ISCTE-IUL
Mestrado em Ciência de Dados

Análise dos stocks prices da NVIDIA através de estratégias estatísticas e um agente de RL

Trabalho de Grupo realizado no âmbito da Unidade Curricular de Otimização de Estratégias Orientada por Dados do 1° ano do Mestrado em Ciência de Dados

Diogo Freitas, 104841, MCD-LCD
daafs@iscte-iul.pt

João Francisco Botas, 104782, MCD-LCD
Joao_Botas@iscte-iul.pt

Rebeca Sampaio, 126628, MCD-LCD
rhmso1@iscte-iul.pt

29 de dezembro 2024
Versão 1.0.0

Índice

1. Introdução ..... 1 / 25
2. Problema do Projeto ..... 1 / 25
3. Dados ..... 2 / 25
    3.1. Aquisição e preparação dos dados ..... 2 / 25
    3.2. Descrição dos dados ..... 2 / 25
    3.3. Estatísticas ..... 3 / 25
        3.3.1. Estatísticas Anuais ..... 3 / 25
        3.3.2. Estatísticas Gerais ..... 5 / 25
4. Metodologia ..... 7 / 25
    4.1. Estratégias de *trading* ..... 7 / 25
        4.1.1. Crossover de Média Móvel Exponencial ..... 7 / 25
    4.2. Modelos de *Machine Learning* (Supervisionada) ..... 9 / 25
        4.2.1. Regressão ..... 9 / 25
        4.2.2. Classificação ..... 11 / 25
            4.2.2.1. Aplicação do modelo ..... 12 / 25
            4.2.2.2. *BackTesting* do modelo de Machine Learning ..... 13 / 25
    4.3. Algoritmo de *Reinforcement Learning* ..... 13 / 25
    4.4. Avaliação, Comparação e *Fine Tune* dos modelos de RL ..... 15 / 25
5. Resultados ..... 17 / 25
6. Conclusões ..... 23 / 25
Anexos ..... 25 / 25
    Anexo A - *Dashboard* com os dados iniciais ..... 25 / 25
    Anexo B - Gráficos EMA para valores da Tabela 5 (restantes) ..... 25 / 25
    Anexo C - Mau exemplo de parâmetros Reinforcemente Learning ..... 27 / 25

1. Introdução

No âmbito da Unidade Curricular de Otimização de Estratégias Orientada por Dados (OEOD), foi-nos proposto o desafio de desenvolver um estudo inovador que alinhasse estratégias estatísticas com algoritmos de *Reinforcement Learning*. Este projeto tem como foco a análise do comportamento das ações da NVIDIA, uma empresa amplamente reconhecida pela sua liderança em tecnologias de ponta, como inteligência artificial, GPUs de alto desempenho e soluções para data centers.

Fundada em 1993, a NVIDIA é uma empresa reconhecida como líder global em tecnologia de computação gráfica, inteligência artificial e soluções de computação acelerada. Desde o lançamento da sua primeira unidade de processamento gráfico (GPU) em 1999, a empresa transformou setores como entretenimento digital, ciência, saúde e automobilismo, posicionando-se como um dos principais impulsionadores da inovação tecnológica no século XXI. Nos últimos anos, a NVIDIA tem se destacado não apenas pela inovação tecnológica, mas também pelo bom desempenho no mercado financeiro, com o valor das suas ações, registando aumentos significativos. Esse crescimento está diretamente relacionado à forte expansão das GPUS de alto desempenho, fundamentais para os domínios da Inteligência Artificial (AI), *Cloud Computing* e *Machine Learning* (ML), além do papel essencial da empresa no desenvolvimento de soluções para data centers e tecnologias de ponta.

O principal objetivo é compreender os fatores que impulsionaram o crescimento da empresa entre 2019 e 2024, desenvolvendo estratégias de negociação automatizada que possam otimizar retornos financeiros. Para isso, integrámos abordagens estatísticas clássicas, como o *crossover* de médias móveis exponenciais, ou mais robustas, através de um modelo preditivo de ML. Isto tudo associado à complexidade de um agente de ***Reinforcement Learning*** capaz de tomar decisões adaptativas baseadas em dados históricos. Este estudo não só explora a viabilidade técnica dessas ferramentas no mercado financeiro, mas também contribui para a aplicação prática de metodologias avançadas no contexto de *Algorithmic Trading* (AT).

A NVIDIA, além da sua relevância tecnológica, reflete a sua alta volatilidade e potencial lucrativo, tornando-se um caso de estudo ideal para explorar a aplicação de técnicas analíticas e preditivas em mercados dinâmicos. Analistas do Citi projetam que as ações da Nvidia podem valorizar até 27% nos próximos 90 dias, impulsionadas por expectativas positivas em relação ao discurso do CEO, Jensen Huang, na CES 2025.

2. Problema do Projeto

A proposta combina estratégias estatísticas e algoritmos de aprendizagem por ***Reinforcement*** para identificar padrões no mercado e aproveitar oportunidades de investimento, com foco na otimização de retornos financeiros e análise dos fatores que influenciam o desempenho das ações da NVIDIA. A empresa, líder em inteligência artificial e GPUs, apresenta um aumento no mercado bastante significativo nestes últimos anos, tornando-se num caso ideal para técnicas avançadas de análise quantitativa e AT.

O projeto apresenta desafios específicos que precisam ser confrontados para garantir o seu sucesso. Um dos principais obstáculos é a previsibilidade limitada do mercado, uma vez que, o desempenho das ações da NVIDIA é influenciado por fatores externos como, por exemplo, anúncios de novos produtos, alterações nas regulamentações e avanços tecnológicos, que podem causar movimentos inesperados. Além disso, as flutuações macroeconómicas, como mudanças nas taxas de juros, inflação e políticas comerciais, desempenham um papel significativo na variação do valor das ações da empresa. Esses desafios ressaltam a complexidade do mercado financeiro e reforçam a importância de desenvolver abordagens robustas, adaptativas e com espírito crítico para lidar com essas incertezas e maximizar os resultados das estratégias de AT.

3. Dados

3.1. Aquisição e preparação dos dados

Para a recolha dos dados históricos das ações da *NVIDIA*, foi utilizada a biblioteca yfinance no *Python*, uma biblioteca utilizada para obter dados financeiros de forma eficiente e acessível. Posteriormente, será realizado um *dashboard* para a melhor compreensão e visualização dos dados (Secção Anexo A), com um gráfico OHLC interativo.

Foram definidos os seguintes parâmetros para o *download*:

- **`ticker`**: `NVDA`, representa as ações da *NVIDIA*.
- **`start_date`**: `2019-01-01`, correspondente ao início do período de interesse.
- **`end_date`**: `2024-11-01`, para garantir que os dados incluam também o dia 31 de outubro de 2024.

Após a execução da função de *download*, os dados foram obtidos num formato inicial não ideal, onde: a coluna **`Ticker`** era repetida para cada coluna e a data era o índice, dificultando certas operações. Este formato necessitou de pequenos ajustes para facilitar a manipulação e análise dos dados. Após o processamento, os dados foram reorganizados, resultando num *dataset* mais limpo e estruturado. A Tabela 1 mostra o resultado final, após o tratamento dos dados:

Tabela 1: Exemplo de dados da *NVIDIA*

Date	Adj Close	Close	High	Low	Open	Volume
2019-01-02	3.378089	3.405500	3.462000	3.251250	3.266000	508752000
2019-01-03	3.173996	3.199750	3.379000	3.192250	3.344750	705552000
2019-01-04	3.377346	3.404750	3.443250	3.242500	3.273500	585620000
2019-01-07	3.556145	3.585000	3.622250	3.410750	3.462500	709160000
			(...)			
2024-10-31	132.750854	132.759995	137.610001	132.110001	137.600006	270039600

3.2. Descrição dos dados

Uma descrição mais detalhada dos dados pode ser visualizada na Tabela 2:

Tabela 2: Descrição das variáveis

Variable	Type	Description
`Date`	`datetime('%Y-%m-%d')`	Representa a data de negociação de cada registo
`Adj Close`	`float`	Refere-se ao preço de fecho ajustado para fatores como dividendos e desdobramentos de ações (*stock splits*).
`Close`	`float`	O preço de fecho da ação no dia de negociação, sem ajustes por eventos como dividendos ou *splits*.
`High`	`float`	O preço mais alto que a ação atingiu durante o dia de negociação.
`Low`	`float`	O preço mais baixo que a ação atingiu durante o dia de negociação.
`Open`	`float`	O preço inicial da ação no início do dia de negociação.
`Volume`	`int`	O número total de ações transacionadas no dia, representando a liquidez e o interesse do mercado. **Ver Dados** 16

É relevante destacar algumas características da variável **`Date`**, que corresponde ao índice das linhas do nosso *dataset*. A variável inicia no dia `2019-01-02` e termina no dia `2024-10-31`. O dia `2019-01-01` não está presente porque corresponde a um feriado, durante o qual o mercado de *stocks* esteve fechado. De forma intuitiva, seria expectável que o número total de linhas do *dataset* fosse 2130, representando todos os dias do intervalo entre `2019-01-02` e `2024-10-31`. Contudo, o *dataset* contém apenas 1469 linhas. Isto deve-se ao facto de o mercado de *stocks* não operar durante os fins de semana e feriados. Por exemplo, como ilustrado em Tabela 1, os dias `2019-01-05` e `2019-01-06` estão ausentes, pois corresponderam a um fim de semana.

3.3. Estatísticas

3.3.1. Estatísticas Anuais

Tabela 3: Tabela de métricas anuais do stock

Year	Annual Return (%)	Avg Price	Max High	Min Low	Avg Volume	Volatility (%)	Trading Days	Days Above Avg Volume	Trend Slope
2019	73.41%	$4.339	$6.045	$3.192	456399667	41,29%	260	94	0.004417
2020	118.02%	$9.86	$14.727	$4.517	480855257	58,84%	261	106	0.026562
2021	124.48%	$19.483	$34.647	$11.567	359558817	45,70%	260	92	0.051868
2022	−51.44%	$18.544	$30.711	$10.813	543163223	64.21%	260	111	−0.036685
2023	246.10%	$36.552	$50.548	$14.034	473557460	49.10%	259	97	0.095347
2024	175.68%	$101.594	$144.42	$47.32	409995164	51.30%	218	98	0.254817

A tabela abaixo apresenta as métricas anuais de 2019 a 2024 para o preço de uma ação, com foco nas variáveis mais relevantes.

1. **`Retorno Anual(%)`**: é calculado como a diferença percentual (Tx. Variação) entre o preço de encerramento do último dia do ano e o primeiro dia do ano. Abaixo, podemos visualizar a fórmula utilizada:

$$\texttt{annual\_return\_pct} = \frac{annual\_stats_{last\_close} - annual\_stats_{first\_close}}{annual\_stats_{first\_close}} \times 100 \tag{1}$$

- O maior **retorno anual** ocorreu em 2023, com 246,10%, indicando um grande crescimento no valor da ação entre o início do ano e o final deste.
- O **retorno anual** mais baixo foi em 2022, com −51,44%, representando uma queda significativa no valor da ação.

2. **`Preço Médio Anual (Preço Ajustado)`**: reflete o valor médio da ação ao longo do ano, levando em consideração os ajustes de dividendos e desdobramentos.
   - Em 2024, o preço médio foi de $101.59, o mais alto da tabela, o que pode indicar um forte desempenho da *NVIDIA* ao longo dos anos, que contribui oara um aumentos das suas ação nos últimos anos.
3. **`Máximo Anual (High) e Mínimo Anual (Low)`**: Esses valores representam o preço mais alto e mais baixo alcançado pela ação ao longo do ano.
   - Em 2023, o **máximo anual** foi de $50.55, que é o valor mais alto entre todos os anos da tabela.
   - O **mínimo anual** mais baixo foi em 2020, com $4.52.
4. **`Volume Médio Anual`**: mostra a média do número de ações negociadas por dia no ano.
   - O maior volume foi em 2020, com 480.855.257 ações negociadas diariamente.
5. **`Volatilidade Anual`**[1]: é uma medida que indica o quanto os retornos diários de um ativo podem variar ao longo de um ano, expressa como uma percentagem, e é calculada multiplicando o desvio padrão dos retornos diários pela raiz quadrada do número de dias úteis no ano. Abaixo podemos visualizar a respetiva fórmula:

$$\sigma_a = \sigma_d \times \sqrt{N} \times 100; \tag{2}$$

Onde,

- N: é o número de dias úteis no ano (Normalmente, é utilizado o valor de 252 dias úteis).
- $\sigma_d = \sqrt{\frac{\Sigma_{k=1}^{N}(R_k - \bar{R})^2}{N-1}}$
  - **Retorno diário**: $R_k = \frac{P_k}{P_{k-1}} - 1$. O grupo já possui o preço ajustado de "fechamento" para cada dia. Assim, basta calcular a taxa de variação entre esses valores, ou seja, `data['Adj Close'].pct_change()`.
  - **Média dos Retornos**: $\bar{R} = \frac{1}{N}\Sigma_{k=1}^{N} R_k$. Esta fórmula consiste em fazer uma média dos $\boldsymbol{R_k}$

[1]Cálculo da Volatilidade Anual: https://www.linkedin.com/pulse/volatilidade-bpiga-nma7f/ | https://opcoespro.com.br/blog/como-calcular-a-volatilidade-historica

A volatilidade foi mais alta em 2022 (64,21%), indicando que a ação teve uma maior variação nos preços durante esse ano.

6. **`Dias de Negociação`**: Este valor indica o número total de dias úteis de negociação para cada ano.
   - A média de dias de negociação foi aproximadamente 260, com exceção de 2024, que registou 218 dias devido à disponibilidade dos dados até 31 de outubro, e de 2020, que teve 261 dias por ser um ano bissexto.
7. **`Dias com Volume Acima da Média`**: Indica o número de dias em que o volume de transações foi superior à média anual.
   - O ano de 2020 teve 106 dias de volume acima da média, o maior valor da tabela.
8. **`Tendência Linear (Slope)`**: reflete a inclinação da linha de tendência do preço de fecho ao longo do ano, usando uma regressão linear.
   - Em 2024, a tendência foi a mais acentuada com um valor de 0.369580, indicando um crescimento acentuado da ação no último ano.
   - Os dados de 2022 indicam uma tendência decrescente, com um único ponto atingindo o valor de **–0.036685**. É interessante notar que outras variáveis dessa linha também apresentam variações significativas. O **retorno anual percentual** é negativo, o que indica que o preço da ação da *NVIDIA* no início do ano era superior ao seu valor no final do ano. Além disso, o **volume de transações** (**`Avg Volume`**) foi maior nesse ano, possivelmente devido ao aumento no número de vendas, o que, por sua vez, pode ter contribuído para um maior número de ações físicas tomadas pelos investidores.
   - Após esta análise, podemos visualizar na **Figura 1** a evolução do **`Close`** ao longo dos anos, com a respetiva linha de tendência. É notório um aumento, quase exponencial, dos valores da variável **`Close`**, mas com uma diminuição dos valores do ano de 2022 para 2023. Não é possível identificar com certeza o motivo, mas pode estar relacionado com a escassez de chips[2] que ocorreu em 2022, ou com o fim de linha da GPU[3] mais utilizada na época, a **RTX 2060**.

