RL Project

Design and Development of Intelligent Trading Agents Based on Reinforcement Learning (RL)

Deep Learning

Course: 2025/2026

Antonio Barea Rodríguez

Javier Ruiz Chicoy

INDEX

1. Introduction and Problem Definition

1.1. Project idea

In this project, we propose developing an intelligent trading system based on Reinforcement Learning (RL). Our goal is to train an agent capable of making buy, sell, or hold decisions on a stock market index (for example, the S&P 500), using historical daily price data.

Unlike traditional approaches based on fixed rules or predictive models, our RL agent learns an optimal decision policy through reinforcement, in order to maximize its accumulated reward, understood as the expected long-term profitability.

The ultimate purpose is to evaluate whether a RL-based trading strategy can be more profitable and efficient in risk management than a benchmark passive strategy, such as the well-known "buy-and-hold" strategy, when tested on data not seen during training.

1.2. Project objective

Main question:

- Can a Reinforcement Learning agent learn a profitable trading policy for a stock market index (e.g., the S&P 500), using only historical daily price data?

Sub-question:

- Is this policy capable of outperforming a passive buy-and-hold strategy in risk-adjusted returns (measured by the Sharpe Ratio) during a trial period not included in the training?

Furthermore, we propose to compare the performance of different families of RL algorithms, such as value-based (DQN) and policy-based or critical-actor (PPO) algorithms, in order to select and justify the most appropriate model.

1.3. Understanding the context

- **Why is this problem important?**
  - Algorithmic trading is a cornerstone of modern financial markets. While many models rely on predictions, algorithmic trading offers a different approach: instead of predicting price, it learns an optimal sequence of decisions (buy, sell, hold) to maximize cumulative reward (profit) over the long term.
- **Who will use the results?**
  - From an academic perspective, the results help demonstrate the viability of real-time automation in financial domains. From an applied perspective, they would be of interest to analysts, investment fund managers, or any entity interested in developing and evaluating automated trading strategies.

1.4. Established success criteria

- **What defines a good solution?**
    - The solution will be considered successful if it meets the following criteria in the test set:
        1. **Positive Net Profitability:**The agent must generate a positive total profit after simulating, if possible, transaction costs (commissions).
        2. **Outperforming the Benchmark:**The agent's strategy must achieve a total return higher than that of the passive "buy and hold" strategy over the same period.
        3. **Risk-Adjusted Return**The Sharpe ratio of the agent must be higher than that of the buy-and-hold strategy.
        4. **Risk Management:**The maximum drop in value from a peak of the agent must be, at a minimum, comparable to (or ideally lower than) that of the benchmark, demonstrating that it does not obtain profitability by assuming excessive risk.

1.5. Data Collection and Restrictions

Data: Only publicly available historical daily price data (Open, High, Low, Close, Volume) will be used (e.g., S&P 500 or Iberdrola via [website address missing]).yfinanceThe study period will be approximately 2016-2024.

Computational Resources: Training and assessment will be limited to what is available on the personal computers of project members, free cloud services (Google Colab) or the cluster provided by the university.

2. State of the Art

The application of Deep Reinforcement Learning (DRL) in financial markets has polarized into two main approaches: value-based methods, whose prime example is**DQN**and policy gradient-based methods, currently dominated by**PPO**Recent literature suggests that the choice between one or the other depends largely on the market regime (trend vs. sideways) and the desired risk management.

2.1. Deep Q-Networks (DQN): Efficiency in Discrete Spaces

Originally proposed by Mnih et al., the DQN algorithm revolutionized the field by combining Q-Learning with deep neural networks and Experience Replay. In the financial context, DQN has become the standard for discrete stock spaces (Buy/Sell/Hold).

We analyzed several 2024 studies that highlight its sampling efficiency: DQN can learn from past experiences stored in its memory thanks to the replay buffer, allowing it to converge faster in data-constrained environments. Comparative research on emerging markets (SBC, 2024) found that DQN variants outperformed PPO in short-term swing trading strategies, demonstrating a greater ability to capture immediate profit opportunities in volatile markets, albeit at the cost of greater instability during training due to overestimation of Q values.

2.2. Proximal Policy Optimization (PPO): Stability and Robustness

In contrast to the instability of value methods, PPO (Schulman et al., 2017) has established itself as the state-of-the-art algorithm in the policy gradient family. Its main innovation, the "clipping" mechanism in the objective function, avoids drastic updates to the agent's policy, ensuring monotonous and safe learning.

In the field of algorithmic trading, research on**NHSJS (2025)**They point out that PPO tends to perform better during periods of clear trends (bull markets). Its stochastic nature allows it to explore more complex strategies and avoid falling into premature local lows. The empirical study showed that, over longer periods, PPO achieved a higher cumulative return (62%) compared to DQN (33%), thanks to its ability to maintain winning positions for longer, although this sometimes entails accepting greater volatility (drawdown).

2.3. Direct Comparison: Stability vs. Aggressiveness

The comparison between PPO and DQN is not arbitrary, as both represent two distinct approaches within Reinforcement Learning applied to algorithmic trading.

1. **Off-policy (DQN) vs. On-policy (PPO):**While DQN reuses old data to optimize efficiency, PPO requires fresh data generated by current policy. Sources suggest that this difference makes DQN more "memoristic" of past patterns, while PPO is more "adaptive" to new market regimes (Borkar & Jadhav, 2024).
2. **Risk Management:**Recent studies suggest that DQN tends to make more frequent and selective trades in sideways markets ("noise"), while PPO excels at capturing large trending moves. Validating which behavior maximizes the Sharpe Ratio across different indices or assets will be the central objective of our research.

3. Data Collection and Characterization

In this section, we detail the process we followed for the acquisition, cleaning, and descriptive analysis of the historical price data we used for the project.

3.1. Data Selection and Acquisition

In line with the restrictions defined in the Introduction (Section 1), we selected historical daily price data (Open, High, Low, Close, Volume) for the period between January 1, 2016 and December 31, 2024.

For the acquisition, we used the library`yfinance`from Python, which facilitates downloading data from Yahoo Finance. Although our initial intention was for the RL agent to focus primarily on a stock market index, we have decided to expand the initial characterization to include five financial instruments and evaluate the adaptability of the methodology to different markets:

1. **S&P 500** (^GSPC)
2. **IBEX 35** (^IBEX)
3. **NASDAQ Composite** (^IXIC)
4. **Bitcoin** (BTC-USD)
5. **United States Natural Gas Fund** (WHO)

The acquisition process we carried out included defining the tickers and the time range. Subsequently, we iterated over each ticker to download the price data, resolving the MultiIndex issue by flattening the columns and renaming them to lowercase for consistency.

```
# --- Parámetros de la Descarga ---
TICKERS = ['^GSPC', '^IBEX', 'BTC-USD', '^IXIC', 'UNG'] #S&P 500, IBEX-35, BTC-USD, NASDAQ y UNG
START_DATE = '2016-01-01'
END_DATE = '2024-12-31'
```

3.2. Cleaning and Descriptive Analysis

After downloading, we performed a data quality inspection. The resulting dataset contains**12,377 records**This analysis revealed two key characteristics in our data:

Integrity

Initially, we looked for null or missing (NaN) values in our dataset in order to remove them. The analysis confirmed that there were no null values in the data obtained from yfinance for the selected period, so it was not necessary to apply any additional imputation or cleaning techniques. This tells us thatThe data only includes the dates on which the market opens (every day in the case of Bitcoin).

Disparate scales

Below, we show in a table some of the results obtained in the`data.groupby`to calculate basic close and volume statistics.

The data shows that the scales of the variables are drastically heterogeneous. While Bitcoin handles an average daily volume exceeding 21 billion monetary units and a high standard deviation (reflecting its high volatility), assets like the Natural Gas Fund (UNG) have an average price of 72.87.

Ticker	Average Price ($)	Standard Deviation (Std)	Average Daily Volume
BTC-USD	22,327.90	22,301.88	21,233,616,964
WHO	72.87	37.66	2,210,967
^GSPC	3,499.22	1,046.78	4,044,900,150
^IBEX	9,186.74	1,105.61	192,063,698
^IXIC	10,397.36	3,978.99	3,678,010,786

These differences in magnitude confirm that feeding a neural network with raw data would greatly complicate model convergence, justifying the need to apply scaling techniques (`StandardScaler`) detailed in section 3.

