
Universidad de San Andrés

Escuela de Administración y Negocios

Magíster en Finanzas

# *Estrategias de negociación utilizando agentes de aprendizaje reforzado*

Autor: Ariel Gonzalo Alonso

DNI: 31.752.630

Director de Trabajo Final de Graduación: Javier Kreiner

Buenos Aires, 25 de mayo de 2022

# Índice

**Resumen**

El objetivo de este trabajo es entrenar y comparar dos agentes de aprendizaje reforzado con distintos niveles de información para que negocien en alta frecuencia un bono argentino. Se busca analizar si existe una diferencia estadísticamente significativa del rendimiento obtenido por parte del agente al que se le entrega el libro de órdenes entero del bono por sobre el otro, al que sólo se le entrega la mejor oferta de compra y venta del libro. Para ello se desarrolla un modelo adaptado al tratamiento de series temporales con las características necesarias para permitir el entrenamiento y posterior predicción de acciones a tomar en la negociación del bono.

# 1. Introducción

En el ámbito de los mercados financieros la penetración de sistemas automatizados de negociación de alta frecuencia está creciendo continuamente, lo que lleva a la industria a dedicar más recursos al desarrollo de estas plataformas. En vista de esto, el presente trabajo busca utilizar una técnica de vanguardia para resolver un problema particular del ámbito financiero, que consta en evaluar la utilidad de tomar en cuenta un libro de órdenes entero de algún activo, en contraposición con sólo considerar el *top* de dicho libro. La motivación que llevó a aplicar la técnica a este tema en particular se basa por un lado en que la información completa de los libros puede ser muy costosa para algunos activos, y por otro en que la idea de conseguir rendimientos extraordinarios con respecto al mercado constituye una expectativa demasiado ambiciosa para una técnica que aún está en su fase incipiente en cuanto a la implementación en mercados financieros, pero cuyo crecimiento es notorio.

La técnica de aprendizaje reforzado es un área de *Machine Learning* que tiene como fundamento central que el entrenamiento de los algoritmos, denominados agentes, se asemeje al mecanismo de aprendizaje de los seres vivos en cuanto a la repetición de experiencias con el objetivo de perseguir una noción de recompensa acumulada.

La afirmación de esta tesis es que un agente de aprendizaje reforzado puede conseguir rendimientos extraordinarios con respecto a otro agente de similares características si se le provee un libro de órdenes de nivel 2 en comparación a uno de nivel 1. El agente es capaz de utilizar la información adicional para tomar mejores decisiones sobre cómo operar en el mercado de bonos argentinos AY24D y obtener rendimientos que se diferencien estadísticamente del otro agente.

Este trabajo presenta la siguiente estructura: la sección 2 contiene la revisión de la literatura, abarcando trabajos tanto sobre aprendizaje reforzado en general como sobre aplicaciones financieras en particular; en la sección 3 se presenta la técnica y el problema particular a resolver; en las secciones 4, 5 y 6 se detallan los datos, la metodología y el modelo que se utilizará para abordar el problema; en la sección 7 se describe la evolución y adaptación que sufre el modelo junto con resultados parciales; en la sección 8 se muestran los resultados finales; en la sección 9 se comentan las conclusiones; y en la sección 10 se mencionan nuevas posibles líneas de investigación asociadas al presente trabajo. Luego se incluyen apéndices donde se explican brevemente

la estructura y el funcionamiento de cada uno de los dos módulos principales del programa, por un lado la extracción y procesamiento de los datos de entrada y por otro los modelos de los agentes de aprendizaje reforzado, y se adjunta el código fuente del proyecto.

# 2. Revisión literaria

Se puede distinguir claramente bibliografía sobre dos temas: los textos asociados a la técnica de aprendizaje reforzado, sin aplicación a un dominio específico, y los que se refieren a aplicaciones financieras, en particular a la negociación algorítmica de alta frecuencia.

## 2.1. Aprendizaje reforzado

Dentro del primer conjunto, se toma como punto de partida el libro clásico sobre aprendizaje reforzado, tanto en su primera edición (Sutton et al., 1998) como en la última (Sutton and Barto, 2018), donde se introducen las nociones sobre este tipo de técnica de inteligencia artificial y se analizan los posibles casos de uso, identificando sus limitaciones, necesidades y características para ser de utilidad en el análisis de un problema relevante para cualquier tipo de industria.

En el artículo (Szepesvári, 2010), se resumen los conceptos y algoritmos principales utilizados en el proceso de aplicar la técnica de aprendizaje reforzado. Se detallan aspectos teóricos de los Procesos de Decisión de Markov (MDP), así como problemas típicos sobre los cuales aplicar esta técnica es adecuado e innovador.

## 2.2. Aplicaciones financieras

Existen diversas publicaciones relacionadas a la aplicación de inteligencia artificial, en particular de aprendizaje reforzado en los mercados internacionales. Muchas de ellas se caracterizan por generar eficiencia en la creación de mercado o la ejecución de órdenes, mientras que otras buscan obtener resultados anormales por sobre el mercado mediante la identificación de patrones y tendencias en los datos de entrada.

(Moody et al., 1998) introduce la aplicación de técnicas de aprendizaje reforzado al dominio de la negociación de instrumentos financieros, más

precisamente al rebalanceo de portafolios de inversión. Asimismo, indica a la técnica de aprendizaje reforzado recurrente, caracterizada por contar con una red de realimentación, como la variante más apta para participar de los mercados financieros dada su capacidad de independizarse de la alta relación de ruido a señal que caracteriza a las series de tiempo en este campo.

(Lee, 2001) muestra que es posible obtener retornos anormales usando agentes de aprendizaje reforzado para el mercado de Corea del Sur. (Lee and Jangmin, 2002) extiende la investigación en este campo planteando un esquema cooperativo de agentes de aprendizaje reforzado bajo una estructura de *Q-Learning*. Este paradigma no requiere un modelo del entorno para ser entrenado, sino que construye un conjunto de reglas que se ejecutan en base al estado actual de otras variables del ambiente en donde se aplica. El esquema de *Q-Learning* se sigue utilizando hoy en día con la diferencia de que para la representación de estados posibles se utilizan complejas redes neuronales, denominándose hoy en día esta técnica *Deep Q-Learning*.

(Deng et al., 2015) introduce la técnica de aprendizaje reforzado Directo para la negociación de alta frecuencia, indicando que se adapta mejor a las condiciones de un problema que pretende obtener retornos en el corto plazo. Marca la diferencia entre métodos basados en el entorno y métodos basados en el agente, resaltando la capacidad de estos últimos de reaccionar con más precisión a la naturaleza de las órdenes en tiempo real. (Deng et al., 2016) amplía esta estrategia, convirtiendo a la red neuronal utilizada en una de tipo multicapa e implementando estrategias de lógica difusa, lo que permite una reducción del ruido propio de los precios minuto a minuto.

(Patel, 2018) diseña una estrategia de creación de mercado utilizando dos agentes de aprendizaje reforzado; uno que identifica tendencias e indica las cantidades de activo a comprar y vender y el otro que ejecuta las órdenes de compra y venta en función de las puntas disponibles para operar. La función del agente que identifica tendencias es particularmente relevante para la implementación del método en la presente tesis.

(Yang et al., 2020) describe detalladamente el paradigma para la aplicación de agentes de aprendizaje reforzado a la administración de portafolios de acciones. Para ello aplica un conjunto de complejos algoritmos con el objetivo de maximizar la relación de *Sharpe* del portafolio, y luego compara los resultados con métodos tradicionales como la técnica de mínima varianza.

# 3. Descripción del problema

En el ámbito de los mercados financieros existe una incesante búsqueda de métodos o técnicas que permitan obtener retornos positivos anormales, lo que se conoce en la jerga como "vencer al mercado". Una de las tendencias crecientes en los últimos años es el uso de la técnica de negociación de alta frecuencia (HFT, del inglés *High Frequency Trading*) para obtener resultados extraordinarios, valiéndose de la mejorada tecnología que va poniéndose a disposición año tras año.

Para la aplicación de las técnicas de negociación de alta frecuencia es necesario disponer de datos del mercado en tiempo real; la información requerida se compone del libro de órdenes del activo a negociar. Dentro de este libro se encuentran las ofertas de compra y venta de los participantes del mercado sobre cada instrumento en particular. Las puntas más competitivas, conocidas como *Top* del libro o "Libro de Nivel 1", dan la pauta más inmediata de a qué precio se puede negociar el activo; sin embargo el libro de órdenes se compone de otras ofertas de compra y venta a precios menos competitivos, conociéndose este conjunto de información como "Libro de Nivel 2". Las contrapartes centrales que organizan la transmisión y recepción de esta información, conocidas como *Exchanges*, comercializan la disponibilidad de estos datos en tiempo real para ser adquiridos por los interesados, a diferentes precios en función de la cantidad de información brindada.

El objetivo de este trabajo será corroborar el aporte de información que realizan los datos de Libro de Nivel 2 por sobre los de Libro de Nivel 1 para tomar decisiones de inversión, mediante la utilización de agentes de aprendizaje reforzado. Se buscará probar significación estadística en el aporte de los datos adicionales que hagan al agente tomar mejores decisiones de inversión para maximizar los retornos.

En la actualidad se utiliza a nivel mundial el protocolo FIX (del inglés *Financial Exchange Protocol*) para establecer la comunicación entre los participantes del mercado y la contraparte central, enviando y recibiendo las órdenes que se emiten con el objetivo de negociar los instrumentos.

En el contexto de la inteligencia artificial se denomina a este tipo de problemas *Pie in the Sky*. Éstos presentan inicialmente un nivel de complejidad que supera la capacidad de los modelos de predicción de arribar a conclusiones convincentes. Sin embargo, en la medida en que estas técnicas continúen mejorando su potencia y capturando de una forma más extensiva el contexto

en el que se desarrolla el problema a resolver, ciertos efectos del entorno que expliquen significativamente al dominio de los datos bajo análisis comenzarán a ser capturados por los modelos, proveyendo crecientes posibilidades de en algún punto realizar un hallazgo que permita ser consistentemente eficaz.

## 3.1. Agentes de aprendizaje reforzado

Esta técnica de inteligencia artificial es la elegida en este trabajo para encarar el problema financiero de conseguir retornos anormales estableciendo una estrategia de negociación en base a los datos disponibles. El paradigma de aprendizaje reforzado (también conocido como aprendizaje por refuerzo) se basa en el estudio de cómo se produce el aprendizaje en seres vivos. Se acepta a nivel biológico que el mecanismo de prueba y error es el proceso natural que siguen todos los seres sobre la faz de la tierra para comprender cómo funciona nuestro entorno en cuanto a la física y cómo interactuar con él. Bajo esta premisa, la intención es replicar este concepto aplicando un conjunto de algoritmos. Para ello se crea un agente virtual que en un principio está despojado de todo conocimiento. Luego se debe modelar el entorno en el que actuará, tal que mediante la realización de acciones sobre este medio, el agente sea capaz de generar un resultado determinado. Para su entrenamiento se define una función de recompensa, la cual asigna valores mayores a resultados deseados y menores a los no deseados.

El esquema básico de esta estructura consta de un agente que realiza acciones sobre un ambiente y recibe como *feedback* una recompensa y el estado resultante de ese ambiente dada la acción que tomó. El agente irá ajustando sus criterios de decisión, englobados en lo que se denomina política, sobre qué acciones realizar en cierto estado del ambiente o universo utilizando un modelo ajustado a sus necesidades, en pos de maximizar una estimación de las recompensas futuras. El conjunto de acciones que puede tomar se condensan en un "espacio de acciones" y los estados en los que se puede encontrar se denominan colectivamente "espacio de estados" (Fig. 1).

Esta técnica se diferencia de otros paradigmas de inteligencia artificial por el hecho de que no está encuadrada dentro del concepto de aprendizaje supervisado o no supervisado, sino que el objetivo de la misma se relaciona con cómo el agente responde a cambios en el ambiente en el que opera, lo cual resulta "a priori" especialmente útil en entornos que evolucionen en el tiempo, tal como los caracterizados por series temporales.

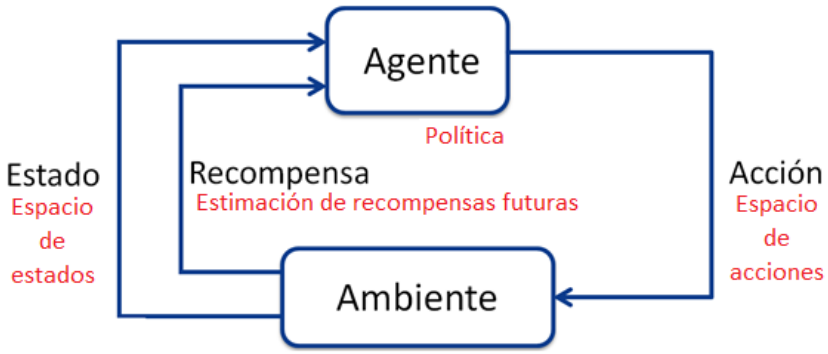



**Figura 1:** Esquema básico de la estructura de aprendizaje reforzado

El proceso de entrenamiento para estos agentes es iterativo y comparativamente extenso con respecto al de agentes que responden a otras técnicas de inteligencia artificial, debido a que la relación entre los conjuntos de datos de entrenamiento *in-sample* y *out-of-sample* es habitualmente mayor para la técnica bajo estudio, tales como agentes representados exclusivamente por redes neuronales o diversas clases de regresión. Esta diferencia se debe en parte a que el ambiente puede presentar estados diferentes en el tiempo, lo cual exige una adaptación a nuevas condiciones. Además, al momento de la inicialización del agente, no se especifican parámetros que simplifiquen el aprendizaje, sino que se modeliza el entorno y se lo deja interactuar con éste para que vaya ajustando sus parámetros internos en pos de maximizar la función de recompensa asociada. Es preciso notar que, en una gran parte de los problemas que se intentan resolver mediante esta técnica, los conjuntos de datos de entrenamiento y de prueba son el mismo, ya que no se intentan realizar predicciones en estos casos sino optimizar un resultado sobre un dominio de datos conocido, generando una sucesión de acciones ideal para ese universo.

Una característica importante de estos agentes es que le asignan un peso distinto de cero a las recompensas futuras, normalmente en contraposición al que le otorgan a las recompensas inmediatas. Como complemento de esta premisa surge la dualidad exploración-explotación, donde al agente se le puede configurar cuántas veces está dispuesto a tomar decisiones que lo depositen en estados inexplorados en búsqueda de recompensas mayores, en detrimento de arribar a un estado ya conocido que ofrece una recompensa positiva, pero relativamente pequeña. Esto se conoce como la adopción de una política estocástica, ya que el comportamiento del agente no será determinístico sino que estará asociado a una función de decisión probabilística que dictará qué

acción debe tomar dado cierto estado de la naturaleza.

## 3.2. Negociación del bono soberano AY24D

Los activos financieros argentinos se caracterizan por no respetar muchas de las asunciones que aplican sobre activos de mercados desarrollados, lo que permite arbitrar sobre ellos en ciertas circunstancias. Este hecho deja abierta la posibilidad para que nuevas técnicas de negociación, las cuales no necesariamente dan buenos resultados en mercados arbitrados, consigan resultados significativamente positivos sobre activos argentinos.

Para este trabajo se optó por el activo financiero más líquido que ofrece nuestro mercado, que es el bono AY24 en su versión en moneda dólar estadounidense (AY24D), dado que permite construir una base de datos relativamente confiable para el entrenamiento del agente de aprendizaje reforzado. Otra gran ventaja que ofrece este activo es que exhibe el menor *spread* entre las puntas compradora y vendedora, lo que facilita enormemente la aplicación de una estrategia de negociación sin una influencia decisiva de los problemas de liquidez asociados a la creación de mercado, como podría ocurrir con otros activos locales.



# 4. Datos

## 4.1. Mensajes FIX

Los datos a utilizar se componen de registros de órdenes de mercado del bono AY24D, las cuales están codificadas mediante el protocolo FIX. La decisión de utilizar este activo para desarrollar el análisis se debe a que cotiza en el mercado local Argentino, a su liquidez y a la disponibilidad de los libros de órdenes. A pesar de que los datos son en su totalidad de carácter público, no están disponibles para el usuario final directamente, sino que se deben conseguir a través de un Agente de Liquidación y Compensación (ALyC). Sin embargo, no es necesaria la autorización escrita por parte de ninguna entidad privada para su utilización, ya que se limitan a información estrictamente de mercado y son de carácter puramente histórico.

Se comenzará a trabajar con series de órdenes del activo AY24D con liquidación T+2, obtenidas directamente de Bolsas y Mercados Argentinos (BYMA). Las series ofrecen datos en tiempo real del libro de órdenes del

mercado sobre el activo de interés con una profundidad de cinco niveles. Se dispone de datos de 5 días calendarios completos, lo que da un volumen aproximado de 100.000 puntos temporales, denominados *ticks*, que contienen la información del libro de órdenes completo.

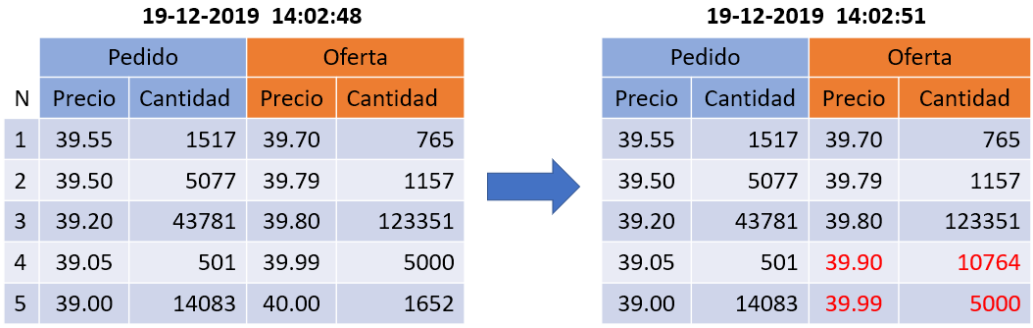
No se necesitará utilizar en un principio métodos de interpolación en los datos dado que las órdenes contienen etiqueta temporal precisa y los agentes que operen con esta información lo harán en tiempo real. En caso de que sea necesario completar ciertas entradas de alguna serie, se optará por definir el mejor criterio de interpolación en base al tipo de dato a rellenar.

Mensajes crudos de ejemplo y el modo de procesarlos para obtener los libros de órdenes finales, junto con una descripción del programa de carga de los datos, pueden encontrarse en el apéndice A del presente trabajo.

## 4.2. Libros de órdenes

Es necesario realizar una adaptación de los datos de registros disponibles a un formato legible por parte de los agentes de aprendizaje reforzado. El volumen de estos datos y sus rudimentarias características exigen un procesamiento en varias etapas, para establecer a cada instante el estado completo del libro de órdenes. Dado que una de las características principales del Protocolo FIX es transmitir la menor cantidad de datos posibles, los mensajes de datos de mercado se dividen esencialmente en dos tipos. Por una parte, los mensajes que dan la información completa del libro de órdenes, llamados *full-refresh* y por otra los que sólo dan la información de cambios en el libro, llamados *incremental-refresh*. El primer tipo de mensajes es relativamente sencillo de incorporar ya que simplemente se debe replicar la estructura de datos, pero el segundo caso exige un tratamiento exhaustivo de los datos ya que es necesario calcular el nuevo libro de órdenes en base a la modificación puntual informada en ese mensaje.

