je, pokud hráč hodí hodnotu 1 či hodnotu 6 vícekrát po sobě. V takovém případě se jeho skór vynuluje.

Pro simulaci hodu kostky jsem použil knihovnu **random**.

```
MonteCarlo.py > ...
1   import random
2
```

Tu používám ve funkci **roll_dice(),** kde si pomocí metody **randint()** vygeneruji číslo v rozsahu od 1 do 6.

```
def roll_dice():
    return random.randint(1, 6)
```

Poté jsem implementoval metodu **play_dice(),** která simuluje hru v kostky mezi dvěma hráči podle pravidel, která jsem popsal v úvodu. Simulace jedné hry probíhá, dokud jeden z hráčů nedosáhne výherního skóre. Na úvod funkce inicializuji skóre a aktuálního hráče, který začne. Pak už jen v cyklu **while** probíhá samotná hra, kde přičítám hráčům skóre a přepínám mezi nimi dle pravidel.

```python
def play_dice(target_score):
    player_scores = [0, 0]  # Inicializace skóre pro 2 hráče
    current_player = 0
    consecutive_six = 0

    while max(player_scores) < target_score:
        # Hod kostkou
        dice_roll = roll_dice()

       #Určení skóre dle hodu
        if dice_roll == 1:
            # Pokud hráč hodí jedničku, tak se jeho skóre resetuje a hraje další hráč
            player_scores[current_player] = 0
            current_player = 1 - current_player
        else:
            # Jinak se přičte ke skóre hráče hodnota kostky
            player_scores[current_player] += dice_roll

            if dice_roll == 6:
                consecutive_six += 1
                if consecutive_six == 2:
                    # Při hození hodnoty 6 vícekrát po sobě se skóre zresetuje a přepne se na dalšího hráče
                    player_scores[current_player] = 0
                    consecutive_six = 0
                    current_player = 1 - current_player
            else:
                consecutive_six = 0

    return current_player
```

Nastavím tedy cílové skóre do proměnné **target_score** a zadám počet simulací, které se mají provést. Také si připravím proměnné **wins_player** a **losses_player** pro postupné sčítání výher a proher obou hráčů. Poté v cyklu for, kterému nastavím range pomocí proměnné **num_simulations**, se volá funkce **play_dice().** Výsledek dané simulace se zapíše do proměnných o výhře a prohře hráčů. Na závěr, když proběhnou všechny simulace, se vypočítá pravděpodobnost výhry obou hráčů. Pravděpodobnost vypočítá jako počet **výher daného hráče / počet her** a uloží do proměnné **probability_player**. Poté se do konzole vypíše statistika o výhrách a prohrách obou hráčů a k tomu pravděpodobnost jejich výhry.

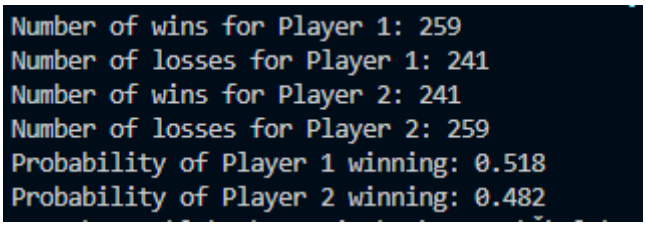
```python
# Cílové skóre hry
target_score = 50

# Simulace hry kostky pomocí metody Monte Carlo
num_simulations = 500
wins_player1 = 0
wins_player2 = 0
losses_player1 = 0
losses_player2 = 0

for i in range(num_simulations):
    winner = play_dice(target_score)
    if winner == 0:
        wins_player1 += 1
        losses_player2 += 1
    else:
        wins_player2 += 1
        losses_player1 += 1

# Výpočet pravděpodobnosti výhry pro jednotlivé hráče
probability_player1 = wins_player1 / num_simulations
probability_player2 = wins_player2 / num_simulations

print(f"Number of wins for Player 1: {wins_player1}")
print(f"Number of losses for Player 1: {losses_player1}")
print(f"Number of wins for Player 2: {wins_player2}")
print(f"Number of losses for Player 2: {losses_player2}")
print(f"Probability of Player 1 winning: {probability_player1}")
print(f"Probability of Player 2 winning: {probability_player2}")
```



Závěr

Povedlo se mi tedy v mém kódu nasimulovat větší množství her hry v kostky mezi dvěma hráči a následně pomocí metody Monte Carlo vypočítat pravděpodobnost výhry jednotlivých hráčů. Využití Monte Carlo je rozsáhlé a může být aplikováno v různých oblastech, včetně financí, fyziky, biologie, inženýrství a mnoha dalších. Pomocí této metody můžeme provádět analýzy rizika, optimalizaci rozhodování a získávat odhady pravděpodobnosti v různých scénářích.