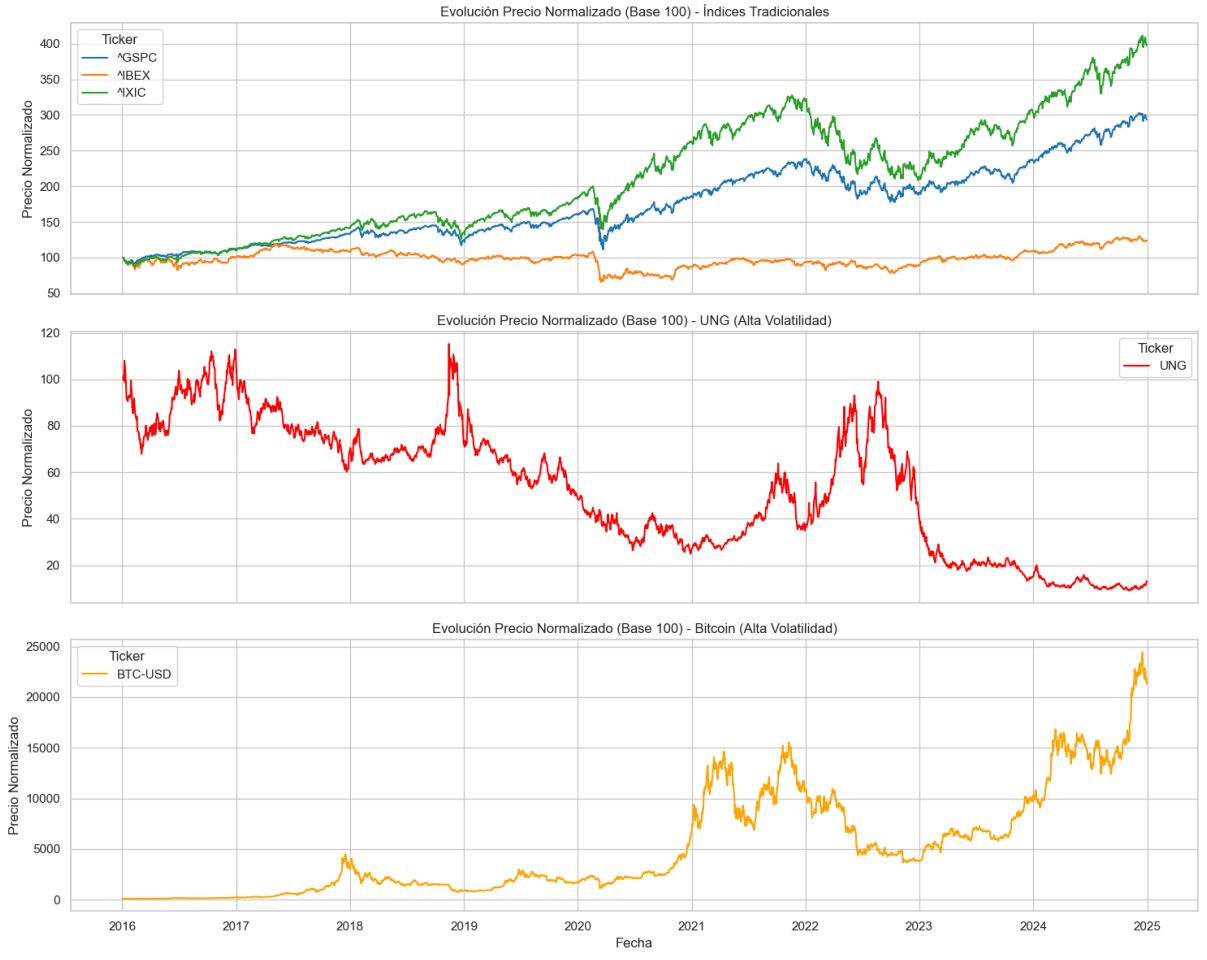
3.3. Exploratory Data Analysis (EDA)

To understand the nature of the environment in which the agent will operate, we performed key visualizations.

3.3.1. Evolution of Normalized Prices

Since absolute prices are visually incomparable, we calculate a normalized price (Base 100 at the beginning of the period).

```
# 1. Crear 'normalized_close' base 100
# (Precio_de_hoy / Primer_precio) * 100
data['normalized_close'] = data.groupby('ticker')['close'].transform(lambda x: (x / x.iloc[0]) * 100)
```

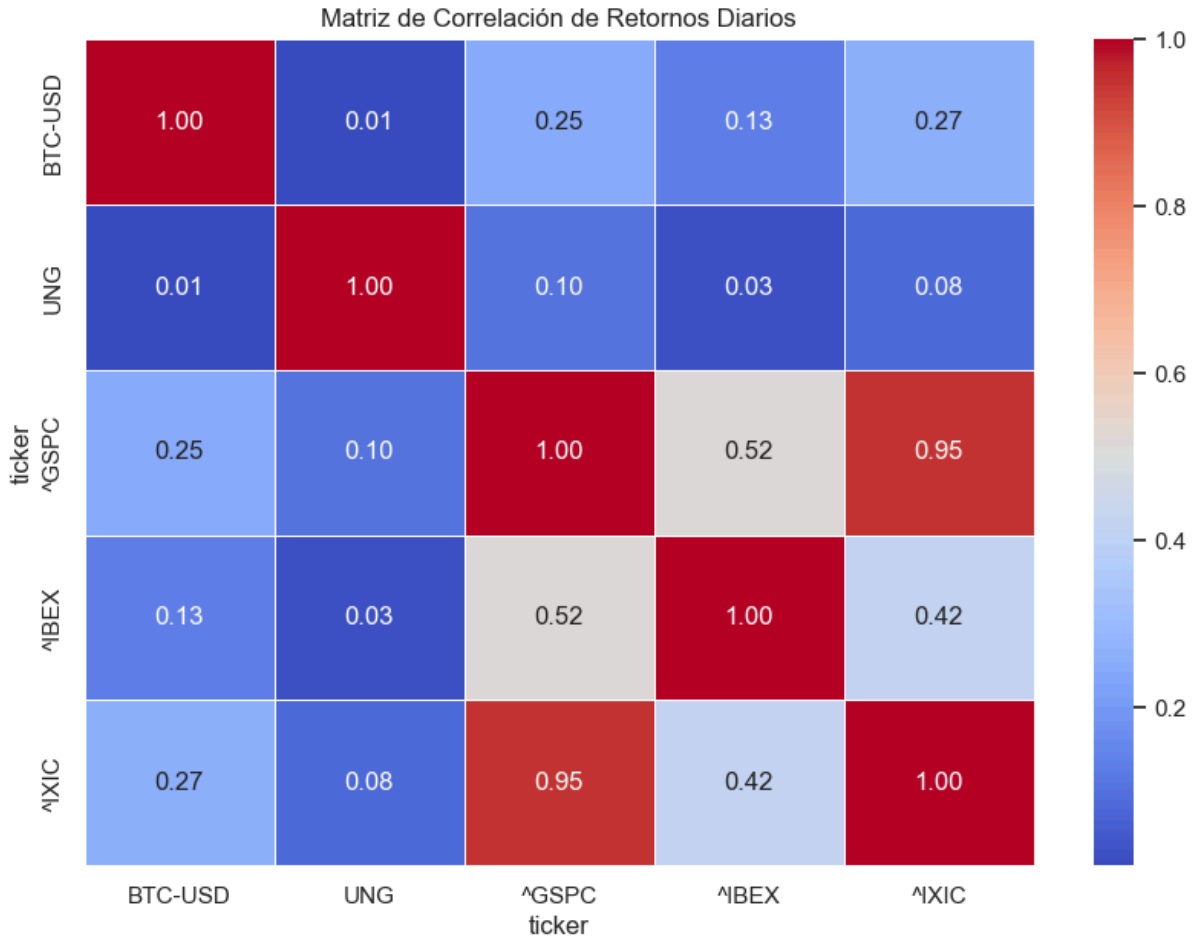


Observations:

- It is evident **extreme volatility** of Bitcoin versus traditional indices.
- The S&P 500 and NASDAQ show a clear and correlated upward trend, superior to that of the IBEX 35.
- Natural gas (UNG) shows a very different behavior, with no clear long-term upward trend and high variance.

3.3.2. Correlation Analysis

We calculate daily returns to analyze how assets move relative to each other. This is crucial for understanding whether the broker could diversify risk.