Una vez procesados los mensajes se obtienen los libros de órdenes finales, ejemplificados en la figura 2, donde se muestra la evolución del estado del libro de órdenes desde un instante al inmediatamente siguiente. Este cambio se produce por la aparición de una nueva oferta de venta del activo a un precio de $39.90 por una cantidad de 10764 unidades. La oferta que se ubicaba en la posición 4 pasa a ubicarse en la posición 5 y la de la opción 5 sale del libro de órdenes de 5 niveles.



19-12-2019 14:02:48

| N | Pedido | | Oferta | |
|---|---|---|---|---|
| | Precio | Cantidad | Precio | Cantidad |
| 1 | 39.55 | 1517 | 39.70 | 765 |
| 2 | 39.50 | 5077 | 39.79 | 1157 |
| 3 | 39.20 | 43781 | 39.80 | 123351 |
| 4 | 39.05 | 501 | 39.99 | 5000 |
| 5 | 39.00 | 14083 | 40.00 | 1652 |

19-12-2019 14:02:51

| Pedido | | Oferta | |
|---|---|---|---|
| Precio | Cantidad | Precio | Cantidad |
| 39.55 | 1517 | 39.70 | 765 |
| 39.50 | 5077 | 39.79 | 1157 |
| 39.20 | 43781 | 39.80 | 123351 |
| 39.05 | 501 | 39.90 | 10764 |
| 39.00 | 14083 | 39.99 | 5000 |

**Figura 2:** Evolución del libro de órdenes de nivel 2

# 5. Metodología

## 5.1. Configuración de la estrategia

La estrategia que se llevará a cabo será proporcionar a dos agentes de aprendizaje reforzado un presupuesto idéntico y limitado de dinero para operar. A uno de ellos se le proveerá de un entorno compuesto por el libro de órdenes completo para cada instante, mientras que al segundo sólo se le dará a conocer la orden más competitiva de compra y de venta (Fig. 3). Esto hace que el primero perciba cambios en el libro de órdenes en todos los pasos, mientras que el segundo sólo verá un cambio si el *top del book* es alterado. Cada agente deberá tomar la decisión de comprar o vender unidades del activo en base a los análisis que realice de acuerdo a la recompensa que reciba por sus acciones.

Las intervenciones del agente en el mercado se llevarán a cabo en sesiones de negociación de duración limitada, donde se le permitirá operar durante un cierto periodo, coincidente con la ventana de tiempo que se determine en cada libro de órdenes. Mediante cada sesión en la que el agente opere, parámetros internos del modelo se irán actualizando para mejorar el valor de la función recompensa obtenida.

A los efectos de simplificar el análisis de comportamiento de los agentes, se tomará la decisión de permitirles negociar solamente una unidad de bono por instante de tiempo, o *tick*. Esto tiene como objetivo minimizar el impacto que tendrían los agentes sobre el mercado, dado que se supondrá que no alteran el ambiente en el que operan. Esta consideración resulta de vital importancia para mantener la simplicidad de los modelos y el procesamiento de los datos utilizados para el análisis, además del hecho de establecer que los agentes no tendrán como objetivo la creación de mercado, sino que simplemente

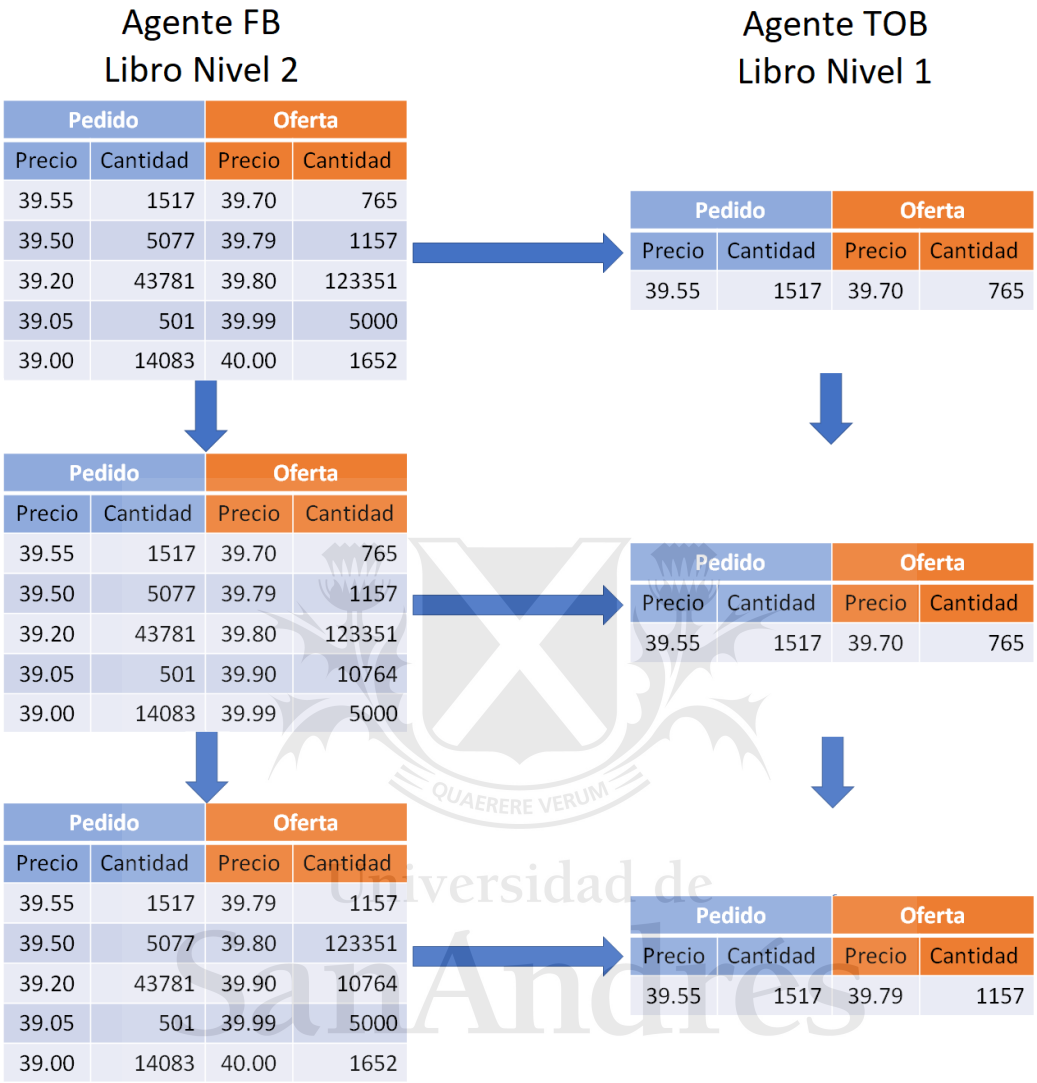



**Figura 3:** Asimetría de información entre los agentes

utilizarán los datos de los libros de órdenes para detectar oportunidades de generar ganancias negociando el bono en el corto y/o mediano plazo, teniendo que comprar al precio de *ask* y vender al precio de *bid*. En la figura 4 se ejemplifican algunas series de tiempo utilizadas con los precios mencionados y se puede apreciar el *spread* que presentan.

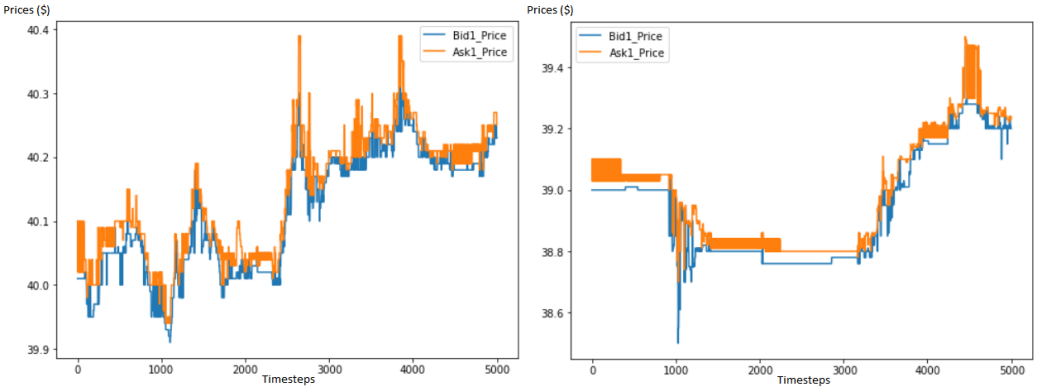



**Figura 4:** Series de ejemplo de precios de *bid* y *ask*

## 5.2. División de los datos y etapas del proceso

Para la creación de los experimentos a realizar se definirá un *set* de datos de entrenamiento, un modelo y un *set* de datos de prueba. Mediante el ajuste de estos tres bloques se buscará lograr entrenar a los agentes tal que tengan la capacidad de predecir las acciones a ejecutar para generar ganancias con cierta eficacia y consistencia. Un experimento tipo consta de tres fases principales: el entrenamiento de los agentes, la predicción de acciones sobre el conjunto de datos de entrenamiento y la predicción de acciones sobre el conjunto de datos de prueba. Es particularmente importante discernir entre la fase de entrenamiento y la de predicción sobre el conjunto de datos de entrenamiento, ya que a diferencia de otros paradigmas de *Machine Learning* como el aprendizaje supervisado, los datos en este caso no están etiquetados y por lo tanto no existe el concepto de acción correcta determinada, sino que la noción que tienen los agentes sobre qué acción será la óptima en cierto estado está representada por una función de densidad de probabilidad sobre las acciones, construida mediante el ajuste de su política.

Durante la fase de entrenamiento los agentes van ajustando esta política al recorrer los datos de entrenamiento de forma iterativa. Una vez finalizada esta fase, proceden a intentar predecir, sobre los mismos datos de entrenamiento, la acción óptima a tomar dado el estado del ambiente. Se espera que logren predecir acciones sobre este conjunto de datos tal que generen ganancias mediante la compraventa de bonos. Finalmente se procede a la fase de predicción sobre los datos de prueba, y se evalúan en este caso también las ganancias o pérdidas generadas por la participación en este nuevo conjunto de datos.

Una vez obtenidos resultados de rendimiento de cada uno de los agentes

tanto sobre los datos de entrenamiento como sobre los de prueba, se analiza si la utilización de los datos del libro de Nivel 2 hacen un aporte significativo al rendimiento del agente al que le fueron informados, por sobre el rendimiento del agente al que sólo le fueron informados los datos del libro de Nivel 1.

# 6. Modelo

## 6.1. Marco general de los modelos de aprendizaje reforzado

Un modelo de aprendizaje reforzado consta de tres capas generales para el entrenamiento de los agentes. La interna, denominada política, representa los métodos utilizados para conseguir un resultado matemático determinado, normalmente la maximización de una función objetivo, y está representada por redes neuronales con distintas características. La capa intermedia es denominada algoritmo, y se encarga de gestionar los resultados parciales que se consiguen al aplicar la política elegida al conjunto de datos de entrada, y forma un criterio general para las decisiones del agente. Por último, la capa más externa es la que determina el ambiente donde se mueve el agente; las características que se le proveen para tomar decisiones; los hiperparámetros de configuración y cómo se define la recompensa asociada a cada acción que tome en determinado estado del universo. Las combinaciones de estas tres capas determinarán lo que llamamos experimentos, los cuales marcarán la hoja de ruta para la evaluación de los resultados.

Los agentes realizarán su entrenamiento por repetición, con patrones de entrenamiento por bloques definidos de acuerdo a un hiperparámetro del modelo. Estos bloques conforman episodios de entrenamiento, durante los cuales una recompensa es entregada al agente, representada por un valor numérico a cada instante. El valor y la forma de otorgar la mencionada recompensa se irá ajustando en función del experimento que se lleve a cabo para determinar de qué forma es más eficiente para el éxito del entrenamiento de los agentes.

La configuración inicial para la operación de los agentes es definida tal que comiencen su sesión de negociación con una cantidad nula de efectivo y puedan comprar o vender, incluso en corto, una unidad del bono por instante de tiempo, teniendo como parámetro configurable un límite en la posición larga o corta de una cantidad definida de bonos.

## 6.2. Algoritmos de entrenamiento

Las acciones que pueden tomar los agentes sobre su entorno son la compra o la venta de unidades de los bonos AY24D, lo que convierte al espacio de acciones en uno de naturaleza finita. Esta característica restringe la variedad de algoritmos a utilizar a un subconjunto, recomendados para un problema de este tipo. Debido a la naturaleza aleatoria en el comportamiento de estos algoritmos, se opta por establecer una semilla ( *"seed"*) fija para permitir reproducir experimentos controlados. Existen varias clasificaciones para este tipo de algoritmos; en la más representativa de ellas, podemos clasificarlos en dos grandes grupos: los de función de valor, que buscan asignar un valor numérico al estado en el que se encuentra el agente para puntuar su rendimiento, y los de *policy gradient*, que trabajan directamente sobre la función de la política, es decir, sobre la distribución de probabilidades sobre las acciones posibles dado un estado de la naturaleza. También existen híbridos que combinan ambas aproximaciones, intentando combinar las ventajas de uno y otro grupo.

- *Deep-Q-Network* (DQN)

  Este algoritmo de función de valor es una adaptación del algoritmo *Q-Learning*, el cual representa mediante una tabla todos los posibles estados del mundo y busca asignarle una recompensa numérica fija a cada acción que se pueda tomar en cada estado. Esto funciona bien para problemas con un espacio finito de estados, donde el mantenimiento de esta tabla de decisión es manejable. Dado que para el problema de negociación algorítmica la cantidad de estados tiende a infinito (ya que los precios y las cantidades de un activo forman conjuntos muy grandes de estados), se trata de simplificar esta tabla mediante funciones que optimicen el camino a seguir en cada uno de esos estados. Para esto se incorporan redes neuronales capaces de modelar el conjunto de estados infinitos y representar mediante funciones el valor asignado a cierto estado y la relación entre la recompensa estimada por ejecutar determinada acción en ese estado sujeto a cierta política. Implementa además un búfer de repetición, que es un mecanismo inspirado en la biología que consiste en utilizar muestras aleatorias de acciones pasadas en lugar de la más reciente acción para tomar decisiones. Esto elimina correlaciones en la secuencia de observaciones y suaviza los cambios en la distribución de los datos.

- *Proximal Policy Optimization* (PPO)

  La idea de este algoritmo de *policy gradient* es estabilizar los criterios de decisión del agente, evitando que tome decisiones muy distintas entre

sí ante estados relativamente similares. Para ello combina la utilización de varios agentes en paralelo y establece una región de confianza sobre la cual construir decisiones, lo que redunda en que no se permitan modificaciones abruptas en su política.

- *Sample Efficient Actor-Critic with Experience Replay* (ACER)

  Este algoritmo combina características de los dos anteriores, siendo un híbrido, como usar varios agentes en paralelo y una región de confianza (como PPO) o tener un búfer de repetición (como DQN). Por sobre todas las ventajas anteriores, es el único de esta lista que es compatible con políticas recurrentes para sus procesos de decisión, lo que lo dotaría de una mayor robustez en el tratamiento de series de tiempo.

## 6.3. Políticas de los agentes

Las políticas utilizadas en este trabajo por los agentes se basan en redes neuronales del tipo perceptrón multicapa; se utilizarán tres variantes de estas políticas:

- Perceptrón Multicapa Simple (MLP)

  Esta política consta de un perceptrón multicapa de 2 capas de 64 neuronas cada una. Su objetivo es maximizar la función objetivo, que es en este problema la recompensa, o valor del portafolio al final de cada episodio.

- Perceptrón Multicapa con normalización de capas (LnMLP)

  Esta variante añade la normalización de las distribuciones de las capas intermedias de la red, lo que produce gradientes más suaves, entrenamiento más rápido y mejor capacidad de generalización.

- Memoria a corto y largo plazo con normalización de capas y extracción de características a través de Perceptrón Multicapa (LnLSTM+MLP)

  Esta es una política de característica recurrente que incorpora los datos de salida de estados anteriores como información de entrada al nuevo estado, con el objetivo de construir sus propios indicadores de largo, mediano y corto plazo para tomar mejores decisiones.

## 6.4. Ambientes del sistema

Al contrario de lo que ocurre con las dos capas más internas del modelo, que están relativamente estandarizadas por el estado del arte del aprendizaje reforzado en general, para la construcción de los ambientes el panorama

se presenta menos definido y es necesario establecer definiciones en muchos aspectos relativos al entrenamiento y a la evaluación de rendimiento de los agentes. En este trabajo se diferencian claramente los ambientes de entrenamiento y de prueba, representando el primero el mayor desafío para lograr el aprendizaje de los agentes y el que más dificultades presenta y configuraciones admite, ya que el proceso de definir una política exitosa es el objetivo final. Se debe definir en éste la forma en que se recorre iterativamente el conjunto de datos de entrenamiento, por ejemplo en bloques más pequeños o considerando todos los datos como un único bloque, lo que convierte a este en otro factor que incide directamente en cómo el agente ajusta su política. El ambiente de prueba por su parte simplemente constará de presentarle a los agentes una serie de datos de un día entero, ya sea utilizando datos de entrenamiento o de prueba, para que realicen la predicción de una acción a realizar en cada *tick* de esa serie, el cual representa el estado del universo que tienen como entrada a cada instante.

La distinción entre los dos ambientes es fundamental para comprender por qué no es automático el buen rendimiento del agente cuando se lo pone a predecir acciones sobre los datos de entrenamiento, es decir, sobre los mismos datos sobre los que fue entrenado. Debido a que las rutinas de entrenamiento están condicionadas por la forma en que los agentes son recompensados, no existiendo en este marco teórico acciones correctas o incorrectas propiamente dichas.



## 6.5. Características del modelo

Como parte del proceso de construcción del ambiente se deben definir puntualmente qué características, también denominadas *features*, serán las que alimenten a los modelos para su entrenamiento, incluyendo posibles combinaciones de ellas para simplificar o complejizar el *set* de datos de entrada, siendo además normalizadas en un paso intermedio. Las características alimentadas al modelo varían para los diferentes experimentos realizados, mencionándose los más relevantes a continuación:

- *"Bid Price"*: Los precios de compra del libro de órdenes, en sus 5 niveles disponibles.
- *"Ask Price"*: Los precios de venta del libro de órdenes, en sus 5 niveles disponibles.
- *"Bid Size"*: Las cantidades de ofertas de compra del libro de órdenes, en sus 5 niveles disponibles.

