Blackjack Decision Advisor: Um software para auxilio na tomada de decição no jogo de Blackjack

Ana Roberta Fornari, Lívia Stefanni Luiz Santos, Erick Augusto e Vinícius Pittoli

RESUMO:

Este projeto consiste em um simulador de Blackjack desenvolvido para auxiliar jogadores na tomada de decisão durante o jogo, por meio de métodos como simulação Monte Carlo, árvore de decisão, regressão logística e cálculos matemáticos, considerando a mão do jogador e a carta visível do dealer para recomendar a melhor ação possível. O foco principal está na aplicação de técnicas de Inteligência Artificial para modelar estratégias eficazes, simulando múltiplas possibilidades de jogadas e oferecendo sugestões baseadas em análises probabilísticas e modelos matemáticos, permitindo que o jogador compreenda suas chances associadas a cada movimento e maximize suas possibilidades de vitória. É importante destacar que, mesmo seguindo a melhor decisão possível, o Blackjack continua sendo um jogo de azar — decisões ótimas maximizam as chances de sucesso no longo prazo, mas não eliminam completamente as perdas. Assim, o projeto tem caráter educativo, com foco em alcançar o melhor desempenho estatístico possível ao longo do tempo.

PALAVRAS-CHAVE: Blackjack, simulador, inteligência artificial, Monte Carlo, árvore de decisão, regressão logística, estratégia, probabilidade, tomada de decisão, desempenho.

ABSTRACT:

This project is a Blackjack simulator developed to assist players in making decisions during the game by employing methods such as Monte Carlo simulation, decision trees, logistic regression, and mathematical calculations, taking into account the player's hand and the dealer's visible card to recommend the best possible action. The main focus is on applying Artificial Intelligence techniques to model effective strategies, simulating multiple possible plays and providing suggestions based on probabilistic analyses and mathematical models, allowing the player to understand the odds associated with each move and maximize their chances of winning. It is important to highlight that even when following the best possible decision, Blackjack remains a game of chance — optimal decisions maximize long-term success but do not eliminate losses entirely. Therefore, this project has an educational purpose, aiming to achieve the best possible statistical performance over time.

KEYWORDS: Blackjack, simulator, artificial intelligence, Monte Carlo, decision tree, logistic regression, strategy, probability, decision-making, performance.

INTRODUÇÃO:

O projeto propõe o desenvolvimento de um simulador de Blackjack com ênfase na aplicação de técnicas de inteligência artificial voltadas à otimização da tomada de decisão. Utiliza modelos de machine learning, como regressão logística e árvore de decisão, treinados com dados simulados de partidas reais disponibilizados via Kaggle.

Para um dos métodos de regressão logística, adotou-se um conceito estratégico conhecido como contagem de cartas (card counting), especificamente o sistema Hi-Lo (High-Low), que atribui valores simples (+1, 0, -1) às cartas para estimar a "contagem verdadeira" (true count). Essa contagem auxilia na identificação de situações em que o baralho está favorável ao jogador, influenciando suas decisões.

O simulador fornece recomendações em tempo real sobre as melhores ações a serem tomadas, considerando a mão atual do jogador e a carta visível do dealer. Entretanto, sempre ressalta-se que o Blackjack é um jogo de azar, no qual mesmo as melhores estratégias possíveis dependem de variáveis inerentes ao acaso.

MATERIAL E MÉTODOS

Para o desenvolvimento do simulador, foi utilizada a linguagem de programação Python na sua versão 3.13.5, com o auxílio das bibliotecas PySide 6.9.1 para a implementação da interface gráfica, Scikit Learn 1.7.0, Pandas 2.3.0 e Numpy 2.3.1 para o tratamento dos dados de treino e treinamento dos modelos de machine learning, além do uso da plataforma Jupyter Notebook para a visualização dos dados. O simulador também incorpora técnicas como simulação Monte Carlo para estimar probabilidades de resultados futuros, cálculo matemático de chances para diferentes combinações de mãos e o uso de tabelas guia disponíveis na internet, amplamente utilizadas como tutoriais para iniciantes no jogo. Os dados de treino foram elaborados por Ho (2021), licenciados sob domínio público e obtidos através da plataforma Kaggle.