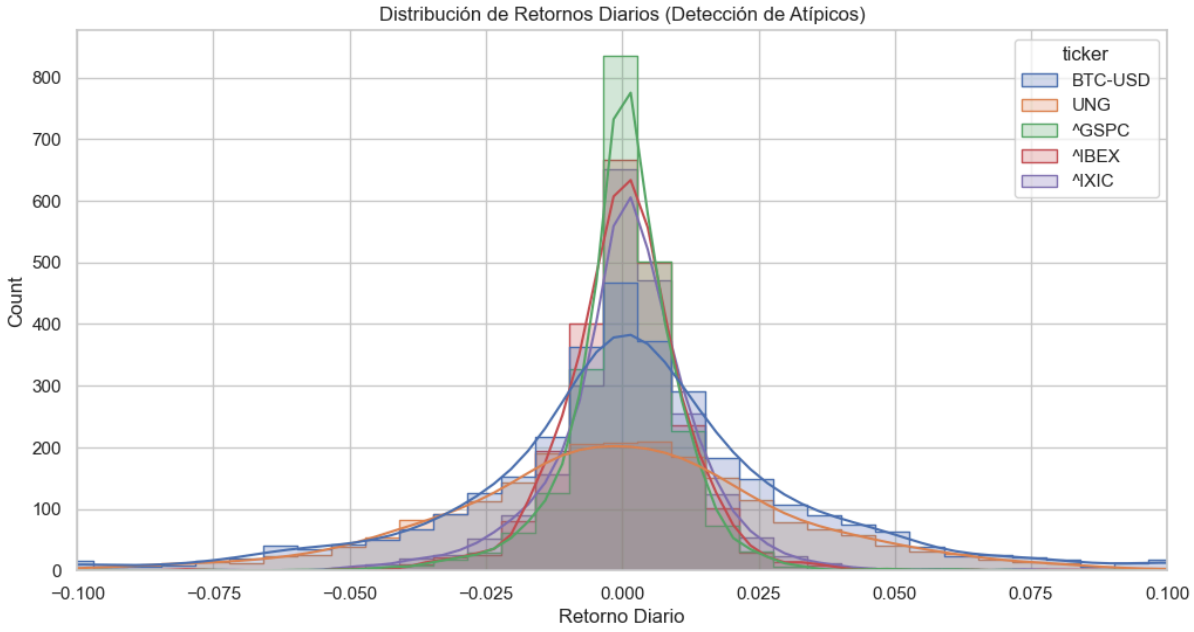


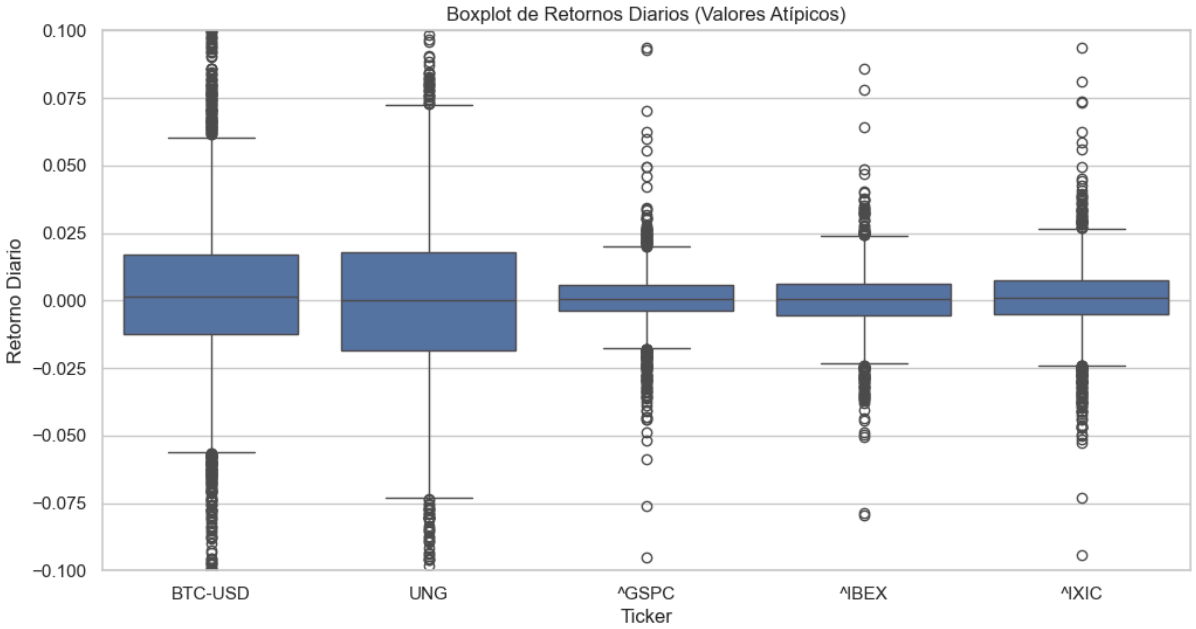
The heat map results indicate:

- High redundancy in the US: The S&P 500 and NASDAQ have a correlation of 0.95, which implies that they provide almost identical information.
- Europe vs. the US: The IBEX 35 has a positive but much weaker correlation (0.52 and 0.42) with US indices.
- **Diversification:**Bitcoin and UNG show very low correlations (<0.30) with traditional indices, suggesting that they are useful assets for portfolio decorrelation.

3.3.3. Detection of Outliers

Finally, we analyzed the distribution of daily returns to verify if they follow a normal distribution (Gaussian bell curve) and performed boxplots to check for outliers.





Visual analysis of the histograms reveals critical structural heterogeneity in asset behavior:

1. **Traditional Indices (^GSPC, ^IBEX, ^IXIC):**They exhibit leptokurtic distributions (highly peaked and narrow around zero). This indicates that the vast majority of their daily movements are small and stable.
2. **Speculative Assets (BTC-USD, UNG):**They exhibit much flatter and more dispersed distributions. This confirms a significantly higher variance; that is, the probability of

experiencing extreme daily swings (of 5%, 10% or more) is common in these markets, unlike in the traditional stock market.

4. Data cleaning and preparation

Reinforcement learning models, especially those using neural networks, require stationary and scaled input data (state space). The transformations performed are detailed below.

4.1. Feature Engineering

Instead of feeding the agent the raw price (which is not capped), we have created technical indicators that provide "context" on the state of the market; these indicators are the ones that are usually used in trading to know whether it is worth buying, selling, or holding.

The following variables were defined for the state vector:

1. `log_ret` **(Logarithmic Returns):** They indicate immediate profit/loss. They are preferable to simple prices because they are stationary.
2. `dist_sma_60` **(Trend):** Percentage distance of the current price from its 60-day moving average. Indicates whether the asset is "expensive" or "cheap" relative to its trend.
3. `rsi` **(Moment):** Relative Strength Index (0-100). Helps the agent detect overbought or oversold conditions.
4. `volatility` **(Risk):** Moving standard deviation of returns (20-day window). Measures current market uncertainty.

```python
def add_features(df):
    """
    Calcula y añade los indicadores técnicos al DataFrame.
    """
    df_feat = df.copy()

    # 1. Retornos Logarítmicos: ln(P_t / P_{t-1}) --> Variable fundamental para la recompensa del agente.
    df_feat['log_ret'] = np.log(df_feat['close'] / df_feat['close'].shift(1))

    # 2. Tendencia (Medias Móviles Simples - SMA) --> SMA de corto plazo (10 días) y largo plazo (60 días)
    df_feat['sma_10'] = df_feat['close'].rolling(window=10).mean()
    df_feat['sma_60'] = df_feat['close'].rolling(window=60).mean()

    # 3. Distancia a la media (Normalizada): ¿Está el precio muy alejado de la tendencia?
    df_feat['dist_sma_60'] = (df_feat['close'] - df_feat['sma_60']) / df_feat['sma_60']

    # 4. Momento (RSI - 14 días)
    df_feat['rsi'] = calculate_rsi(df_feat['close'], period=14)

    # 5. Volatilidad (Riesgo) --> Desviación estándar móvil de 20 días de los retornos
    df_feat['volatility'] = df_feat['log_ret'].rolling(window=20).std()

    # 6. Limpieza de Nulos Generados --> Eliminamos las filas iniciales que tienen NaNs por el cálculo de ventanas
    df_clean = df_feat.dropna()

    return df_clean
```

4.2. Data splitting (Train/Test Split) with Buffer

To prevent data leakage (future data leaks), we implement a strict time division:

- **Training:**Data prior to**01-01-2023**.
- **Test:**Data from the**01-01-2023**from now on.