Figura 1: Preços da variável `Close` e a sua linha de tendência (por ano)

[2]Noticia sobre a escassez de Chips: https://tecnoblog.net/noticias/nvidia-escassez-chips-ate-2022-atraso-compra-arm/

[3]Noticia sobre o fim de linha da RTX 2060: https://www.tecmundo.com.br/voxel/254189-fim-linha-nvidia-encerra-producao-placas-rtx-2060.htm

3.3.2. Estatísticas Gerais

Tabela 4: Análise de Retornos: Médias Esperadas e Risco

Soma dos retornos diários	Retorno Esperado	Asset Expected Return	Volatilidade	Sharpe Ratio	CAGR
4.457 (3)	0.003 (4)	0.000002 (5)	0.033 (6)	0.091 (7)	0.002 (8)

1. **`Soma dos retornos diários`**: acumulação simples das variações percentuais do preço de uma ação ao longo de um período. Podemos visualizar a fórmula abaixo (3):

$$\Sigma_{k=1}^{n} R_k = \Sigma_{k=1}^{n} \left(\frac{P_k}{P_{k-1}} - 1 \right) \vee \Sigma_{k=1}^{n} R_k = \Sigma_{k=1}^{n} \left(\log\left(\frac{P_k}{P_{k-1}} \right) \right) \tag{3}$$

- Onde:
  - R_k é o retorno diário do dia k;
  - P_k é o preço do dia k; P_{k-1} é o preço do dia $k-1$;
  - n é o número total de períodos (dias) analisados.

A `soma dos retornos diários` da *NVIDIA* foi de 4.457 durante o período analisado. Este valor indica uma variação líquida acumulada de aproximadamente 4.46% nos retornos percentuais diários do preço da ação ao longo do tempo. Embora este valor seja positivo, indicando um desempenho global favorável no período, ele não reflete diretamente o crescimento composto do ativo, mas apenas a soma simples das variações diárias.

2. **`Retorno Esperado`**: uma métrica financeira que estima o retorno médio de um ativo ao longo de um período, com base no desempenho histórico. (4)

$$\mu = \frac{Soma\ dos\ Retornos\ Diários}{n} \tag{4}$$

- Onde:
  - n é o número total de períodos (dias) analisados.
  - O `retorno esperado` de 0.003 (ou 0.3%) indica uma previsão de crescimento moderado no preço da ação da *NVIDIA*, sugerindo um ganho diário contínuo, o que, ao ser acumulado, pode resultar em um aumento substancial ao longo do tempo.

3. **`Asset Expected Return`**: representa o retorno médio esperado de um ativo por unidade de tempo, geralmente com base em dados históricos. Este valor é calculado como a média dos retornos diários (ou de outro intervalo temporal) ao longo do período analisado. Podemos visualizar a fórmula abaixo (5):

$$\frac{Retorno\ Esperado}{n} \tag{5}$$

- Onde:
  - n é o número total de períodos (dias) analisados.

O `retorno médio diário` esperado de 0.000002 sugere um crescimento marginal no preço da ação da *NVIDIA*, que, embora pequeno, pode acumular-se de forma significativa ao longo do tempo.

4. **`Volatilidade`**: uma medida da variabilidade dos retornos de um ativo, indicando o quão "errático" ou incerto é o seu comportamento ao longo do tempo. Podemos visualizar a fórmula abaixo (6):

$$\sigma = \sqrt{\frac{\Sigma(R_k - Retorno\ Esperado)^2}{n}} \tag{6}$$

- Onde:
  - n é o número total de períodos (dias) analisados.
  - R_k é o retorno diário do dia k;

Um valor de 0.033 (ou 3.3%) significa que os retornos diários da ação da *NVIDIA* têm uma variação média de 3.3% em torno da média. Quanto maior a volatilidade, maior o risco associado ao ativo, pois os preços podem oscilar significativamente, o que pode representar oportunidades ou desafios para os investidores.

5. **`Sharpe Ratio`**: uma medida de desempenho ajustada pelo risco, utilizada para avaliar o retorno de um ativo ou portfólio em relação ao risco que ele apresenta. Ele é calculado subtraindo a taxa livre de risco do retorno do ativo e dividindo o resultado pela volatilidade (desvio padrão) do retorno do ativo. Podemos visualizar a fórmula abaixo (7):

$$SR = \frac{Retorno\ Esperado - Taxa\ Livre\ de\ Risco}{Volatilidade} \tag{7}$$

- **`Taxa Livre de Risco`**: $(1 + 0.02)^{\frac{1}{360}} - 1$
  - A `Taxa Livre de Risco` é o retorno que um investidor pode esperar de um investimento considerado sem risco de crédito, como os títulos do governo de curto prazo. Esses investimentos são vistos como seguros, pois o risco de inadimplência é mínimo. A taxa livre de risco serve como base para comparar outros investimentos, pois oferece um retorno sem risco, e qualquer ativo mais arriscado deve gerar um retorno maior para compensar o risco adicional.

Um valor de 0.091 é consideravelmente baixo, o que sugere que o retorno extra do investimento não é suficiente para compensar o nível de risco envolvido. Em termos práticos, esse valor implica que o risco assumido não é adequadamente recompensado com o retorno gerado. O `Sharpe Ratio` idealmente deveria ser superior a 1, sendo que um valor abaixo de 1 pode ser visto como um sinal de que o investimento não é suficientemente atraente para justificar o risco.

6. **`CAGR`**: uma métrica utilizada para calcular a taxa de crescimento média de um investimento ao longo de um período de tempo específico, assumindo que o crescimento ocorre de forma constante a cada ano. Em outras palavras, o `CAGR` descreve a taxa de retorno de um investimento, considerando que ele cresceu a uma taxa constante ao longo do período. Podemos visualizar a fórmula abaixo (8):

$$CAGR = \left(\frac{P_{\text{Final}}}{P_{\text{Inicial}}}\right)^{\frac{1}{n}} - 1 \tag{8}$$

- Onde:
  - P_{Final} é o valor final do investimento;
  - P_{Inicial} é o valor inicial do investimento;
  - n é o número de anos.

O `CAGR` de 0.002, ou 0.2%, indica que, ao longo do período considerado, o valor do investimento cresceu a uma taxa média anual de 0.2%. Em outras palavras, se o investimento tivesse crescido a uma taxa constante de 0.2% ao ano, o valor final seria o mesmo que o observado no final do período.

---

7. **Conclusão**:

- Embora a *NVIDIA* apresente uma soma positiva de retornos diários (4.457), o retorno esperado é relativamente baixo, com um valor de 0.003, indicando que o ativo não tem um desempenho excepcional em termos de rentabilidade anual. O `Sharpe Ratio` de 0.091 sugere que o risco associado ao ativo não está suficientemente compensado pelo retorno, o que indica que, apesar do crescimento potencial, o ativo apresenta um retorno inadequado em relação ao risco que envolve. O `Asset Expected Return`, por sua vez, é quase irrelevante, com um valor muito baixo, indicando que o retorno do ativo em relação ao seu valor esperado é praticamente insignificante. A `volatilidade` moderada (0.033) deve ser considerada pelos investidores, pois ela implica que o preço do ativo tem oscilações que podem impactar o desempenho no curto prazo, sendo importante para os investidores avaliarem o risco de flutuações nos preços.
- Em suma, apesar de um retorno diário positivo, a *NVIDIA* apresenta uma relação risco-retorno que pode ser considerada subótima, com a `volatilidade` e a expectativa de retorno relativamente baixas, o que pode torná-la uma opção de investimento menos atrativa para perfis que buscam um desempenho ajustado ao risco mais favorável.

4. Metodologia

4.1. Estratégias de *trading*

Para começar a nossa análise, foi necessário escolher a nossa estratégia de *trading* para ser dado como entrada no modelo de *Reinforcement Learning* (RL). Para isso, decidimos utilizar duas estratégias distintas na sua robustez para posteriormente compará-las e fazer um *trade-off* entre complexidade e explicabilidade das estratégias. Talvez valha mais a pena ter um algoritmo menos complexo, mais fácil de explicar, e que tenha uma *performance* inferior a algo mais robusto e que não produza resultados muito mais interessantes que uma estratégia simples.

A discussão do *trade-off* entre complexidade e explicabilidade com a performance será incorporado na função `get_state()` que será explicado mais à frente na Secção 4.3.

4.1.1. Crossover de Média Móvel Exponencial

Como primeira estratégia decidimos utilizar uma estratégia clássica e bastante simples; a média móvel exponencial (EMA)[4]. É uma variação da média móvel simples (SMA)[5], mas que dá destaque a preços mais recentes, pois utiliza a função exponencial no seu cálculo. Essa característica torna a EMA particularmente útil para mercados dinâmicos e em constante mudança.

No nosso caso, esta estratégia mostra-se interessante por diversos motivos, já enunciados na Secção 3:

- Como os *stocks* da *NVIDIA*, no período considerado, têm um crescimento bastante alto, é importante dar maior destaque aos preços mais recentes, mais próximos do fim da série a 31 de outubro de 2024. Isso permite que esta estratégia se adapte melhor às tendências mais recentes.
- O método de EMA reage a grandes variações de preço mais rápido em comparação ao método de SMA. Algo bastante positivo para mercados voláteis, em que bruscas ou eventos inesperados (como lançamentos de GPUs ou mudanças legislativas) podem indiciar instantes de *buy* ou *sell*;
- Embora seja mais "sofisticada" que a SMA, a EMA continua a ser uma estratégia relativamente simples de explicar e de implementar.

O cálculo da EMA é realizado pela função `ewn()`[6], através da seguinte fórmula:

$$\text{EMA}_t = \text{Close Price}_{\text{today}} \times \left(\frac{\text{Smoothing} = 2}{1 + \text{Days}}\right) + \text{EMA}_{\text{yesterday}} \times \left(1 - \frac{\text{Smoothing} = 2}{1 + \text{Days}}\right) \tag{9}$$

O quociente $\frac{\text{Smoothing}=2}{1+\text{ Days}}$ corresponde ao fator de suavização α, onde o valor de *Smoothing* é fixado em 2 por convenção, mas que poderia ser aumentado se quiséssemos dar ainda mais importância aos instantes mais recentes no EMA. Já o *Days* é o valor do `span`, que corresponde ao número de dias que estão no período para a média móvel exponencial.

Como estamos a adotar uma estratégia de *crossover*, é necessário ter duas linhas para identificar pontos de interseção entre as duas, chamados de *crosses*. Assim, teremos de ter dois valores de ***Days***, com valores de dias distintos para a média móvel, um para captar movimentos mais rápidos e ser mais sensível a variações (***fast***) e outro para ter uma visão mais a longo prazo, suavizando as oscilações de preços a curto prazo (***slow***). Por indução, devem-se utilizar intervalos de dias pequenos se quisermos investimentos a curto prazo (8/20-*days*), ou intervalos maiores se quisermos investimentos a longo prazo, numa série maior (50/200-*days*). No nosso caso, optámos começar pelos valores de 50 e 200 em EMA, tendo mudado estes mais para a frente.

Nesta estratégia, como já foi referido, queremos identificar pontos de interseção entre as duas retas de ***fast*** e ***slow***. Quando a reta ***slow*** (`span`=200) cruza e ultrapassa a reta ***fast*** (`span`=50), esse ponto é designado de *death cross* e o mercado entra em *danger zone*; quando o inverso acontece, a interseção é designada de *golden cross* e o mercado entra numa *opportunity zone*[7]. Na Figura 2 podemos observar estes pontos e zonas.

[4] EMA em stock trading: https://www.investopedia.com/terms/e/ema.asp
[5] SMA em stock trading: https://www.investopedia.com/terms/s/sma.asp
[6] Função *pandas* para EMA: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.ewm.html
[7] Explicação de *crosses*: https://www.investopedia.com/terms/g/goldencross.asp

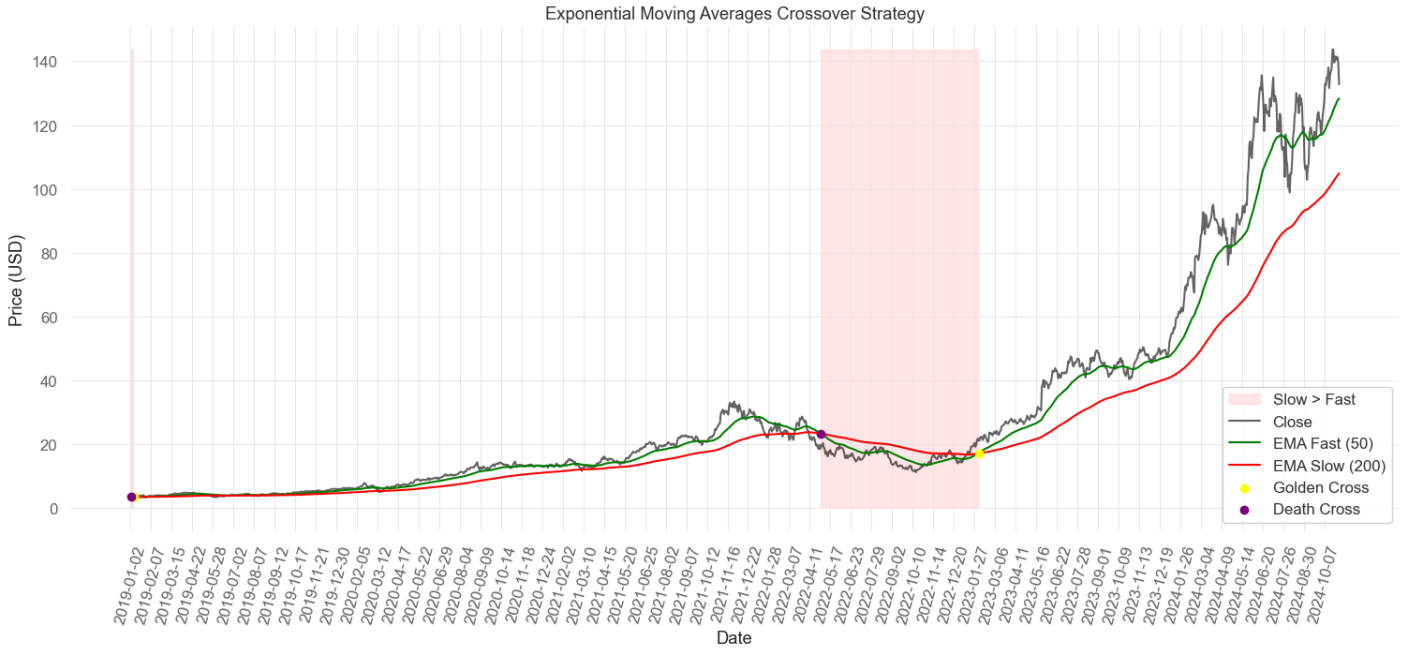

Figura 2: Estratégia de *Crossover* EMA(*fast*:50, *slow*:200)

