Reproduzindo Deep Reinforcement Learning aplicado ao problema de Optimal Stopping em ações Brasileiras

João Lucas de Moraes Barros
Cadorniga
Insper
São Paulo
joaolmbc@al.insper.edu.br

***Abstract*—This work investigates whether a Deep Reinforcement Learning (DRL) agent can improve the optimal stopping (exit) decision in equity trading when combined with a simple, fixed entry rule. Building on a recent DRL approach for optimal stopping, we reproduce its methodology using daily data from 45 liquid Brazilian stocks between 2019 and 2024 and train a dueling Double DQN with LSTM architecture to decide when to exit long positions. For out-of-sample backtesting, position entries are triggered only when a 2-day RSI falls below 5, and the learned exit policy is evaluated on the Ibovespa index and two representative stocks (PETR4 and ITUB4) in 2025, against three benchmarks: an RSI mean-reversion strategy, buy-and-hold, and a CDI cash strategy. Overall, results suggest that DRL-based optimal stopping has high potential but this version exhibits limited robustness and dependence on the RSI input.**

Keywords—Reinforcement Learning, Optimal Stopping, Algorithmic Trading, Deep Learning, Backtesting.

I. Introdução

Estratégias de trading algorítmico consistem de dois componentes principais de decisão: quando entrar em uma posição (comprar) e quando sair da mesma (vender). Atualmente, encontrar um bom momento para criar uma posição é o menor entre os dois desafios [3]. O uso de indicadores técnicos, como o "*Relative Strength Index*" (RSI), o "*Moving Average Convergence Divergence*" (MACD), entre outros sinais (por exemplo, utilizando modelos de Machine Learning), conseguem identificar boas condições de entrada [3]. Até opções pouco embasadas em teoria, como a apresentada por Larry Connors no livro *Short-Term Trading Strategies That Work* (2009) [2], a qual utiliza apenas RSI de período 2, não produzem pontos de entrada significativamente ruins.

No entanto, o problema de saída, também conhecido como "*optimal stopping*", continua sendo uma das maiores dificuldades de estratégias de trading algorítmico. Saber o momento certo de vender uma posição, balanceando o possível ganho futuro com o risco, pode ser o grande diferencial de uma estratégia superior. Métodos tradicionais, como *Stop Losses, Take Profits,* ou *Trailing Stops*, podem ser rígidos demais e ter dificuldades de se adaptar em momentos diferentes do mercado.

O uso de Reinforcement Learning (RL) no mercado financeiro possui extensivas pesquisas [4, 5]. No entanto, com frequência aplicando modelos em todas as etapas do processo de trading. RL aplicado apenas para *optimal stopping* possui trabalhos importantes nos últimos anos, como *A. Fathan et al* no artigo "Deep reinforcement learning for optimal stopping with application in financial engineering" [1], a maior inspiração deste trabalho. Mesmo assim, é incomum encontrar a mistura de modelos de inteligência artificial para saída com indicadores já conhecidos para entrada, testando um modelo híbrido para backtestings reais, e mais raramente ainda em ações do mercado brasileiro.

Portanto, o objetivo deste trabalho é realizar um experimento aplicando Deep Reinforcement Learning para treinar um agente que realiza *optimal stopping* no mercado brasileiro. Para tal tarefa, é proposta a reprodução da metodologia apresentada por Fathan [1] em ações brasileiras, combinado com uma medida de RSI para entrada de posições, e comparando com 3 benchmarks: uma estratégia apenas usando o RSI, o *CDI* (*risk-free rate*), e *Buy & Hold*.

Como ferramentas, este trabalho utiliza *Python* com a biblioteca *backtrader* para todas as simulações. Além disso, são usadas bibliotecas como *torch, gymnasium, pandas, numpy e yfinance* para obtenção e tratamento de dados, desenvolvimento do ambiente de RL e treinamento do modelo.

II. Metodologia

A metodologia do trabalho envolve replicar o que foi realizado por [1], porém treinado com dados de ações brasileiras e testado no índice Ibovespa.