1. Método da Simulação de Monte Carlo

Analisando estatisticamente as diferentes ações possiveis em uma mão de Blackjack, a ideia central é simular repetidamente o desfecho do jogo para cada desição (HIT, STAND, SPLIT, DOUBLE) a partir da mão inicial do jogador e a mão visivel do Dealer, estimando as probabilidades de vitoria, empate e derrota, assim descobrindo qual ação é a mais favorável ao jogador.

Temos uma função (simular_acoes) que executa 10 mil resultados (reduzida para mil casos por questões de processamento), estimando as probabilidades reais associadas a cada decisão, considerando a variabilidade do jogo e o embaralhamento aleatório.

2. Método de Estimativa de EV + Simulação Monte Carlo para probabilidade de vitoria, empate e derrota.

No estudo de decisões no Blackjack, o conceito fundamental é o Valor Esperado (EV). O EV é uma medida estatística que indica o valor médio esperado de uma ação ao longo do tempo, considerando todos os possíveis resultados e suas probabilidades.

Matematicamente calculado como:

EV = $\sum$ P(i)$x\ G(i)$, onde P(i) é a probabilidade do resultado i ocorrer, e G(i) é o ganho ou a perda associada ao resultado i.

Esta tabela (representada como um dicionário no arquivo *MonteCarloTabela.py*) foi utilizada para determinar a decisão com maior probabilidade de ocorrer. Após essa decisão, aplicou-se a simulação Monte Carlo para estimar a probabilidade de vitória, empate ou derrota da mão correspondente.

3. Tratamento do conjunto de dados

O conjunto de dados utilizado para o treinamento dos modelos de inteligência artificial apresenta o resultado de 50 milhões de partidas simuladas, em um arquivo no formato CSV, onde foram utilizados 8 baralhos comuns de 52 cartas cada (2 a 10, Ás, Rei, Dama e Valete para os naipes Copas, Espadas, Ouro e Paus), com uma penetração máxima de 338 cartas até o reembaralhamento. Os dados foram organizados de forma que cada linha representa uma partida, e cada coluna uma das variáveis da partida, sendo elas: "shoe_id" (qual dos 8 baralhos foi utilizado na partida), "cards_remaning" (número de cartas restantes na pilha), "dealer_up" (carta visível do *dealer*), "initial_hand" (duas cartas iniciais do jogador), "dealer_final" (as cartas do *dealer* no final da partida), "dealer_final_value" (soma dos pontos das cartas do *dealer*), "player_final" (as cartas do jogador no final da partida), "player_final_value" (a soma dos pontos das cartas do jogador no final da partida), "actions_taken" (conjunto de ações tomadas pelo jogador durante a partida, sendo representadas por uma letra entre "H", "S", "D", "P", "R", "I" e "N"), "run_count" e "true_count" (que representam valores obtidos através da estratégia de *card-counting*) e, por fim, "win" (taxa de lucro relativa a aposta inicial).

Isto posto, das colunas presentes no arquivo CSV original, foram utilizadas apenas "dealer_final", "player_final", "actions_taken" e "win" (e "true_count" para o treinamento de modelos utilizando a estratégia de *card-counting*). Foram consideradas apenas as primeiras 25000 partidas a fim de manter o escopo da aplicação reduzido, e evitar o *over-fitting* dos modelos em relação aos dados, e então filtramos as partidas para considerarmos apenas aquelas que obtiveram resultado positivo (em que a coluna *"win"* seja maior ou igual a 1).

Assim sendo, para facilitar o treinamento dos modelos de IA, dividimos cada linha do conjunto de dados em várias linhas que possuam apenas uma ação, a soma dos valores das cartas na mão do jogador no momento em que a ação foi tomada, a carta relevada do *dealer* e o resultado *"win"* através do excerto de código apresentado no código 1.

