Simulare Monte Carlo - Blackjack

1. Ce este algoritmul Monte Carlo?

Algoritmul Monte Carlo este o metodă de simulare stocastică bazată pe simulări aleatorii repetate, utilizată pentru a estima soluții pentru probleme complexe. În cazul jocului Blackjack, simulările determină probabilitatea ca jucătorul să câștige atunci când alege să tragă o carte ("hit") sau să stea ("stand"), în funcție de următorii parametri:

- **Scorul jucătorului** înainte de a lua o decizie.
- **Cărțile vizibile ale dealer-ului** (numite "upcard").
- **Acțiunea jucătorului** ("hit" sau "stand").

2. Cum funcționează algoritmul Monte Carlo în acest caz?

a. Inițializarea simulărilor

Sunt definite structuri de date care vor stoca numărul de câștiguri și numărul total de simulări pentru fiecare combinație posibilă de:

- **Sumă a cărților jucătorului**: între 2 și 21.
- **Upcard-ul dealer-ului**: între 1 și 10.
- **Acțiunea**: "hit" sau "stand".

b. Simularea unui joc de Blackjack (`simulate_blackjack`)

Implementare:https://github.com/dariadragomir/Blackjack_Monte_Carlo_Simulation/blob/main/blackjack_monte_carlo.py

Această funcție modelează un singur joc:

1. **Generarea mâinilor inițiale**:
   - Se selectează două cărți pentru jucător și două pentru dealer dintr-un pachet aleatoriu.
   - Se calculează scorurile inițiale pentru jucător și dealer.
2. **Determinarea acțiunii jucătorului**:
   - Jucătorul decide aleatoriu dacă face **"hit"** (mai trage o carte) sau **"stand"** (se oprește).
3. **Rezolvarea jocului**:
   - Dacă jucătorul face **"hit"**, trage o carte și verifică dacă a depășit 21 sau a ajuns la 21.
   - Dealer-ul joacă conform regulilor standard (trebuie să continue să tragă până ajunge la cel puțin 17).
4. **Determinarea câștigătorului**:
   - Dacă jucătorul depășește 21, pierde automat.

- Dacă dealer-ul depășește 21, jucătorul câștigă.
- În alte cazuri, câștigătorul este cel care are scorul mai mare.

c. Simularea pe scară largă (`monte_carlo_blackjack`)

Această funcție rulează simulările de Blackjack de un număr foarte mare de ori (`N_SIMULATIONS = 10**8`):

1. **Actualizarea statisticilor**:
   - După fiecare joc, rezultatul este înregistrat în `probabilites` și `number_simulations`, în funcție de scorul inițial, upcard-ul dealer-ului și acțiunea luată.
2. **Calcularea probabilităților**:
   - Pentru fiecare combinație posibilă de parametri, probabilitatea este calculată ca raportul dintre numărul de câștiguri și numărul total de simulări: P=Numărul de câștiguri/Numărul total de simulări

De exemplu:

```
Player's Score: 14
  Dealer Upcard: 1 | Hit Probability: 16.1% | Stand Probability: 8.3%
  Dealer Upcard: 2 | Hit Probability: 19.5% | Stand Probability: 11.9%
```

Metoda Monte Carlo utilizează un principiu de bază din teoria probabilităților și statistica inferențială: dacă avem o serie de observații independente ale unei variabile aleatoare, atunci media acestor observații converge în mod probabilistic către valoarea așteptată a variabilei aleatoare (2. Teorema Limită Centrală).

1. Fundamentul teoretic al metodei Monte Carlo

Presupunem că vrem să aproximăm probabilitatea unui eveniment P(A). Aceasta este valoarea teoretică pe care încercăm să o estimăm. Prin simulare, generăm un eșantion aleator de N observații, fiecare având aceeași probabilitate P(A) de succes. Fie variabila aleatoare Xi, care ia valoarea 1 dacă evenimentul A apare și 0 altfel. Observațiile sunt **independente și identic distribuite (i.i.d.)**.