Buffer Mechanism: Since our indicators require a historical window (e.g., a 60-day moving average), if we were to abruptly start counting the data on January 1st, we would lose the first 60 days of the test set while the averages are being calculated. To address this, we implemented a 90-day buffer: we added the last 90 days of the training set to the beginning of the test set before calculating the indicators, and then trimmed the excess. This ensures that the broker can operate from the first day of the test period with valid data.

```
--- Procesando datos con fecha de corte: 2023-01-01 ---
Filas Crudas Train: 9634
Filas Crudas Test : 2743

Calculando indicadores técnicos...

--- Dimensiones Finales (Listos para usar) ---
Train Set Final: 9339 filas (Se eliminaron nulos iniciales)
Test Set Final : 2743 filas (Completo desde 2023-01-01)

Verificación del primer día de Test (sin NaNs):
Price             close        sma_60       rsi
Date
2023-01-01  16625.080078  17157.376758  46.05346
```

4.3. Normalization (Scaling)

Neural networks converge best when the inputs have a mean of 0 and a standard deviation of 1. We use`StandardScaler`from the scikit-learn library.

Data Leakage Prevention: It is essential to adjust the scaler (`fit`) using only the training data. Subsequently, we use those parameters (mean and standard deviation learned in training) to transform the test data. If we scaled using the entire dataset, we would be passing information about the future (test distribution) to the agent during training.

```
# 1. FIT (Aprender media/std) SOLO en TRAIN
scaler.fit(train_scaled.loc[mask_train, feature_cols]) # feature_cols son las variables que forman el "ESTADO" (no incluimos close)

# 2. TRANSFORM (Aplicar) en TRAIN y TEST
train_scaled.loc[mask_train, feature_cols] = scaler.transform(train_scaled.loc[mask_train, feature_cols])
test_scaled.loc[mask_test, feature_cols] = scaler.transform(test_scaled.loc[mask_test, feature_cols])
```

4.4. Exporting Datasets

Finally, the processed data were exported in individual CSV files per asset (e.g.BTC-USD_train.csv, BTC-USD_test.csvThese files contain the normalized variables ready to be ingested by the Gymnasium environment in the modeling phase.

```
Datos guardados para BTC-USD:
   - Train: data_processed/BTC-USD_train.csv (2498 filas)
   - Test : data_processed/BTC-USD_test.csv  (730 filas)
Datos guardados para UNG:
   - Train: data_processed/UNG_train.csv (1703 filas)
   - Test : data_processed/UNG_test.csv  (501 filas)
Datos guardados para ^GSPC:
   - Train: data_processed/GSPC_train.csv (1703 filas)
   - Test : data_processed/GSPC_test.csv  (501 filas)
Datos guardados para ^IBEX:
   - Train: data_processed/IBEX_train.csv (1732 filas)
   - Test : data_processed/IBEX_test.csv  (510 filas)
Datos guardados para ^IXIC:
   - Train: data_processed/IXIC_train.csv (1703 filas)
   - Test : data_processed/IXIC_test.csv  (501 filas)
```

5. Agent Modeling and Evaluation

Once the data was prepared, we entered the core phase of the project: the construction of the Artificial Intelligence agents. In this section, we detail the architecture of the selected algorithms, the design of the financial simulation environment, and the initial validation against the benchmark index (S&P 500).

5.1. Algorithm Selection: Value vs. Policy

To address the sequential decision problem in financial markets, we have selected two algorithms that represent the main families of Deep Reinforcement Learning:

1. **DQN (Deep Q-Network):**A value-based approach. The agent learns a function Q(s, a) that estimates the expected future reward of taking a specific action in a given state.
    - Implementation: It uses a discrete action space. We have defined 5 levels of aggressiveness to avoid simplistic binary decisions:`[Hold, Strong Buy (100%), Strong Sell (100%), Partial Buy (50%), Partial Sell (50%)]`.
2. **PPO (Proximal Policy Optimization):**An approach**Actor-Critic**(Policy-Based). The agent directly learns a stochastic policy$\pi(a|s)$that maps states to probabilities of action.
    - Implementation: Uses a contiguous action space. The agent emits a floating-point value within the range.[-1, 1], which allows for infinite granularity in position management (e.g., investing 12.5% of capital).

5.2. Trading Environment Design (`TradingEnv`)

The environment is the critical component that translates financial data into the standard format that RL agents (inherited from`gymnasium`) can interpret. The design decisions implemented in the class are then justified.`TradingEnv`:

Observation Space

The agent not only receives the technical indicators calculated during the feature engineering phase, but also the portfolio context. It is crucial for the agent to know if they are already invested in order to decide whether to sell or hold.

To facilitate the convergence of the neural network, we normalized the internal variables of the portfolio to the same range as the technical indicators:

```
# Normalización para ayudar a la red neuronal
# Estimamos un máximo de acciones posible para normalizar entre 0 y 1 aprox.
max_possible_shares = (self.initial_balance / current_price) * 5
shares_norm = self.shares_held / max_possible_shares if max_possible_shares > 0 else 0
balance_norm = self.balance / self.initial_balance
```

Reward Function

Instead of using simple percentage change, we opted for Logarithmic Returns ($Rt = ln(\frac{Vt}{Vt-1})$This metric is preferable in mathematical models because of its property of time additivity and symmetry with respect to gains and losses.

Additionally, we introduced a Reward Shaping mechanism to incentivize the decision: if the agent makes a high conviction trade (action > 50%) and the result is positive, the reward is increased by an extra 10%.

5.3. Training Configuration (S&P 500)

The training was performed on the dataset`GSPC_train.csv`for a total of**1,500,000 steps**(timesteps). Since the S&P 500 is an asset with moderate volatility and a stable historical trend, it was not necessary to apply aggressive Early Stopping techniques in this initial phase, allowing agents to explore the entire state space.

Key hyperparameters:

- **DQN:**Learning rate of1e-4and an experience buffer of 50,000 transitions to break the temporal correlation of the data.
- **PPO:**Learning rate of3e-4, taking advantage of its greater inherent stability through the clipping mechanism.

```python
# DQN: Deep Q-Network
model_dqn = DQN(
    "MlpPolicy",
    env_dqn,
    verbose=0,
    learning_rate=1e-4,
    buffer_size=50000,
    exploration_fraction=0.2
)
```

```python
# PPO: Proximal Policy Optimization
model_ppo = PPO(
    "MlpPolicy",
    env_ppo,
    verbose=0,
    learning_rate=3e-4
)
```

5.4. Validation results in S&P 500

After training, we evaluated the models on the test set (`GSPC_test.csv`), which corresponds to the period**2023-2024**, a cycle characterized by a strong upward recovery (Bull Market).

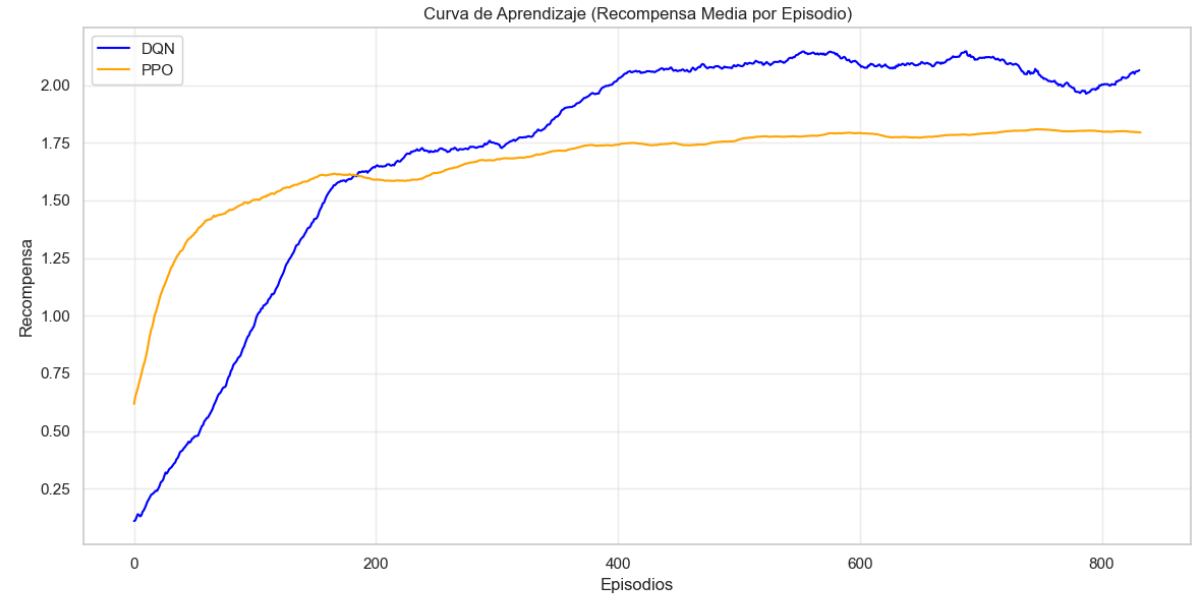
Results Table and Graphs (S&P 500 Test)