- *"Ask Size"*: Las cantidades de ofertas de venta del libro de órdenes, en sus 5 niveles disponibles.
- Efectivo (*"Agent Cash"*): Efectivo que posee el agente.
- Cantidad de bonos (*"Agent Bond Quantity"*): Posición del agente en bonos, positivo si está largo y negativo si está corto.
- Desequilibrio de mercado (*"Size Imbalance"*): El indicador simplificado que busca unificar en una sola característica toda la profundidad del libro de órdenes, ya sea de nivel 1 o de nivel 2. Para calcular esta característica en el caso del libro de nivel 1, simplemente se realiza la resta de las cantidades de venta menos la de compra del precio más competitivo:

  $$Equivalente_1 = BidSize_1 - AskSize_1$$

  En el caso del libro de nivel 2 se estima el equivalente como

  $$Equivalente = Equivalente_1 + \sum_{i=2}^{i=5} (BidEquivalente_i - AskEquivalente_i)$$

  donde

  $$BidEquivalente_i = exp(-(BidPrice_1 - BidPrice_i)/\tau) * BidSize_i$$

  $$AskEquivalente_i = exp(-(AskPrice_i - AskPrice_1)/\tau) * AskSize_i$$

  De esta manera se ponderan las cantidades de cada línea de compra y venta en función de la distancia al precio más competitivo, disminuyendo su influencia a medida que se alejan de ese precio, modulado por una constante $\tau$ parametrizable. Lo que se busca con esta simplificación es condensar la información del libro entero de órdenes en un solo valor para ambos casos, reduciendo la cantidad de características iniciales a una única, con una representatividad mayor que utilizar todos los valores como características separadas.
- *MACD*: Este indicador técnico para medir *momentum* se calcula como

  $$MACD = EMA_{12} - EMA_{26}$$

  donde EMA representa la media móvil exponencial de la cantidad de períodos denotada en el subíndice, los que en este trabajo estarán representados por cada tick del libro de órdenes. Dado que los datos

utilizados son de mayor frecuencia que los períodos habituales utilizados por este indicador, se corregirá el *span* que determina la cantidad de datos considerados para calcular la media multiplicándolo por una constante $\mu$ parametrizable, y utilizando el precio medio entre compra y venta, resultando el indicador adaptado

$$MACD_\mu = EMA_{12\mu} - EMA_{26\mu}$$

- *MACD Signal*: La señal del MACD busca representar el *momentum* del propio indicador $MACD_\mu$ a través de la media móvil exponencial de 9 períodos, como

$$MACD_\mu Signal = EMA_9(MACD_\mu)$$

- *MACD* simplificado: La información provista por la característica $MACD_\mu$ y la $MACD_\mu Signal$ se condensa en una nueva característica que toma valores categóricos del conjunto `{-1; 0; 1}` en función de cuándo y en qué sentido la serie $MACD_\mu$ cruza a la $MACD_\mu Signal$. Si el cruce es hacia arriba, se asignará un 1; si es hacia abajo un $-1$ y si no se produce cruce en ese punto de la serie, se asignará el valor 0.

- *RSI*: Es un indicador técnico de tipo oscilador que busca expresar la fuerza relativa que tiene una serie de precios relacionando los cambios de precios en los períodos alcistas y bajistas pasados. La idea central es estimar cuándo un activo está sobrecomprado o sobrevendido, y usar esta información como una señal para operar en consecuencia. Toma valores entre 0 y 100 y se calcula de la siguiente manera:

$$RSI = 100 * EMA_U(n)/(EMA_U(n) + EMA_D(n))$$

  donde $U = P_i - P_{i-1}$ y $D = P_{i-1} - P_i$, con $P_i$ el precio de el tick presente y $P_{i-1}$ el precio del tick anterior, asumiendo que ante una diferencia positiva del precio se utiliza $U$, y ante una negativa $D$. Se define $n = 14$ por convención y se aplica la corrección temporal mediante el multiplicador $\mu$, el cual es el mismo que para el indicador *MACD*, por consistencia temporal, resultando el indicador a utilizar como característica

$$RSI_\mu = 100 * EMA_U(n\mu)/(EMA_U(n\mu) + EMA_D(n\mu))$$

- *RSI* simplificado: El indicador RSI puede convertirse en una variable categórica que toma valores en el conjunto de enteros `{-1; 0; 1}`, casos que corresponden a estar sobrecomprado, neutral, o sobrevendido el

activo, respectivamente. Esta conversión se determina usando un umbral simétrico parametrizable, que transforma, para un valor ejemplo de 30, a los valores de la serie RSI que están por debajo de 30 en un 1, a los que están entre 30 y 70 en 0 y a los que están por sobre 70 en $-1$.

- *"Bid-Ask spread"*: Es la diferencia de precio entre las ofertas de compra y de venta más competitivas.
- Características discretizadas: Tanto la variable *"Size Imbalance"* como *"Bid-Ask spread"* pueden ser discretizadas para reducir a un número finito y parametrizable la cantidad de valores numéricos que pueden tomar, con el objetivo de reducir la cantidad de estados posibles a los que se enfrenten los agentes.

## 6.6. Hiperparámetros del modelo

Durante la etapa de entrenamiento, se configura a través de hiperparámetros el ambiente preciso en el cual se produce el entrenamiento. A continuación se detallan los principales:

- Multiplicador EMA de los indicadores (*"Indicators EMA span multiplier"*, "$\mu$"): Debido a que la naturaleza de los indicadores MACD y RSI *vanilla* es utilizar medias móviles con precios de cierre diarios de activos, para compensar el hecho de que estamos trabajando con datos de alta frecuencia, multiplicamos las *EMA* (*exponential moving average*) por una constante $\mu$.
- Frontera RSI (*"RSI threshold"*): Para simplificar el indicador RSI debemos establecer una frontera numérica para generar la variable categórica "RSI simplificado". Este parámetro determina ante qué valor del indicador original, el modificado toma los valores correspondientes a la transformación.
- Horizonte de recompensa (*"Reward Horizon"*): Este parámetro establecido en *ticks* se utiliza para determinar el plazo al cual se liquidan posiciones abiertas o se calculan las recompensas de los agentes, dependiendo de la configuración puntual del experimento. Su influencia es mayúscula dado que va a impactar en la capacidad del agente de conseguir realizar operaciones que le reporten una ganancia, modulando el horizonte de inversión dentro de la sesión de negociación.
- Tamaño de la serie de datos de entrada diaria (*"Entries Retained"*): Determina cuán larga será la serie de tiempo de cada día, medida en *ticks* del libro de órdenes.

- Cantidad de entradas removidas (*"Entries Removed"*): Se elimina una cierta cantidad de *ticks* al inicio y al final de la serie de tiempo de cada día para eliminar aberraciones cerca del inicio y el cierre de la rueda.

- Fechas a usar como datos de entrenamiento y prueba: De los 5 días de datos de libro de órdenes disponibles, se selecciona un subconjunto para formar parte del *set* de datos de entrenamiento (hasta 4 días); un día para representar un *set* de datos de validación (hasta 1 día) y un día restante para representar el *set* de datos de prueba.

- *Bid-Ask spread*: Determina si se añadirá la característica del mismo nombre al conjunto de datos de entrada al sistema.

- Simplificación de indicadores técnicos (*"Simplified Indicators"*): Determina si se realizará una simplificación de las características de indicadores técnicos, pasando de tener $MACD_\mu$, $MACD_\mu Signal$ y $RSI_\mu$ a tener $MACD_\mu$ simplificado y $RSI_\mu$ simplificado.

- Desequilibrio de mercado (*"Market Imbalance"*): Este parámetro activa o desactiva la simplificación del libro de órdenes a una única característica numérica, llamada *"Size Imbalance"* que represente las cantidades de compra y venta del libro en ese instante.

- Discretizar (*"Discretize"*): Variable que determina si se discretizan las características *Size Imbalance* y *Bid-Ask Spread*.

- Cantidad de intervalos de discretización (*"Discretization Quantity"*): Variable entera que determina la cantidad de intervalos en los que se discretizarán las dos características mencionadas en el hiperparámetro precedente.

- Longitud del episodio de entrenamiento (*"Episode Length"*): Valor numérico que determina la cantidad de ticks del set de datos de entrenamiento que se considerarán para definir un episodio de entrenamiento. Una vez culminado un episodio, se ajusta la política del agente en base a las recompensas recibidas durante el transcurso de éste.

- Límites de posición (*"Position Limits"*): Determina si se establecen límites a las posiciones de efectivo y bonos del agente.

- Máximo efectivo (*"Max Cash"*): Posición en efectivo máxima, positiva o negativa, que se le admite al agente.

- Máxima cantidad de bonos (*"Max Bond Quantity"*): Posición en bonos máxima, en largo o en corto, que se le admite al agente.
- Penalización por exceder la posición máxima (*"Position Penalty"*): Determina si se penaliza al agente mediante una recompensa negativa si supera los límites de posición máxima positiva o negativa de bonos.
- Monto de la penalización por exceder la posición máxima (*"Position Penalty Amount"*): Determina en qué magnitud numérica se penaliza al agente si excede la posición en bonos admitida.
- Penalización por no operar (*"Trading Penalty"*): Resuelve si se penaliza al agente por no operar durante un episodio de entrenamiento.
- Monto de la penalización por no operar (*"Non Trading Penalty Amount"*): Establece la recompensa negativa numérica por no operar durante un episodio de entrenamiento.
- Potenciación de las recompensas positivas (*"Positive Reward Booster"*): Determina si las recompensas positivas serán multiplicadas por una constante para sobreponderarlas por sobre las negativas.
- Factor de potenciación de recompensas positivas (*"Reward Boost Factor"*): Establece el valor numérico que multiplica a las recompensas positivas.
- Recompensa inmediata (*"Reward Immediate"*): Determina si se activan las recompensas inmediatas al agente.
- Recompensa al final del episodio (*"Reward End of Episode"*): Determina si se se activa la recompensa al final del episodio al agente.
- Penalización por negociar en alta frecuencia (*"High Freq Penalty"*): Determina si se aplica una penalidad por negociar demasiado frecuentemente.
- Cantidad de *ticks* de alta frecuencia (*"High Freq Ticks"*): Establece el plazo medido en *ticks* durante el cual el agente es penalizado si negocia más de una vez dentro de éste.
- Monto de la penalidad por alta frecuencia (*"High Freq Penalty Amount"*): Establece una recompensa negativa para el agente en caso de negociar sucesivamente en el plazo definido por el hiperparámetro precedente.

- Modelo (*"MODEL"*): Determina qué algoritmo de aprendizaje reforzado se utilizará para el experimento (DQN, PPO2, ACER).
- Semilla del modelo (*"Model Seed"*): Establece la semilla que utilizará el modelo, garantizando control sobre el entrenamiento y asegurando reproducibilidad de los resultados.
- Pasos (*"Timesteps"*): Es la cantidad de *ticks* totales que entrenará el modelo, independientemente de la longitud del episodio. Por ejemplo, para una cantidad de 100k *timesteps* y una longitud de episodio de 1k *ticks*, el entrenamiento constará de 100 episodios, la razón de ambas magnitudes.

# 7. Cronología de experimentos y evolución de los modelos

En la presente sección se describirá el proceso de ajuste del modelo hasta llegar a su versión final. El camino recorrido para determinar las características y los valores finales de los hiperparámetros consta de la realización de experimentos cada vez más complejos, añadiendo, quitando y modificando características y haciendo variar los valores de estos hiperparámetros para evaluar el rendimiento a cada paso.

## 7.1. Ambientes y experimentos iniciales

Como configuración de experimento inicial se opta por realizar el entrenamiento de los agentes en episodios cortos cuyo punto de inicio es escogido aleatoriamente, utilizando series de longitud definida por el hiperparámetro "Horizonte de recompensa", el cual se ajusta en 10 *ticks*, y se itera sobre el conjunto de datos completo hasta alcanzar un tiempo de entrenamiento definido. En cada instante de tiempo o *tick* que el agente entrena, se le asigna una recompensa inmediata en base a la acción que elija, de compra, venta, o no operación, la cual es calculada como el precio al que va a poder liquidar su posición en el instante siguiente menos el precio que pagó o cobró por el bono en el instante actual. Se utiliza la serie de libro de órdenes de un día calendario como conjunto de datos de entrenamiento y de otro día como conjunto de prueba. La elección tanto del día de entrenamiento como del de prueba es parametrizable.

La definición de esta recompensa inicial busca confirmar que los agentes son capaces de entender que si operan van a tener una pérdida inmediata.

Dado que todos los movimientos de precios están reflejados en la serie de datos utilizada, si existe un cambio en un precio o una cantidad del libro de órdenes, el agente lo verá como el estado siguiente, haciendo imposible que obtenga una recompensa positiva por operar, ya que lo que puede comprar en el instante presente lo podrá vender al instante siguiente a un precio indefectiblemente más bajo que el que pagó, y viceversa en caso de que venda. Los resultados de este experimento confirman que los agentes no intervienen en el mercado, eligiendo siempre no operar.

El paso siguiente es modificar la estrategia de recompensa, y establecer una única recompensa al final del episodio, que se calcule como el valor del portafolio en ese instante. Es decir, si el agente tiene efectivo y bonos, se calcula este valor de portafolio como el precio de liquidación de los bonos sumado al efectivo que posea. Este escenario busca que los agentes detecten algún tipo de tendencia en los datos para elaborar estrategias que les reporten ganancias. Los primeros experimentos demuestran que los agentes no son capaces de identificar oportunidades tal de generar consistentemente un valor del portafolio positivo luego de un día de negociación, y buscan operar lo menos posible para no pagar el costo del *spread* entre ofertas de compra y venta, incluso cuando la recompensa se da al final del episodio. Aquí se empieza a evidenciar que con horizontes de inversión cortos, es prácticamente imposible obtener una ganancia operando el bono, ya que el movimiento de los precios requiere cierta cantidad mínima de *ticks* para que el precio de compra futuro supere al precio de venta actual, permitiendo que una operación de compraventa sea redituable. Por lo tanto se considerará incrementar el hiperparámetro Horizonte de Recompensa para los próximos experimentos (Fig. 5).

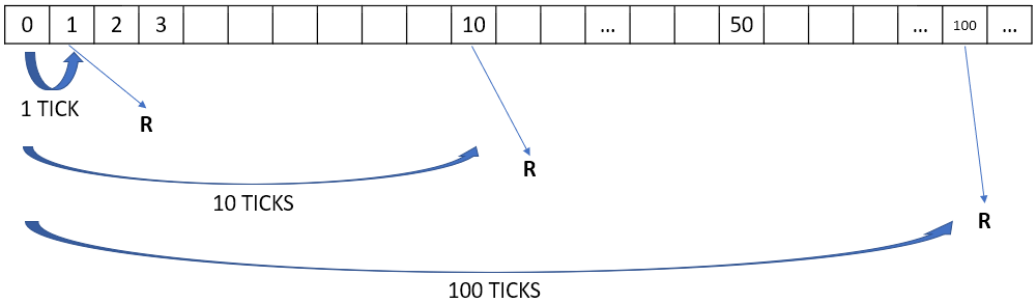



**Figura 5:** Recepción de una recompensa "R" al final del episodio con horizonte parametrizable

En esta instancia se debe poner el foco en la convergencia de la política de los agentes, tal que se esté entrenando una cantidad de tiempo suficiente

para asegurar que la toma de decisiones tenga consistencia. Para verificar esta convergencia en el caso del modelo PPO2 utilizado para este experimento, debemos observar cómo evolucionan algunos parámetros internos del modelo (Fig. 6), donde una recompensa estable al final del episodio de entrenamiento (*"episode_reward"*) y una entropía del modelo (*"entropy_loss"*) que tienda a 0 aseguran que la política ya es prácticamente determinística por lo que las decisiones de los agentes están determinadas en base a los estados que ha experimentado, y por lo tanto que se ha reducido al mínimo la probabilidad de explorar nuevas acciones y se explotarán los resultados conocidos para maximizar el valor de la recompensa del episodio. Estos mismos conceptos son por su parte aplicables al algoritmo ACER, que comparte ciertas características con PPO2 que permiten evaluar su convergencia de manera análoga. En el caso de DQN, el algoritmo produce, como herramienta para chequear convergencia, un gráfico de pérdida (*"loss"*), el cual representa una medida de distancia entre los pesos de la red que estima la función de pares estado-acción, comparando los pesos actuales con los pesos que el algoritmo calcula como valores óptimos, siendo estos últimos también dinámicos durante el proceso de entrenamiento.

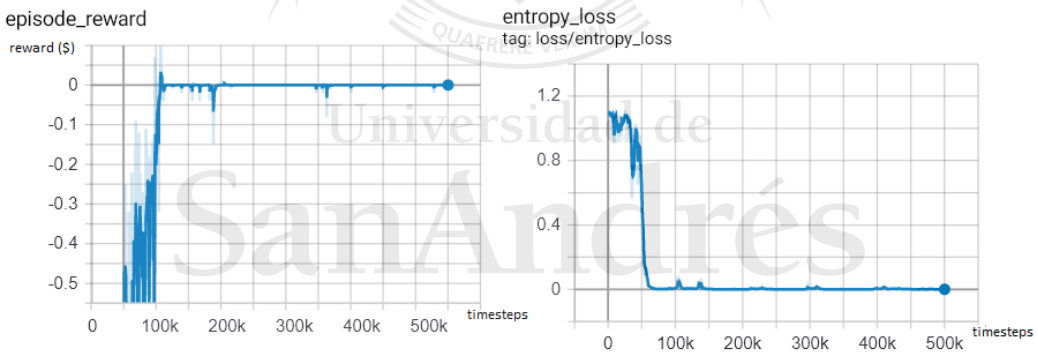



**Figura 6:** Parámetros internos del modelo durante el entrenamiento del agente PPO2

Se observa que la entropía del modelo desciende bruscamente alrededor de la muestra número 60.000 y coincidentemente la recompensa se acerca a cero, al haber aprendido el agente a aplicar la acción de esperar en cada instante, independientemente del estado del libro de órdenes.

La predicción de los modelos sobre el conjunto de datos de prueba, para el algoritmo PPO2 con un Horizonte de recompensa = 50 (Fig. 7) muestra que el agente no opera en el mercado dado que considera que mantenerse al margen es la mejor estrategia para maximizar su recompensa.

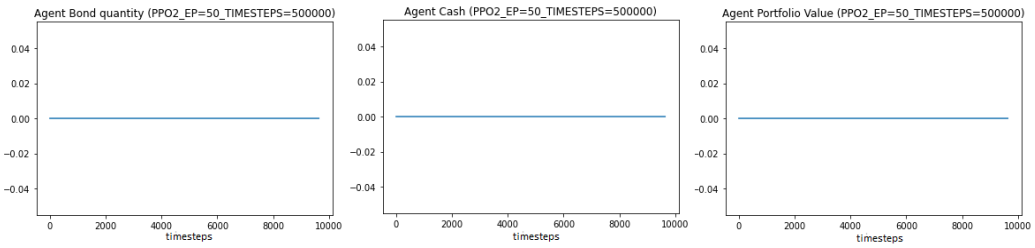



**Figura 7:** Rendimiento del agente PPO2

Es preciso destacar que la configuración del horizonte de inversión, que es la variable que determina cuántos *ticks* durará cada episodio de entrenamiento, genera dos cambios fundamentales en el ambiente. Por un lado, aumenta el plazo de negociación de los agentes, por lo que tienen más tiempo para esperar que sus posiciones se conviertan en redituables antes de ser liquidadas; pero por otro, la convergencia de sus políticas es mucho más lenta, requiriendo estos experimentos un tiempo de entrenamiento sustancialmente mayor, lo que dificulta su evaluación. Esta consecuencia negativa se debe a que la cantidad de recompensas observadas por el agente está determinada por la razón entre la cantidad total de *ticks* de entrenamiento y la longitud del episodio.