Estimăm P(A) folosind media eșantionului:

$$\hat{P}(A) = \frac{1}{N} \sum_{i=1}^{N} X_i.$$

1. **Valoarea așteptată**:

$$\mathbb{E}[\hat{P}(A)] = P(A).$$

2. **Varianța**: $$\text{Var}(\hat{P}(A)) = \frac{\text{Var}(X_i)}{N} = \frac{P(A)(1-P(A))}{N}.$$

   Varianța scade invers proporțional cu N, ceea ce înseamnă că estimarea devine mai precisă pe măsură ce numărul de simulări crește.

2. Teorema Limită Centrală

Conform Teoremei Limita Centrala, pentru un număr mare de simulări (N mare), distribuția estimatorului $\hat{P}(A)$ devine aproximativ normală, indiferent de distribuția lui Xi.

Adică: $$\hat{P}(A) \sim \mathcal{N}\left(P(A), \frac{P(A)(1-P(A))}{N}\right).$$

3. Inegalitatea Cebâșev

O altă metodă de a cuantifica eroarea estimării folosește **Inegalitatea Cebâșev**, care oferă o limită pentru probabilitatea ca media eșantionului să se abată de la valoarea așteptată cu mai mult de o toleranță ϵ>0:

$$P(|\hat{P}(A) - P(A)| \geq \epsilon) \leq \frac{\text{Var}(\hat{P}(A))}{\epsilon^2}.$$

Aceasta înseamnă că probabilitatea ca estimarea să fie departe de valoarea reală P(A) scade proporțional cu creșterea numărului de simulări N. Cu alte cuvinte, creșterea N reduce incertitudinea în estimare.

4. Inegalitatea Chernoff-Hoeffding

$$P(|\hat{P}(A) - P(A)| \geq \epsilon) \leq 2\exp\left(-2N\epsilon^2\right).$$

Această inegalitate arată că probabilitatea unei deviații mari scade **exponențial** cu numărul de simulări N. Este mai precisă decât Cebâșev pentru variabile limitate, cum ar fi cazul lui Xi, care este binară (0 sau 1).

Inegalitatea Chernoff - Hoeffding:
Fie $X_1, \dots, X_n$ variabile aleatoare independente și identic distribuite, i.e. $a \leq x_i \leq b$, $\forall i = \overline{1,n}$.
Atunci $\forall \varepsilon > 0$, $P(\overline{S}_n - \mu) \geq \varepsilon \leq 2e^{-\frac{2n\varepsilon^2}{(b-a)^2}}$.

În cazul nostru, putem modela rezultatul unei mâini de Blackjack după executarea unei acțiuni (de „hit" sau „stand") folosind o variabilă aleatoare Bernoulli de probabilitate p, deoarece rezultatul poate fi „câștig" sau „pierdere".

$X \sim \text{Bernoulli}(p) = \begin{pmatrix} 0 & 1 \\ 1-p & p \end{pmatrix}$ $\quad 0 = $ „pierdere"
$1 = $ „câștig"
$0 \leq x_i \leq 1$

Determinăm numărul de simulări necesare pentru a prezice rezultatul cu o marjă de eroare $\varepsilon = 0{,}05$ cu un nivel de încredere $\mathcal{L} = 0{,}98$ (98%) pentru o anumită mână:

$$P(|\overline{S}_n - \mu| < \varepsilon) \geq 1 - 2e^{-2n\varepsilon^2} \geq \mathcal{L}$$

$$\Leftrightarrow e^{-2n\varepsilon^2} \leq \frac{1-\mathcal{L}}{2} \Leftrightarrow 2n\varepsilon^2 \geq -\ln\left(\frac{1-\mathcal{L}}{2}\right)$$

$$\Leftrightarrow n \geq \frac{1}{2\varepsilon^2}\ln\left(\frac{2}{1-\mathcal{L}}\right) \Leftrightarrow n \geq \frac{1}{2\cdot(0{,}05)^2}\cdot\ln\left(\frac{2}{1-0{,}98}\right)$$

$$\Leftrightarrow n \geq \frac{1}{2\cdot\frac{25}{10^4}}\cdot\ln\left(\frac{2}{0{,}02}\right) \Leftrightarrow n \geq \frac{10^4}{5\cdot 10}\cdot\ln(10^2) \Leftrightarrow \boxed{n \geq 921}$$