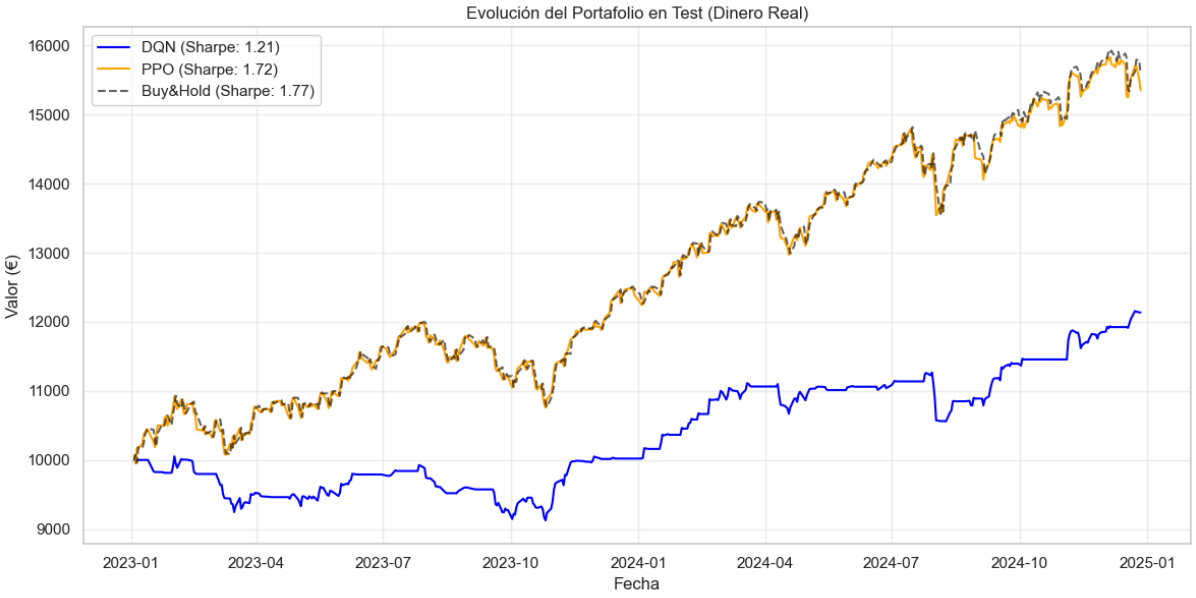
The quantitative results obtained are summarized in the following table:

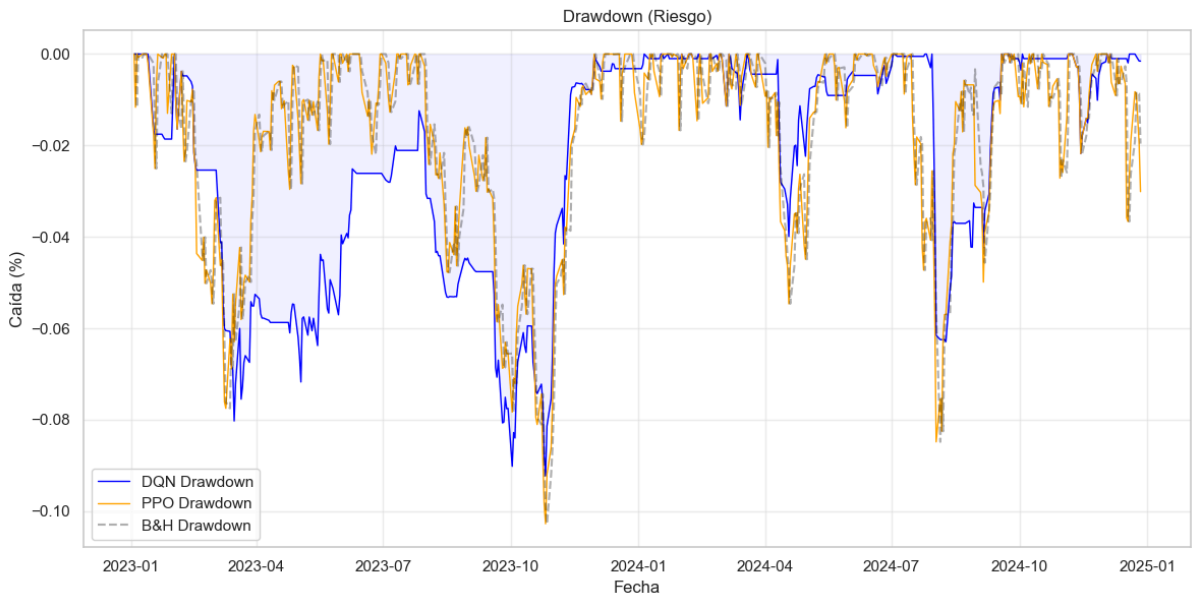
Metric	Benchmark (Buy & Hold)	DQN Agent	PPO Agent
Capital Final	**15.446 €**	12.136 €	15.355 €
Profitability %	**54,5 %**	21,4 %	53,6 %
Sharpe Ratio	**1,77**	1,21	1,72
Max Drawdown	-10,3 %	**-9,2 %**	-10,3 %

Before discussing the quantitative values shown in the table, the validation graph, which contains 3 panels, can be seen on the next page:

1. Learning Curve → Monitors the average reward per episode during training
2. Portfolio Evolution → Compares the accumulated monetary performance of the agents against the passive strategy (Buy & Hold) in the test set.
3. Drawdown → Graphically represents risk by measuring the percentage drop in capital relative to its historical maximum accumulated at each moment.


Curva de Aprendizaje (Recompensa Media por Episodio)
DQN
PPO
Recompensa
Episodios
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0
200
400
600
800


Evolución del Portafolio en Test (Dinero Real)
DQN (Sharpe: 1.21)
PPO (Sharpe: 1.72)
Buy&Hold (Sharpe: 1.77)
Valor (€)
Fecha
9000
10000
11000
12000
13000
14000
15000
16000
2023-01
2023-04
2023-07
2023-10
2024-01
2024-04
2024-07
2024-10
2025-01


Drawdown (Riesgo)
DQN Drawdown
PPO Drawdown
B&H Drawdown
Caída (%)
Fecha
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
2023-01
2023-04
2023-07
2023-10
2024-01
2024-04
2024-07
2024-10
2025-01

Execution Analysis

1. **PPO vs. Benchmark:**The PPO agent achieved almost identical performance to the passive (Buy & Hold) strategy, with a return of 53.6% compared to the market's 54.5% and a Sharpe Ratio of 1.72. This indicates that the agent correctly learned to identify the prevailing upward trend and remain invested most of the time, managing the position efficiently thanks to its continuous trading space.
2. **DQN Subperformance:**The DQN broker achieved a positive return (21.4%), but significantly lower than the market. Its discrete nature (with swings from 0% to 50% or 100%) and its difficulty in converging on stable policies in noisy environments led it to be overly conservative or exit the market prematurely, missing out on much of the 2023 rally.
3. **Partial Conclusion:**In a market with a strong upward trend and relatively low volatility, like the recent S&P 500, it's extremely difficult to beat the Buy & Hold strategy. However, the fact that PPO has matched the market validates that the trader is capable of learning a useful strategy and not losing capital.

Open Question: What would happen if we subjected these same agents to hostile environments, with extremely high volatility or bearish trends where "Buy and Hold" can be a losing strategy? We will address this in the following experiments with Bitcoin and Natural Gas.

6. Validation in High Volatility Environments (Stress Testing)

After analyzing the basic functioning of market participants in an upward trend environment (S&P 500), we proceed to subject them to extreme market conditions. The objective is to answer the question:**Can Reinforcement Learning outperformbenchmarkWhen is the passive "Buy and Hold" strategy highly risky or unprofitable?**

To do this, we selected two assets with risk profiles opposite to the S&P 500:

1. **Bitcoin (BTC):**High bidirectional volatility, with explosive rises and severe falls (>70%).
2. **Gas Natural (UNG):**Long-term structural downward trend and high volatility, a hostile environment for passive investment.