A. Dados Utilizados

Toda a coleta de dados foi feita utilizando a biblioteca *yfinance*. Em seu paper, *Fathan et al* [1] usou dados diários de 111 ações do S&P500 (as 500 empresas mais valiosas da bolsa americana) de 2014 a 2019. Desses, 60 ações de 2014-2016 para treino e o restante para teste. Visando criar um ambiente semelhante, levando em consideração a menor variedade de ações no mercado brasileiro, mas mantendo a divisão de *Out-of-Sample* para integridade dos testes, foram coletados dados de 45 das mais conhecidas ações do mercado brasileiro (apêndice 1) de diferentes ramos. Os dados coletados também foram diários, de 01/01/2019 até 31/12/2024, divididos em treino, validação e teste (70%, 15% e 15% respectivamente) para a rede neural. Para Backtesting, foram usados dados do índice Ibovespa de 2025 até o presente (visando testar um instrumento diferente, mas que ainda tenha a mesma base das ações que

o agente usou para aprender), mas também alguns testes utilizando ações da lista original.

Visto que a proposta deste trabalho é utilizar o índice RSI para entrada, o primeiro teste para geração de episódios foi gerá-los apenas quando houvesse um sinal de entrada, mantendo fidelidade com os cenários reais do Backtesting. No entanto, isso causaria a existência de apenas aproximadamente 1,300 episódios únicos, o que não é suficiente para o modelo generalizar, rapidamente causando overfitting. Portanto, foi adotada a mesma metodologia de [1], de gerar episódios com samples aleatórios e únicos dos dados para gerar 15,000 episódios.

Cada episódio possui 12 dias de "aquecimento" de dados anteriores ao dia da entrada da posição (seguindo [1] para "aquecer" a arquitetura de Long-Short Term Memory, ou LSTM), e com uma duração para o futuro de 20 dias, que foi estabelecido como o máximo tempo de hold, com o objetivo de o agente aprender a procurar o momento certo de venda, e não segurar a posição indefinidamente.

Para o conjunto de testes do modelo, a geração utilizou apenas episódios que começam com sinal de entrada de RSI de 2 dias menor que 5. Esses números foram escolhidos baseado na estratégia de Larry Connors, e não foi realizada busca de hiperparâmetros ótimos.

B. *Arquitetura do Agente de Reinforcement Learning*

O agente escolhido foi a iteração mais simples utilizada em [1]: um DDQN com duas redes LSTM e "*dueling architecture*". Não foi realizada busca de hiperparâmetros, todos foram escolhidos como clássicos valores gerais de Reinforcement Learning de trabalhos anteriores (detalhes de hiperparâmetros no apêndice 2).

A entrada do modelo tem tamanho 19, consistindo de:

- Os últimos 15 preços da ação (janela de *lookback* escolhida baseado em [1]);
- O *PnL* (profit and loss) percentual (em decimal) do dia atual em relação ao momento de entrada;
- O tempo restante de trade (normalizado pelo número máximo de dias), para prover ao agente a noção de tempo;
- O RSI do dia atual (normalizado de 0-1);
- A volatilidade dos últimos 15 preços.

Todos os parâmetros de entrada foram baseados em [1].

C. *Ambiente*

Como citado na seção anterior, portanto, a observação do estado do ambiente é de dimensão 1x19, de episódios de 20 dias máximos e 12 dias de "aquecimento" da rede, nos quais nenhuma decisão era realizada, apenas dados ingeridos.

O espaço de ação do ambiente é de tamanho 2: vender (1) ou não a posição (0). Assim, a saída do modelo é a ação com maior Q-value.

A função de reward foi escolhida também alinhada com o trabalho de *Fathan et al* [1]. No momento em que o agente decide encerrar a posição, a recompensa é calculada como o retorno percentual simples da operação:

$$R_t = \frac{P_{exit} - P_{entry}}{P_{entry}}$$

Caso a decisão do agente seja de não terminar a posição, a recompensa é 0. Essa estrutura esparsa tem o objetivo de forçar o agente a aprender o melhor momento de venda que otimize o lucro e equilibre o custo de oportunidade do presente com o futuro.

No entanto, é importante notar que existem outras ideias que não foram experimentadas no presente trabalho, como incluir um "horizonte" de passado e futuro para subtrair do retorno do momento de saída o melhor retorno. O objetivo dessa estratégia é penalizar o agente que deixa de ganhar dinheiro no futuro ou no passado. No entanto, essa função de recompensa não foi testada nessa iteração.