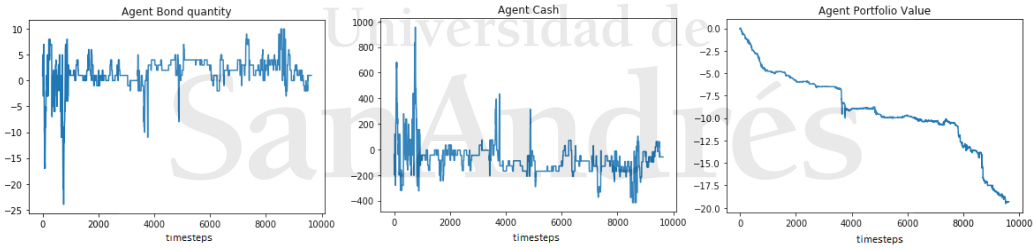



**Figura 8:** Rendimiento del agente DQN con política no estabilizada

Se puede observar que el agente DQN entrenado por un tiempo insuficiente (Fig. 8) presenta comportamientos erráticos y en prácticamente ninguna intervención en el mercado resulta en un aumento del valor del portafolio, a pesar de que debe destacarse que la tenencia de bonos en el tiempo oscila en torno a cero sin haberle instruido al agente a hacer esto, lo cual es deseable y auspicioso.

Estos resultados, si bien representan "a priori" un resultado trivial, sientan bases sólidas para buscar nuevas características a incorporar al modelo e hiperparámetros a ajustar en pos de que los agentes eventualmente comiencen

a detectar alguna característica en los datos que les permita operar y obtener recompensas positivas consistentemente.

## 7.2. Series de datos *dummy*, incorporación de indicadores técnicos, penalización por no operar y simplificación de características

Para comprobar la capacidad de los agentes de detectar patrones simples, se crean series artificiales de datos con formas triviales, como rampas, triángulos y dientes de sierra, y se entrena a los agentes sobre ellos para comprobar qué resultados arrojan sus predicciones. Los primeros resultados no fueron convincentes para ninguno de los 3 algoritmos (DQN, PPO2, ACER), tomando comportamientos que no se ajustan a las acciones que se esperaría realicen.

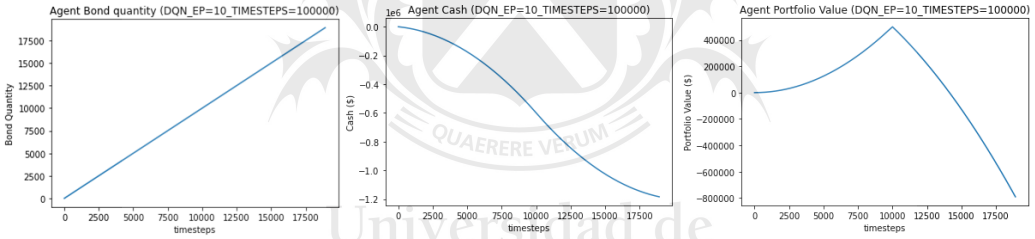



**Figura 9:** Serie *dummy* (diente de sierra) - Agente DQN

El agente DQN (Fig. 9) sólo parece aprender la parte ascendente y no la descendente de una serie triangular, resultando en que siempre escoge la acción de comprar, lo que provoca una brusca caída en el valor de su portafolio una vez que la serie entra en la etapa descendente.

Estos resultados llevan a pensar en que la forma de analizar la serie de los agentes y la información contenida en ellas no resulta suficiente para entender la correlación temporal entre *ticks*. Si bien los algoritmos computan en sus capas internas variables intermedias para poder en teoría realizar estas conexiones, no se evidencia en los resultados obtenidos. Por este motivo se decide la incorporación externa de indicadores técnicos a las series de datos, como el MACD y el RSI, en sus variantes $MACD_{\mu}$, $MACD_{\mu}Signal$ y el $RSI_{\mu}$ tal como son descriptos en la sección 6.5, fijando $\mu = 50$, valor que permite obtener series poco ruidosas sobre los datos de entrenamiento, pero con suficientes cambios de tendencia para ser de utilidad.

Los primeros resultados obtenidos sobre datos *dummy* son positivos, reconociendo los agentes que las características de los indicadores son claramente la guía a tomar en cuenta para decidir sus acciones, ya que para estas series ficticias marcan la tendencia unívocamente. Se puede notar una diferencia llamativa entre los algoritmos DQN (Fig. 10) y ACER (Fig. 11) en su comportamiento sobre la serie de datos de diente de sierra, habiendo sido entrenados sobre una serie de datos triangular únicamente. El primero se adapta inmediatamente a los cambios de tendencia mientras que el segundo presenta cierto ruido en sus acciones ante estos cambios, perdiendo un poco de rendimiento.

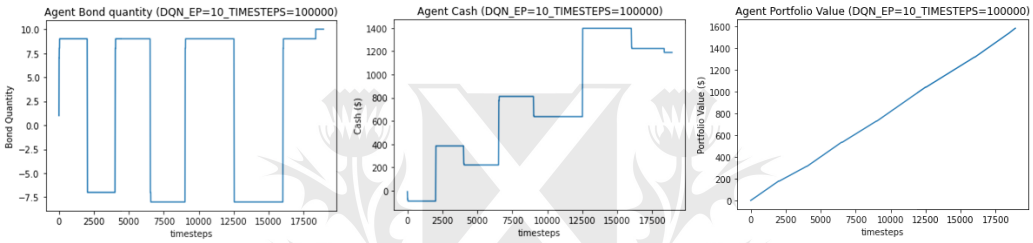



**Figura 10:** Serie *dummy* (diente de sierra) - Agente DQN

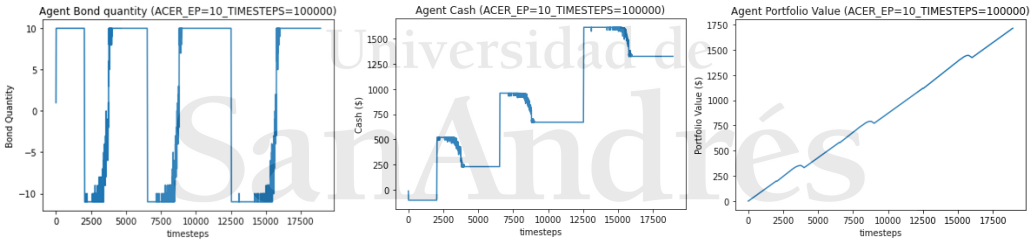



**Figura 11:** Serie *dummy* (diente de sierra) - Agente ACER

Retornando a entrenar al agente sobre datos reales, los resultados arrojan que una vez que la política converge el agente no opera, continuando en la situación de los experimentos iniciales.

En paralelo con esta inclusión de indicadores se realiza la simplificación de las variables del libro de órdenes de precio en cantidad y se unifican en una llamada "Desequilibrio de mercado" según lo descripto en la sección 6.5. La motivación de este procedimiento se basa en el hecho de que reducir la cantidad de características que alimentan a los modelos y por ende la dimensionalidad del problema disminuye drásticamente el tiempo de entrenamiento

hasta que la política converja, y por otra parte esta nueva característica relaciona de una forma coherente las distintas magnitudes que hacen al libro de órdenes, valores que por sí solos poca información le dan al agente, ya que no tiene una manera confiable, ni explícita ni implícita, de relacionarlas e interpretarlas.

La situación de no intervenir en el mercado motiva la introducción de una penalidad a la recompensa del agente, la cual es aplicada al final del episodio si se comprueba que ninguna acción dentro del mismo fue una compra o una venta. Esto resulta en una situación en la que el agente tenga que decidir si seguir sin operar aceptando la penalidad u operar el bono, incluso cuando sabe que va a ir a pérdida.

Las primeras conclusiones de esta nueva penalidad se dan en la fase de entrenamiento del agente PPO2 (Fig. 12), donde se comprueba que no se llega a la convergencia de la política ya que la medida de entropía se mantiene en magnitudes positivas apreciables. Esto marca que las predicciones del agente resultan ser preponderantemente aleatorias, y los resultados se tornan no representativos. Esto se exacerba si la penalidad numérica es relativamente grande, ya que fomenta más intervención del agente en el mercado con resultados negativos.

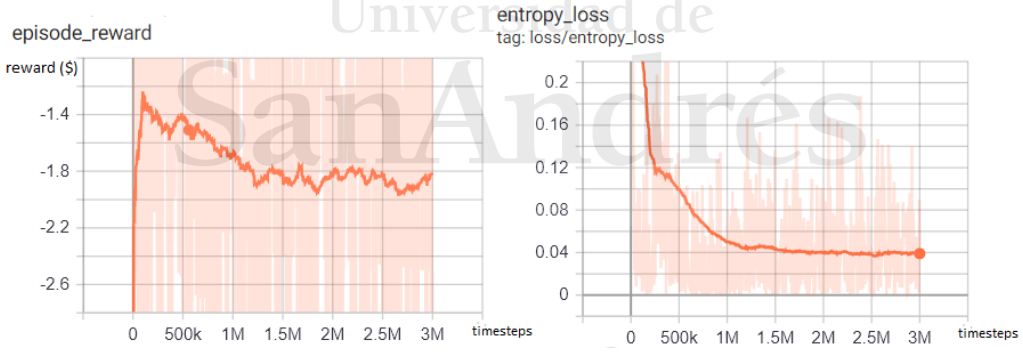



**Figura 12:** Pérdida de entropía positiva del agente PPO2

Con la filosofía de seguir simplificando características, se observa que los indicadores técnicos pueden ser convertidos a magnitudes que representen más directamente lo que intentan predecir. En el caso del $MACD_\mu$ y el $MACD_\mu Signal$, lo que habitualmente un negociador evalúa es dónde se dan los cruces entre ambas series, y toma este punto en el tiempo como señal de compra o venta, por lo que se crea una nueva serie llamada "MACD Simplificado" que reemplace a las dos. Algo similar ocurre con el $RSI_\mu$, para

el cual se toma que pasado cierto límite hacia arriba o hacia abajo de su valor, el activo se encuentra sobrevendido o sobrecomprado, reemplazando esta serie por una de tipo categórica, que se denomina "RSI simplificado". El detalle de la construcción de estas series se describe en la sección 6.5.

Con la remoción de la penalidad por no operar y las simplificaciones de las características de los indicadores técnicos los agentes retornan al comportamiento pasivo en el mercado, y no operan, en tanto y en cuanto hayan entrenado un tiempo suficiente para estabilizar su política.

## 7.3. Nuevas características, discretización y cambio en la concepción de las recompensas

En busca de alternativas para fomentar la negociación por parte de los agentes, se implementa una potenciación a las recompensas positivas que ellos reciben. Mediante una regla que multiplica estas recompensas para los episodios de entrenamiento donde obtienen ganancias el objetivo radica en que los agentes den mayor importancia a estas experiencias en particular. Los primeros resultados indican que esta regla no modifica el comportamiento pasivo, pero de todas maneras se mantiene ya que en combinación con otras estrategias podría resultar efectiva.

Se propone la remoción de características que podrían no ser esenciales como la cantidad de efectivo y la tenencia de bonos que tiene el agente en su poder, ya que no deberían afectar directamente las decisiones que tome con respecto a la operación del bono porque no son variables asociadas a la recompensa que recibe. Como simplificación adicional de características se quitan los precios de compra y venta más competitivos ("Bid1_Price" y "Ask1_Price") y se convierten en una nueva característica de *spread* que resulta la resta de ambas, información condensada que en definitiva parece representar una mejor manera de entregar información al agente, ya que los precios en valor absoluto, más allá de ser útiles para calcular las posiciones, no aportarían información necesariamente útil para tomar decisiones de negociación.

Estas modificaciones generan resultados notables sobre el conjunto de datos de entrenamiento (Fig. 13) y también sobre el de prueba (Fig. 14). El patrón de compraventa de bonos es relativamente lógico, con una intervención en el mercado esporádica y balanceada, y el valor del portafolio comienza a resultar positivo en algunas realizaciones del experimento, si bien es pertinente aclarar que esta magnitud es altamente variable y toma valores positivos

como negativos con la reiteración del experimento, debido a la aleatoriedad inherente en el entrenamiento de los modelos.

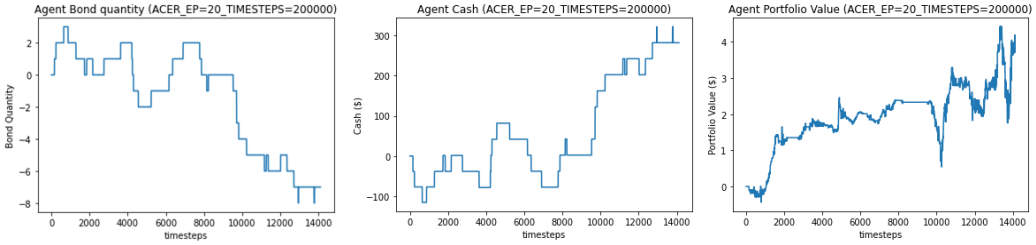



**Figura 13:** Resultados sobre el conjunto de datos de entrenamiento - Agente ACER

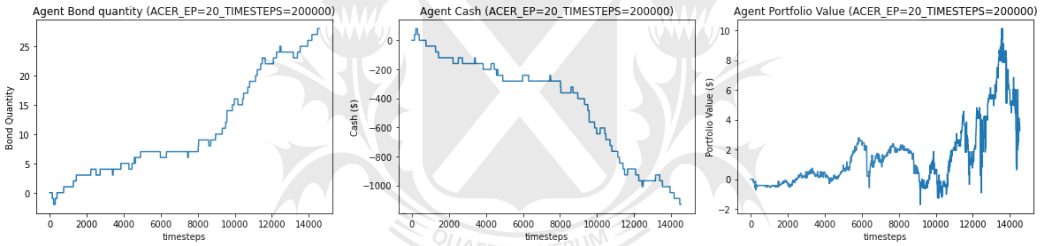



**Figura 14:** Resultados sobre el conjunto de datos de prueba - Agente ACER

Si bien los algoritmos de entrenamiento de aprendizaje reforzado tienen en teoría la capacidad de manejar una cantidad de estados infinitos debido a la utilización de redes neuronales para la estimación de parámetros, se propone la discretización de ciertas características numéricas que pueden tomar una cantidad de valores grande, como son el desequilibrio de mercado ("Size Imbalance") y el "Bid-Ask *spread*". Mediante los hiperparémtros "Discretize" y "Discretization Quantity" se habilita esta modificación de las variables y se determina en cuántos niveles discretos serán recategorizadas. De esta forma se busca reducir además el tiempo de entrenamiento requerido para estabilizar la política de los agentes.

A pesar de que los resultados de rendimiento no son significativamente diferentes, se realizan dos pruebas de entrenamiento y se comparan los resultados, utilizando 5 y 10 niveles de discretización respectivamente para las variables mencionadas. Se observa que si bien existe cierta similitud en las decisiones del agente entrenado en ambos experimentos, pequeñas diferencias en cuándo y qué acción toma entregan resultados de valor de portafolio muy disímiles al final del día (Figs. 15, 16). Por su parte el resultado de los agentes

sobre los datos de prueba es relativamente alentador (Fig. 17), observándose que la sucesión de acciones se parece muy poco a las que realizó sobre los datos de entrenamiento, es decir, que el proceso de entrenamiento no induce al agente, a través de la construcción de su política, a replicar comportamientos adquiridos independientemente del estado de las variables del universo.

Se retoma la observación detallada de los datos de entrenamiento internos del modelo (Fig. 18) y se nota que en algunos episodios de entrenamiento el agente logra recompensas positivas, multiplicadas como se estableció por un factor parametrizable, para que las tome más en cuenta, pero son muy pocas; en cuanto a la entropía, sigue estando consistentemente por sobre cero. Para intentar confirmar que el modelo no esté subentrenado, se procede a entrenarlo durante un tiempo sustancialmente mayor y se observa que la estrategia elegida por los agentes pasa a ser la de no operar, confirmando que algunos resultados parciales obtenidos que parecían ser positivos estaban influenciados por algún estado temporal de la política interna que luego fue desestimado como el óptimo al seguir entrenando.