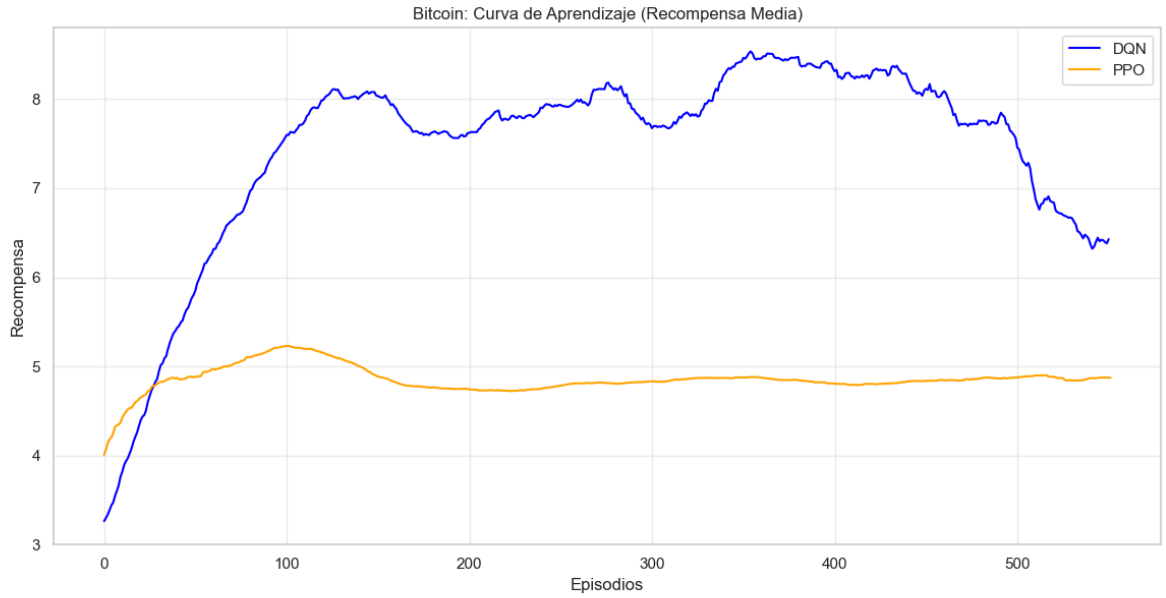
6.1. Experiment 1: Bitcoin (Euphoria and Panic Management)

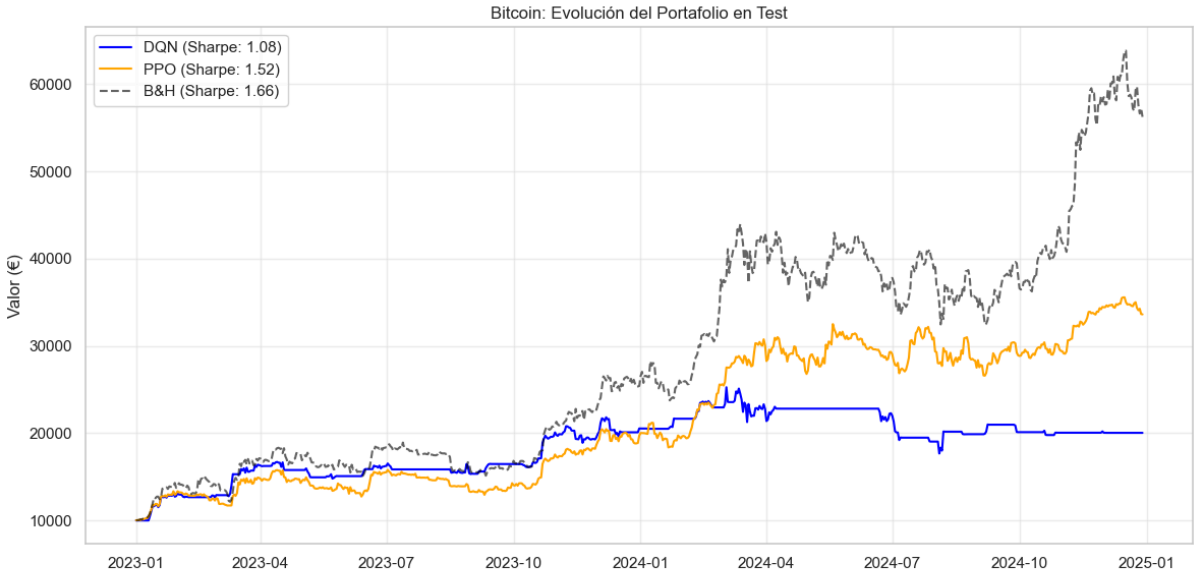
We have retrainedDQN and PPO agents from scratch using the Bitcoin dataset (`BTC-USD_train.csv`), maintaining the base hyperparameters but adapting the environment to the volatility of the crypto asset.

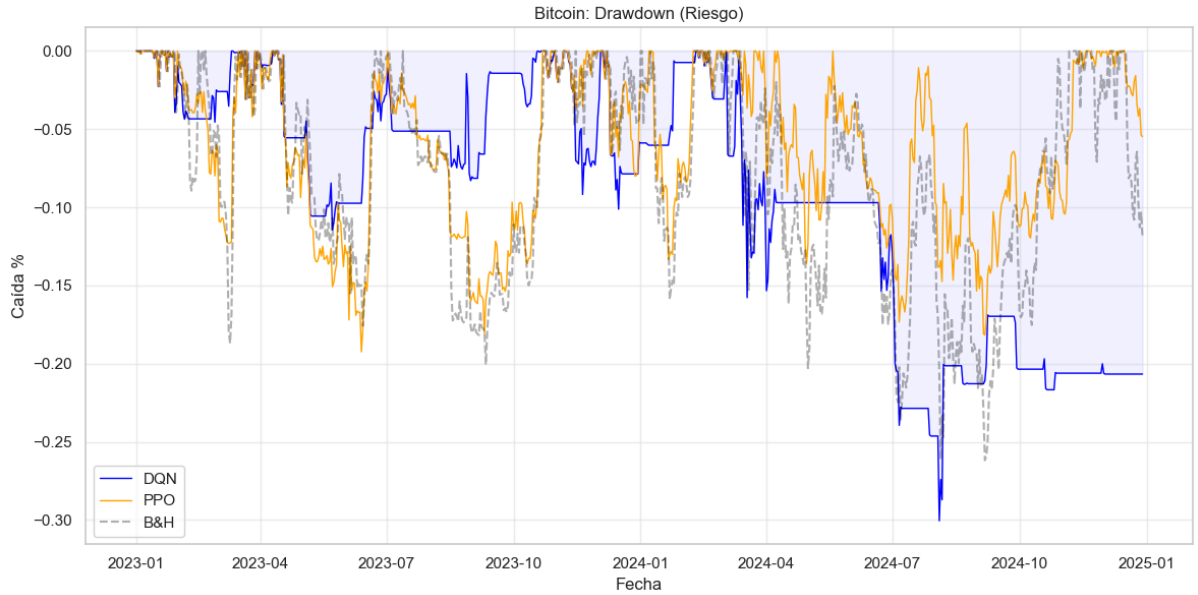
Results table and graphs (Bitcoin Test)

Metric	Benchmark (Buy & Hold)	DQN Agent	PPO Agent
Capital Final	**55.725 €**	20.044 €	33.628 €
Profitability %	**457,2 %**	100,4 %	236,3 %
Sharpe Ratio	**1,66**	1,08	1,52
Max Drawdown	-26,2 %	-30,1 %	**-19,2 %**

Before discussing the quantitative values shown in the table, the following page displays the validation graph for the same 3 panels as with the S&P 500.


Bitcoin: Curva de Aprendizaje (Recompensa Media)
DQN
PPO
Recompensa
Episodios
3
4
5
6
7
8
0
100
200
300
400
500


Bitcoin: Evolución del Portafolio en Test
DQN (Sharpe: 1.08)
PPO (Sharpe: 1.52)
B&H (Sharpe: 1.66)
Valor (€)
10000
20000
30000
40000
50000
60000
2023-01
2023-04
2023-07
2023-10
2024-01
2024-04
2024-07
2024-10
2025-01


Bitcoin: Drawdown (Riesgo)
Caída %
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
DQN
PPO
B&H
2023-01
2023-04
2023-07
2023-10
2024-01
2024-04
2024-07
2024-10
2025-01
Fecha

Execution Analysis

In this experiment, we observed an expected phenomenon, due to the vertical rise that Bitcoin suffered in the validation set period (457% rise), none of the agents managed to surpass or equal the benchmark.