D. *Processo de Treinamento e Teste da Rede Neural*

O treinamento do modelo do agente foi feito com os 15,000 episódios citados na seção de *Coleta de Dados*, durante 5 *epochs*. Após cada *epoch*, o modelo era aplicado ao conjunto de validação. Ao final do treinamento, ele foi testado no conjunto de testes (mais detalhes de hiperparâmetros no apêndice 2).

E. *Processo de Backtesting e Benchmarking*

Como citado anteriormente, o Out-of-Sample (OOS) Backtesting foi realizado em dados do índice de Ibovespa de 2025 (Janeiro à Outubro), e de duas ações escolhidas do conjunto (PETR4 e ITUB4), no mesmo período.

O agente de RL foi comparado com três benchmarks:

1. Estratégia de reversão à média com RSI por Larry Connors: com período de 2 dias e upper e lower bounds de 80 e 5, respectivamente;
2. Buy and Hold: comprar posição inteira no primeiro dia e vender no último;
3. *Risk-free*: manter todo o caixa rendendo a taxa CDI.

Para esses experimentos, as seguintes premissas foram adotadas:

- Todas as estratégias possuem Carry de CDI diário calculado com uma taxa anual fixa de 14.15% (aproximando o comportamento da taxa em 2025);
- Todas as estratégias tiveram custos de emolumentos de $0.005\% + 0.0275\%$ integrado como comissão da biblioteca *backtrader*;
- O agente RL manteve 15 dias de espera para possuir *lookback* completo dos dados e uma duração de trade máxima de 20 dias para evitar grandes perdas. Outras estratégias mais robustas podem ser implementadas no futuro.
- Foram computadas diversas métricas sobre cada experimento (Sharpe, CAGR, Drawdown, etc.) utilizando a biblioteca. Detalhes podem ser encontrados no código-fonte.

III. Resultados

Os resultados deste trabalho podem ser divididos em duas seções, referentes ao treinamento do agente de RL, de como o agente se comportou em um cenário real de *backtesting*.

A. *Treinamento do Modelo*

Após o treinamento do modelo de 5 *epochs*, o agente atingiu um reward médio por episódio (que equivale a um trade) de aproximadamente 0.70% no conjunto de treinamento e 0.80% no conjunto de validação, como é possível analisar no gráfico abaixo:

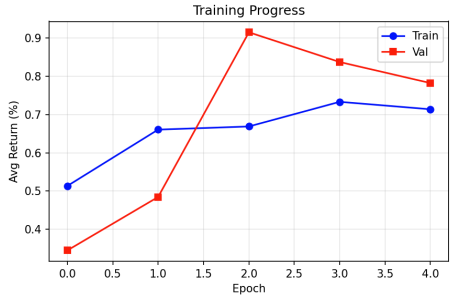

Fig. 1. Retorno médio (recompensa) ao longo das epochs

A recompensa média do conjunto de validação é maior que o conjunto de treinamento não é usual, mas pode ser explicada pela divisão de treino-teste, nos quais podem ter sido separados episódios em que o agente tende a se sair melhor.

Já no conjunto de testes, vemos um resultado mais próximo do esperado, com um retorno médio de 0.25%. O agente foi comparado com outras estratégias simples de parada nesse mesmo conjunto. Lembrando que esse conjunto de episódios foi gerado com a condição de entrada do RSI, diferente do conjunto de treinamento, para se aproximar ao cenário real.