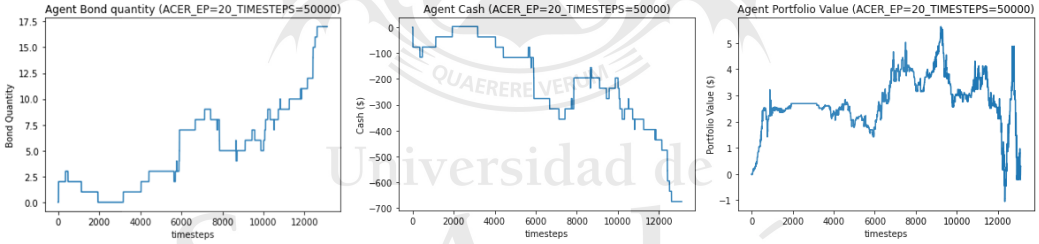



**Figura 15:** Rendimiento sobre datos de entrenamiento - 5 niveles de discretización - Agente ACER

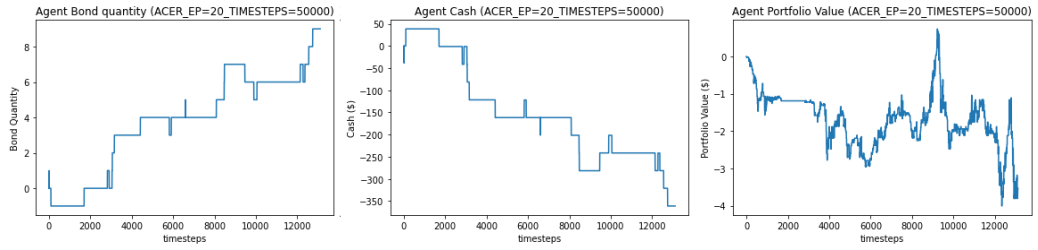



**Figura 16:** Rendimiento sobre datos de entrenamiento - 10 niveles de discretización - Agente ACER

La próxima modificación al modelo consta de una redefinición importante de la recompensa otorgada a los agentes. Se pasa de otorgar una recompensa

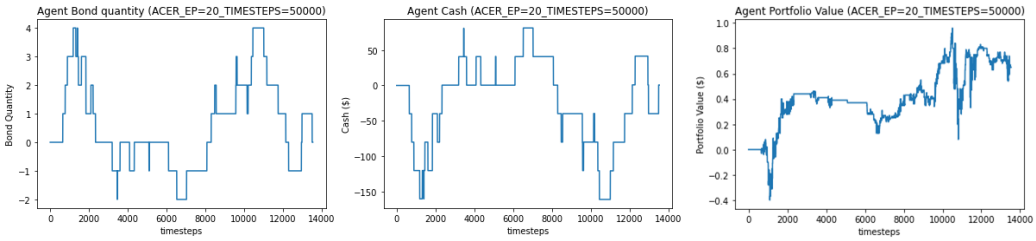



**Figura 17:** Rendimiento sobre datos de prueba - Agente ACER

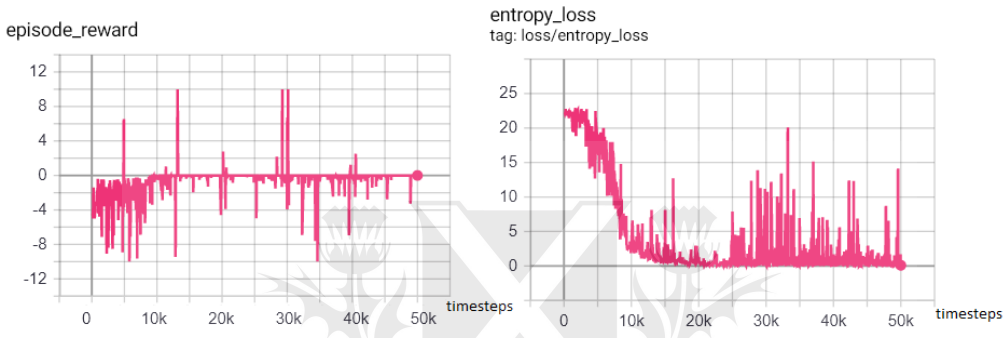



**Figura 18:** Entrenamiento incompleto (entropía elevada) - Agente ACER

única al final del episodio de entrenamiento a entregar una en cada instante. El valor de esta recompensa instantánea estará dado por la resta de dos precios; se sustrae el precio en el momento presente de compra o venta del precio de liquidación de esa posición en un instante futuro, determinado por el hiperparámetro "Horizonte de recompensa". Mediante esta modificación, los agentes estarán recibiendo una recompensa que representa su ganancia realizada en cierto horizonte, lo que individualiza a cada acción de compra o venta sin estar intrínsecamente relacionada a las inmediatamente precedentes o posteriores. Esto se logra mediante un cálculo interno en el entrenamiento del modelo, disponibilizando el precio futuro de compra y venta en cierto instante determinado por el hiperparámetro en cuestión, tal de poder calcular durante el entrenamiento del agente a cada paso este valor. Es importante notar que estos datos con información del futuro sólo estarían disponibles en un entorno exclusivo al entrenamiento del modelo y no al momento de evaluar el rendimiento del algoritmo, lo que implicaría que exista una violación al principio de la no utilización de datos futuros al momento de entrenar. En otras palabras, no se agregan como una característica adicional que alimenta al modelo en esta instancia.

Los primeros resultados con esta modificación marcan que sobre un deter-

minado de conjunto de datos de entrenamiento, el agente aprende a predecir correctamente acciones tal de maximizar el valor del portafolio, sobre todo hacia el final del día donde hay una caída significativa del precio del bono (Fig. 19). No ocurre lo mismo sobre los datos de prueba (Fig. 20), donde el rendimiento es negativo, y aquí es donde se empieza a manifestar un fenómeno muy poco deseable, que se trata de que un agente intente replicar una serie de acciones similar sobre un conjunto de datos de entrenamiento distinto, independientemente del estado del universo en ese momento. Este comportamiento se observa particularmente al final de la serie, donde la acción de vender toma mayor preponderancia repitiéndose sucesivamente, lo que hace que el agente quede siempre con una posición en corto en bonos hacia el final del día.

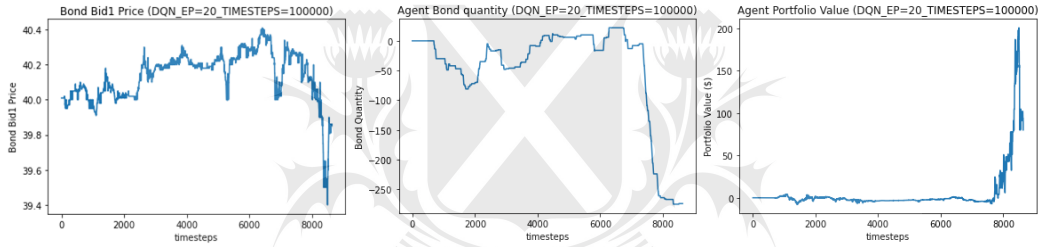



**Figura 19:** Rendimiento sobre datos de entrenamiento (horizonte=20) - Agente DQN

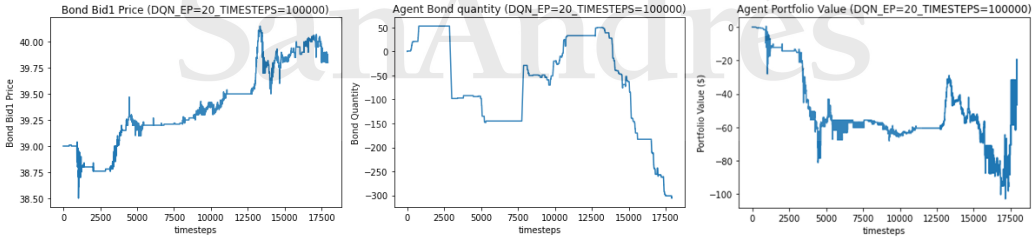



**Figura 20:** Rendimiento sobre datos de prueba (horizonte=20) - Agente DQN

Se observa sobre los parámetros internos del modelo que la recompensa total por episodio ("episode_reward") comienza a tomar valores positivos consistentemente, lo que evidencia que el agente está entendiendo qué acciones tomar dado cierto estado individual de la naturaleza, simplemente conociendo experiencias pasadas, y desligándose de alguna forma de la noción de serie temporal que los libros de órdenes representan. El cálculo de la recompensa total del episodio en este contexto se toma como la suma aritmética de las

recompensas individuales que recibe el agente en cada instante de tiempo. En comparación con un experimento donde el parámetro de horizonte se fija en 100 *ticks*, se observa que la magnitud de las recompensas toma valores más positivos aún (Fig. 21).


**Figura 21:** Comparación de recompensas por episodio de entrenamiento (horizonte=20 vs 100) - Agente DQN

En este punto debe tenerse cuidado en no estar utilizando un *set* de datos que sea favorable para que el agente elija cierta acción por sobre otra, por ejemplo si el agente se entrena sobre una serie de precios predominantemente creciente como la de la figura 20 existe el riesgo de que la política entre en un estado espurio donde determina que la acción de comprar es la más beneficiosa, independientemente del estado del universo.

Se verifica que estos resultados, que en principio parecían prometedores, no pueden ser reproducidos sobre otros conjuntos de datos de entrenamiento y de prueba utilizando información del libro de órdenes de otros días calendarios, lo que evidencia una falta de robustez del modelo y lleva a buscar nuevas alternativas para mejorarlo. Se realizan pruebas exhaustivas también sobre el algoritmo ACER donde se observan comportamientos extraños sobre el valor linealmente decreciente de la recompensa del episodio (Fig. 22), y se llega además a la conclusión de que con suficiente cantidad de tiempo de entrenamiento, los agentes recurren nuevamente a la decisión de nunca operar.

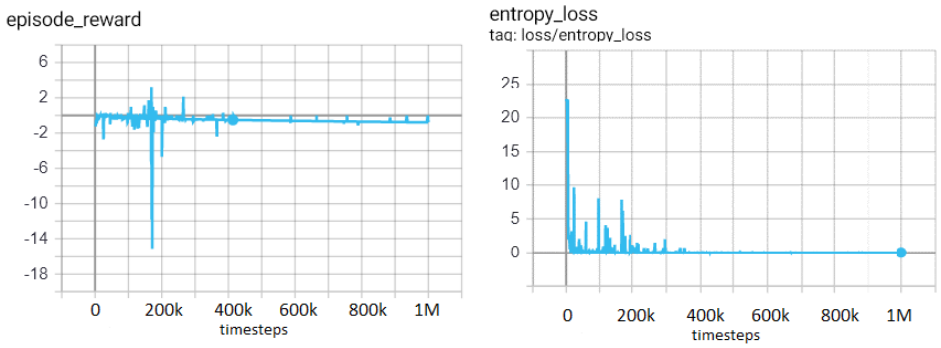



**Figura 22:** Comportamiento espurio con recompensa decreciente - Agente ACER

## 7.4. Ampliación de los conjuntos de datos y penalización a las posiciones abiertas

Para intentar solucionar el inconveniente de estar utilizando algún tipo de serie de precios que fomente a los agentes a inclinarse por una acción única a realizar, se actualiza el ambiente de entrenamiento incorporando toda la información disponible, asignando datos de 4 días calendarios, de elección parametrizable, como conjunto de datos de entrenamiento, y el restante día calendario como conjunto de datos de prueba. Esta modificación conlleva cambios importantes en la forma en que se recorren los datos durante el entrenamiento ya que se tiene que tener especial cuidado en no atravesar la nueva serie de precios unificada pasando de un día a otro en un mismo episodio, lo que no reflejaría un escenario real de *trading*.

Para este experimento se establece un episodio de longitud corta (10 *ticks*) con el objetivo de que los agentes no tengan la posibilidad de asociar un día calendario específico a la serie en la que están entrenando al representar ésta sólo una pequeña parte del día entero, y adicionalmente se fija un horizonte de recompensa (20 *ticks*) un tanto mayor que el episodio, con el objetivo de capturar tendencias de un plazo corto pero no tanto como para no poder encontrar oportunidades de realizar operaciones de compraventa redituables. Se implementa en esta instancia además el entrenamiento paralelo de los agentes (*top_of_book* ("TOB") y *full_book* ("FB")) para poder empezar a comparar el rendimiento de ambos. Es pertinente aclarar que la elección del día de prueba es rotativa para poder analizar la robustez del modelo y no se considera en esta instancia que la utilización de datos de un día futuro sobre la predicción de un día anterior corresponda a una violación al principio de no utilizar datos aún no conocidos durante el entrenamiento, ya que cada serie

de tiempo se tratará como una serie temporalmente independiente, debido a que los conjuntos de datos diarios disponibles no representan necesariamente días calendarios contiguos.

Los resultados permiten observar cómo el agente *FB* es el único que negocia, mientras que el *TOB* llegó a una política de mantenerse al margen del mercado. Se observa que para ciertos días de entrenamiento el agente *FB* predice acciones de compra o venta, logrando resultados mixtos en los distintos días de entrenamiento (Figs. 23, 24, 25, 26), y perdiendo dinero en el dia de prueba (Fig. 27).

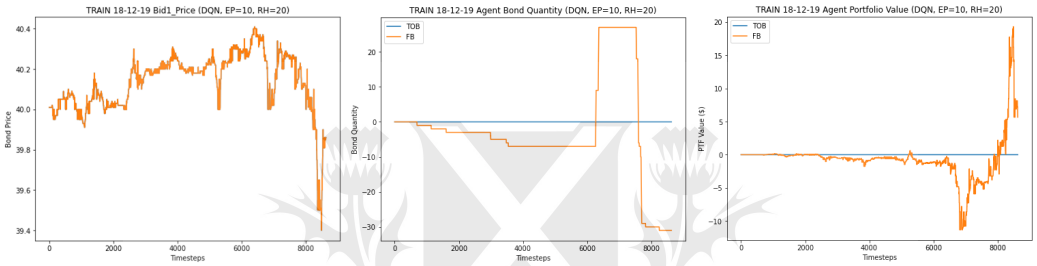



**Figura 23:** Predicciones sobre datos de entrenamiento - Día 18/12/19 - Agente DQN

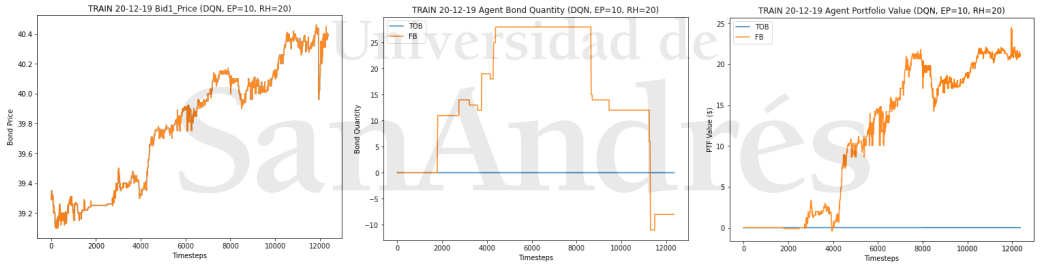



**Figura 24:** Predicciones sobre datos de entrenamiento - Día 20/12/19 - Agente DQN

Una consecuencia no deseada observada en estos resultados son los cambios extremos en la posición en bonos, tanto en largo como en corto, que adopta el agente que negocia. Lo ideal sería mantener la cantidad de bonos en un nivel bajo para mantener la volatilidad controlada, y debido a que en el ambiente en el que se desarrolla el experimento no se limita mediante una restricción explícita la tenencia de bonos, se crea una situación de este tipo, donde el agente no es consciente de las implicaciones negativas de su tenencia. Por lo tanto se decide en este punto restaurar como una característica alimentada al modelo la cantidad de bonos en tenencia, pero en lugar de forzar

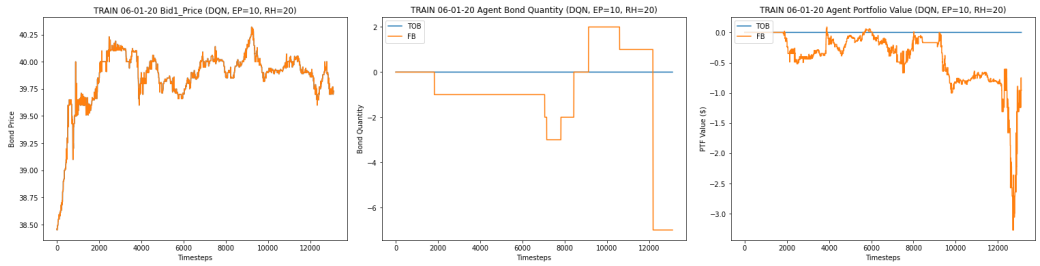



**Figura 25:** Predicciones sobre datos de entrenamiento - Día 06/01/20 - Agente DQN

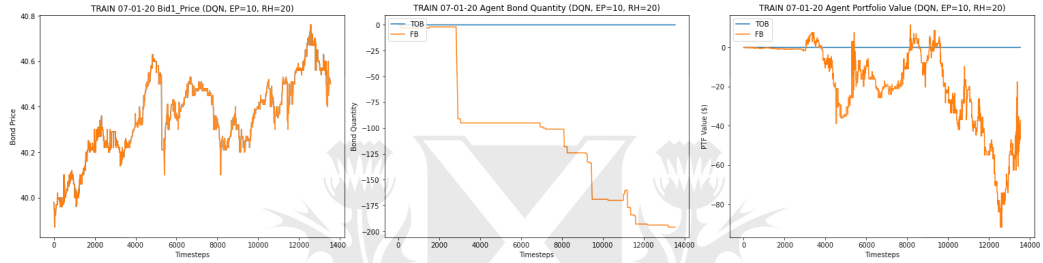



**Figura 26:** Predicciones sobre datos de entrenamiento - Día 07/01/20 - Agente DQN

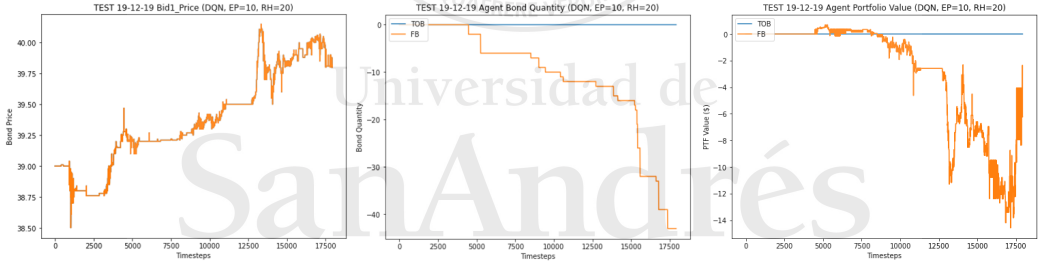



**Figura 27:** Predicciones sobre datos de prueba - Día 19/12/19 - Agente DQN

que su tenencia no supere cierta cantidad, se propone un cambio paradigmático: Penalizar con recompensas negativas a los agentes que superen cierto nivel de tenencia de bonos, complementando la noción de recompensa, hasta ese momento sólo dada por el resultado de liquidar la posición recientemente abierta en el futuro. Ahora se suma una nueva componente a esa recompensa, lo que se espera que regule exitosamente su tenencia, determinada por los hiperparámetros *"Position Penalty"* y *"Position Penalty Amount"*, fijados en 5 y -5 respectivamente. En otras palabras, cada vez que el agente tenga una posición en corto o en largo de 6 o más bonos, recibirá una penalidad de -5, impactando severamente el valor de recompensa acumulada y por lo tanto

su política de decisión.

Las primeras evaluaciones acerca del entrenamiento de los agentes demuestran que han aprendido exitosamente a mantener su tenencia dentro de los límites, y además evidencian particularidades con respecto a la frecuencia de negociación. El patrón de compraventa sobre datos de prueba (Fig. 28) muestra que los agentes son más propensos a comprar y vender que cuando no estaban restringidos por la posición de bonos en cartera. Esto genera indefectiblemente que paguen con más frecuencia el costo del *spread* por lo que su rendimiento resulta muy negativo. Por otra parte, el algoritmo DQN aprende con mayor rapidez a operar dentro de los límites de tenencia para no ver penalizada su recompensa, mientras que el algoritmo ACER evidencia en la figura 29 que el agente TOB, representado por el trazo celeste, aprendió recién a la mitad del proceso de entrenamiento que no debía salirse de los límites impuestos por la limitación de tenencia, mejorando sustancialmente el valor de recompensa del episodio, mientras que el representado por el trazo rojo nunca lo aprende. A pesar de este comportamiento al entrenar, el agente ACER no fue capaz de mantenerse dentro de los márgenes de tenencia permitidos al momento de predecir sobre el conjunto de datos de prueba.