- **PPO vs. Benchmark:** The PPO broker demonstrated solid and consistent performance, achieving a return of 236.3%, with a Sharpe Ratio of 1.52 and a maximum drawdown of -19.2%, significantly lower than that of the Buy & Hold strategy. While it failed to surpass the extreme market returns in a strongly bullish cycle like Bitcoin's 2023–2024, PPO stands out for its superior risk management, mitigating losses and smoothing volatility thanks to its continuous trading space and more stable policies.
- **DQN Subperformance:**The DQN broker achieved positive results (100.4% return), but these were significantly lower than both PPO and the benchmark. Its lower Sharpe ratio (1.08) and high drawdown (−30.1%) indicate an unstable policy and poor adaptation to Bitcoin's high volatility. The discrete nature of DQN and its difficulty in capturing non-stationary dynamics led to inefficient entries and exits, penalizing capital growth.

In an extremely volatile asset with an explosive upward trend like Bitcoin, beating the Buy & Hold strategy is especially difficult during such a steep rise. However, PPO's performance validates that the broker is capable of learning a sound trading strategy, prioritizing risk management over pure profit maximization, while DQN shows clear limitations in these types of complex financial environments.

6.2. Experiment 2: Natural Gas (Managing Bearish Trends)

This scenario represents the most hostile environment for any investor: a market with a structurally negative trend.

We used the United States Natural Gas Fund (UNG), a fund that tracks the price of natural gas. During the test period (2023-2024), this asset suffered a massive devaluation, losing more than 60% of its value.

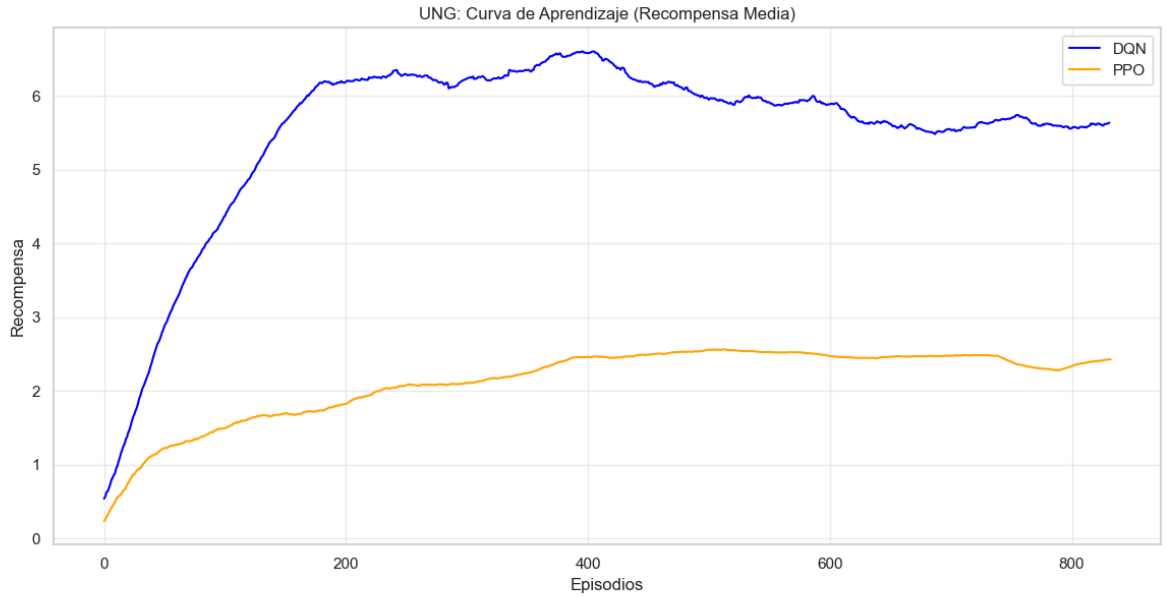
The Technical Challenge: Unlike the S&P 500 or Bitcoin, where the market trend itself helps to generate profits, our situation here is completely different.

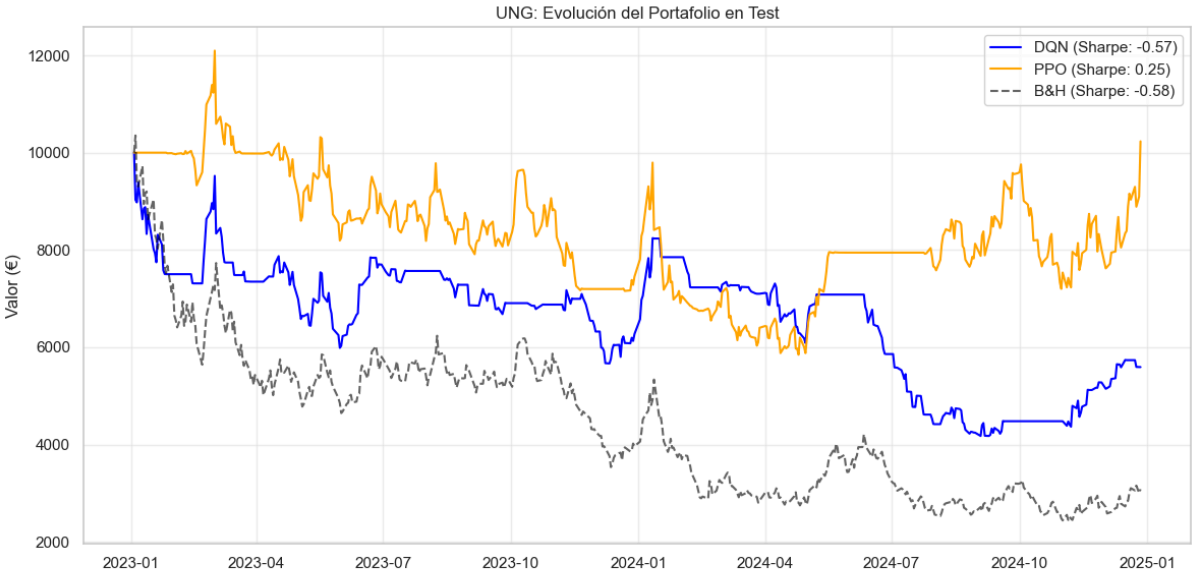
- **Benchmark Failure:**The passive "Buy and Hold" strategy is a guarantee of severe losses in this context.
- **Survival Goal:**The primary goal of our agents is changing radically. We are no longer focused on maximizing aggressive growth, but rather on capital preservation. We want to assess whether agents are able to detect asset declines and make the decision to sell everything before a drop and buy everything before a rise, trading only on very clear, one-off rebounds.

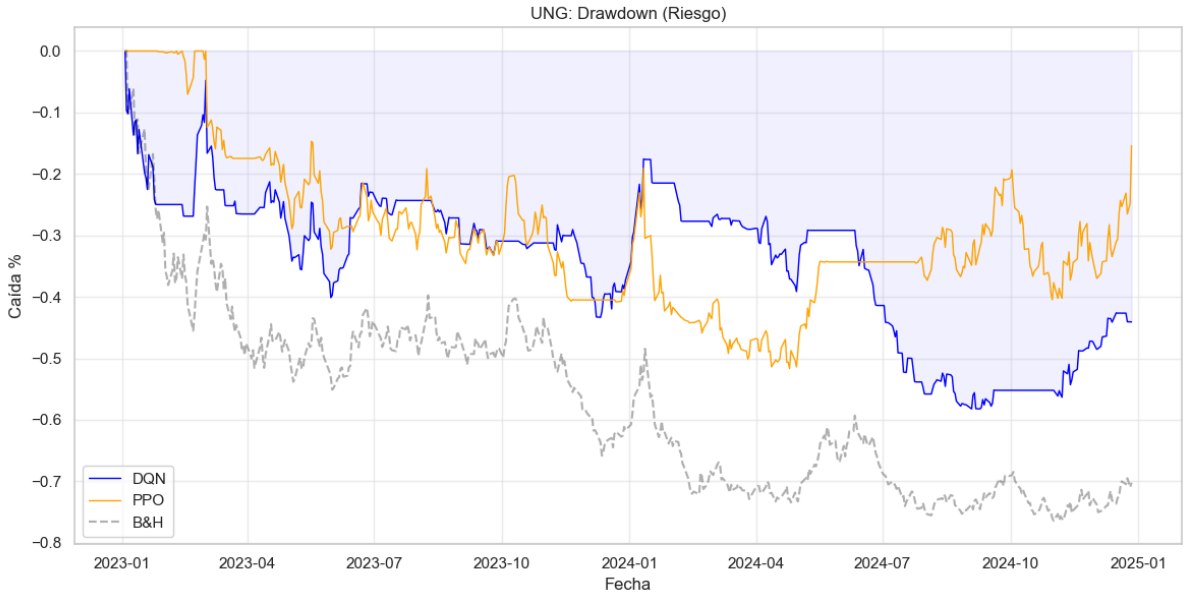
Results table and graphs (UNG Test)

Metric	Benchmark (Buy & Hold)	DQN Agent	PPO Agent
Capital Final	3.527 €	5.593 €	**10.233 €**
Profitability %	-64,7 %	-44,1 %	**+2,3 %**
Sharpe Ratio	-0,58	-0,57	**+0,25**
Max Drawdown	-76,4 %	-58,2 %	**-51,7 %**

Before discussing the quantitative values shown in the table, on the next page you can see the validation graph of the same 3 panels as with the S&P500 and Bitcoin.