TABELA I. Comparação de estratégias no conjunto de teste

Estratégia	Retorno Médio (%)	Mediana do Retorno	Desvio Padrão do Retorno	Win Rate (%)	Média de duração (dias)
RL Agent	0.26	2.10	9.58	60.33	22.46
Fixed 5%	-0.34	3.69	11.24	55.65	13.58
Stop Loss -2%	-0.37	-2.72	9.87	26.74	9.81
Trailing 3%	-0.12	-1.59	7.94	36.52	7.31
RSI Exit >70	-0.88	-1.02	13.21	46.20	17.59

O agente de RL apresentou desempenho melhor que as outras estratégias de parada, as quais todas, em média, tiveram retornos negativos. Vale notar também que o seu desvio padrão não foi muito diferente das outras estratégias, mas seu Win Rate foi maior. Além disso, os trades tiveram duração média superior a 20 dias (o limite de duração do treino não foi aplicado no teste). Uma possível explicação pode ser que o agente tendeu a demorar a vender em episódios nunca vistos antes. A duração dos trades também pode ser observada no gráfico:

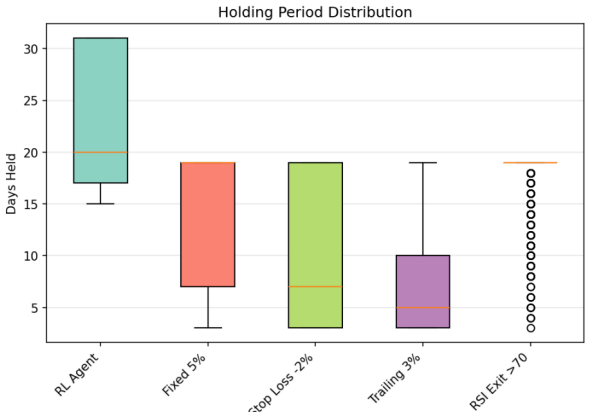

Fig 2. Trades do agente de RL tendem a ser bem mais longos.

B. *Backtesting OOS*

Após aplicar todas as estratégias nos dados do índice Ibovespa, no período de Janeiro à Outubro de 2025 (10 meses de dados), as principais métricas encontradas foram:

TABELA II. Comparação de estratégias no backtesting

Estratégia	Retorno Total (%)	Sharpe	Max Drawdown (%)	Win Rate (%) e (Número de Trades)	Profit Factor	Duração Média (dias)
RL Agent	16.52	1.08	250.17	60 (5)	7.68	7.2
RSI-only	13.26	0.52	250.85	75% (4)	5.82	5.82
Buy & Hold	21.04	N/A	N/A	100 (1)	N/A	N/A
CDI-only	11.08	N/A	N/A	N/A	N/A	N/A

O agente obteve resultados superiores à maioria dos benchmarks, mas não conseguiu operar melhor que Buy & Hold do índice. Seu Sharpe foi razoável, e o número de trades e *max drawdown* foram muito semelhantes a estratégia com apenas RSI. Outras estratégias de entrada podem ser testadas para entender como o agente se comporta quando realiza trades mais frequentes. É também interessante notar que os trades tiveram duração significativamente mais curta que no conjunto de teste do modelo. Uma possível explicação é que o conjunto anterior poderia ser muito diferente e o modelo não teve boa capacidade de generalização, optando por manter posições pelo máximo possível.

Outro aspecto que indica pouca robustez do modelo é que, ao reproduzir a simulação repetidas vezes para o índice, os resultados variaram. O retorno total esteve em um intervalo de 11% (que seria inferior a estratégia apenas com RSI) a 20% (comparável ao Buy & Hold). Essa variância alta pode tentar ser remediada com um modelo de

arquitetura mais robusta, ajustes na função de recompensa ou, principalmente, mais dados de treinamento.

Para entender se o modelo também teria desempenho razoável em cenários diferentes e não apenas no índice que tendeu a subir, as estratégias também foram aplicadas às ações PETR4 e ITUB4, que tiveram distribuições diferentes no mesmo período:

TABELA III. Comparação de estratégias no backtesting de PETR4

Estratégia	Retorno Total (%)	Sharpe	Max Drawdown (%)	Win Rate (%) e (Número de Trades)	Profit Factor	Duração Média (dias)
RL Agent	0.76	-0.68	1636	71.4 (7)	0.65	11
RSI-only	0.83	-0.67	1756	62.5 (8)	0.66	8.6
Buy & Hold	-13.02	N/A	N/A	0 (1)	N/A	N/A
CDI-only	11.08	N/A	N/A	N/A	N/A	N/A