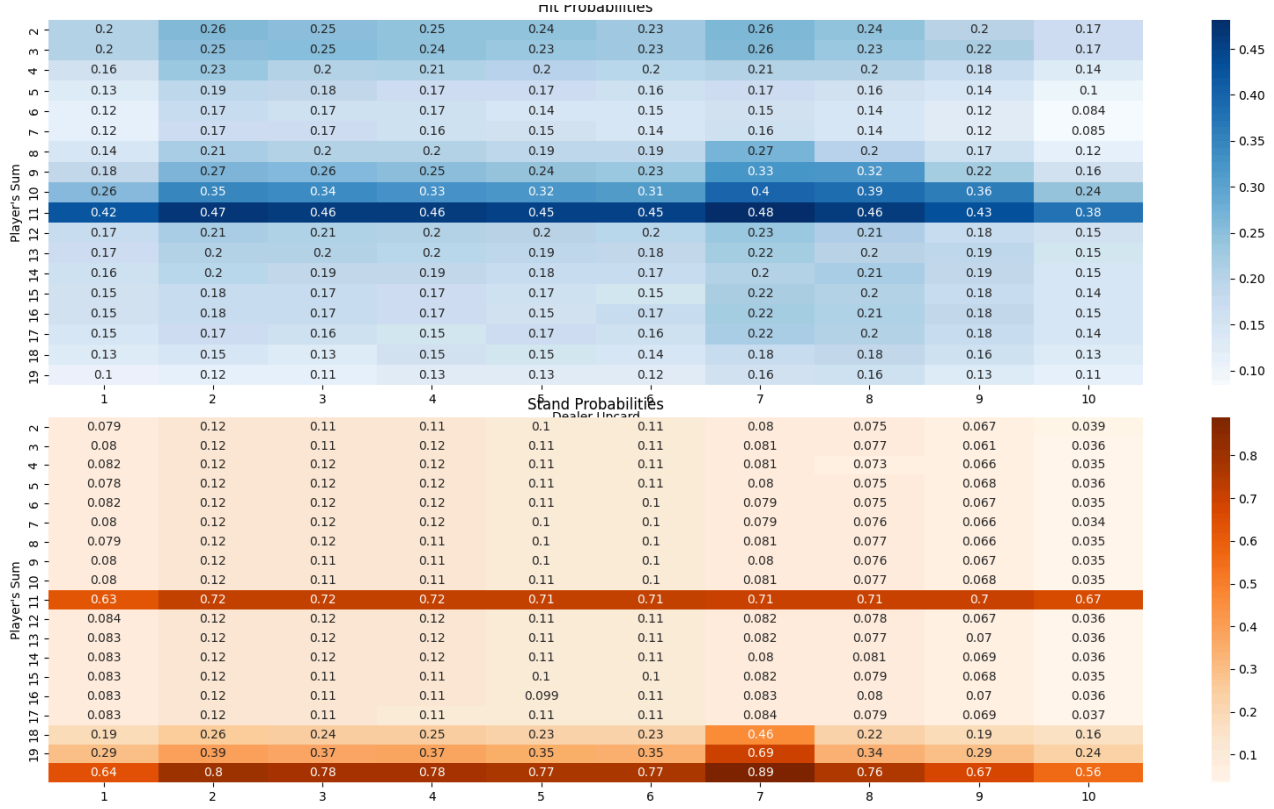
Sunt necesare cel puțin 921 simulări pentru fiecare mână, pentru $\varepsilon = 0{,}05$ și $\mathcal{L} = 0{,}98$.