Para o cálculo dos retornos da estratégia é utilizada a fórmula já calculada anteriormente na Secção 3.3, o R_k, bem como o valor do *Signal*; que é 1, caso o EMA *fast* seja maior que o EMA *slow*, ou seja, um *golden cross*; ou 0 se o EMA *slow* for superior ao EMA *fast* (*death cross*) Assim, o retorno da estratégia é apresentado pela seguinte fórmula:

$$\text{StratReturns}_t = R_t \times \text{Signal}_{t-1} \tag{10}$$

Pela fórmula (10) é possível observar que o *Signal* funciona como um "interruptor" que liga ou desliga para os retornos, em movimentos de subida ou descida. Se o *Signal*=0 num instante t, o retorno da estratégia será nulo para o dia seguinte $t+1$ e, por conseguinte, não será contabilizado nos retornos acumulados da estratégia, descritos pela seguinte fórmula:

$$\text{CumReturns}_t = \prod_{i=1}^{t}(1 + \text{StratReturns}_i) \tag{11}$$

Para os parâmetros de EMA(*fast*=50, *slow*=200) obtivemos um valor de retornos acumulados de 33.38. Para termos de comparação, serão testados vários valores de `span` (Tabela 5), principalmente na curva *slow*, para percebermos o impacto que terá nos retornos acumulados. A razão para isto está no facto de que, quanto maior for o intervalo de datas considerado para o cálculo da média móvel, mais suavizada a curva resultante se torna. Como resultado, a EMA *slow* tende a distanciar-se da linha de `Close` ao longo da série temporal, especialmente depois de 2023 (Figura 2).

Tabela 5: Vários valores de `span` para a estratégia de *Crossover* de EMA

EMA *fast*	EMA *slow*	CumReturns
50	200	33.38
50	150	36.39
50	140	**39.19**
50	130	34.91
40	140	32.65
20	140	23.09

Pelos valores da Tabela 5, é possível verificar que os valores sugeridos inicialmente não refletiam tão bem os retornos acumulados, sendo necessário reduzir o número de dias da média móvel *slow*, a fim de obter-se um valor de retorno acumulado maior e de ter a área da *danger zone* mais adequada, ou até mais *danger zones* e *crosses* (Secção Anexo B). Na Figura 3 temos a estratégia com os parâmetros de (*fast*:50, *slow*:140).

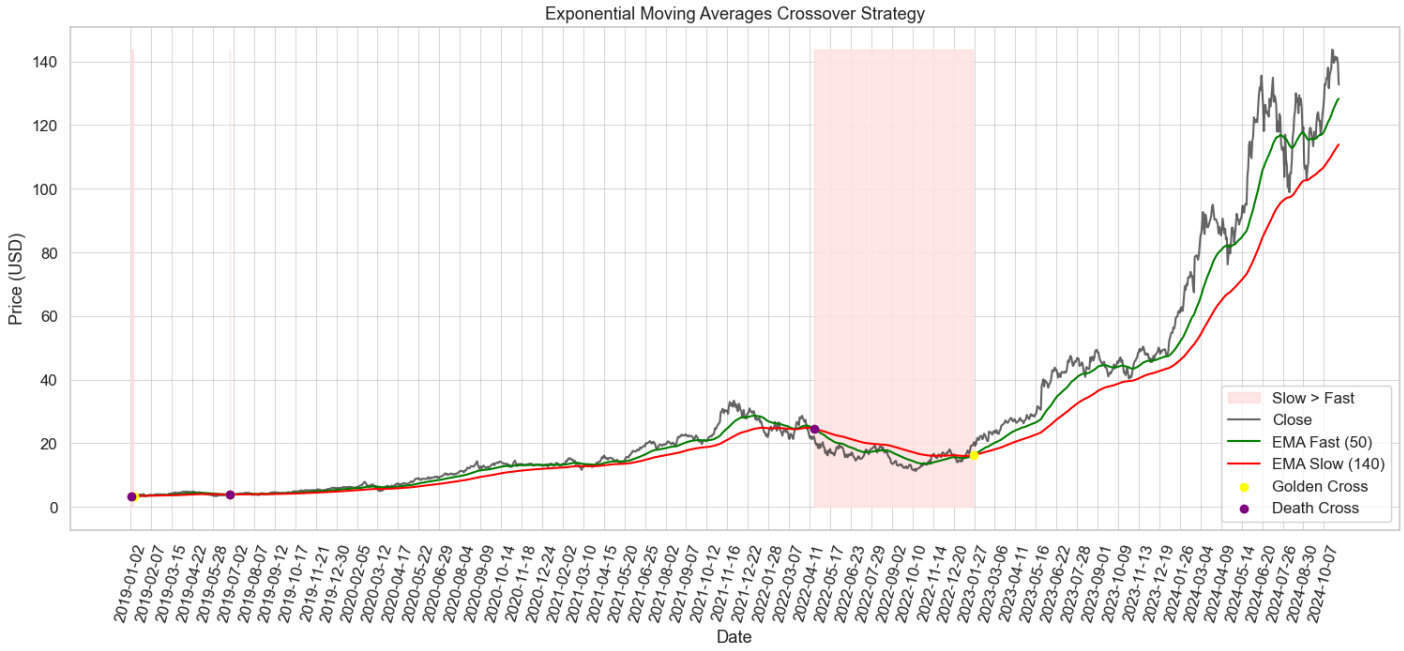

Figura 3: Estratégia de *Crossover* "otimizada" -> EMA(*fast*:50, *slow*:140)

4.2. Modelos de *Machine Learning* (Supervisionada)

Antes de avançarmos para os modelos de **reinforcement learning** (Secção 4.3), será realizado um teste utilizando modelos de **machine learning** tradicionais. Esta abordagem será dividida em duas partes:

1. **ML Regressão** (Secção 4.2.1): Será aplicado um modelo de regressão com o objetivo de prever o valor da variável `Close` do mercado de ações.
2. **ML Classificação** (Secção 4.2.2): Desenvolveremos um modelo de classificação com três classes — **Buy**, **Sell**, e **Hold** — para tentar identificar os momentos ideais de compra e venda de ações, visando maximizar os lucros.

4.2.1. Regressão

Na escolha do modelo de *machine learning*, o grupo decidiu implementar um modelo de **rede neuronal** **LSTM**[8], amplamente reconhecido pela sua robustez e capacidade de captura de padrões complexos temporais. O objetivo principal deste modelo é prever o preço da variável **`Close`** com base em variáveis históricas e indicadores financeiros. As variáveis preditivas utilizadas foram as seguintes:

- **Retornos**: Calcula a variação percentual diária do preço da variável `Close`.
- **Lags**: Criação de variáveis que representam os preços da variável `Close` dos dias anteriores (*lags* de 1 a 4 dias).
- **Volatilidade**: Calcula o desvio padrão dos retornos diários com uma janela de 5 dias.
- **Momentum**: Calcula a diferença entre o preço da variável `Close` atual e o de 5 dias atrás, capturando a tendência recente do mercado.

No pré-processamento dos dados, foram eliminadas as linhas que continham valores nulos, a fim de evitar que esses dados faltantes prejudicassem o desempenho do modelo. Além disso, a variável `close` foi normalizada utilizando o **MinMaxScaler**, que escala os valores entre 0 e 1, garantindo que as variáveis estejam no mesmo intervalo e, assim, facilitando a convergência do modelo durante o treino.

Para a divisão dos dados em treino e teste, optou-se por uma distribuição de **80%** para treino e **20%** para teste. As variáveis preditivas, que incluem os **`lags`**, **`volatilidade`** e **`momentum`**, foram armazenadas na variável **`X`**, enquanto o preço da variável **`Close`** foi armazenado em **`Y`**, que representa o alvo da previsão.

Para a criação do modelo de **rede neuronal LSTM**, foram definidas três camadas com as seguintes características:

[8]Base para a criação do nosso modelo de *Machine Learning* de **Regressão**: https://www.simplilearn.com/tutorials/machine-learning-tutorial/stock-price-prediction-using-machine-learning

- **`Camada de Entrada (LSTM)`**: Composta por **64 unidades** e função de ativação *ReLU* (Rectified Linear Unit), uma camada recorrente que permite capturar dependências temporais nos dados, essencial para séries temporais.
- **`Camada Oculta`**: Com **32 unidades**, também utilizando a função de ativação *ReLU*, o que permite à rede aprender representações mais complexas dos dados.
- **`Camada de Saída`**: Uma única unidade que tem como objetivo prever o valor contínuo do preço de fechamento da ação.

A rede foi compilada utilizando o **Otimizador Adam**, que ajusta os pesos de maneira eficiente durante o treino, e a **Função de perda MSE (Erro Quadrático Médio)**, que mede a diferença entre as previsões do modelo e os valores reais, visando minimizar esse erro.

No treino do modelo, foram escolhidos parâmetros que garantissem um bom desempenho sem comprometer a eficiência computacional. Optou-se por **50** ***epochs***, o que permite ao modelo aprender de forma adequada sem sobrecarregar o tempo de treino. O ***Batch Size*** foi definido como **32**, o que oferece um equilíbrio entre eficiência computacional e precisão na atualização dos pesos durante o treino. Após aplicar o modelo, o resultado pode ser visualizado na **Figura 4**, onde comparamos as previsões do preço da variável **`Close`** com os valores reais.

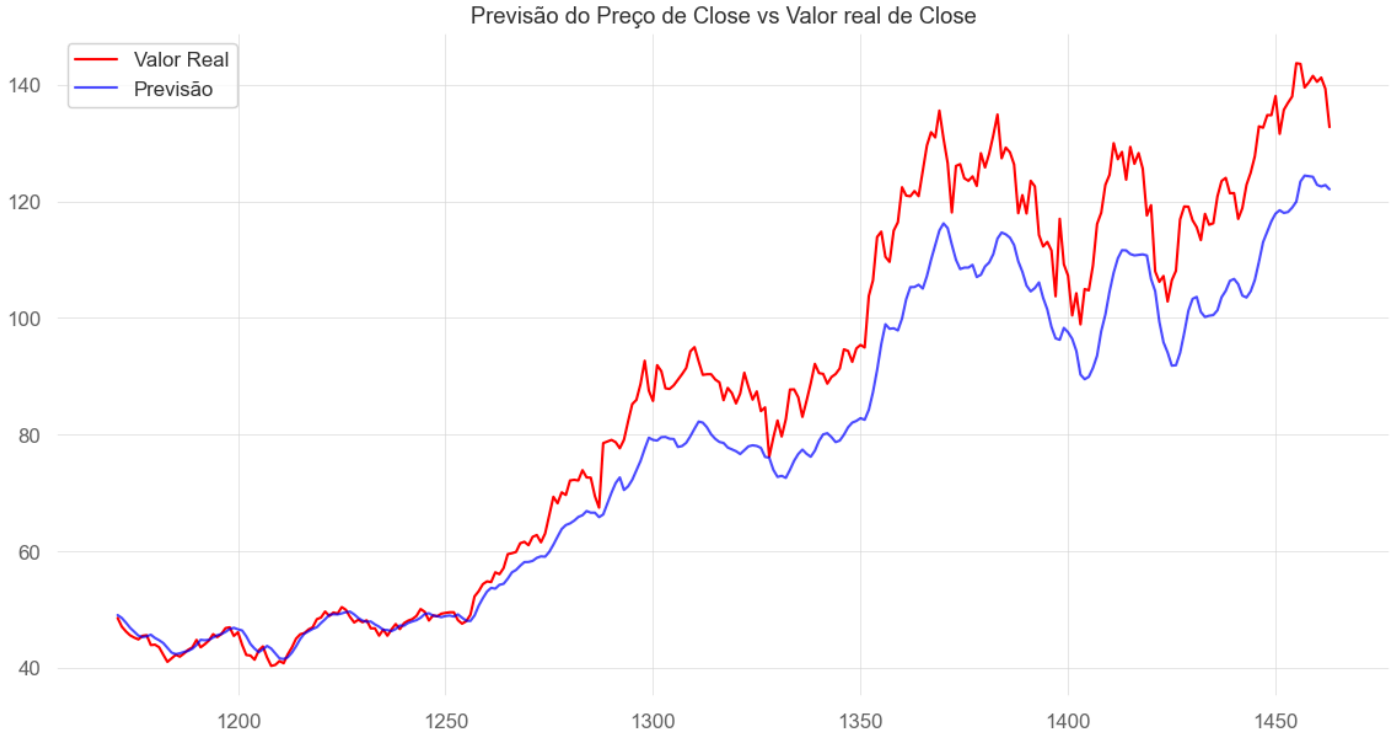

Figura 4: Previsão do preço da variável **`Close`** (conjunto teste) VS Valor real da variável **`Close`**

Para uma análise mais detalhada, iremos combinar os valores das métricas, calculadas na **Listing 1**, com a visualização do gráfico da **Figura 4**.

```
Mean Squared Error (MSE): 132.95054324484673
Mean Absolute Error (MAE): 9.053899865893644
Root Mean Squared Error (RMSE): 11.530418173025934
R-squared (R²): 0.8765712665219383
```

Listing 1: Parâmetros do *LSTM model*

Os resultados sugerem que o modelo pode estar a sofrer de *overfitting*, como indicado pelo alto valor de R^2 (**0.8766**), que revela um bom ajuste no treino, mas levanta dúvidas sobre a generalização para novos dados. O uso de *features* como **lag** e **momentum**, derivadas da variável alvo (**`Close`**), pode estar a exacerbar este comportamento, criando relações artificiais no treino. A **Figura 4** também sugere *overfitting*, enquanto os erros absolutos (MAE e RMSE) apontam uma margem de erro significativa nas previsões. Assim, priorizar um modelo de classificação aparenta ser a decisão adequada para melhorar os resultados do nosso objetivo.

