APRENDIZAJE REFORZADO PARA ESTRATEGIAS DE TRADING

Por

Darío Castro González

Fernando De Santos Franco

Sergio Daniel Dueñas Godínez

André 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 Q-Learning 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 según las ganancias netas tras cada operación, considerando comisiones por transacción.

El agente utiliza una política epsilon-greedy para equilibrar la exploración y la explotación durante el entrenamiento, asegurando que se exploren nuevas estrategias en fases iniciales y se prioricen las más efectivas a medida que aprende.

Esta entrega describe los componentes clave del sistema, los resultados preliminares y los próximos pasos para mejorar el modelo, como la implementación de Deep Q-Learning para manejar mejor los espacios de estado continuos.

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.
3. **qlearning.py**: Implementación del Q-Learning.
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

- **Clase TradingEnvironment**: Define el entorno de simulación donde el agente interactúa. Los estados se construyen a partir de indicadores técnicos y variables como el saldo disponible y la cantidad de acciones poseídas. La recompensa se basa en las ganancias obtenidas y penalizaciones asociadas a comisiones por transacción.

3.3 qlearning.py

- **Clase QLearningAgent**: Implementa el algoritmo Q-Learning, caracterizado por la actualización iterativa de la tabla Q utilizando la ecuación de Bellman. El agente sigue una política epsilon-greedy para equilibrar la exploración y la explotación de estrategias.
- **Exploración y Explotación**: Se implementa una estrategia de exploración decreciente mediante un factor de descuento, asegurando que el agente explore nuevas estrategias durante las fases iniciales del entrenamiento.

3.4 training.py

- **Función train_agent**: Coordina el entrenamiento del agente en el entorno. Registra las recompensas acumuladas y reduce gradualmente la tasa de exploración.

4. Resultados Preliminares

- El agente fue entrenado en el entorno con 1000 episodios, mostrando una mejora progresiva en la recompensa acumulada.
- La política aprendida muestra potencial para capturar tendencias positivas, aunque la implementación inicial se limita a posiciones largas.

5. Discusión

- **Fortalezas**: La simplicidad del algoritmo Q-Learning permite interpretaciones claras de la política aprendida.
- **Limitaciones**: La escalabilidad es un desafío debido a la representación tabular de la política. Además, la falta de generalización podría limitar el desempeño en escenarios no observados durante el entrenamiento.

6. Conclusión

La implementación actual sienta una base sólida para explorar técnicas avanzadas como el Deep Q-Learning. La efectividad preliminar del agente en entornos simulados valida el potencial del Aprendizaje por Refuerzo en estrategias de trading.

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-42e2-bed6-611b9b5f915f/content