TABELA II. Comparação de estratégias no backtesting de ITUB4

Estratégia	Retorno Total (%)	Sharpe	Max Drawdown (%)	Win Rate (%) e (Número de Trades)	Profit Factor	Duração Média (dias)
RL Agent	14.37	1.09	143.82	50 (4)	2.78	3.5
RSI-only	14.14	1.10	186.06	66.7% (3)	14.38	5.3
Buy & Hold	41.65	N/A	N/A	100 (1)	N/A	N/A
CDI-only	11.08	N/A	N/A	N/A	N/A	N/A

No cenário de um instrumento que teve queda no período (PETR4), tanto o agente quanto a estratégia de RSI-only quase não conseguiram retorno positivo. Ambas tiveram perdas muito grandes, apesar de ganharem mais do que perderam. A duração média de trade foi ligeiramente mais longa que o experimento anterior, mas comparável. O Buy & Hold, como esperado, teve apenas uma perda enorme com a queda da ação.

Ademais, no cenário de um instrumento em alta (ITUB4), os resultados voltaram a ser semelhantes com o Ibovespa. No entanto, novamente também não alcançou o ganho do Buy & Hold.

A alta semelhança entre os resultados do agente com a estratégia RSI-only são interessantes, pois podem indicar ineficiência no treinamento do modelo. Ele não foi capaz de generalizar melhor que a simples métrica, e não conseguiu lidar tão bem com cenários diversos. Ao final, aparenta ter "convergido" para ser muito semelhante a simples técnica de RSI-only. Uma possível explicação pode estar ligada a entrada da rede neural, que inclui o RSI do dia. Isso pode ter enviesado a rede neural e essa informação pode ter ganho importância alta, ditando as decisões, como a estratégia RSI-only.

Todos os gráficos de cada estratégia foram omitidos para evitar poluição da seção, mas podem ser encontrados no apêndice 3.

IV. Conclusão e Trabalho Futuro

Treinar um agente de Reinforcement Learning no problema de *optimal stopping* para aplicá-lo a um sistema de trades é uma estratégia que demonstra potencial, com retornos positivos que batem o CDI em cenários de instrumentos que estão subindo. No entanto, não operou muito bem com instrumento que está em baixa.

Ademais, o agente ainda apresenta algumas inconsistências. As diferenças de alguns resultados, como a duração dos trades e o retorno médio entre os conjuntos de validação, teste e backtesting podem indicar dificuldade de generalização do modelo para mais cenários. Ao fazer simulações consecutivas com os mesmos dados, o agente apresentou variância maior que o esperado, também indicando instabilidade. O agente acabou obtendo desempenho muito semelhante à estratégia simples de RSI, mostrando que essa medida pode ter possuído alta influência na tomada de decisão.

Para trabalhos futuros, existem limitações claras que podem ser exploradas para melhorias no desempenho do agente:

1. Aumentar o conjunto de dados, para realizar treinamentos mais robustos e possivelmente com episódios apenas com sinal de entrada;
2. Alterar a função de recompensa, para considerar "penalizar" o agente comparando sua saída com a melhor saída em um horizonte de dias;
3. Realizar backtesting mais robusto com mais instrumentos para mais dados de comparação;
4. Alterar a entrada do modelo, pois o valor de RSI pode o estar enviesando;
5. Realizar demais testes estatísticos, como Diebold-Mariano para comparar estratégias semelhantes (agente versus RSI-only) e simulação de Monte Carlo para melhor testar a variância do modelo em um grande número de cenários.

No final, conclui-se que esse tipo de agente tem alto potencial ao apresentar resultados satisfatórios com treinamento de complexidade baixa e possíveis mudanças que podem melhorar suas métricas significativamente.

Todo o código do projeto pode ser encontrado em: https://github.com/JoaoLucasMBC/rl-algotrading

Referências

[1] A. Fathan and E. Delage, “Deep reinforcement learning for optimal stopping with application in financial engineering,” arXiv preprint arXiv:2105.08877, 2021. doi: 10.48550/arXiv.2105.08877.

[2] L. Connors and C. Alvarez, “Short-Term Trading Strategies That Work”. MarketPlace Books, 2009.

[3] R. Ikeda, “Algotrading class”. Insper, 2025.

[4] Z. Zhang, S. Zohren, and S. Roberts, “Deep reinforcement learning for trading,” arXiv preprint arXiv:1911.10107, 2019. doi: 10.48550/arXiv.1911.10107.