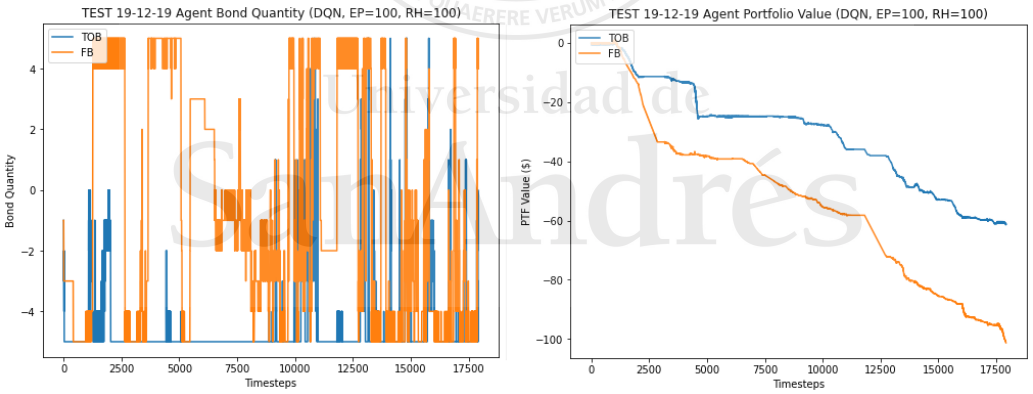



**Figura 28:** Predicciones sobre datos de prueba con limitación de tenencia - Agente DQN

Para intentar limitar la operación sucesiva de los agentes se implementa un tipo de penalización que castigue a la recompensa de los agentes si operan el bono, tanto para compra como para venta, antes de cierta cantidad de *ticks* desde la última operación. Esto tiene como objetivo generar una reticencia en los agentes a encadenar operaciones en *ticks* demasiado cercanos, limitando las potenciales pérdidas por pago del *spread*. Las características de la penalización están controladas por los hiperparámetros *"High Freq Penalty"*,

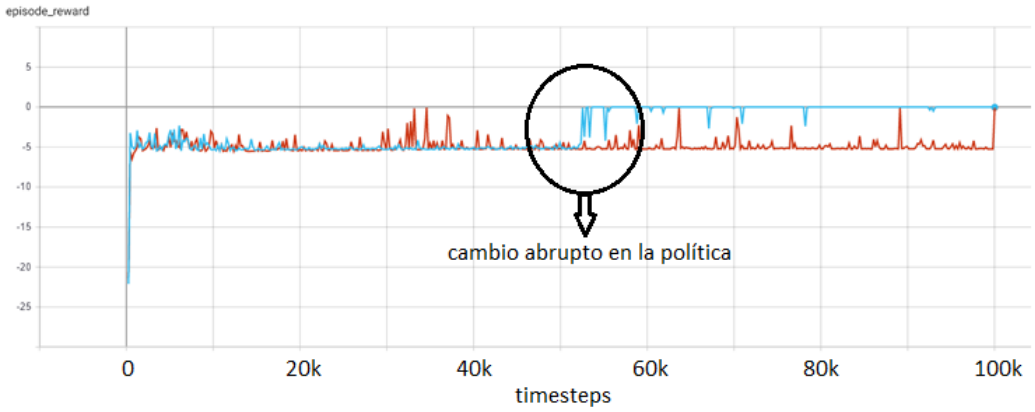



**Figura 29:** Detalle de cambio de política a mitad del entrenamiento - Agente ACER TOB (celeste) vs FB (rojo)

*"Position Penalty Amount"* y *"High Freq Ticks"*, que determinan respectivamente si se activa este tipo de penalidad, qué magnitud tendrá esta penalidad si se incurre en ella, y cuántos *ticks* deben dejarse pasar sin operar para no ser sancionado. De los resultados sobre datos tanto de entrenamiento (Fig. 30) como de prueba (Fig. 31) se observa que esta nueva penalidad reduce drásticamente la cantidad de operaciones de los agentes, y adicionalmente restringe indirectamente la tenencia máxima que alcanzan, dejando sin efecto a la penalidad por cantidad de bonos en cartera. El agente TOB no opera y el FB realiza algunas operaciones pero sin lograr resultados consistentemente positivos.

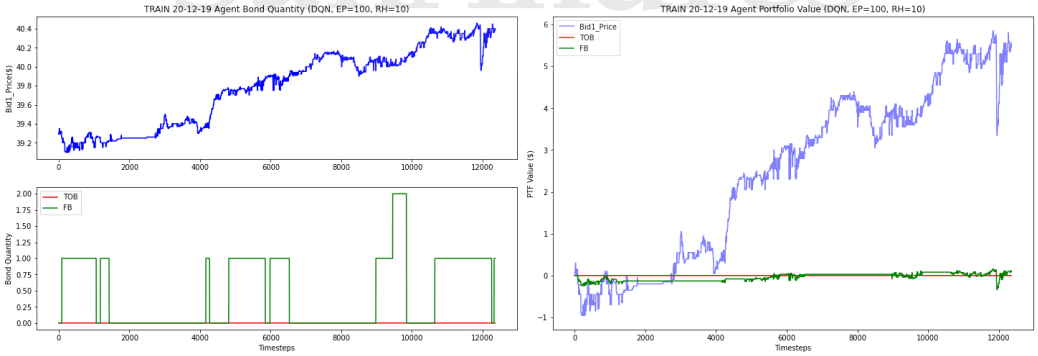



**Figura 30:** Predicciones sobre datos de entrenamiento con penalidad por alta frecuencia - Agente DQN

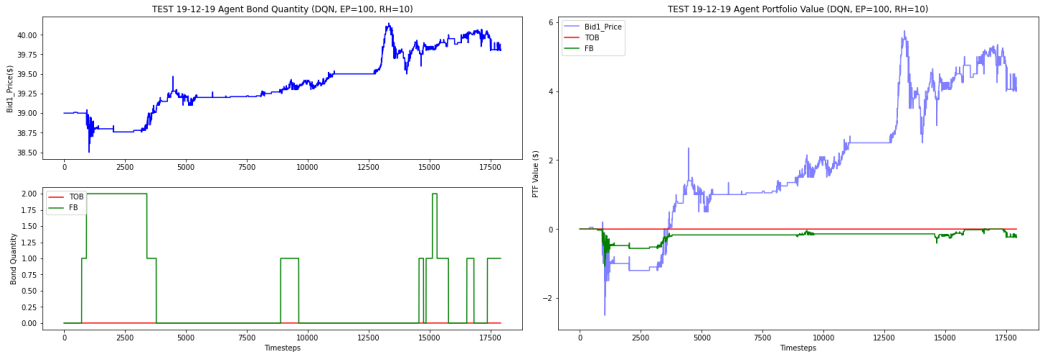



**Figura 31:** Predicciones sobre datos de prueba con penalidad por alta frecuencia - Agente DQN

## 7.5. Liquidación anticipada de la posición y redefinición y reducción de la cantidad de datos de entrenamiento

Atentos a la volatilidad en el valor del portafolio que presentan los agentes, se decide liquidar las posiciones abiertas en un futuro próximo, sin necesidad de esperar al fin de la rueda de negociación. Al hacer coincidir el horizonte de liquidación con el horizonte que determina la recompensa recibida por el agente ante cada operación, se asegura que si la recompensa fue positiva ante una operación, el resultado de ésta en cuanto a valor de portafolio, sea de igual magnitud y signo. Esto a su vez reduce el volumen de tenencia de bonos en todo momento, ya que se liquidan las posiciones antes en el tiempo y permiten individualizar aún más cada operación que realiza el agente.

Los primeros experimentos arrojan resultados mixtos, donde se observa que sobre uno de los días de entrenamiento (Fig. 32) el rendimiento de los agentes es exactamente el deseado, comprando ante una subida de precios y vendiendo ante una caída. También se puede observar que la tenencia de bonos tiene una media cercana a cero, ya que al operar poco, la posición global en bonos retorna a cero cumplido el horizonte de tiempo. Para los demás días de entrenamiento (Figs. 33, 34, 35) y para el día de prueba (Fig. 36) los resultados no son deseables, operando mucho y casi siempre perdiendo. El efecto de liquidar la posición en un horizonte determinado genera que la tenencia máxima de bonos se vea afectada por éste, fijado en 100 ticks, debido a que ante una sucesión de acciones de compra infinita, cumplido el tiempo determinado por el horizonte desde la primera compra, ese bono se venderá a la vez que se compra uno nuevo, cancelando el cambio en la posición.

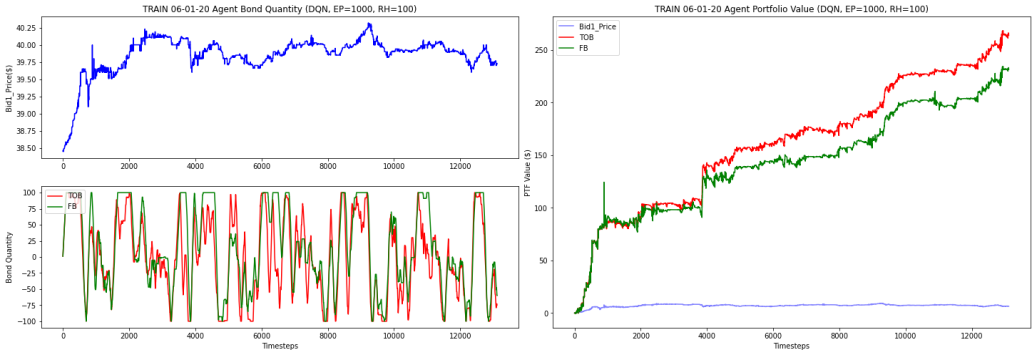



**Figura 32:** Predicciones sobre datos de entrenamiento - Resultados positivos - Día 06/01/20 - Agente DQN

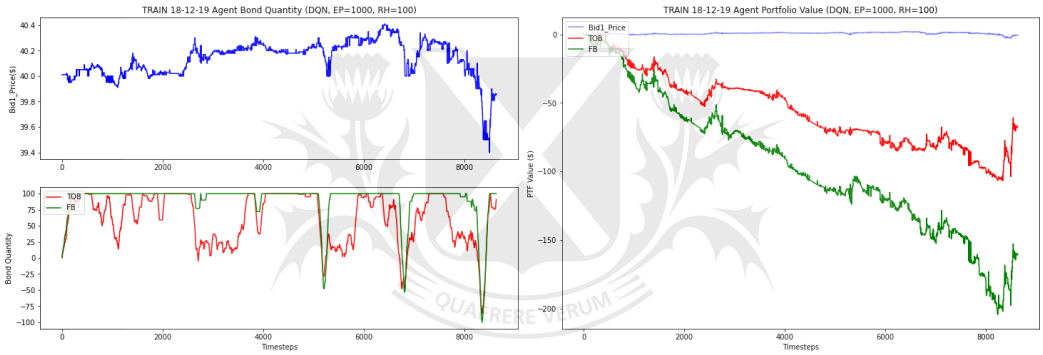



**Figura 33:** Predicciones sobre datos de entrenamiento - Resultados negativos - Día 18/12/19 - Agente DQN

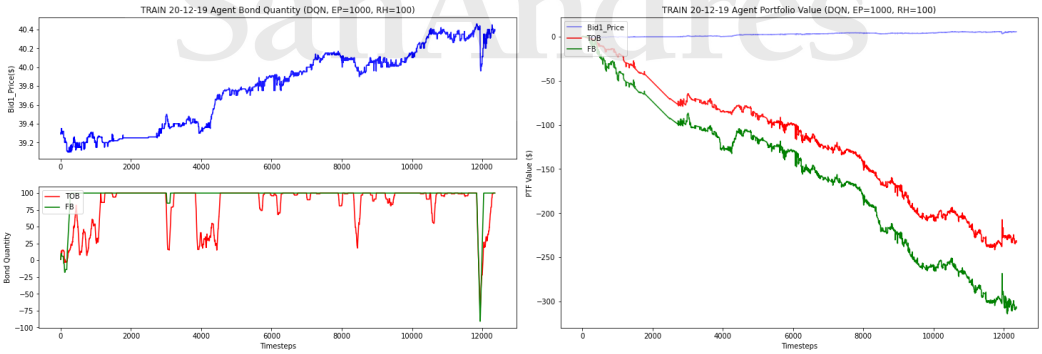



**Figura 34:** Predicciones sobre datos de entrenamiento - Resultados negativos - Día 20/12/19 - Agente DQN

Estos resultados llevan a pensar que el entrenamiento generó una adaptación al subconjunto de datos del día donde se consiguieron buenos resultados,

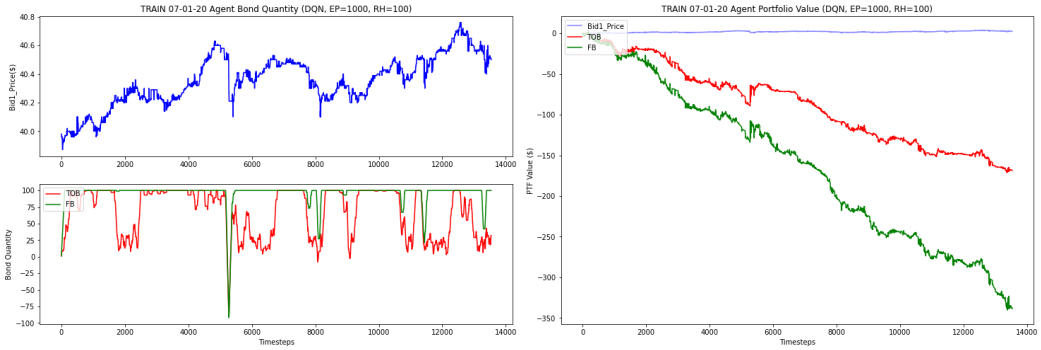



**Figura 35:** Predicciones sobre datos de entrenamiento - Resultados negativos - Día 07/01/20 - Agente DQN

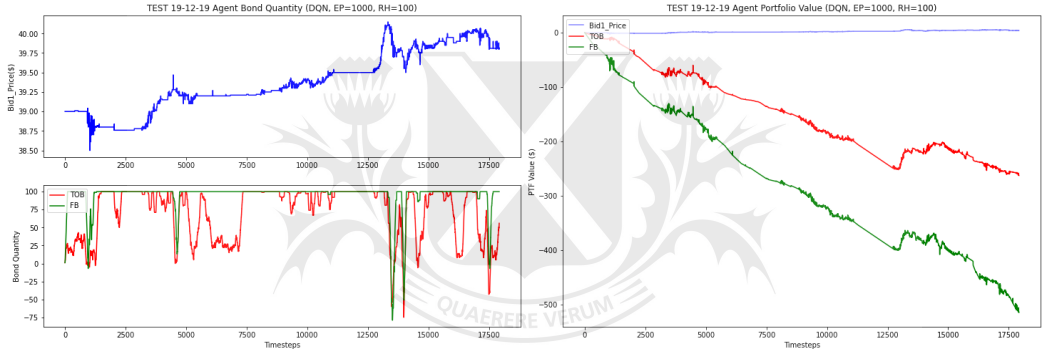



**Figura 36:** Predicciones sobre datos de prueba - Resultados negativos - Día 19/12/19 - Agente DQN

y no consiguió adecuar su política a las series de precios de los demás días de entrenamiento. Esto puede haberse dado porque el primer *set* de datos que recorrió fue el de ese día, actualizando su política bruscamente al principio y no permitiendo el margen de ajuste suficiente para corregirla y adecuarse a la característica de las otras series de entrenamiento. De todas formas, los parámetros internos del modelo evidencian que las recompensas finales de los episodios de entrenamiento son consistentemente positivas (Fig. 37).

Para comprobar el indicio del experimento precedente, se procede a entrenar sobre un día solo y verificar qué condiciones tienen que darse para lograr buenos resultados. Se llevan a cabo experimentos con diferentes configuraciones, alternando patrones de recompensa y días rotativos de entrenamiento, obteniendo resultados positivos en la predicción sobre los mismos datos de entrenamiento en un muy escaso porcentaje de los intentos.

Por ese motivo se decide reducir la longitud de las series de entrenamien-

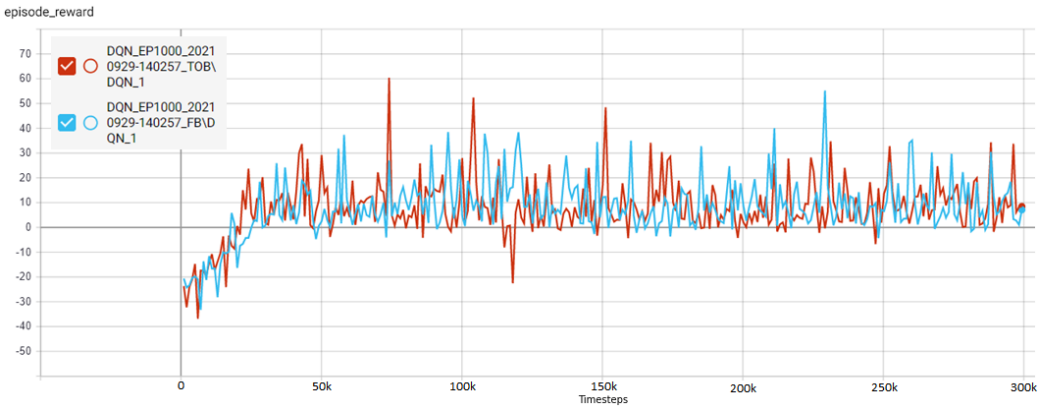



**Figura 37:** Recompensas por episodio de entrenamiento - Parámetro interno del modelo DQN

to a aproximadamente un 5 % de su longitud original, quedándose sólo con una porción del día, buscando que el modelo entrene aún más veces sobre los mismos datos y permita descartar problemas de modelos subentrenados, situación latente en todos los experimentos que altera la correcta evaluación del rendimiento de los agentes. Los primeros resultados obtenidos con esta modificación son alentadores: los agentes logran predecir adecuadamente para la serie de datos de entrenamiento reducida (Fig. 38), aunque no así para la serie de prueba completa (Fig. 39).

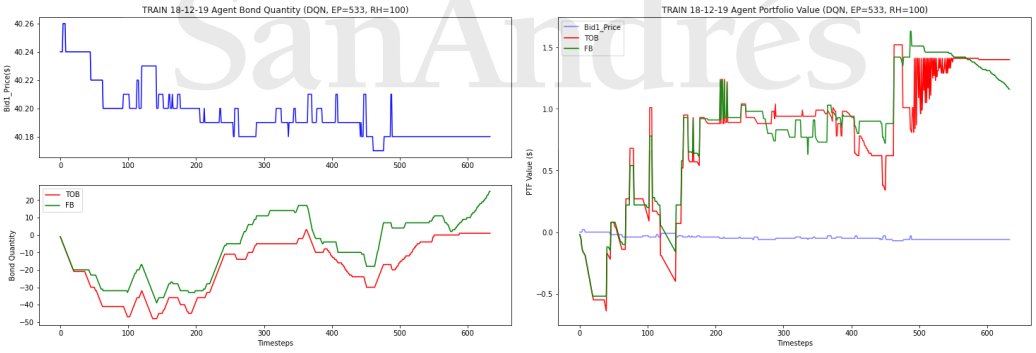



**Figura 38:** Predicciones sobre serie de datos de entrenamiento reducida al 5 % - Agente DQN

Se incrementa la longitud de la serie de entrenamiento a un 20 % para verificar si la capacidad de predecir sobre esa misma serie se mantiene, y se comprueba que esto resulta afirmativo en la mayoría de los casos donde se entrena durante un tiempo suficiente para lograr estos resultados (Fig.