UNG: Curva de Aprendizaje (Recompensa Media)
DQN
PPO
Recompensa
Episodios
0
1
2
3
4
5
6
0
200
400
600
800


UNG: Evolución del Portafolio en Test
DQN (Sharpe: -0.57)
PPO (Sharpe: 0.25)
B&H (Sharpe: -0.58)
Valor (€)
2000
4000
6000
8000
10000
12000
2023-01
2023-04
2023-07
2023-10
2024-01
2024-04
2024-07
2024-10
2025-01


UNG: Drawdown (Riesgo)
DQN
PPO
B&H
Caída %
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
2023-01
2023-04
2023-07
2023-10
2024-01
2024-04
2024-07
2024-10
2025-01
Fecha

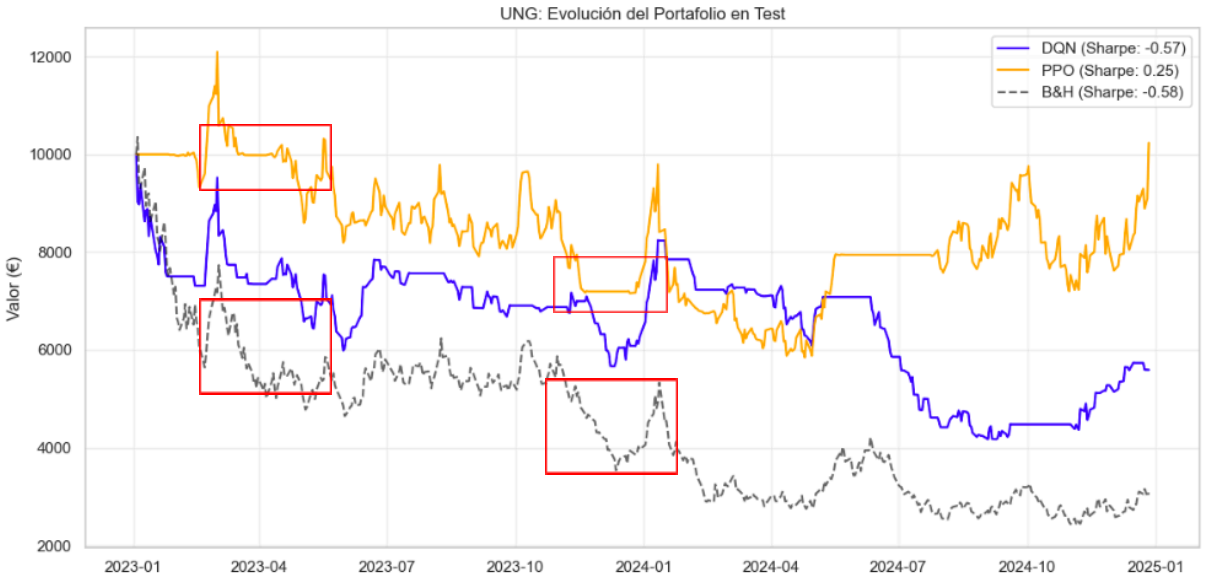
Execution Analysis

The results at UNG are conclusive and show us the best result so far in this project:

1. **DQN (Mitigated Loss):**DQN managed to lose less than the market (-44% vs. -64%), but it couldn't avoid capital destruction. Its low-interest ("all or nothing") nature worked against it in a noisy market with a constant downward trend.
2. **PPO (Positive Profitability in Crisis):**The PPO agent achieved something we did not expect: generating profits (+2.3%) on an asset that lost 65% of its value.
    - He learned to stay out of the market (in cash) during downturns, only entering on high-probability rebounds and managing position size thanks to his continuous space for action.
    - Its positive Sharpe Ratio (0.25) compared to the disastrous -0.58 of the market demonstrates that Reinforcement Learning is capable of finding autonomous survival strategies in environments where a human or passive investor would have capitulated.

When we did this example, wewe imaginedWe expected both agents to outperform the benchmark due to the bearish market for this ETF. However, we were very surprised to find that PPO even managed to generate profits while taking on some risk.

If we look at the following image, we see how PPO manages to learn how to withdraw all his money by predicting market downturns and even manages to reinvest it during upturns, generating much more profit during bullish periods and much less loss during bearish periods.



7. Final conclusions of the project

The integration of experiments conducted on the S&P 500 (bull market), Bitcoin (highly volatile asset) and Natural Gas (bear market) allows us to draw three main conclusions about the viability of Reinforcement Learning in algorithmic trading.

7.1. Limitations in markets with an upward trend

In environments with a marked positive trend, such as the S&P 500 and Bitcoin during the analyzed period, we observed that the passive strategy (Buy & Hold) is very difficult to surpass in terms of absolute profitability.

Our results show that any attempt at active management by agents (such as reducing exposure or temporarily exiting the market with the aim of repurchasing at lower prices) involved a relevant opportunity cost by losing exposure during prolonged upward stretches.

In this context, the PPO agent, while not outperforming the benchmark in total return, did demonstrate its usefulness from a risk management perspective. In the case of Bitcoin, it captured a significant portion of the rally (+236%), reducing the maximum drawdown by approximately 7% compared to the market. This suggests that, for more risk-averse investors, a Reinforcement Learning-based agent can be a reasonable alternative to passively holding highly volatile assets.

7.2. Capital protection in bear markets (UNG)

The experiment with Natural Gas (UNG) is one of the project's most significant outcomes. In this scenario, the passive strategy inevitably led to a significant capital loss (−65%), reflecting a clearly unfavorable environment.

Faced with this situation, the PPO agent was able to decouple its behavior from the underlying asset, not only preserving capital but also achieving a positive return (+2.3%). Analysis of the learned policy indicates that the agent autonomously adopted a liquidity-preference strategy, remaining out of the market during prolonged downturns on more than one occasion and buying back in during upward movements, thus increasing its capital and avoiding losses.

This result reinforces the idea that Reinforcement Learning can provide added value in bear markets or crisis situations, where passive and static strategies show clear limitations.

7.3. Better performance of PPO compared to DQN

Across all the scenarios analyzed, we observed that PPO (Proximal Policy Optimization) exhibited more consistent behavior than DQN. The discrete nature of DQN's action space proved too rigid for financial environments with high levels of noise and volatility, resulting in less stable policies and more pronounced drawdowns.

Conversely, PPO's continuous action space allowed for more gradual risk exposure management, facilitating both survival in bear markets and more efficient participation in bull markets. This flexibility was key to explaining its better overall performance.

General conclusion

Based on the results obtained, we conclude that Reinforcement Learning is not a tool designed to systematically beat the market in every context. However, it does prove to be an effective approach for dynamic risk management. In bull markets, it tends to act as a stabilization mechanism that sacrifices some returns in exchange for lower risk exposure, while in bear markets it demonstrates a clear ability to preserve capital and even attempt to generate value, outperforming traditional passive strategies.

Literature

Borkar, S., & Jadhav, A. (2024). Reinforcement Learning Techniques for Stock Trading: A Survey of Current Research. The Journal of Financial Data Science.

Botía Blaya, J. A. (2025). Machine Learning Extensions: Class Material and Reinforcement Learning Topic. University of Murcia.

Google (2025). Gemini.

NHSJS (2025). Comparative Analysis of the Effectiveness of Deep Reinforcement Learning Models for Trading in the Stock Market. National High School Journal of Science.

OpenAI (2025). ChatGPT.

ResearchGate (2024). A Comparative Study of Deep Reinforcement Learning Models: DQN vs PPO vs A2C. (Preprint).

SBC (2024). Reinforcement Learning for Automated Investment in the Brazilian Stock Market: A Comparative Study of DQN, PPO, and Their Recurrent Versions. Sociedade Brasileira de Computação.