[5] A. Millea, “Deep reinforcement learning for trading—A critical survey,” Data, vol. 6, no. 11, p. 119, 2021. doi: 10.3390/data6110119.

Apêndice 1 - Ações utilizadas

Os instrumentos da bolsa brasileira utilizados neste trabalho foram:

- PETR3.SA, PETR4.SA, PRIO3.SA, BBAS3.SA, ITUB3.SA, ITUB4.SA, BBDC3.SA, BBDC4.SA, SANB11.SA, VALE3.SA, SUZB3.SA, GGBR4.SA, USIM5.SA, CSNA3.SA, LREN3.SA, MGLU3.SA, ARZZ3.SA, PCAR3.SA, BHIA3.SA, ENEV3.SA, EQTL3.SA, ELET3.SA, ELET6.SA, CMIG4.SA, CPLE6.SA, WEGE3.SA, EMBR3.SA, RAIL3.SA, AZUL4.SA, RADL3.SA, ABEV3.SA, BEEF3.SA, JBSS3.SA, SMTO3.SA, VIVT3.SA, TIMS3.SA, TOTS3.SA, CYRE3.SA, MRVE3.SA, EZTC3.SA, CSAN3.SA, UGPA3.SA, SMFT3.SA, HAPV3.SA, RENT3.SA

Apêndice 2 - Hiperparâmetros

Os hiperparâmetros utilizados para o modelo e o treinamento foram:

- Modelo:
  - Gamma: 0.99
  - Epsilon start: 1.0
  - Epsilon minimum: 0.01
  - Epsilon decay: 0.995
  - Target network update: every step (soft update)
  - Soft update factor (tau): 0.001
  - Replay memory size: 10000
  - Learning rate: 0.0001
  - Batch size: 64
- Arquitetura:
  - LSTM hidden size: 128
  - Number of LSTM layers: 2
  - Dropout: 0.5
- Dados:
  - Start date: 2019-01-01
  - End date: 2024-12-31
  - Train / validation / test split: 70% / 15% / 15%
- RSI e Episódios:
  - RSI period: 2
  - RSI entry threshold: 5
  - Maximum holding period: 20 days
  - LSTM warm-up: 12 days
- Treino:
  - Epochs: 5
  - Save checkpoint every epoch