Código 1: Trecho do código utilizado para o tratamento dos dados

```
# Divide as ações dos players em vários estados de jogo
for i in range(0, len(df.index)):
    linha = df.loc[i]

    for j in range(0, len(linha['actions_taken'])):
        mao = linha['player_final'][j]
        acoes = linha['actions_taken'][j]

        for k in range(0, len(acoes)):
            acao = acoes[k]

            df = pd.concat([df, pd.DataFrame([{'dealer_final': linha['dealer_final'],
'player_final': sum(mao[:2+k]), 'actions_taken': acoes[k], 'win': linha['win']}])],
ignore_index=True)
```

Finalmente, os dados são emabaralhados e divididos em subconjuntos de treino (80% do conjunto de dados) e teste (20% do conjunto de dados) através do método *train_test_split()* da biblioteca Scikit Learn. Por fim, estes subconjuntos são embaralhados de forma aleatória.

4. Treinamento dos modelos de Machine Learning

Como modelos de machine learning, optou-se pelo uso dos modelos de classificação baseados em Regressão Logística e Árvore de Decisão implementados na biblioteca Scikit Learn. Cada um foi treinado duas vezes, uma ignorando e a outra levando em consideração a estratégia de *card-counting*. Como conjunto de características, foram utilziadas as colunas "dealer_final", "player_final" e "true_count", com o valor da coluna "win" sendo utilizado como peso para as amostras e a coluna "actions_taken" como a saída esperada. Por fim, obteve-se a precisão dos modelos de classificação em relação ao subconjunto de dados de teste através do método "score()", e então os modelos foram exportados no formato de arquivos PKL através do método "pickle.dump()", parte da biblioteca-padrão da linguagem Python, para que possa ser integrado ao simulador.

RESULTADOS

1. Treinamento dos Modelos de Machine Learning

Após o treinamento, podemos observar a precisão dos modelos, em relação aos subconjuntos de dados teste, na tabela 1.

Tabela 1: Pontuação dos modelos em relação ao conjunto de dados de teste

Modelo	Estratégia de Card Counting	Precisão obtida
Regressão Logística	Não	78,34 %
Regressão Logística	Sim	77,91 %
Árvore de Decisão	Não	91,94 %
Árvore de Decisão	Sim	90,15 %

Podemos observar que há uma pequena diferença entre a precisão dos modelos com e sem card-counting, provavelmente decorrente do embaralhamento da ordem das linhas dos casos de treino e de teste, mas que não representa uma diferença significativa na prática. Assim sendo, podemos, também, visualizar o aumento de precisão do modelo de Árvore de Decisão em relação ao modelo de Regressão Logística proveniente da forma como estes tratam as fronteiras entre os dados. Uma vez que o modelo de Árvore de Decisão define as classes de forma determinística, e a Regressão Linear, de forma probabilística.

2. Jogando sem auxílio da ferramenta X com auxílio da ferramenta

Realizamos uma breve análise comparativa entre dois jogadores que utilizaram a plataforma e tiveram suas decisões confrontadas com as recomendações do sistema. Cada um jogou 20 mãos: o Jogador 1 acertou 70% das decisões conforme a estratégia da máquina, mas terminou com um prejuízo de 500 fichas; já o Jogador 2 acertou 60% das decisões, porém obteve um lucro de 1100 fichas. O resultado evidencia que, além da

aderência à estratégia ideal, fatores como sorte, variação do jogo e decisões em momentos-chave também influenciam significativamente o desempenho final. Podendo notar na mão 16 do grafico 2, onde o Jogador 2 ganhou 450 fichas (split quaduplo ganhando 3 e fazendo um Blackjack), e diversas ocasiões onde o jogador teve ganhos enquanto o sistema não teve, evidenciando sorte.

