APRENDIZAJE REFORZADO PARA ESTRATEGIAS DE TRADING

Por

Darío Castro González

Fernando De Santos Franco

Sergio Daniel Dueñas Godínez

Andre Yahir González Cuevas

Flavio Maximiliano Herrada Avalos

Ernesto Morales Mozka

Ingeniería Financiera

Instituto Tecnológico y de Estudios Superiores de Occidente

2025

Revisado por

Luis Felipe Gómez Estrada

1. Introducción

El Aprendizaje por Refuerzo (Reinforcement Learning, RL) es una técnica efectiva para la toma de decisiones secuenciales en entornos complejos como los mercados financieros. A diferencia de otros enfoques de Machine Learning, el RL permite a un agente aprender a través de la interacción con el entorno, ajustando sus decisiones según las recompensas obtenidas.

Este proyecto utiliza el algoritmo Proximal Policy Optimization (PPO) para desarrollar un agente capaz de tomar decisiones de trading basadas en indicadores técnicos como SMA, RSI y MACD. Los estados se construyen a partir de estos indicadores, mientras que las acciones corresponden a comprar, vender o mantener activos. La recompensa se define a partir de múltiples factores:

- Ganancia/pérdida individual por operación de venta.
- Situación del mercado en compras (por ejemplo, RSI bajo).
- Penalización o recompensa por mantener una posición en función de la variación de precio.
- Cambio global en el valor del portafolio ponderado.

El agente se entrena con un total de 2 millones de pasos de simulación, lo que le permite identificar patrones más profundos y robustos. La estrategia PPO, basada en políticas, ofrece mayor estabilidad frente a cambios súbitos del entorno y una mejor adaptación a entornos financieros no estacionarios.

Esta entrega describe los componentes clave del sistema, los resultados preliminares y los próximos pasos para mejorar la generalización y robustez del agente.

2. Estructura del Código

El proyecto se organiza en archivos principales:

1. **data_base.py**: Descarga y procesamiento de los datos históricos financieros.
2. **environment.py**: Definición del entorno de trading, usando OpenAi gym 3
3. **qlearning.py**: Implementación del deep Q-Learning con stable-baselines3.
4. **training.py**: Entrenamiento y evaluación del agente.

3. Descripción de los Módulos

3.1 data_base.py

- **Función download_market_data**: Descarga datos históricos desde Yahoo Finance mediante la API de yfinance. La precisión y disponibilidad de los datos son factores críticos para garantizar la efectividad del agente.

- **Función calculate_indicators**: Utiliza la librería ta para calcular indicadores técnicos como las medias móviles simples (SMA), el índice de fuerza relativa (RSI) y el MACD, que son fundamentales para la toma de decisiones basada en tendencias y momentos del mercado.

3.2 environment.py

- **TradingGymEnv:** Un entorno personalizado de OpenAI Gym que simula operaciones de trading. **Características clave:**
  - Representación del estado que combina precio, indicadores técnicos y estado del portafolio.
  - Espacio de acción discreto (Mantener, Comprar, Vender).
  - Mecanismo de recompensa basado en cambios en el valor del portafolio y costos de transacción.

3.3 deepQLearning.py

- **Implementa el algoritmo PPO de la librería Stable Baselines3 para el entrenamiento del agente..**
- **Estrategia** avanzada de exploración:
  - Exploración epsilon-greedy.
  - Reducción gradual de la tasa de exploración.
  - Replay buffer para un aprendizaje estable.
- Uso de redes neuronales para aproximar los valores de estado-acción.

3.4 training.py

- Coordina el **entrenamiento, evaluación y visualización** del modelo.
- Genera **métricas de rendimiento** y gráficos.
- Guarda el modelo entrenado y los datos de desempeño.

4. Enfoque técnico.

- Algoritmo de Aprendizaje
  - **Proximal Policy Optimization (PPO)**, que utiliza una política basada en redes neuronales con exploración epsilon-greedy.
    - Precio actual.
    - Indicadores técnicos (SMA_50, SMA_200, RSI_14, MACD).
    - Número de acciones en posesión.
    - Saldo actual.

- **Hiperparámetros Clave**
  - Tasa de aprendizaje: 1e-3
  - Tasa de exploración: 1.0 → 0.05
  - Factor de descuento (gamma): 0.99
  - Tamaño del replay buffer: 10,000
  - Batch size: 64

5. Conclusión

La implementación con PPO representa una mejora sustancial sobre enfoques anteriores como DQN, al ofrecer mayor estabilidad, mejor convergencia y adaptación continua al entorno. La redefinición de la función de recompensa introduce un aprendizaje más contextualizado y enfocado en decisiones eficientes de trading.

