# O Aprendizado de Máquina no Jogo de Blackjack: Uma Análise de Q-Learning

---

## 1. Introdução

Neste projeto, implementamos o algoritmo de aprendizado por reforço Q-Learning para desenvolver um agente autônomo capaz de jogar Blackjack. O objetivo principal foi criar uma política de jogo que maximize a taxa de vitórias a longo prazo, superando estratégias de jogo mais simples, como uma política aleatória e a estratégia fixa do próprio cassino.

O Blackjack foi escolhido devido à sua natureza estocástica, onde o resultado de cada rodada depende tanto das decisões do jogador quanto das cartas sorteadas, tornando-o um ambiente ideal para testar a capacidade de um agente de Q-Learning de aprender a tomar decisões ótimas em condições de incerteza.

---

## 2. Metodologia

A implementação do projeto foi dividida em três partes principais:

1. **Ambiente de Jogo (BlackjackEnv.py):** Criamos um ambiente que simula as regras do Blackjack. O estado do jogo é definido por uma tupla: (soma_da_mão_do_jogador, valor_da_carta_do_dealer, tem_ás_utilizável). As ações possíveis do agente são "pedir carta" (hit) ou "parar" (stick). A recompensa é atribuída ao final de cada rodada: +1 por vitória, -1 por derrota e 0 por empate. Para enriquecer o aprendizado, recompensas parciais foram adicionadas.
2. **Agente de Aprendizado (QLearningAgent.py):** O algoritmo Q-Learning foi implementado para aprender a política ótima. A tabela Q armazena o valor esperado de cada ação em cada estado. A política de exploração-explotação epsilon-greedy foi utilizada, onde o agente balanceia entre explorar novas ações e explotar as ações que ele já conhece como as melhores. O valor de epsilon decai com o tempo para que o agente se torne mais confiante em suas decisões.
3. **Avaliação e Análise (main.py):** Para avaliar o desempenho do agente, ele foi treinado com diferentes configurações de hiperparâmetros (taxa de aprendizado alpha, fator de desconto gamma, decaimento de exploração epsilon_decay e número de episódios). O desempenho de cada configuração foi comparado com duas políticas de base: uma **política aleatória** (que toma decisões aleatoriamente) e uma **política de cassino** (que para em 17 ou mais).

A avaliação foi realizada com uma média de 10 sementes de jogo para garantir que os resultados fossem robustos e não dependessem de uma sequência de cartas aleatória favorável.

---

## 3. Análise de Resultados

A tabela a seguir resume os resultados de desempenho, mostrando a taxa de vitória (%) para o agente de Q-Learning em comparação com as políticas de base.

| ALPHA | GAMMA | EPS_DECAY | EPISÓDIOS | Q-LEARN WIN% | RANDOM WIN% | CASINO WIN% |
|---|---|---|---|---|---|---|
| 0.10 | 0.90 | 0.9995 | 1000000 | 48.48 | 30.92 | 47.62 |
| 0.10 | 0.90 | 0.9999 | 5000000 | 48.71 | 30.96 | 47.62 |
| 0.20 | 0.90 | 0.9995 | 1000000 | 48.37 | 31.67 | 47.62 |
| 0.05 | 0.90 | 0.9995 | 1000000 | 48.45 | 31.41 | 47.62 |
| 0.10 | 0.99 | 0.9995 | 1000000 | 48.47 | 31.66 | 47.62 |
| 0.10 | 0.80 | 0.9995 | 1000000 | 46.23 | 31.26 | 47.62 |
| 0.30 | 0.90 | 0.9997 | 2000000 | 45.84 | 31.41 | 47.62 |
| 0.01 | 0.90 | 0.9999 | 5000000 | 47.09 | 30.88 | 47.62 |
| 0.10 | 0.90 | 1.0000 | 5000000 | 46.36 | 31.02 | 47.62 |
| 0.10 | 0.90 | 0.9990 | 500000 | 45.96 | 31.90 | 47.62 |
| 0.50 | 0.90 | 0.9990 | 200000 | 45.79 | 31.61 | 47.62 |
| 0.20 | 0.99 | 0.9997 | 2000000 | 45.95 | 31.51 | 47.62 |
| 0.10 | 0.50 | 0.9995 | 1000000 | 46.55 | 30.97 | 47.62 |
| 0.05 | 0.95 | 1.0000 | 3000000 | 47.28 | 31.63 | 47.62 |

### 3.1. Análise Geral

Os resultados demonstram, de forma consistente, que o agente de Q-Learning superou significativamente a política aleatória em todas as configurações, com uma taxa de vitória média de **47.16%**, comparada à taxa de **31.08%** da política aleatória.

O ponto mais crucial é a comparação com a política de cassino. O agente de Q-Learning demonstrou ser capaz de aprender uma estratégia mais eficiente, com um desempenho que **superou a taxa de vitória do cassino em 1.65%** na sua melhor configuração.

### 3.2. Impacto dos Hiperparâmetros

- **alpha (Taxa de Aprendizado):** Os valores de alpha em torno de **0.1** e **0.2** demonstraram os melhores resultados. Valores muito altos (ex: 0.50) ou muito baixos (ex: 0.01) levaram a um desempenho inferior, sugerindo que uma taxa de aprendizado moderada é ideal para que o agente encontre a política ótima sem oscilar ou demorar demais para convergir.
- **gamma (Fator de Desconto):** O gamma de **0.90** apresentou os melhores resultados. Valores mais altos (0.99) podem ter incentivado o agente a buscar recompensas futuras de forma excessiva, enquanto valores mais baixos (0.50) o tornaram "míope" para as recompensas de longo prazo, impactando negativamente o desempenho.
- **epsilon_decay (Decaimento de Exploração):** A taxa de decaimento de **0.9999** com um número maior de episódios (5,000,000) resultou no melhor desempenho geral. Isso indica que a **exploração prolongada** foi crucial para que o agente visitasse um número maior de estados e encontrasse uma política mais robusta, especialmente em um ambiente complexo como o Blackjack.

## 4. Conclusão

Os resultados deste projeto demonstram a eficácia do algoritmo Q-Learning para resolver problemas de aprendizado por reforço em jogos estocásticos. O agente foi capaz de aprender uma estratégia de jogo que não apenas se mostrou superior a uma abordagem aleatória, mas também superou a estratégia fixa do próprio cassino.

Este projeto destaca a importância da otimização de hiperparâmetros e da exploração do ambiente de jogo para alcançar uma política de aprendizado robusta. As vitórias sobre o cassino, embora pequenas, representam uma prova do potencial do aprendizado por reforço em encontrar políticas ótimas mesmo em ambientes com uma vantagem natural da casa.