Apêndice 3 - Gráficos de Backtesting

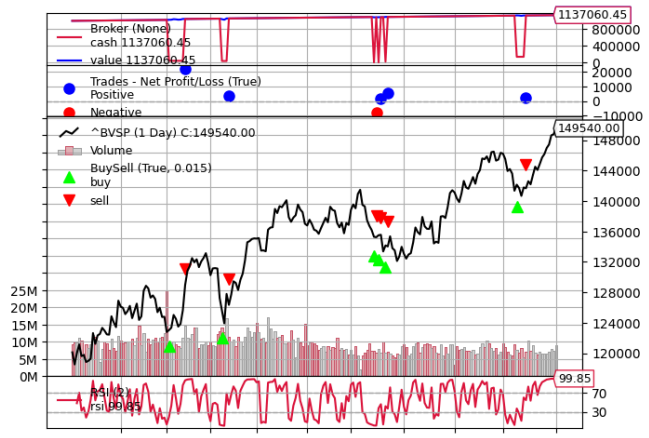

Fig 3. Backtesting do agente RL em Ibovespa

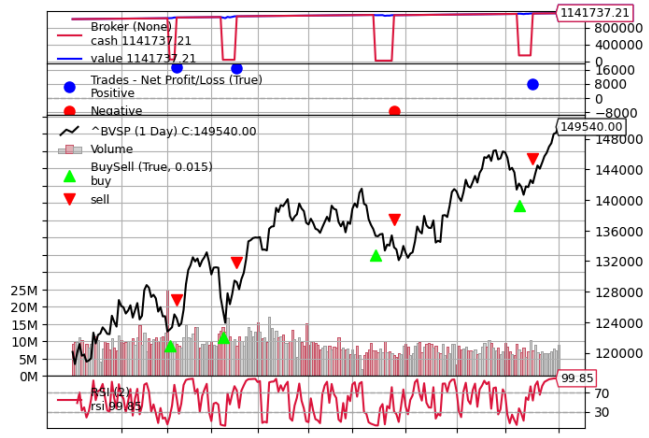

Fig 4. Backtesting do RSI-only em Ibovespa

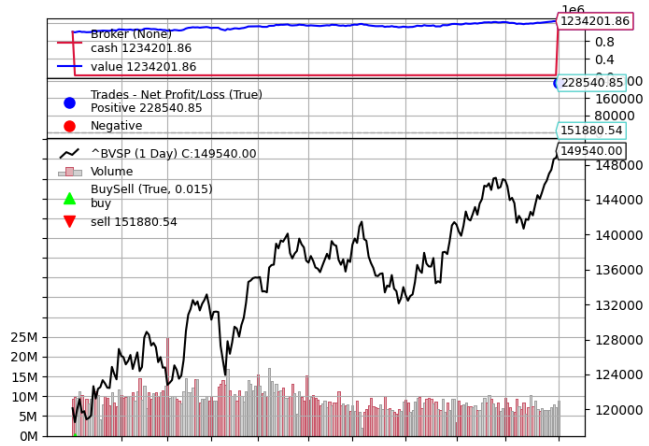

Fig 5. Backtesting do Buy & Hold em Ibovespa

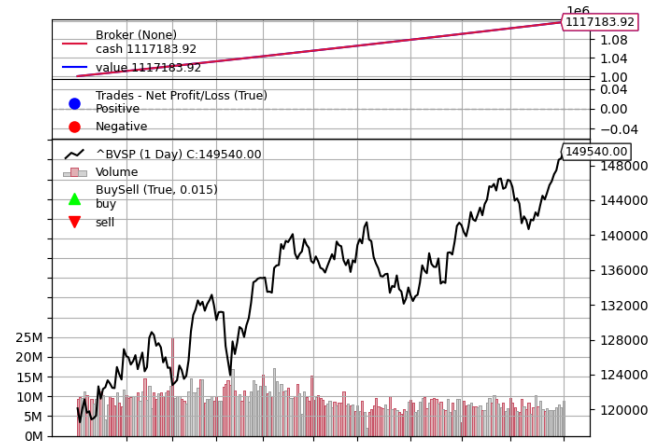

Fig 6. CDI no período (Jan. - Out. 2025)