4.2.2. Classificação

Neste modelo, a variável `Close` é utilizada como alvo, mas com um objetivo distinto: decidir se deve comprar, vender ou manter uma ação. Para isso, foi empregado o modelo XGBoost para classificação multiclasse, utilizando os parâmetros `eval_metric='mlogloss'` e `objective='multi:softmax'`. Foram definidos três possíveis estados para o modelo: **Buy**, **Sell** e **Hold**. Para otimizar a performance do modelo, foi realizada uma procura aleatória (`random_search`) nos hiperparâmetros, o que permitiu encontrar os valores mais adequados para o problema.

O tratamento dos dados, as features e a divisão entre os conjuntos de treino e teste seguiram a mesma abordagem da Secção 4.2.1, com a diferença de que foi introduzido um **threshold**. Este **threshold** foi definido previamente para que o modelo pudesse gerar labels para treino. Após várias testes, o grupo determinou que o valor de 1,5% seria o mais apropriado para esta métrica, levando à seguinte lógica para atribuição das *labels*:

1. Se o preço da variável `Close` do próximo dia for **superior** ao preço da variável `Close` de hoje, multiplicado por $1 + threshold$, a label para o dia de hoje será **Buy**: $Preço_{amanhã} > Preço_{Hoje} \times (1 + threshold)$;
2. Se o preço da variável `Close` do próximo dia for **inferior** ao preço da variável `Close` de hoje, multiplicado por $1 + threshold$, a label para o dia de hoje será **Sell**: $Preço_{amanhã} < Preço_{Hoje} \times (1 - threshold)$;
3. Caso nenhuma das condições anteriores se aplique, a label será **Hold**.

Após a criação e aplicação do modelo aos dados, obtivemos as seguintes métricas:

Após aplicar o modelo, o grupo criou uma *confusion matrix* para realizar uma análise preliminar dos resultados. Ao analisar a *confusion matrix*, é evidente que os resultados não são ideais, embora seja importante destacar que as labels utilizadas não são 100% confiáveis, pois foram definidas manualmente. Além disso, é relevante examinar as métricas e os melhores valores dos hiperparâmetros apresentados na Listing 2. Para avaliar a robustez do modelo, será realizado um teste na Secção 4.2.2.1, permitindo tirar conclusões mais precisas.

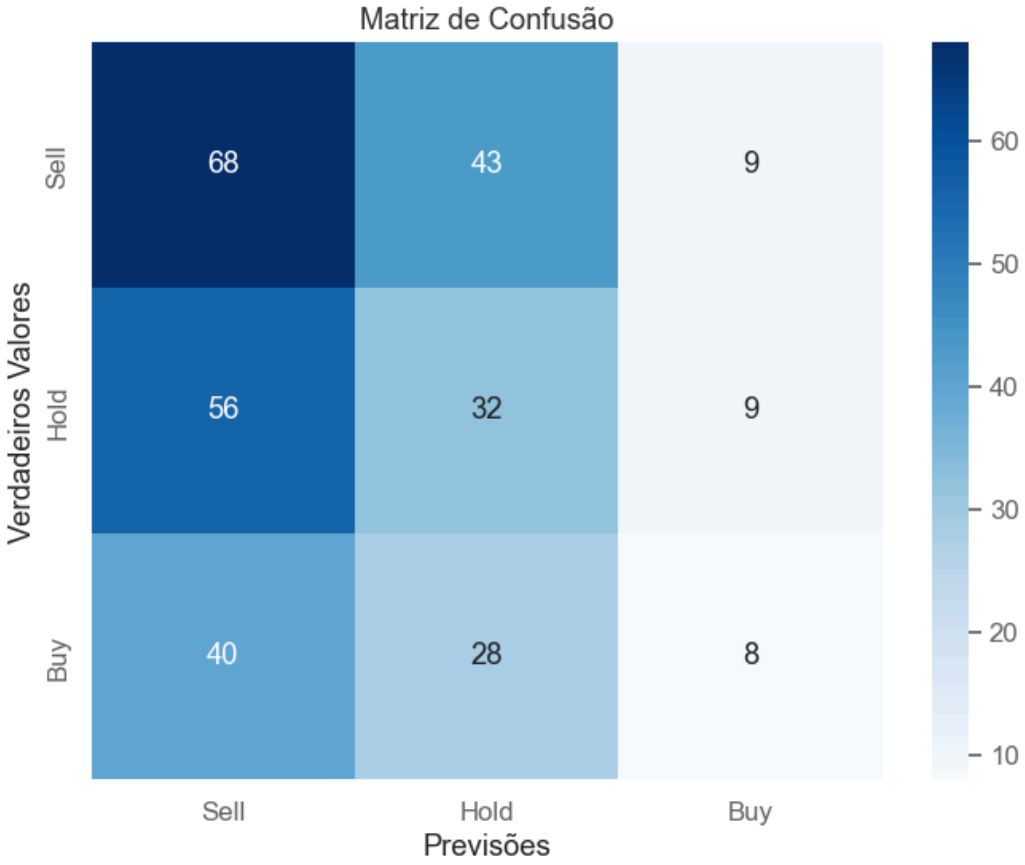

Figura 5: *Confusion Matrix* (conjunto teste)

```
Melhores hiperparâmetros encontrados: {'subsample': 0.9, 'n_estimators': 300, 'max_depth': 7,
'learning_rate': 0.2, 'colsample_bytree': 0.8}
Acurácia: 0.36860068259385664
Precisão: 0.3443353552280855
Recall: 0.33394224392596605
F1-Score: 0.31857866151155295
```

Listing 2: Métricas e hiperpâmetros do modelo de classificação

4.2.2.1. Aplicação do modelo

Para realizar este teste, utilizaremos um orçamento de $1000. É importante destacar algumas limitações deste método, que, para serem corrigidas, exigiriam mais tempo e testes. Primeiramente, por ser um modelo de *Machine Learning*, é necessário dividir os dados em conjunto de treino e teste, o que significa que só será possível avaliar o desempenho do modelo após a execução das ações. Além disso, o modelo pode sugerir a compra de uma ação, mas, devido à limitação do orçamento, pode não ser viável, já que o modelo não foi treinado para considerar essa restrição. Mesmo assim, vamos observar o comportamento do modelo e tirar algumas conclusões. Para uma visualização representativa das recomendações do modelo, podemos utilizar a Figura 6.

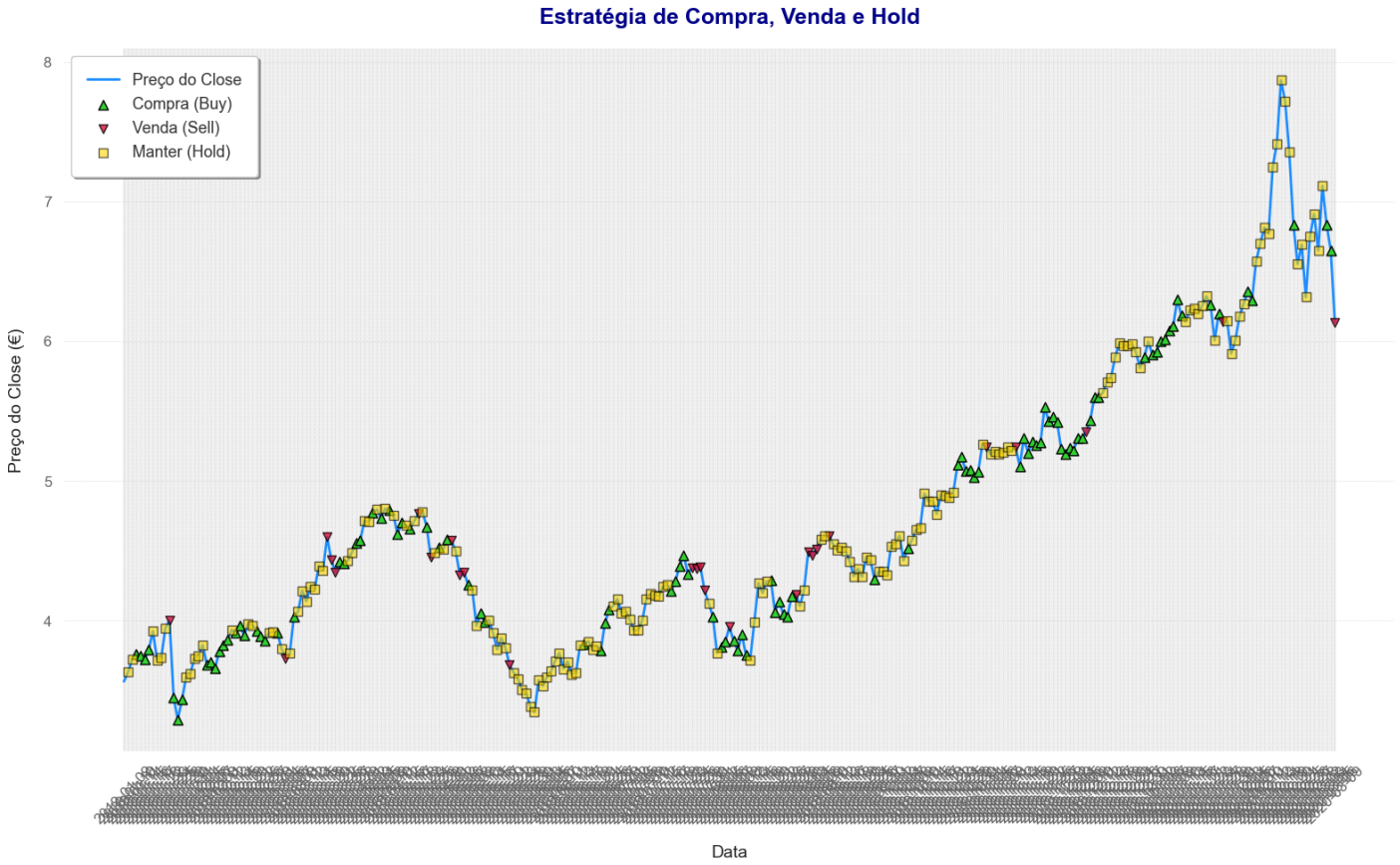

Figura 6: Estratégia de **Compra**, **Venda** e **Hold**: Resultados no Período Final do Dataset

Analisando a Figura 6, é evidente o elevado número de **Hold** registrados, o que é esperado, pois o modelo tende a manter as posições em períodos de incerteza ou estabilidade. Além disso, é importante observar que existem muitos casos de **Buy** seguidos de **Sell**, o que pode refletir a tentativa de aproveitar movimentos de curto prazo no mercado.

Nesse contexto, é relevante destacar a estratégia de tomada de decisão adotada, levando em consideração a **taxa de transação** de 0,1%, que impacta diretamente os resultados de cada operação, reduzindo o lucro potencial de compras e vendas frequentes. Tendo isso em conta, a estratégia de tomada de decisão adotada foi a seguinte:

- $if\ Budget > 0 \land label = Buy$: Compramos todas as ações possíveis e atualizamos o nosso ***Budget*** da seguinte forma:
  - $Budget_{\text{new}} = Budget_{\text{old}} - (quantity_{\text{shares purchased}} \times price_{\text{shares}} \times (1 + fee_{\text{transaction}}))$
- $if\ Shares > 0 \land label = Sell$: Vendemos todas as nossas ações possíveis e atualizamos o nosso ***Budget*** da seguinte forma:
  - $Budget_{\text{new}} = Budget_{\text{old}} + (quantity_{\text{shares in stock}} \times new\ price_{\text{shares}} \times (1 - fee_{\text{transaction}})$
- $else$: Fazemos **hold** quando não houver **budget** suficiente para comprar ações, quando não tivermos ações para vender ou quando a *label* for **hold**.

Após aplicar o algoritmo, é essencial analisar os resultados obtidos, e para isso utilizaremos *backtesting*. As diversas métricas de *backtesting* que serão mencionadas ao longo do relatório estão explicadas e detalhadas na seção Secção 4.4. Na Secção 4.2.2.2, será possível visualizar o comportamento do modelo durante o período de investimentos.

4.2.2.2. *BackTesting* do modelo de Machine Learning

De forma geral, os resultados obtidos durante este período foram positivos. Inicialmente, o nosso *budget* era de \$1000, e ao final do período o modelo concluiu com \$1952, resultando assim em um lucro de \$952, por outras palavras, um retorno sobre o investimento **ROI** (14) de 95,30%. Vale ressaltar que o cálculo do *budget* final leva em consideração o preço das *shares* ainda em *stock*. Além disso, não basta apenas observar o lucro; é fundamental analisar e interpretar outras métricas, cujas explicações detalhadas podem ser encontradas na Secção 3.3.2 e na Secção 4.4.

Outras métricas interessantes de se comentar podem ser visualizadas na Listing 3:

```
Drawdown Máximo: -21.98%
CAGR: 0.78%
Value at Risk (VaR 5%): 3.82%
```

Listing 3: Métricas e hiperpâmetros do modelo de classificação

No **backtesting** do modelo, o **Drawdown Máximo** (15) de −21,98% reflete a maior perda observada durante o período de teste, indicando que o modelo enfrentou uma queda considerável em algum momento. O **CAGR** (8) de 0,78% sugere que o crescimento anual composto foi relativamente baixo, apontando para um retorno modesto ao longo do tempo. O **Value at Risk (VaR) de 5%** (16), de 3.82% o que significa que, com uma probabilidade de 5%, o portfólio poderia sofrer uma perda superior a 3,82% $(0,0382 \times \$1000 = \$38.2)$ em um determinado período. Esses indicadores mostram que, embora o modelo tenha apresentado ganhos, o risco de perdas significativas foi relevante durante o período analisado. Para visualizar a evolução do *backtesting* do modelo, podemos visualizar na Figura 7.

Figura 7: *Backtesting* do modelo de *Machine Learning*

Ao analisar a Figura 7, é claro que o nosso *budget* seguiu uma evolução predominantemente linear, com alguns picos de aumento e decréscimo, especialmente no final. Contudo, é importante refletir sobre essas visualizações, pois as ações da *NVIDIA* apresentaram uma tendência constante de valorização, o que facilita a geração de lucro pelo modelo ao investir nessas ações. Para uma análise mais precisa, é necessário combinar essas observações com métricas previamente discutidas, como a Listing 3.

Um exemplo relevante é o *DrawDown* (15), que indica as perdas durante o período de teste. Ao examinar os valores, é evidente que houve períodos em que ocorreram perdas significativas, o que sugere que, em algumas situações, este modelo pode não ser o mais adequado. Tendo isso em mente, na próxima Secção 4.3, iremos explorar uma técnica que acreditamos ser mais apropriada para este tipo de problema: **Reinforcement Learning**.