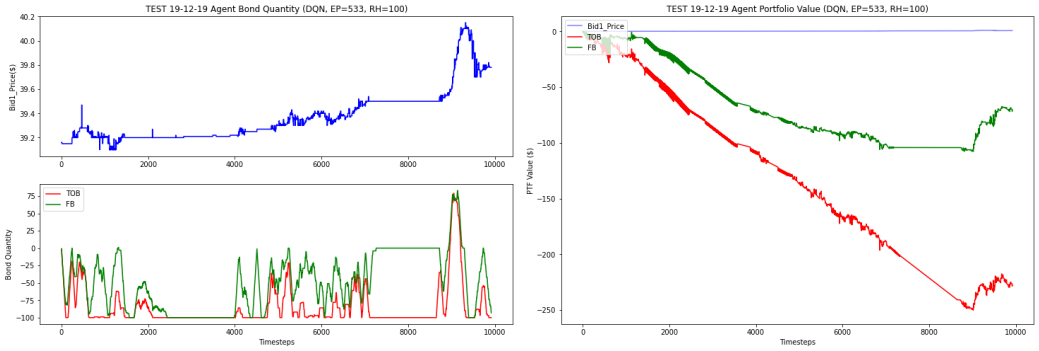



**Figura 39:** Predicciones sobre serie de datos de prueba completa - Agente DQN

40). Para los datos de prueba, la predicción sigue entregando pérdidas. Adicionalmente, a esta altura del proceso de encontrar un modelo que permita comparar el rendimiento del agente TOB contra el del agente FB, se empieza a evidenciar que es en general el FB el que consigue imponerse al TOB, aunque por escaso margen y en condiciones de ganancia tanto como de pérdida.

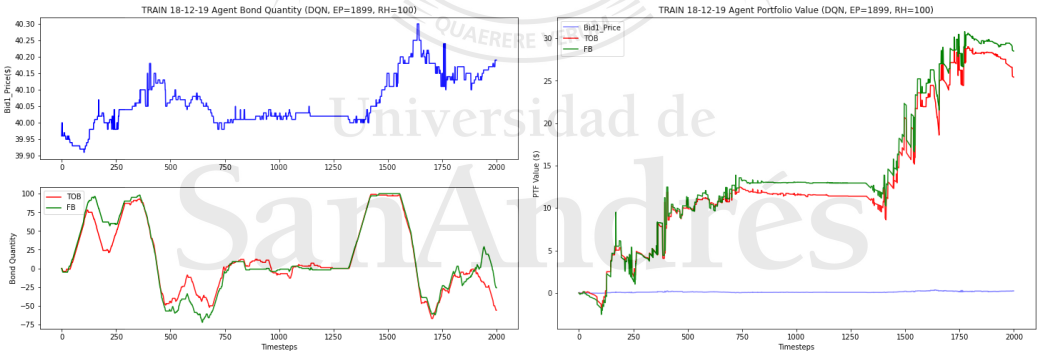



**Figura 40:** Predicciones sobre serie de datos de entrenamiento reducida al 20 % - Agente DQN

Con el objetivo de incorporar al modelo final todos los días de entrenamiento disponibles, y debido al fenómeno descripto anteriormente acerca de la preponderancia en el ajuste de la política de los datos de entrenamiento del primer día sobre el que se entrena, se decide unificar todas las series de datos de entrenamiento con longitud reducida en un único bloque con el fin de evitar que los primeros datos sobre los que se entrena definan prematuramente la dirección que tome la política de los agentes. Esto se basa en el modo de actualización de los pesos de las redes que dictan la política del

modelo, los cuales son ajustados luego de cada episodio de entrenamiento. De esta manera, haciendo al agente experimentar la totalidad de los datos de entrenamiento antes de producir un ajuste de la política, se elimina el sesgo de favorecer a un día particular de entrenamiento. Para esto se combinan en una única serie de precios todos los días de entrenamiento, tomando especial cuidado en no tomar posición en el bono tal que la posición no pueda ser liquidada dentro del mismo día y en el plazo dictado por el horizonte de inversión. Para lograr esto se impide al agente operar cuando quedan menos *ticks* que el horizonte de recompensa definido hasta el final del día calendario. Adicionalmente es necesario redefinir a la longitud del episodio de entrenamiento tal que se recorra al conjunto de datos entero antes de realizar ajustes en la política.

Los resultados de esta aproximación no son positivos para los días de entrenamiento reducidos (Figs. 41, 42, 43, 44), pero al sumar dos series de datos *dummy*, monótonas creciente y decreciente, respectivamente, se observa que para éstos patrones los agentes sí reconocen la tendencia y ajustan sus predicciones para generar ganancias (Figs. 45, 46). El patrón de tenencia sobre la serie monótonamente creciente evidencia un retardo en tomar una posición en el bono que reporte ganancia, lo que sugiere que un posible problema de los agentes sería reaccionar a tiempo a los cambios de tendencia en los precios para ajustar su posición. Esto se debe en parte a la característica intrínseca del modelo que permite a los agentes operar sólo una unidad de bono por unidad de tiempo, pero es sólo uno de los posibles motivos de este retardo.

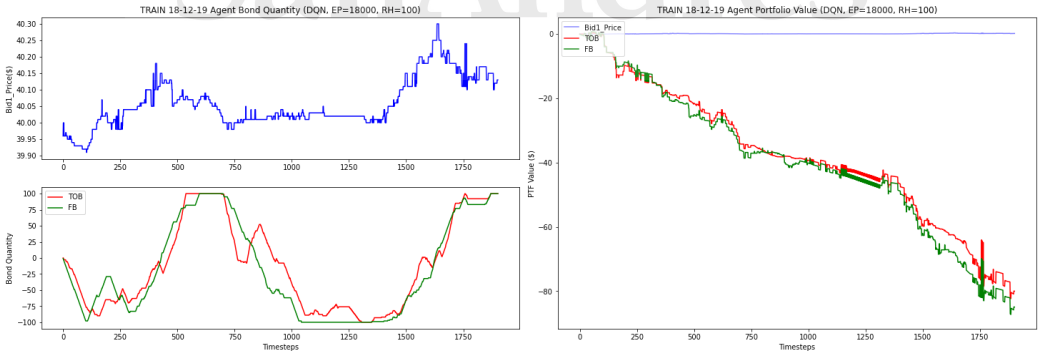



**Figura 41:** Predicciones sobre datos de entrenamiento - Día 18/12/19 - Agente DQN

En una línea de investigación alternativa se busca verificar cómo afecta al modelo el tiempo de entrenamiento del que se dota a los agentes. Se utiliza

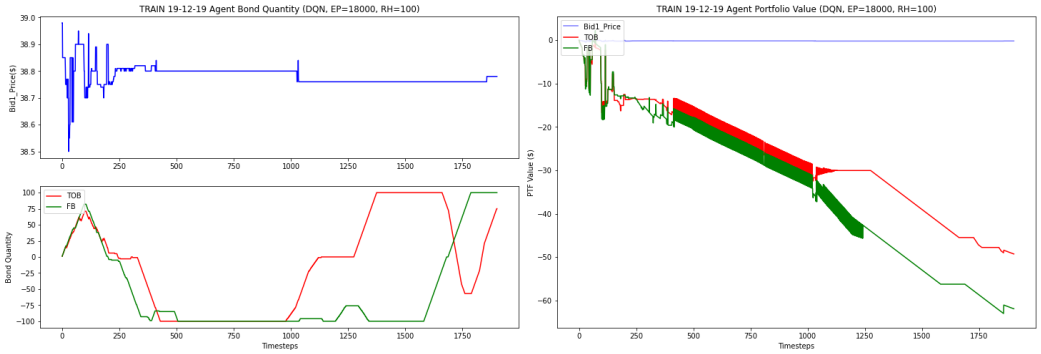



**Figura 42:** Predicciones sobre datos de entrenamiento - Día 19/12/19 - Agente DQN

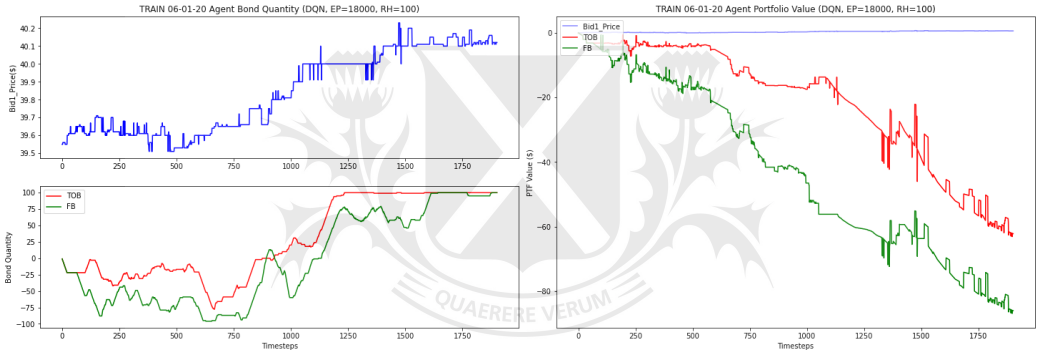



**Figura 43:** Predicciones sobre datos de entrenamiento - Día 06/01/20 - Agente DQN

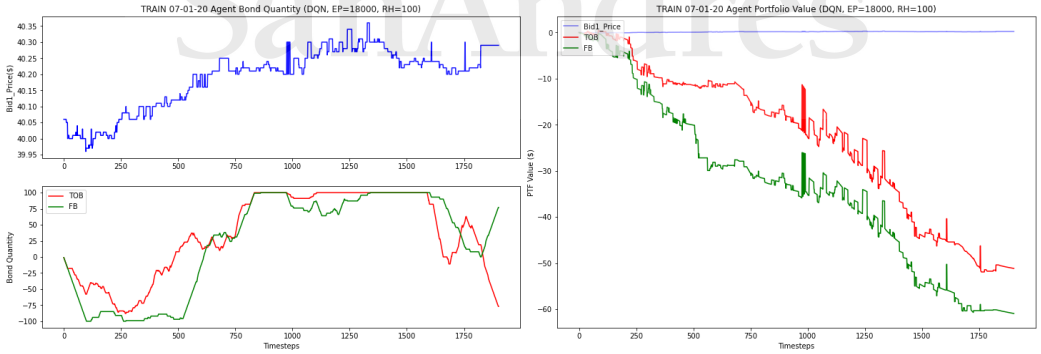



**Figura 44:** Predicciones sobre datos de entrenamiento - Día 07/01/20 - Agente DQN

un sólo día de entrenamiento y un día de prueba, y se registra el valor del portafolio al final del día tanto para el agente TOB como para el FB. Los

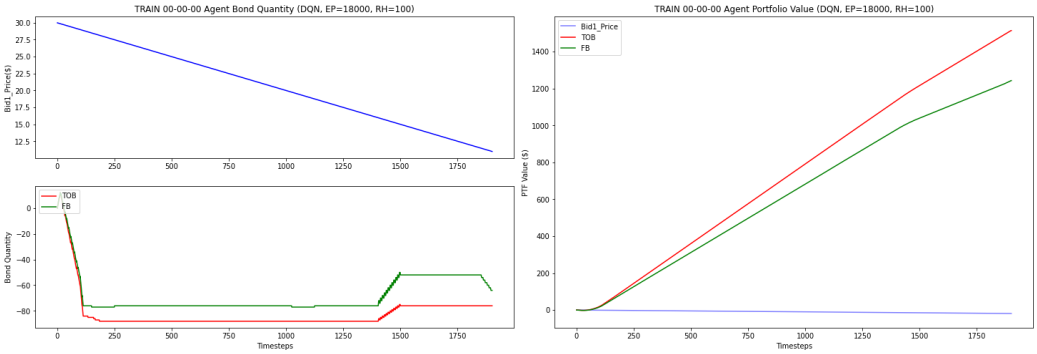



**Figura 45:** Predicciones sobre datos de entrenamiento - Serie *dummy* monótonamente decreciente - Agente DQN

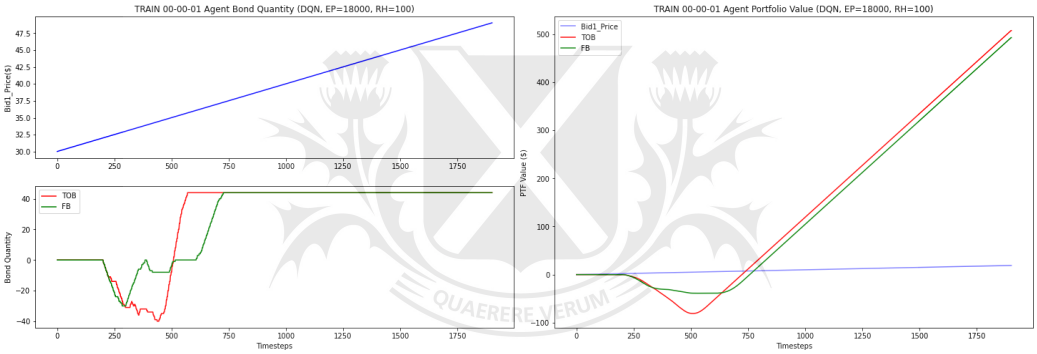



**Figura 46:** Predicciones sobre datos de entrenamiento - Serie *dummy* monótonamente creciente - Agente DQN

resultados de este nuevo enfoque (Fig. 47) exhiben un rápido aprendizaje de los modelos y una subsecuente estabilidad alrededor de cierta política, lo cual es esperable, y permiten comparar con más claridad el único resultado relevante de la actividad de los *bots*, que es en definitiva qué valor de portafolio alcanzan al final de la rueda de negociación, independientemente de las posiciones intermedias. Se evidencia que para el día de entrenamiento ambos agentes (TOB y FB) consiguen, a pesar de las oscilaciones en su rendimiento en función del tiempo de entrenamiento, una ganancia mayor a cero en varias situaciones. Como ensayo de comparación se invierten los días de entrenamiento y de prueba, y se observa (Fig. 48) que en este caso, la conclusión es diferente: los agentes no consiguen resultados significativamente superiores sobre los datos de entrenamiento que sobre los de prueba. Este fenómeno parece confirmar una sospecha que está presente desde el inicio de los experimentos acerca de que algunas series son más fáciles de explotar que otras, dejando de lado las triviales series *dummy*. Lo esperable sería que los

agentes puedan predecir mejor sobre las series de entrenamiento y obtener resultados superiores, independientemente de los datos que se utilicen como entrenamiento y prueba.

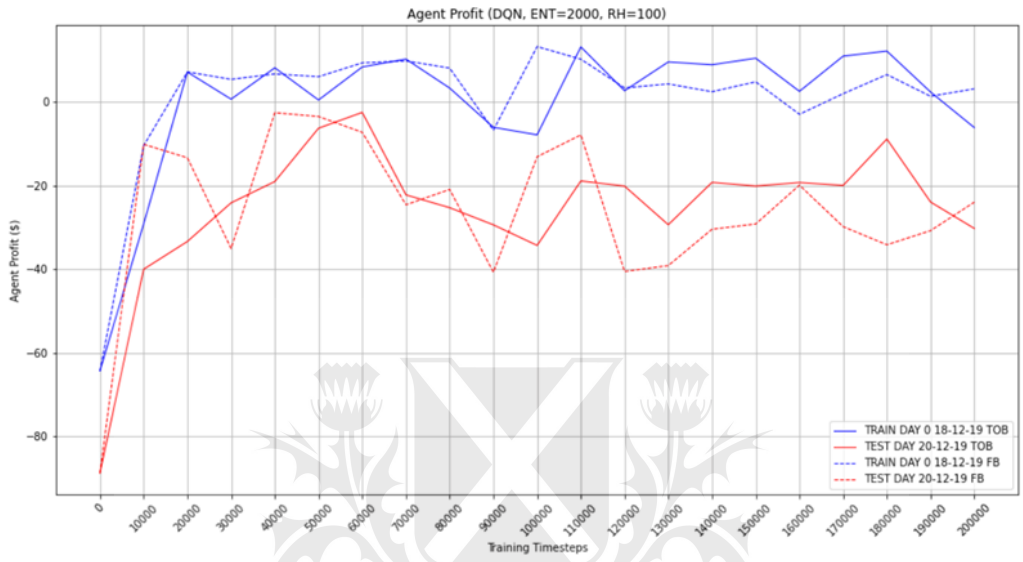



**Figura 47:** Valor de portafolio final en función de la cantidad de *timesteps* de entrenamiento - Entrenamiento 18/12/19 y prueba 20/12/19 - Agente DQN

Para tener una idea de cómo se comportarían los agentes incorporando las demás series de datos al entrenamiento, se repite el experimento anterior con las series reducidas de 2000 a 500 muestras cada una, y se observa una naturaleza un tanto confusa en los resultados (Fig. 49). Curiosamente los agentes lograron predecir mejor sobre la serie de prueba, representados estos resultados por las líneas amarillas, que sobre varias series de entrenamiento, lo que sugiere que el modelo no está siendo capaz de reconocer las acciones más beneficiosas sobre estas últimas. Los resultados de las predicciones sobre la serie de entrenamiento del día 18/12/19 siguen generando resultados relativamente buenos, mientras que para las otras tres series de entrenamiento, además de ser más inestable, el rendimiento es malo, no pudiendo apreciarse en ningún caso una diferencia notoria entre ambos agentes TOB y FB.

## 7.6. Otros experimentos

Varios ensayos fueron realizados en conjunto con los descriptos en las secciones anteriores que por no dar resultados positivos fueron descartados eventualmente.