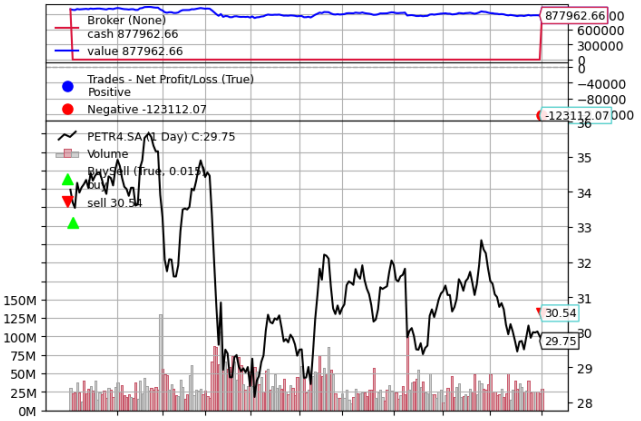

Fig 9. Backtesting de Buy & Hold em PETR4

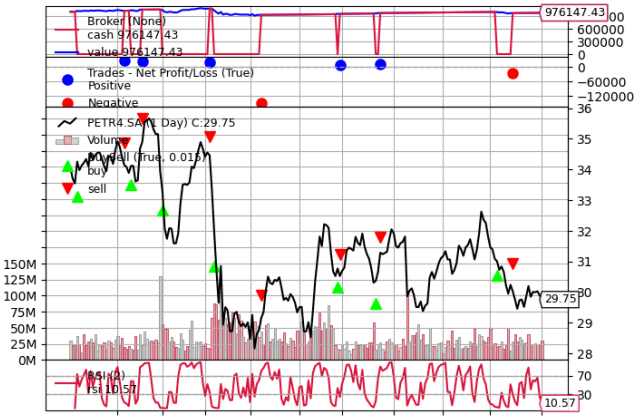

Fig 7. Backtesting do agente RL em PETR4

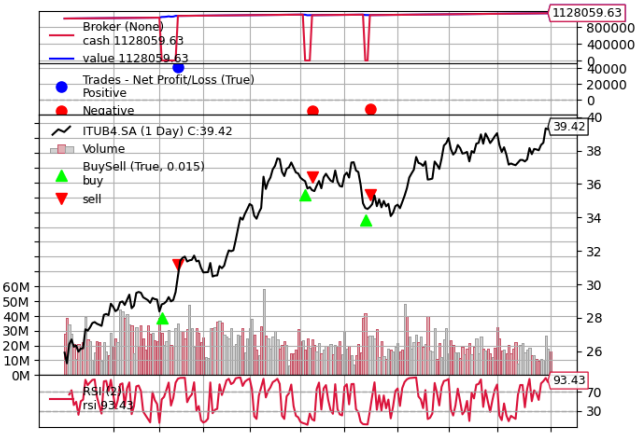

Fig 10. Backtesting do agente RL em ITUB4

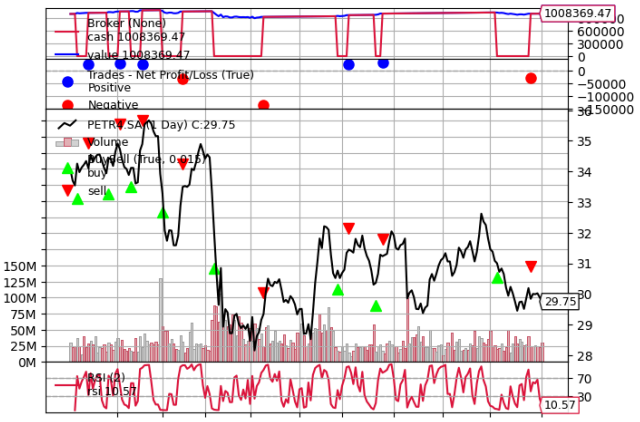

Fig 8. Backtesting de RSI-only em PETR4

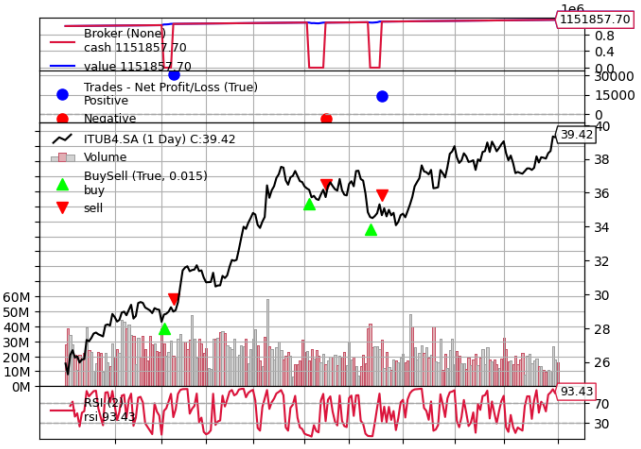

Fig 11. Backtesting de RSI-only em ITUB4

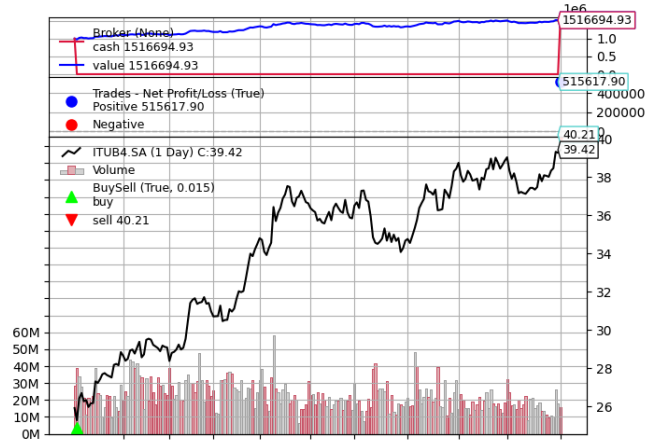

Fig 12. Backtesting de Buy & Hold em PETR4