4.3. Algoritmo de *Reinforcement Learning*

Para a componente de *Reinforcement Learning* iremos treinar um agente para tomar decisões baseado nos dados históricos da NVIDIA. Para isso, vamos utilizar o algoritmo de *Q-Learning*:

$$Q^{\text{new}}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \times \left(r_t + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right) \tag{12}$$

Definição do *environment*:

- $a_t(\text{action})$ -> corresponde à ação que o agente pode tomar estando num estado s_t. O `num_actions` corresponde a 3 porque são as 3 ações que ele pode tomar:
  - 0 = Hold, que significa manter a posição atual sem realizar nenhuma operação;
  - 1 = Buy, que representa a compra do ativo;
  - 2 = Sell, que corresponde à venda do ativo.
- $s_t(\text{state})$ -> representa a condição atual do mercado baseado nas estratégias estatísticas de *trading* (Secção 4.1). Através de uma função `get_state()` obtemos a discretização do espaço dos estados.

Como foi referido anteriormente, foi discutido o *trade-off* entre a complexidade e a explicabilidade com a performance e concluímos que seria melhor utilizar, nesta função de `get_state`, a primeira estratégia das EMAs apenas. Isto deve-se ao facto das duas estratégias revelarem resultados bons o suficiente, mas com grandes margens de diferença a nível de complexidade de implementação e explicabilidade. Nesse sentido, o *Backtest* da estratégia de *Machine Learning* permanece como está na .

$$\text{int}(\text{EMA}_{\text{fast}} \times 10 + \text{EMA}_{\text{slow}} \times 10) \tag{13}$$

A função retorna um estado, que é um número inteiro representando a combinação das duas EMAs. O valor de cada uma da EMA é multiplicado por 10 para os valores não serem demasiado baixos e existir uma maior distribuição de estados diferentes.

- `q_table` é uma matriz inicializada com zeros de dimensões $\text{num\_states} = 100$ x $\text{num\_actions} = 3$, ou seja, uma tabela que armazenará os valores Q para cada combinação de estado e ação. O algoritmo Q-Learning irá preencher essa tabela com valores que representam a "qualidade" ou a expectativa de recompensa de cada ação em um determinado estado.
- ***Alpha*** (α) -> corresponde à taxa de aprendizagem, ou seja, é um equilíbrio entre velocidade e estabilidade. No nosso contexto, o valor inicial foi de 0.1 no ***baseline model***, sendo posteriormente ajustado, como irá ser referido na Secção 4.4.
- ***Gamma*** (γ) -> corresponde ao fator de desconto, que controla a importância das recompensas futuras em relação às recompensas imediatas. Iniciámos o ***baseline model*** com um valor de 0.95, que será "refinado" na Secção 4.4.
- ***Epsilon*** (ε) -> é a taxa de exploração, inicializada como 1.0 (irá passar pelo mesmo processo de ajuste na Secção 4.4). Por outras palavras, no início, o agente explorará aleatoriamente as ações disponíveis, sem confiar apenas na `Q-table`, daí o valor de 1. O valor de `epsilon` diminui ao longo do tempo, à medida que o agente se torna mais confiante nas suas decisões, o que é controlado pela taxa de `epsilon_decay`, iniciada como 0.995, que serve para reduzir gradualmente a taxa de exploração, permitindo que o agente explore no início e, aos poucos, foque em estratégias aprendidas: **`exploit`**.
- ***reward*** (r_t) -> a *reward* corresponde ao desempenho que do agente ao longo do tempo. Foram utilizados diferentes tipos de *reward* diferentes:
  - **1º caso**: *reward* só é calculada no momento da venda e reflete no lucro ou prejuízo, com base no balanço final e inicial da transação;
  - **2º caso**: *reward* é calculada a partir do Sharpe Ratio (7), onde o objetivo é maximizar os lucros ao mesmo tempo que o risco é minimizado.

Por fim, temos as variáveis `initial_balance` que é definida com um valor inicial de carteira de $1000; `balance` que corresponde ao valor da carteira enquanto o agente é treinado -> este valor a cada início de episódio é igual ao

`initial_balance`; `position` que indica se o agente está sem ativos na sua posse (0), ou se está a "segurar" ativos (1) -> é inicializada como 0; `transaction_fee` é definida como 0.001, o que significa uma taxa de 0.1% sobre cada transação realizada.

Parâmetros Q-Learning *baseline model*:

```python
num_states = 100
num_actions = 3  # 0 = Hold, 1 = Buy, 2 = Sell
q_table = np.zeros((num_states, num_actions))
alpha = 0.1
gamma = 0.95
epsilon = 1.0
epsilon_decay = 0.995
min_epsilon = 0.01
initial_balance = 1000
transaction_fee = 0.001 # 0.001*100 = 0.1% de taxa
episodes = 500 # número de episódios do algoritmo de RL
```

Listing 4: Parâmetros do *baseline model*

4.4. Avaliação, Comparação e *Fine Tune* dos modelos de RL

Após executarmos os modelos criados, foi necessário utilizar métricas de comparação entre modelos e dar *tuning* ao(s) melhor(es) modelo(s).

Como métricas de *Backtesting*[9] e de comparação utilizámos a evolução do saldo/*balance*, o retorno sobre o investimento (ROI), o *Maximum Drawdown* (MDD) e o *Value at Risk* (VaR).

A evolução do saldo/*balance* é, como o próprio nome indica, as variações da carteira do investidor no processo de aprendizagem do algoritmo. Esta evolução será observada nos resultados com um gráfico do último episódio (500).

O *Return on Investment* (ROI) é uma métrica financeira que mede a eficiência ou lucratividade de um investimento. É dado em termos percentuais e informa o retorno obtido em relação ao custo inicial do investimento.

$$\text{ROI} = \frac{\text{balanço final} - \text{balanço inicial}}{\text{balanço inicial}} \times 100 \tag{14}$$

O *Maximum Drawdown* (MDD)[10] é uma métrica utilizada para avaliar a maior perda de uma série de retornos de investimentos em relação a um pico anterior. Ele mede o risco de uma estratégia de investimento ou de um portfólio, onde reflete o pior cenário de perda, durante um período específico.

O MDD é medido como uma percentagem negativa, que indica a perda máxima relativamente ao valor máximo histórico. Assim, um MDD de 0% indica que o valor do portfólio nunca diminuiu em relação ao seu pico anterior e simboliza um investimento constante. Da mesma forma, quanto maior o MDD (mais negativo), maior será a perda em relação ao ponto máximo atingido e, consequentemente, maior será o risco.

$$\text{MDD} = \min\left(\frac{V_t - V_{\text{Peak}}}{V_{\text{Peak}}}\right) \tag{15}$$

O *Value at Risk* (VaR) é uma métrica estatística que serve para medir o risco de perda de um portfólio ou investimento, em um determinado intervalo de tempo e com um nível de confiança específico. A métrica responde à pergunta: "Qual é a perda máxima esperada para o meu portfólio, em condições normais de mercado, com uma determinada probabilidade?"

Por exemplo, se o investimento inicial for de \$1000 e o valor de VaR for 2.11% isso indica que a perda diária máxima do saldo não excederá os 2.11%, com intervalo de confiança de 95%. Ou seja, em 5% dos piores dias, a perda máxima no portfólio seria de $0,0211 \times \$1000 = \$21.1$.

[9] Métricas de avaliação de performance: https://blankly.finance/list-of-performance-metrics/
[10] Explicação detalhada do MDD: https://www.investopedia.com/terms/m/maximum-drawdown-mdd.asp

$$\text{VaR}_{(\alpha=0.95)}\ (\%) = -\text{Quantile}(\text{DailyReturns}) \times 100 \tag{16}$$

Para refinar os modelos (*fine tune*), foram ajustados os valores dos parâmetros na Listing 4, com base na sua *performance*. Inicialmente, foi dada uma lista que continha valores distintos para as várias métricas presentes na fórmula do *Q-Learning*, onde eram executadas todas as combinações possíveis através da biblioteca *itertools* do *Python*. Contudo, como esta abordagem tem uma grande complexidade temporal, foi ajustada para um *Random Search*, que otimiza a escolha dos parâmetros com base nos resultados obtidos ao longo dos episódios.

Os vários valores testados foram os seguintes:

```
alpha_space = [0.1, 0.2, 0.5]
gamma_space = [0.9, 0.95, 0.99]
epsilon_space = [1.0, 0.7, 0.5]
epsilon_decay_space = [0.995, 0.99]
```

Listing 5: Parâmetros que serão alterados no *fine-tune*

Na fase de *fine-tune* serão submetidas duas propostas:

- Uma focada na maximização do *Final Balance*, em que serão utilizados os parâmetros seguintes (após a execução do *Random Search*):

```
alpha = 0.5
gamma = 0.95
epsilon = 1.0
epsilon_decay = 0.995
min_epsilon = 0.01
```

Listing 6: Parâmetros do *tunning model* focado no Final Balance

- Outra focalizada na maximização do *Sharpe Ratio*, onde serão utilizados estes parâmetros (após a execução do *Random Search*)

```
alpha = 0.5
gamma = 0.95
epsilon = 0.5
epsilon_decay = 0.99
```

Listing 7: Parâmetros do *tunning model* focado no *Sharpe Ratio*

Com toda a metodologia apresentada, serão apresentados os resultados obtidos dos modelos de *Reinforcement Learning*, bem como todas as várias tentativas e ajustes efetuados. Os parâmetros para esses ajustes foram descritos nesta secção (Secção 4.4).

5. Resultados

Nesta secção serão apresentados os resultados dos algoritmos treinados por um agente de *Reinforcement Learning* e depois serão ajustados os hiperparâmetros desses modelos. De notar que:

> "Os valores apresentados podem sofrer algumas alterações ao executar o código porque a escolha do modelo optar por *exploitation* ou *exploration* sugere valores aleatórios, que tem implicação direta na Q-Table."

Após executarmos o 1º algoritmo de *Reinforcement Learning* obtivemos os seguintes resultados:

- final balance : $926.96
- ROI $= -7.30\%$
- MDD $= -17.27\%$
- VaR (95% confiança) $= 0\%$

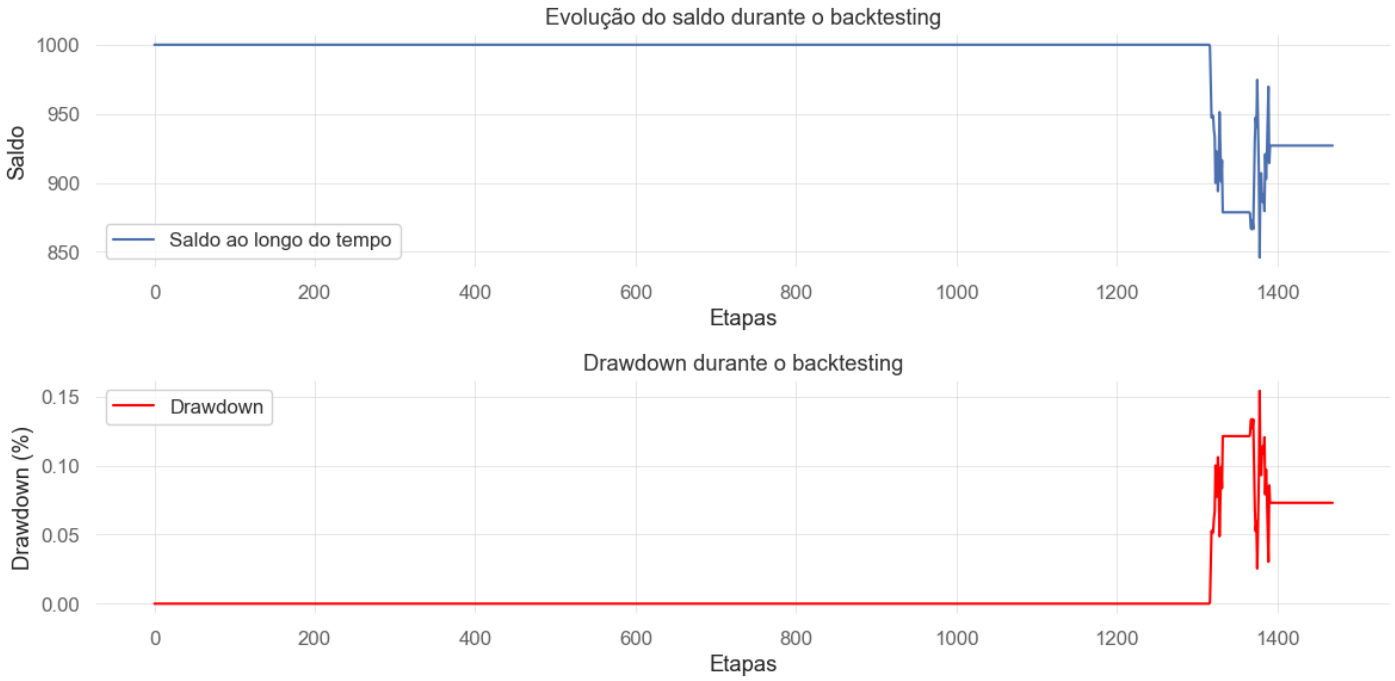

Figura 8: *Backtest* do 1º caso r_t: através do balanço (parâmetros *baseline*) 4

Através destas métricas conseguimos perceber que o modelo gerou uma perda líquida ao longo do período de treino e execução. Conseguimos ver isto, quer pelo valor final de carteira (inferior ao inicial de $1000), quer pelo ROI (-7.30%).

O MDD de −17.27% representa um valor moderado que remete o modelo foi mais conservador e que enfrentou menos volatilidade, mas não foi capaz de se recuperar das perdas para gerar lucros.

O valor de VaR já seria de se esperar que 0 porque houve prejuízo. Contudo, esta métrica não traduz bem os maus resultados do modelo.

Ao observar a Figura 8, é possível verificar que o modelo fica bastante tempo a dar "Hold" e perto do final ele corre um risco baixo para tentar gerar lucro. Porém, acaba por tomar decisões fracas e comprar o ativo numa altura de baixa, que origina numa cadeia sucessiva de dificuldade em gerar lucro.