Los resultados de esta nueva implementación se evaluaron a lo largo de las distintas etapas del desarrollo. En la última simulación, el desempeño del modelo se midió utilizando las siguientes métricas:

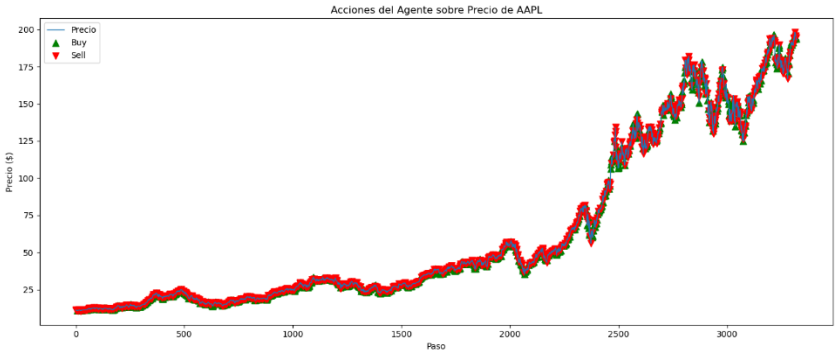
Full Dataset Performance:

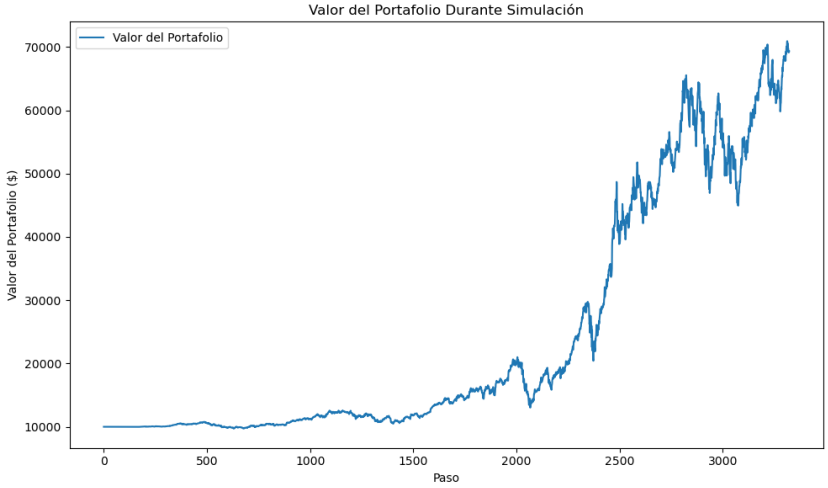
- Total Reward: 794.3453
- Sharpe Ratio: 0.6845
- Sortino Ratio: 0.8436
- Calmar Ratio: 0.4154
- Maximum Drawdown: 37.98%
- Win/Loss Ratio: 1.09
- Annualized Return: 15.77%
- Total Profit: $58953.71
- Final Portfolio Value: $68953.71

El agente entrenado con PPO logró un rendimiento sólido, alcanzando un retorno anualizado del 15.77% y un beneficio total de $58,953.71, elevando el portafolio final a $68,953.71.

Las métricas de riesgo-retorno son positivas pero mejorables: un Sharpe Ratio de 0.68 y un Sortino Ratio de 0.84 reflejan ganancias consistentes, aunque con una volatilidad considerable. El Maximum Drawdown del 37.98% indica riesgo elevado, y un Win/Loss Ratio de 1.09 sugiere una ligera ventaja en la frecuencia de aciertos.

Se graficaron los resultados obtenidos con los cambios hechos. La primera ilustra la evolución del valor del portafolio a lo largo del tiempo, mientras que la segunda detalla las acciones del agente sobre el precio de AAPL, destacando sus decisiones de compra, venta y mantenimiento en respuesta al mercado.





El diseño modular del sistema permite seguir iterando sobre los modelos, entornos y funciones objetivo, sirviendo como una base sólida para la exploración de estrategias más complejas en trading cuantitativo y deep reinforcement learning.

8. Referencias

- Fan, J., Wang, Z., Xie, Y., & Yang, Z. (2020). A Theoretical Analysis of Deep Q-Learning. Retrieved from https://proceedings.mlr.press/v120/yang20a
- Hyungjun, P., Min-Kyu, S., Dong-Gu, C. (2020). An intelligent financial portfolio trading strategy using deep Q-learning. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0957417420303973
- Yahoo Finance (2025). yfinance 0.2.54. Retrieved from https://pypi.org/project/yfinance/
- Darío López Padial (n.d.). Technical Analysis Library in Python. Retrieved from https://technical-analysis-library-in-python.readthedocs.io/en/latest/
- Morales, S. O. (s/f). MODELO DE APRENDIZAJE REFORZADO APLICADO AL TRADING DE BITCOIN. Edu.co. Recuperado el 11 de febrero de 2025, de https://repository.eafit.edu.co/server/api/core/bitstreams/45d87400-5d0f-42e2bed6611b9b5f915f/content