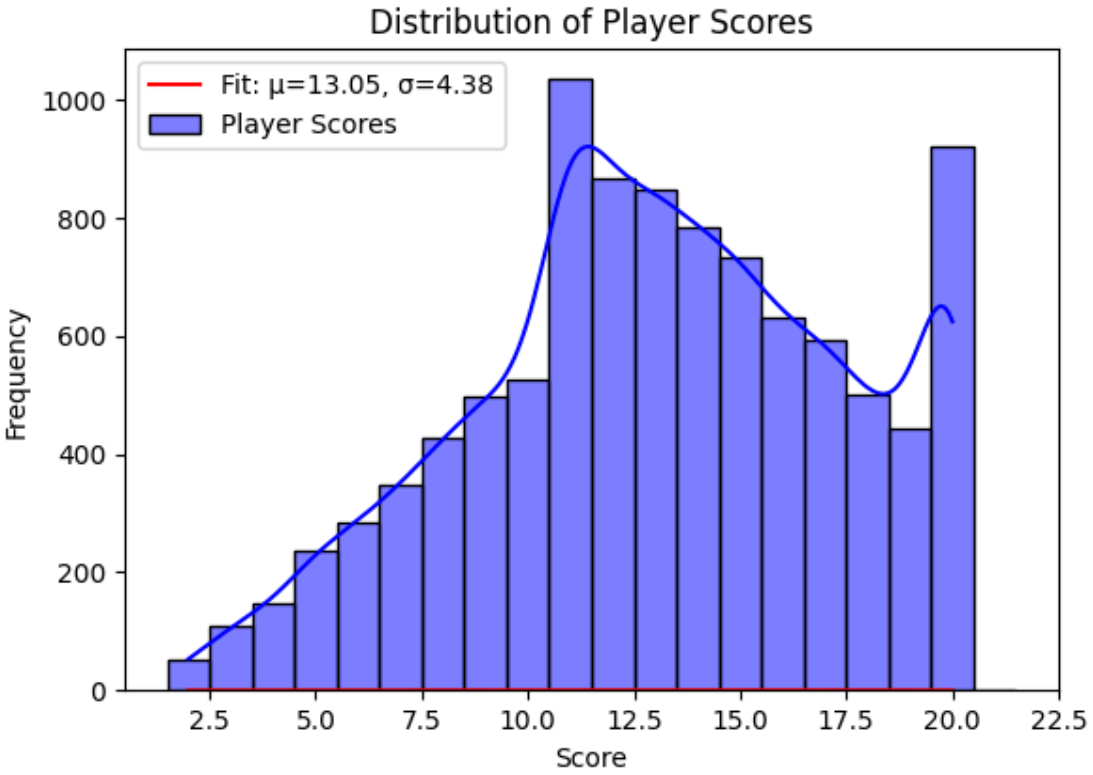
Bibliografie:

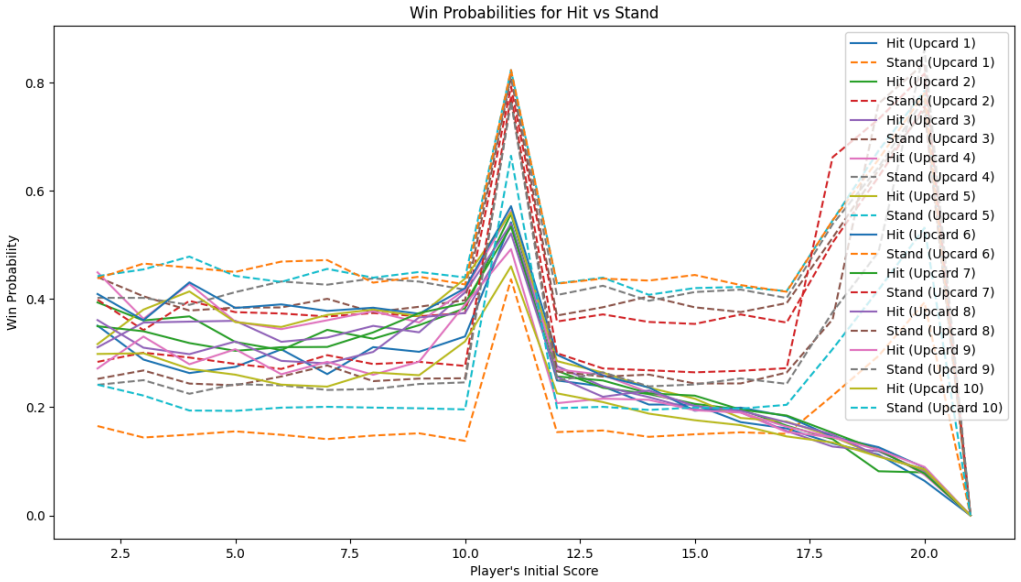
https://ro.wikipedia.org/wiki/Legea_numerelor_mari

http://math.etc.tuiasi.ro/rstrugariu/cursuri/SPD2018/c5.pdf

https://en.wikipedia.org/wiki/Monte_Carlo_method







Proiect realizat de :
Dragomir Daria Nicoleta, grupa 252
Jilavu Izabela Maria, grupa 251
Soare Alex Antonio, grupa 251