- final balance : $1113.08
- ROI = 11.31%
- MDD = −6.21%
- VaR (95% confiança) = 0%

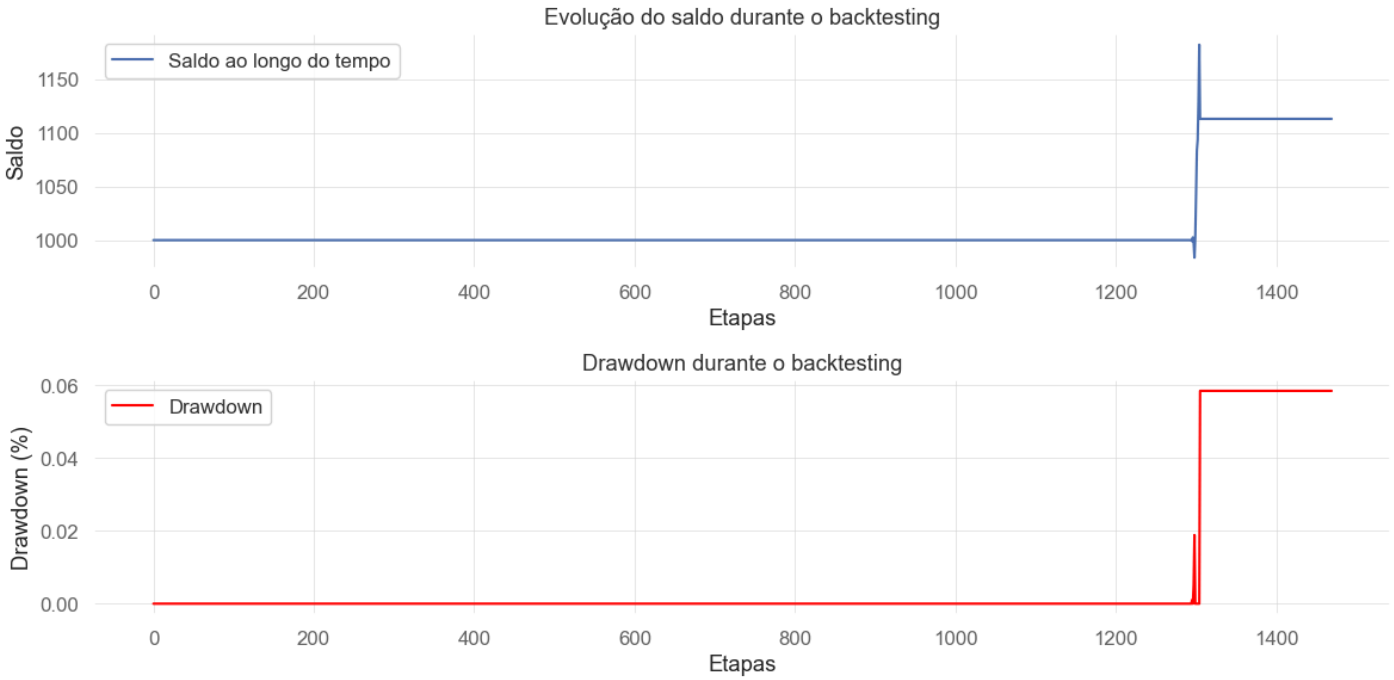

Figura 9: *Backtest* do 2º caso r_t: através do *Sharpe Ratio* (parâmetros *baseline*) 4

O modelo apresentou um *final balance* superior ao inicial de 1000, que indica um lucro líquido no período de *backtesting*. Apesar do resultado positivo, o aumento no valor da carteira é modesto, pois a estratégia continua a ser bastante conservadora e sem muito risco (bastante tempo a dar "Hold" ao ativo). O ROI de 11.31%, à semelhança do *final balance*, demonstra que o modelo foi capaz de gerar um retorno positivo sobre o capital investido.

O MDD de −6.21% indica que o modelo conseguiu limitar as perdas em relação ao pico acumulado do saldo. Este valor relativamente baixo sugere um gerenciamento de risco eficaz, com pouca exposição a grandes flutuações no mercado. Todavia, este valor pode ser "enganador" porque, no período todo de teste, o investimento foi muito reduzido.

O Value at Risk em 0%, para um intervalo de confiança de 95%, que indica que o modelo operou de forma extremamente "preguiçosa" no dia-a-dia, sem apresentar grandes riscos diários. No entanto, como o VaR é uma métrica de curto prazo, ele não reflete totalmente os ganhos ou perdas acumulados.

Os gráficos estão muito refletidos nas métricas apresentadas porque não existiu uma flutuação de investimentos. Por outras palavras, o modelo foi bastante controlado na sua aprendizagem, não

---

Os resultados com os valores de parâmetros do *baseline model* ficaram aquém do esperado, talvez devido ao valor *learning rate* (α) ser baixo. O valor de 0.1 provavelmente impediu o modelo de atualizar a Q-Table de forma significativa a cada iteração, resultando numa convergência lenta ou numa falta de adaptação aos padrões do mercado. Um *learning rate* baixo pode limitar a capacidade do modelo de aprender rapidamente com as recompensas recebidas, especialmente em cenários onde as condições mudam frequentemente, como no nosso caso de *stock trading*.

Após o *Backtesting* foram atualizados os hiperparâmetros do algoritmo de *Reinforcement-Learning*, através dos valores do *Random Search* (Listing 5) e focados em duas métricas diferentes: o *final balance* e *Sharpe Ratio*.

Começaremos por ver os valores obtidos focados para o Final Balance:

- final balance : $1315.21
- ROI = 31.52%
- MDD = −19.96%
- VaR (95% confiança) = 0%

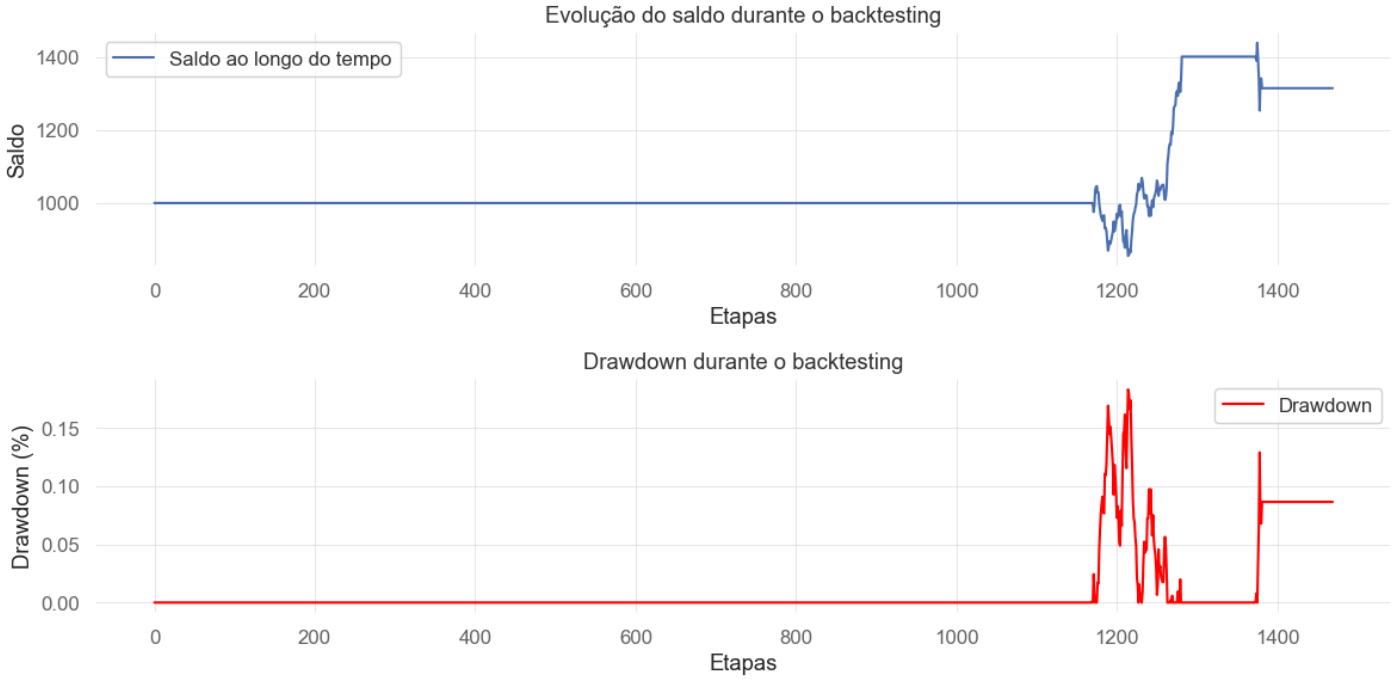

Figura 10: *Backtest* do 1º caso r_t: através do *Final Balance* (parâmetros *tunning* Final Balance) 6

O *final balance* de $1315.21 representa um aumento de 31.52% (ROI) em relação ao valor inicial de $1000. Esse resultado indica que o modelo conseguiu gerar lucros substanciais ao longo do período de *backtesting*, o que demonstra um desempenho positivo.

O MDD de −19.96% evidencia que, em algum momento, o modelo enfrentou uma queda considerável no saldo durante a fase de treino. Contudo, o valor está dentro de um intervalo razoável para estratégias de risco moderado, o que sugere uma gestão adequada de perdas.

O VaR de 0% (a 95% de confiança) implica que o modelo teve um risco muito baixo de apresentar perdas no *final balance* ao longo do tempo. Embora isso seja um bom sinal, também pode sugerir que o modelo priorizou a estabilidade em detrimento de estratégias mais "agressivas".

A partir de aproximadamente 1100 dias, observa-se uma inclinação significativa no saldo, sugerindo que o modelo identificou padrões ou oportunidades no mercado que, consequentemente, resultou numa sequência de operações lucrativas. Após o pico, o saldo estabiliza, que pode refletir decisões mais seguras após ser atingido um nível de lucro confortável.

- final balance : $8273.28
- ROI = 727.33%
- MDD = 44.91%
- VaR (95% confiança) = 0.03%

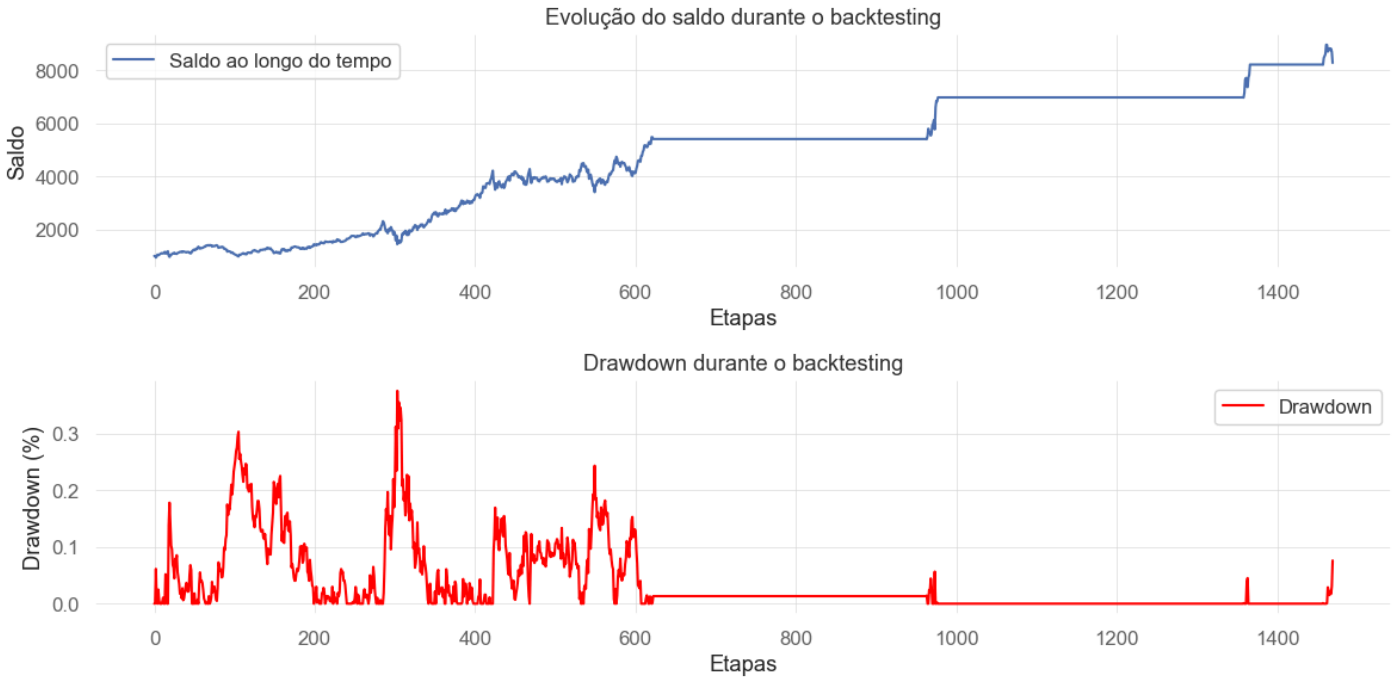

Figura 11: *Backtest* do 2º caso r_t: através do *Final Balance* (parâmetros *tunning Final Balance*) 6

Os valores do saldo final e do ROI foram surpreendentemente positivos, tendo sido obtido um retorno sobre o investimento de mais de 700%.

Um *Maximum Drawdown* de −44.91% indica que o modelo enfrentou momentos de alta volatilidade e risco durante o treinamento. Apesar de ser um valor elevado, o saldo final sugere que o modelo conseguiu superar as perdas temporárias para alcançar resultados positivos nos passos seguintes. Isso mostra que, embora mais agressivo, o modelo foi resiliente.

O VaR de 0.03% (a 95% de confiança) implica que o modelo apresentou baixo risco de perdas severas no saldo final. Isso contrasta com o MDD mais elevado e indica que as perdas mais extremas foram pontuais e não impactaram o valor da carteira de forma contínua.

Após o ajuste dos hiperparâmetros através no *Random Search*, os resultados apresentados refletem uma melhoria significativa no desempenho do modelo focado no *final balance*.

Este ajuste de parâmetros, em particular o aumento no $\alpha = 0.5$, parece ter permitido ao modelo aprender de forma mais eficiente durante o treino. Além disso, o `epsilon_decay` (0.995) e o `min_epsilon` (0.01) garantiram um equilíbrio adequado entre exploração e exploração, permitindo ao modelo explorar oportunidades no início e refinar sua política no final.

Embora o MDD mostre que ainda houve momentos de maior risco, o saldo final e o ROI positivos demonstram que a estratégia geral com estes parâmetros foi bem sucedida. Isto deve-se ao facto de estarmos a focar-nos no valor do *final balance* apenas, em que o modelo não dá um peso significativo ao risco. Ajustes na medida do risco podem explorar uma maior diversificação para reduzir o *Drawdown* enquanto mantêm a lucratividade.