Gráfico 1: Variação de Fichas do Jogador 1 x Variação das fichas usando o Sistema

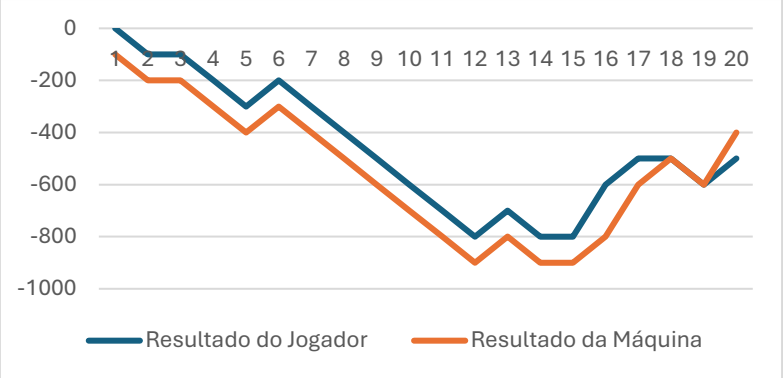
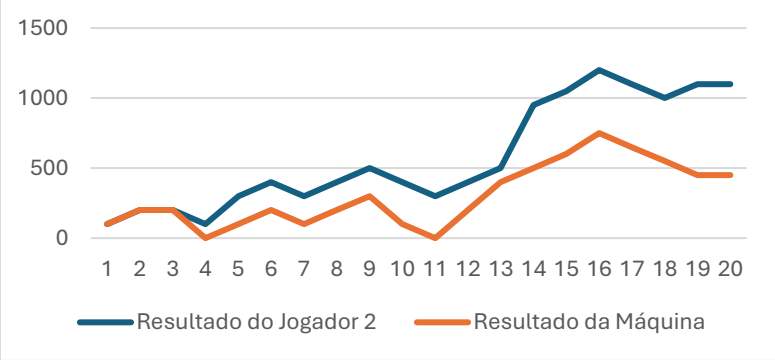


Gráfico 2: Variação de fichas do Jogador 2 x Variação de fichas usando o Sistema



CONCLUSÃO

Com o desenvolvimento do *Blackjack Decision Advisor*, conseguimos demonstrar a viabilidade de aplicar técnicas de inteligência artificial na criação de ferramentas capazes de recomendar ações aos jogadores com base em análises estatísticas e probabilísticas. No entanto, os resultados também evidenciaram o fator sorte, altamente influente no jogo: mesmo com o sistema otimizando suas decisões para maximizar as chances de lucro a longo prazo, ele ainda pode apresentar prejuízo em algumas sessões, como observado no Gráfico 1. Isso reforça que o Blackjack permanece um jogo de variância. Curiosamente, no Gráfico 2, o Jogador 2 — mesmo tomando menos decisões alinhadas com a estratégia ideal — obteve um desempenho superior ao do sistema, evidenciando que, em curto prazo, a sorte pode se sobrepor à estratégia.

REFERENCIAS

BLACKJACKINFO.COM. Blackjack Strategy and Odds. Disponível em: https://www.blackjackinfo.com/. Acesso em: 3 jun. 2025.

HO, Dennis. 50 Million Blackjack Hands. 2021. Disponível em: https://www.kaggle.com/datasets/dennisho/blackjack-hands/. Acesso em: 13 jun. 2025.

STAKE COMMUNITY. Vale a pena seguir a tabela do blackjack? Stake Community, [2023?]. Disponível em: https://stakecommunity.com/topic/30385-vale-a-pena-seguir-a-tabela-do-blackjack/. Acesso em: 3 jun. 2025.

THORP, Edward O. Beat the Dealer: A Winning Strategy for the Game of Twenty-One. New York: Vintage Books, 1966. Acesso em: 19 jun. 2025.

1. Supervised Learning - scikit-learn 1.7.0 documentation. 2025. Disponível em: https://scikit-learn.org/stable/supervised_learning.html. Acesso em: 3 jul. 2025.