- final balance : $2081.65
- ROI = 108.16%
- MDD = −84.33%
- VaR (95% confiança) = 0.04%

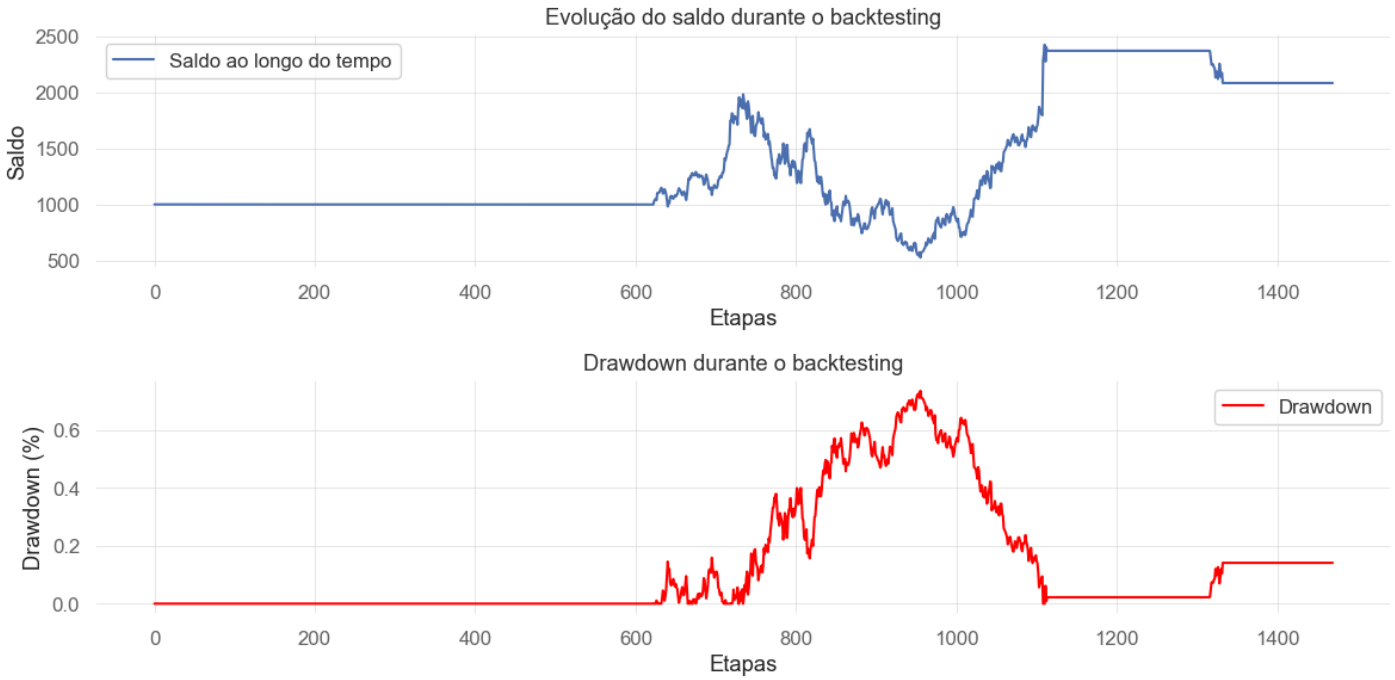

Figura 12: *Backtest* do 1º caso r_t: através do *Sharpe Ratio* (parâmetros *tunning Sharpe Ratio*) 7

O saldo final de $2081.65 indica um crescimento significativo em relação aos $1000 iniciais, o que demonstra que o modelo conseguiu gerar lucro substancial. Podemos ver isto refletido no ROI de 108.16%, que diz que o investimento foi duplicado no período analisado.

O MDD de −84.33% é extremamente elevado, o que significa que, em determinado momento, o saldo do modelo caiu para apenas 15.67% do capital inicial antes de se recuperar. Este valor alto de *drawdown* indica que o modelo esteve exposto a decisões ou condições de mercado muito desfavoráveis, colocando o capital num risco risco significativo.

O VaR de 0.04% (a 95% de confiança) sugere um risco baixo de perdas severas no saldo consolidado, mas essa métrica não captura adequadamente o risco elevado representado pelo MDD. Por essa razão, o VaR deve ser interpretado com cuidado neste contexto, já que não reflete os momentos de grande volatilidade enfrentados pelo modelo antes de chegar a uma estabilidade (nas últimas etapas do modelo; em que o saldo manteve-se mais ou menos constante).

Para além disso, pela Figura 12, conseguimos perceber que há um momento do tempo em que são efetuadas várias ações em que o modelo explora mais e onde o risco aumenta. Após esta fase, os investimentos estabilizam e o saldo não é tão afetado, talvez pela capacidade de aprendizagem através da Q-table ter mais peso.

- final balance : $1279.15
- ROI = 27.91%
- MDD = −25.42%
- VaR (95% confiança) = 0%

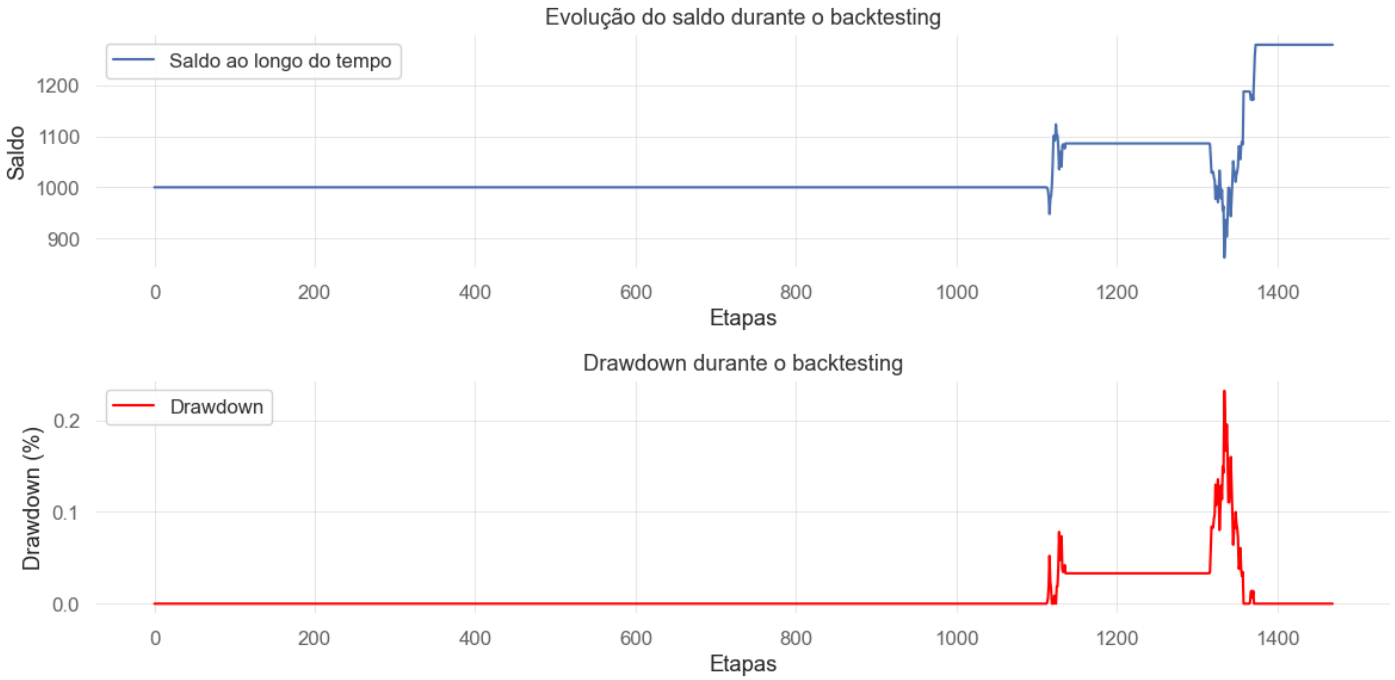

Figura 13: *Backtest* do 2º caso r_t: através do *Sharpe Ratio* (parâmetros *tunning Sharpe Ratio*) 7

O *final balance* de $1279.15 representa um crescimento em relação ao inicial de $1000, indicando que o modelo conseguiu gerar lucro ao longo do período. Este resultado traduz num desempenho razoável, com um retorno de 27.91%. O que mostra que o modelo teve um crescimento limitado no capital inicial, mas que pode ser mais adequado para cenários em que há preservação de capital e de risco.

O *Drawdown* máximo de −25.42% reflete um período significativo de perdas temporárias durante o *backtesting*. Esse valor sugere que o modelo foi enfrentou dificuldades em alguns momentos, chegando a perder até um quarto do capital inicial, antes de se recuperar.

O VaR de 0% (a 95% de confiança) indica que, no saldo final, o modelo apresentou um risco reduzido de perdas severas. Isto significa que, ao longo do período analisado, o saldo manteve-se suficientemente estável no momento de maior impacto sobre os resultados.

Após analisar os resultados focados no *Sharpe Ratio*, podemos concluir que os modelos apresentaram um comportamento mais estável e menos arriscado em comparação aos anteriores, demonstrando uma melhor preservação do capital, ao gerenciar o risco de forma mais eficaz. Além disso, os valores configurados para `epsilon` (0.5) e `epsilon_decay` (0.99) proporcionaram uma maior "confiança" na Q-table, favorecendo decisões mais consistentes e equilibradas ao longo do tempo.

6. Conclusões

Pelos resultados obtidos, concluímos que a *NVIDIA* é uma empresa em constante evolução[11], sustentada pela sua incrível tecnologia focada em inteligência artificial e pelo desenvolvimento das *GPUs* mais potentes do mercado. Consequentemente, as ações da *NVIDIA* têm apresentado um crescimento exponencial[12] (Visualizar Anexo A) resultado do elevado prestígio que a empresa tem alcançado.

Dado este contexto, já era de se esperar que investir na *NVIDIA* pudesse ser considerado "seguro" – ainda que seja sempre necessário ter em mente que o mercado acionista envolve riscos inerentes. O impressionante crescimento da *NVIDIA* reflete-se numa elevada valorização das suas ações, tornando mais previsível o seu comportamento futuro e oferecendo, aparentemente, uma oportunidade atrativa para investidores.

Com base nas nossas análises e resultados, as estatísticas (Secção 3.3) demonstram que, apesar de ser relativamente "seguro" investir na *NVIDIA*, tal não se traduz, necessariamente, em lucros significativos em larga escala. Para testar esta premissa, foram exploradas duas estratégiaspara prever o comportamento das ações da NVIDIA:

1. **Modelos de machine learning - Secção 4.2 (regressão - Secção 4.2.1 e classificação - Secção 4.2.2)**
2. **Reinforcement learning - Secção 4.3** (apoiados pela estratégia de **crossover de média móvel exponencial - Secção 4.1.1).**

Modelos de Machine Learning

- O modelo de regressão mostrou-se inadequado para este contexto, pois o objetivo era criar um sistema capaz de comprar e vender *shares*, algo inviável para este tipo de abordagem. Estes modelos limitam-se à previsão de valores, como a variável `close`, utilizada neste caso. Embora promissora, esta abordagem enfrenta desafios significativos e permanece em desenvolvimento[13]. Mesmo com modelos avançados como o LSTM, o problema de overfitting manteve-se evidente (ver Figura 4). Este resultado era esperado, dada a alta volatilidade do mercado acionista, que dificulta a previsão precisa dos seus valores.
- Por outro lado, o modelo de classificação revelou-se mais adequado, proporcionando resultados interessantes, apesar de diversas limitações. As *labels* tiveram de ser definidas manualmente pelo grupo, utilizando um *threshold*, o que dificultou determinar a melhor altura para comprar ou vender. Além disso, este modelo não consegue adaptar-se ao longo do tempo, sendo treinado apenas uma vez – como discutido na Secção 4.2.2.2. Ainda assim, o desempenho foi surpreendente, alcançando um lucro de $952, um resultado que excedeu as expectativas do grupo, embora tenhamos plena consciência de que não é viável para aplicação no mundo real.

Modelos de Reinforcement Learning

- Este modelo foi consideravelmente mais trabalhoso do que o de *Machine Learning*. Inicialmente, foi necessário definir uma estratégia, o que exigiu um estudo detalhado sobre o comportamento das ações da *NVIDIA*. Após esta análise, o grupo concluiu que a melhor abordagem seria utilizar a EMA (9), dado o crescimento exponencial das ações da empresa nos últimos meses. Na Tabela 5, são apresentados vários valores de `span` aplicados na estratégia de *crossover*, sendo fundamental realizar múltiplos testes para maximizar o **CumReturn** (11). Após diversas iterações, determinou-se que os parâmetros ideais eram: EMA *fast* de 50 e EMA *slow* de 140.
- Com a estratégia definida, passámos finalmente para a criação do modelo de *reinforcement learning*. Para tal, estabelecemos algumas bases, como o número de ações possíveis (**Buy**, **Sell**, **Hold**), e implementámos e adaptámos a

[11] https://www.linkedin.com/pulse/ascens%C3%A3o-exponencial-da-nvidia-conquistas-passadas-e-ewaldo-del-valle-gchzf/
[12] https://pt.investing.com/equities/nvidia-corp-technical
[13] https://repositorio.ulisboa.pt/handle/10400.5/19517

estratégia mencionada anteriormente. De seguida, optámos pelo algoritmo de *Q-learning*, tendo criado 2 diferentes: o primeiro com **reward** do balanço atual - inicial, e o segundo com **reward** do **sharpe ratio**..

- A seguir, chegámos à fase mais desafiante deste trabalho: a definição dos hiperparâmetros. Para tal, foram definidas 3 modelos:
  - O modelo baseline (Listing 4) não teve um desempenho satisfatório, chegando até a gerar prejuízo quando comparado aos modelos de machine learning.
  - Para os restantes modelos, estes foram baseados na técnica de *random search*, com o objetivo de realizar o *tuning* dos hiperparâmetros. Consequentemente, chegámos aos seguintes resultados:
    - **Modelo focado em Lucro** (Listing 6): Este modelo apresentou um risco elevado, como evidenciado pelas várias tentativas do grupo ao rodar o código, que resultaram sempre em resultados insatisfatórios, como os demonstrados na Figura 21. Contudo, apesar desses desafios, foi este o modelo que obteve o melhor resultado em termos de lucro, conforme ilustrado em Figura 11.
    - **Modelo focado em *Sharpe Ratio*** (Listing 7): Ao contrário do outro modelo, este tinha um foco principal e ser seguro, sendo também a recomendação principal do *Fine-Tune Algorythm*. Estes hiperparâmetros estavam mais focados em garantir a obtenção de lucro. O objetivo foi atingido, já que ambas as versões do modelo geraram lucro, sendo que uma delas superou o modelo de **machine learning** em termos de lucro, embora com um desempenho inferior aos hiperparâmetros otimizados para maximizar o lucro.
  - Estes modelos destacaram-se por serem mais apropriados para este tipo de problema, pois têm a capacidade de melhorar continuamente com o passar do tempo, uma característica que os modelos de machine learning não conseguem replicar de forma eficaz.

Reflexão Final

Concluímos que, embora os modelos desenvolvidos ao longo deste trabalho tenham apresentado lucros, é importante destacar que a NVIDIA é um caso especial, cuja previsibilidade facilitou a obtenção de resultados positivos. Este facto contribuiu significativamente para os lucros registados. Assim, ressaltamos que estes modelos não são suficientemente robustos para serem aplicados no mundo real com garantias de lucro, uma vez que apresentam limitações, especialmente no caso dos modelos de **machine learning**, em que os resultados dependeram, em parte, de "sorte".

Adicionalmente, concluímos que os modelos de **reinforcement learning** são mais adequados para este tipo de problemas, pois têm a capacidade de se adaptar e melhorar com o tempo, algo que os modelos de **machine learning** não conseguem fazer da mesma forma. O código deste trabalho encontra-se no seguinte repositório: GitHub.

Figura 14: Logotipo da *NVIDIA*

Anexos

Anexo A - *Dashboard* com os dados iniciais

Figura 15: Gráfico OHLC (*Dashboard*)

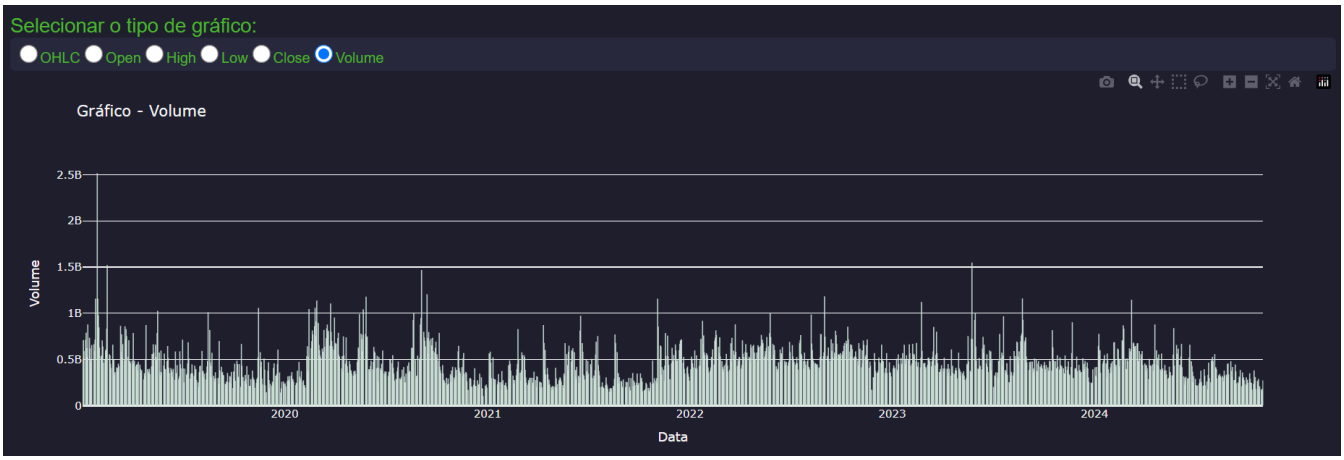

Figura 16: Variável `Volume` (*Dashboard*)

Anexo B - Gráficos EMA para valores da Tabela 5 (restantes)

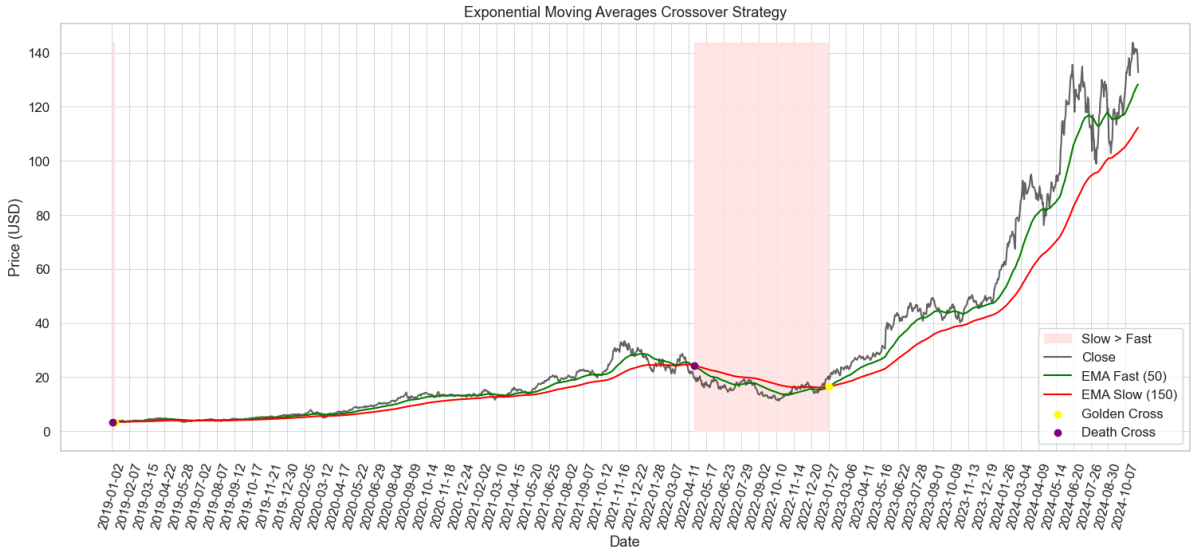

Figura 17: Estratégia de *Crossover* EMA(*fast*:50, *slow*:150)

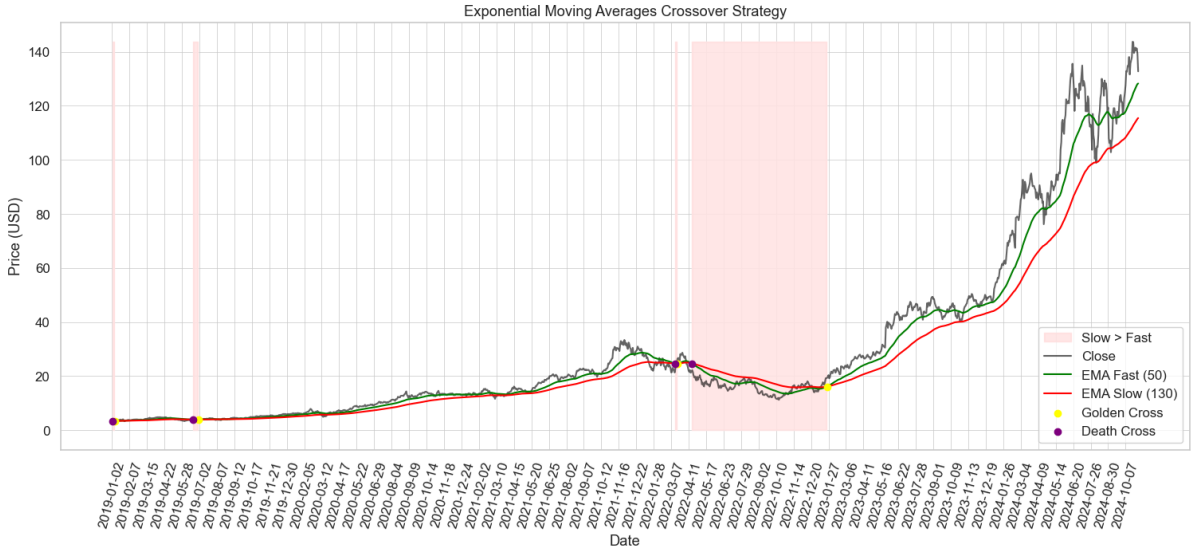

Figura 18: Estratégia de *Crossover* EMA(*fast*:50, *slow*:130)

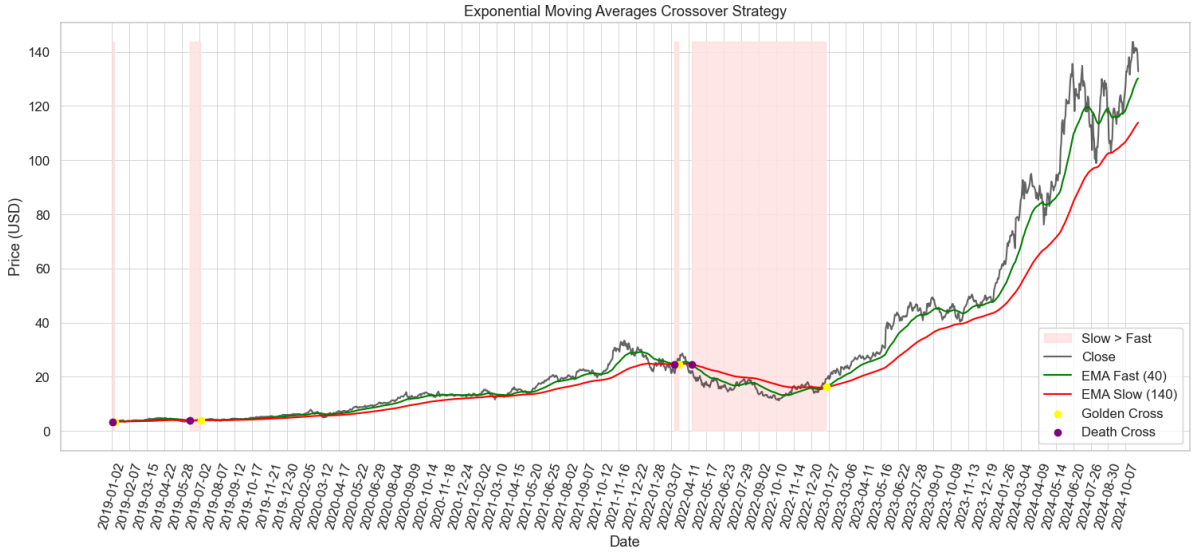

Figura 19: Estratégia de *Crossover* EMA(*fast*:40, *slow*:140)

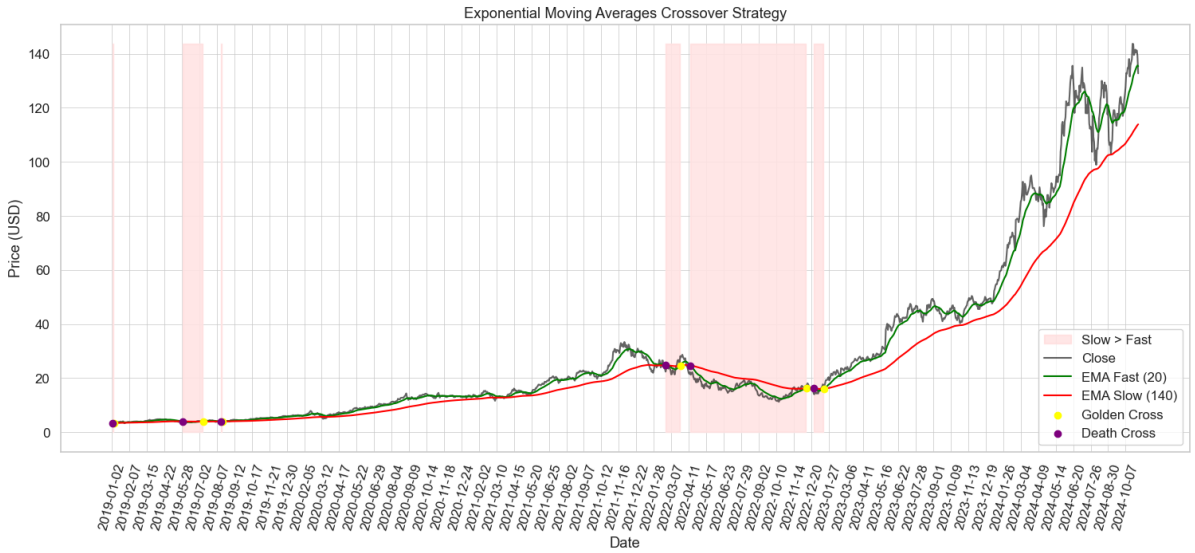

Figura 20: Estratégia de *Crossover* EMA(*fast*:20, *slow*:140)

Anexo C - Mau exemplo de parâmetros Reinforcemente Learning

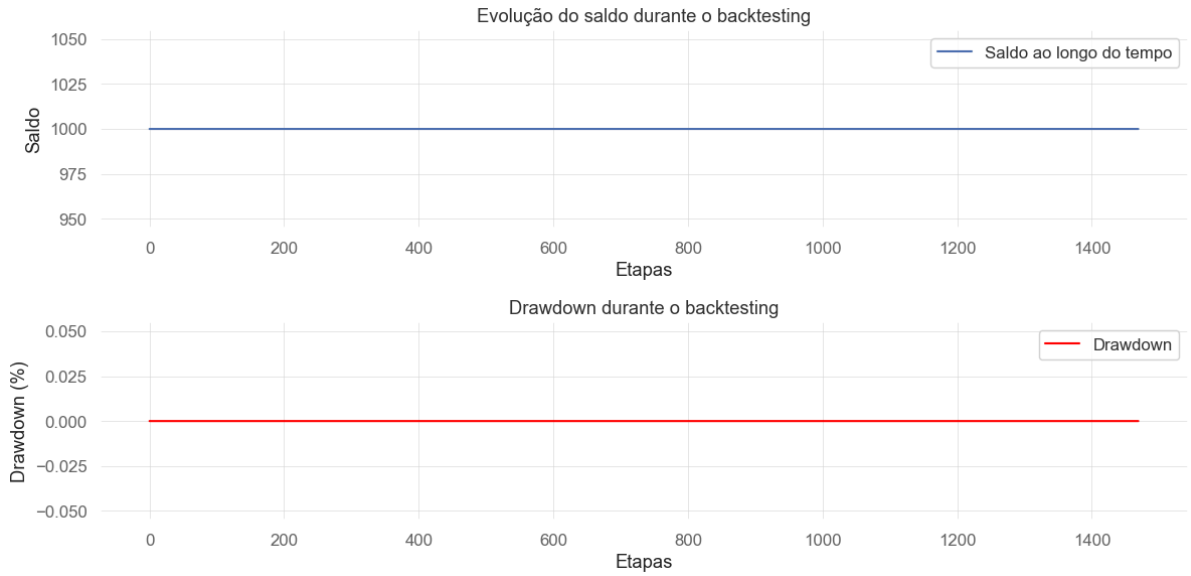

Figura 21:

```
Final balance: 1000
Return on Investment (ROI): 0.00%
Maximum Drawdown (MDD): 0.00%
Value at Risk (95% confidence): 0.00%
```

Listing 8: Métricas de um modelo de **Reinforcement Learning** quando a taxa de aprendizagem (*learning rate*) se aproxima de 0.