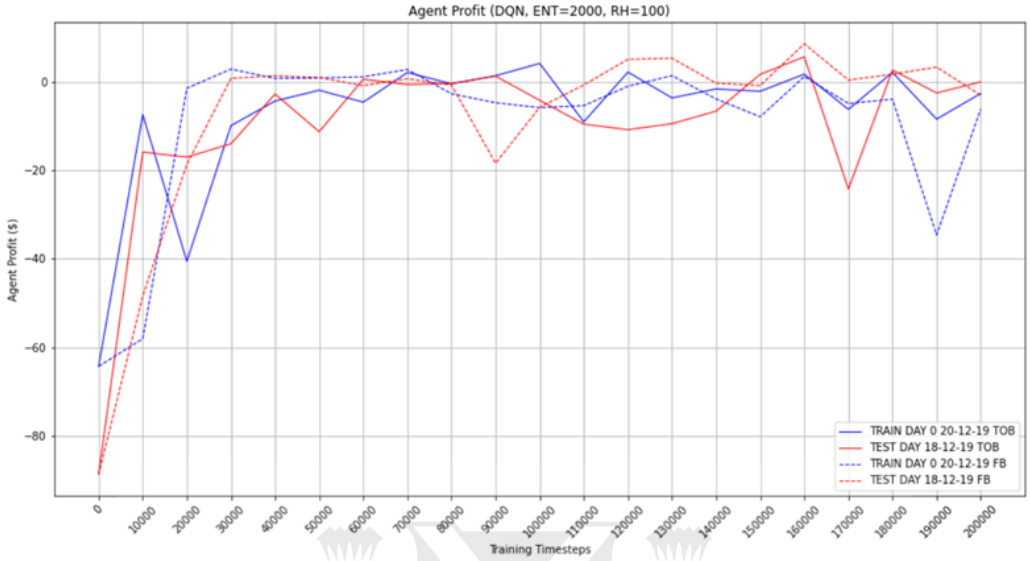



**Figura 48:** Valor de portafolio final en función de la cantidad de *timesteps* de entrenamiento - Entrenamiento 20/12/19 y prueba 18/12/19 - Agente DQN

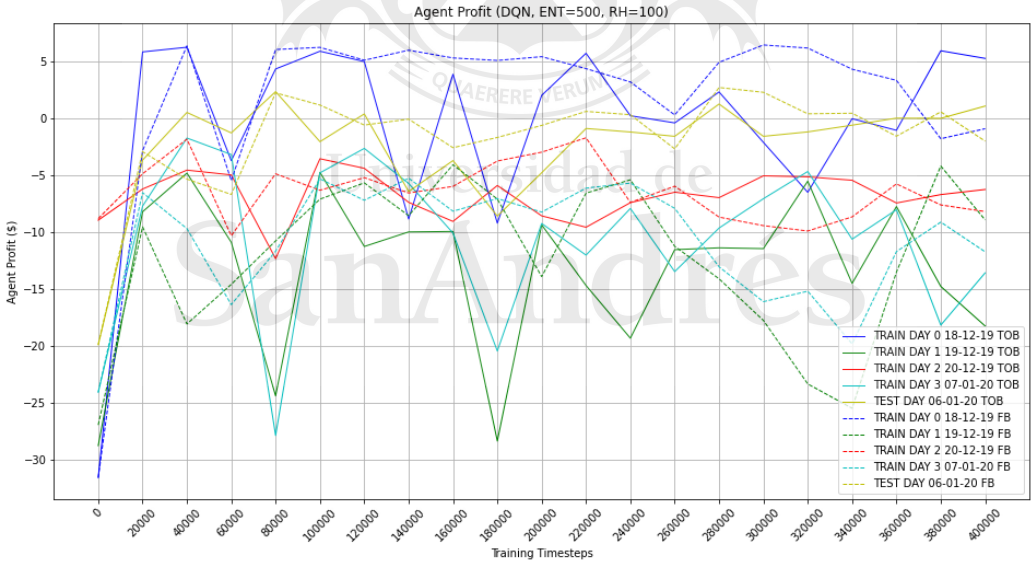



**Figura 49:** Valor de portafolio final en función de la cantidad de *timesteps* de entrenamiento - Todos los días - Agente DQN

En primera instancia se intentó proveer como característica adicional al modelo al precio del bono en un instante futuro, para que pueda inferir en base a este dato si le conviene o no operarlo en el presente. Esto lógicamente hace al modelo muy dependiente de esta única *feature*, ya que le garantiza "a

priori" conseguir resultados positivos. El problema que genera esta modificación es que al momento de predecir sobre datos de prueba, esta variable no va a estar disponible, por lo que habría que generar algún tipo de predicción sobre este precio futuro, lo que convertiría al problema en uno de distinta naturaleza, y escapa al objetivo del presente trabajo.

En algunos experimentos, incluyendo esta característica adicional en el modelo, se observaron comportamientos extraños que no se notaron con otras configuraciones del modelo, como por ejemplo la tenencia de bonos alrededor de una media distinta de cero (Fig. 50), lo que resulta peculiar dado que esa posición en bonos no garantiza una ganancia debido a una subida de precios futura. También se intentó simplificar esta característica adicional proveyendo al agente simplemente con la diferencia de precios entre los *bid* y *ask* actuales y futuros junto con los spreads, para que obtenga una característica aún más evidente de cuándo le conviene comprar y cuándo vender, pero todos estos intentos no justificaron en cuanto a resultados la adopción de estas nuevas *features*, por lo que también se decidió desestimarlas.

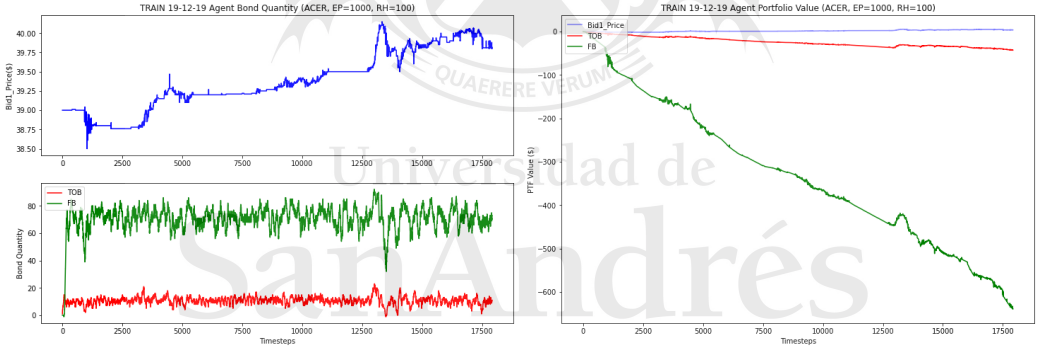



**Figura 50:** Tenencia promedio distinta de cero - TOB vs FB - Agente DQN

Atentos a la imposibilidad de generar ganancias en un corto plazo debido a que para que el precio de *bid* futuro supere al de *ask* pagado en una eventual compra había que esperar varios *ticks*, se intentó realizar una implementación desde un punto de vista del tratamiento de señales, que consistió en submuestrear las series de datos para obtener cambios de mayor frecuencia, mediante un parámetro que determine la relación de muestreo. Esto significa tomar muestras equiespaciadas de la serie de precios para entrenar sobre ellas, pudiendo en la práctica recorrer una mayor amplitud temporal con la misma cantidad de puntos de la serie, aunque con la consecuente pérdida de información intermedia. Además se fijó en 1 el horizonte de recompensa, lo que equivaldría a permitirle al agente operar sólo cada X *ticks* en la serie

original, donde X es justamente el valor de este horizonte. Los resultados de los experimentos con esta modificación arrojaron resultados no concluyentes con respecto al comportamiento de los agentes, por lo que fue desestimada.

Durante gran parte de la cronología de experimentos se probó incluyendo como características del modelo a la tenencia instantánea de bonos y de efectivo, así como el valor del portafolio, por si este conjunto de información extra como parte del estado del universo generaba algún comportamiento diferenciador en las decisiones de los agentes al momento de predecir sobre alguna serie de precios. Los resultados indican que en general esta inclusión es detrimental al rendimiento del modelo, principalmente porque al añadir más características se incrementa notablemente el tiempo requerido para que la política alcance una convergencia aceptable.

Un último intento de cambiar el paradigma del modelo se produjo incorporando al modelo series de precios del activo AY24C, *ticker* que permite comercializar el mismo bono AY24D pero en el mercado estadounidense, dando lugar a posibles arbitrajes entre los precios del bono cotizante en diferentes mercados. Si bien el agente es capaz de comprender que los precios del bono bajo el *ticker* C serían siempre menores a los del *ticker* D, y a partir de esto generar ganancias, esto no representaba el espíritu del arbitraje ya que no alcanza con observar los precios de ambos activos sino que además es necesario ponderar el costo de poseer el efectivo en distintos mercados financieros.



## 7.7. Ajuste del modelo final

Para determinar los parámetros del modelo definitivo para obtener y exponer los resultados del presente trabajo, se establece una estructura de 3 días de entrenamiento, un día de validación y un día de prueba. En este caso se respeta la cronología de los días y se utilizan las series de libros de órdenes de los días 18, 19 y 20 de diciembre de 2019 como *sets* de entrenamiento, la serie del día 6 de enero de 2020 como *set* de validación y la del 7 de enero como *set* de prueba.

Partiendo de esta estructura, se realizan experimentos variando los parámetros disponibles del modelo haciendo un barrido sobre los más críticos, como el horizonte de inversión, los multiplicadores de los indicadores, o la simplificación de características y se registran los resultados del valor del

portafolio al final del día de negociación, tanto para los conjuntos de entrenamiento como para el de validación. Hasta esta instancia no se hará uso del conjunto de datos de prueba, reservándolo para la sección de resultados.

La siguiente combinación de parámetros, detallados en la sección 6.6, produce los resultados más estables y con una intervención del agente moderada sobre los *sets* de entrenamiento y de validación, por lo que será la utilizada para generar los resultados finales del presente trabajo:

- Multiplicador EMA de los indicadores (*"Indicators EMA span multiplier"*, "$\mu$"): 10. Este valor asegura un equilibrio entre tener una señal relativamente poco ruidosa y conseguir suficientes situaciones de cruce entre las señales $MACD_\mu$ y $MACD_\mu Signal$.
- Frontera RSI (*"RSI threshold"*): 40. Este valor genera una cantidad equilibrada de situaciones donde la característica simplificada adopta valores distintos de cero.
- Horizonte de recompensa (*"Reward Horizon"*): 100. Un número menor a este genera poca intervención de los agentes en el mercado ya que no se le da tiempo a la serie de precios a generar una situación donde se pueda generar una ganancia, y uno mayor trae el problema de poder negociar muy pocas veces dentro de una determinada rueda.
- Tamaño de la serie de datos de entrada diaria (*"Entries Retained"*): 1000. Se reduce a un número más manejable el tamaño de las series diarias, para permitir al modelo entrenar una suficiente cantidad de veces sobre cada *tick* del libro de órdenes.
- Cantidad de entradas removidas (*"Entries Removed"*): 4000. Se descartan estas muestras del inicio y del final de las series para descartar efectos no deseados de inicio o de final de día.
- *Bid-Ask spread*: Activado. Se utiliza esta característica para alimentar al modelo, y no se le proveen los precios de las puntas de *bid* y *ask*.
- Simplificación de indicadores técnicos (*"Simplified Indicators"*): Activado. Al reducir en uno el número de características que recibe el modelo, se simplifica el entrenamiento.
- Desequilibrio de mercado (*"Market Imbalance"*): Activado. Se genera una única *feature* para el agente FB condensando toda la información del libro de órdenes de nivel 2.

- Discretizar (*"Discretize"*): Desactivado. Esta transformación sobre algunas características no ha probado ser efectiva.
- Longitud del episodio de entrenamiento (*"Episode Length"*): 3000. Esto representa la suma de los 3 días de longitud 1000 sobre los que se entrena al modelo, tal que recorra todo el *set* de entrenamiento antes de terminar el episodio.
- Límites de posición (*"Position Limits"*): Desactivado. Se ha logrado mantener acotada la posición en bonos de los agentes mediante otras estrategias, como la de liquidar la posición en un horizonte determinado.
- Penalización por no operar (*"Trading Penalty"*): Desactivado.
- Potenciación de las recompensas positivas (*"Positive Reward Booster"*): Desactivado.
- Recompensa inmediata (*"Reward Immediate"*): Activado.
- Recompensa al final del episodio (*"Reward End of Episode"*): Desactivado.
- Penalización por negociar en alta frecuencia (*"High Freq Penalty"*): Desactivado.
- Modelo: DQN. La utilización de los modelos PPO2 y ACER no ha probado ser efectiva en este problema. El modelo PPO2 suele entrar en la estrategia de no operar en la gran mayoría de los casos, independientemente de la configuración del ambiente utilizada, y el ACER sufre el inconveniente de que la política que construye cae frecuentemente en estados espurios desde los cuales no es capaz de generar resultados estables, haciendo dificultosa su evaluación. Por su parte, DQN ha demostrado que tiene una alta eficiencia para definir su política incluso con una cantidad relativamente baja de *timesteps* de entrenamiento, a pesar de no conseguir resultados consistentemente positivos.

A partir de estos parámetros, se busca encontrar la cantidad de tiempo óptima a entrenar para maximizar el rendimiento sobre el set de validación. A esta estrategia se la conoce en el dominio de *Machine Learning* como *"Early Stopping"* y busca no sobreentrenar al modelo, evitando que las decisiones en la predicción se adecúen en demasía al conjunto de datos de entrenamiento. En la figura 51 se observa que con una cantidad de *timesteps* igual a 60.000, se

logra el mejor rendimiento promedio sobre el conjunto de datos de validación, por lo que se utilizará como parámetro en la generación de resultados sobre el conjunto de datos de prueba.


**Figura 51:** Rendimiento de los agentes sobre el *set* de validación en función de la cantidad de *timesteps* de entrenamiento



# 8. Resultados

La predicción de los agentes *top of book* -TOB- y *full book* -FB-, entrenados sobre el conjunto de datos de prueba se mide por el saldo de efectivo al finalizar la sesión, ya que liquidan las posiciones en bonos en tiempo y forma. Se realizan varias sesiones de entrenamiento independientes con distintas semillas (*seed*) del modelo, para reducir la aleatoriedad en los pesos iniciales de las redes de los algoritmos, observándose los resultados en la figura 52.

El promedio de rendimiento del agente FB es de $ -3.65 con un desvío estándar de $7.86, mientras que el agente TOB tiene una media de $ -3.51 con un desvío de $8.15, resultando negativo el saldo final que logra cada uno de los agentes a través de sus predicciones al final del día de prueba, independientemente de la semilla del modelo. Debido a la oscilación en los resultados puntuales de los modelos para diferentes *seeds*, se evidencia que no existe una diferencia significativa en el rendimiento de ambos agentes de


**Figura 52:** Rendimiento de los agentes sobre datos de prueba en función del parámetro *seed* de los modelos DQN de entrenamiento

aprendizaje reforzado, entrenados bajo el modelo establecido en este trabajo.

# 9. Conclusiones

Debido a la inexistencia de diferencias estadísticamente significativas entre los resultados obtenidos por el agente que contaba con la información del libro de órdenes entero (*Full Book* o FB) con respecto al agente que sólo recibía las órdenes más competitivas (*Top of Book* o TOB), se concluye que no se puede probar que contar con información de libro de órdenes de nivel 2 mejore los resultados de un agente de aprendizaje reforzado, en el marco de la configuración de modelos utilizada y las características alimentadas a éstos.

Con respecto a la utilización de modelos de aprendizaje reforzado para este tipo de experimentos, la naturaleza de esta técnica de inteligencia artificial se basa en explotar la repetición del estado del universo en el que se encuentra para tomar mejores decisiones. En el caso de las series temporales, esta ventaja se ve disminuida ante la imposibilidad de realizar predicciones precisas sobre datos futuros que estén estadísticamente relacionados con los del pasado, cercenando la capacidad de reconocer estados puntuales que in-

diquen una tendencia al alza o a la baja de los precios del bono AY24D utilizando como *input* al sistema únicamente al libro de órdenes.

Al realizar los experimentos descriptos en la sección 7 se presentaron dificultades de distinta índole que obligaron a implementar o desestimar ciertas estrategias del ambiente de entrenamiento para lograr mejores resultados al momento de tener que realizar predicciones. La refinación de este ambiente, que incluye fundamentalmente la entrega de recompensas a los agentes, la estructura de los conjuntos de datos de entrenamiento y reglas algorítmicas puntuales para evitar comportamientos extraños al momento de las predicciones, fue un proceso iterativo que dio forma al modelo final utilizado. Aún así, éste no está exento de tener que lidiar con series temporales fundamentalmente ruidosas y de complejo análisis desde el punto de vista estadístico, lo que explica los resultados obtenidos.

# 10. Nuevas líneas de investigación

La ventaja de las técnicas de aprendizaje reforzado se centra en reconocer estados ya experimentados y construir una política de acciones que maximice las recompensas sobre éstos, lo que podría ser un atributo que dote de capacidades de generar ganancias a los agentes en el dominio de la creación de mercado, ya que conociendo la composición de un libro de órdenes en un instante puntual del tiempo y junto con otros indicadores relevantes, se podría tomar una decisión sobre la colocación de puntas de compra y venta y potencialmente construir una política que funcione consistentemente. Para llevar adelante esta idea habría que analizar cómo un agente de aprendizaje reforzado puede intervenir en el mercado para colocar estas puntas; redefinir completamente los ambientes de entrenamiento y de prueba y las acciones que se pueden realizar en ellos y cómo impactan a éste; y finalmente establecer nuevas métricas para evaluar el rendimiento, comparando con otros métodos más tradicionales utilizados en la creación de mercado a modo de *benchmark*. Una posibilidad de continuar por esta línea de investigación sería partiendo de (Selser et al., 2021), donde se describe una estrategia de creación de mercado utilizando precios sintéticos, pero utilizando los libros de órdenes con datos reales, como los tratados en el presente trabajo.

Otra línea de investigación posible es utilizar otros activos potencialmente correlacionados entre sí, que pueden ser las especies AY24 o AY24C ya que comparten activo subyacente con el AY24D, como así también otros bonos argentinos o activos de otra clase. Mediante la incorporación de nuevas series

temporales de libros de órdenes se estaría dotando a los agentes de un mayor volumen de datos para intentar tomar decisiones de negociación, habilitando la posibilidad de realizar arbitrajes estadísticos entre las distintas especies.

En el caso de optar por mantener la pregunta de la presente tesis e intentar encontrar si el libro de nivel 2 aporta información adicional relevante, es una posibilidad utilizar datos de menor frecuencia, como datos de libros de órdenes en un único momento de una rueda de negociación diaria. La dificultad que presenta esta aproximación al problema es la disponibilidad de datos históricos de estas características. Otra posibilidad es aplicar diferentes técnicas de inteligencia artificial a las series de precios de este trabajo, particularmente de aprendizaje no supervisado, en función del estado del arte en este campo del conocimiento.

# Referencias


Yue Deng, Youyong Kong, Feng Bao, and Qionghai Dai. Sparse coding-inspired optimal trading system for hft industry. *IEEE Transactions on Industrial Informatics*, 11(2):467–475, 2015.

Yue Deng, Feng Bao, Youyong Kong, Zhiquan Ren, and Qionghai Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3): 653–664, 2016.

Jae Won Lee. Stock price prediction using reinforcement learning. In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)*, volume 1, pages 690–695. IEEE, 2001.

Jae Won Lee and O Jangmin. A multi-agent q-learning framework for optimizing stock trading systems. In *International Conference on Database and Expert Systems Applications*, pages 153–162. Springer, 2002.

John Moody, Lizhong Wu, Yuansong Liao, and Matthew Saffell. Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, 17(5-6):441–470, 1998.

Yagna Patel. Optimizing market making using multi-agent reinforcement learning. *arXiv preprint arXiv:1812.10252*, 2018.

Matias Selser, Javier Kreiner, and Manuel Maurette. Optimal market making by reinforcement learning. *arXiv preprint arXiv:2104.04036*, 2021.

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard S. Sutton, Andrew G. Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

Hongyang Yang, Xiao-Yang Liu, Shan Zhong, and Anwar Walid. Deep reinforcement learning for automated stock trading: An ensemble strategy. *Available at SSRN*, 2020.

# A. Procesamiento de mensajes crudos del protocolo FIX

En este apéndice se muestran ejemplos del formato de los mensajes del protocolo FIX que se utiliza como dato de entrada y se provee del código fuente del programa para procesarlos.

Existen dos tipos de mensajes básicos de este protocolo para construir el libro de órdenes, los que se denominan *full-refresh* e *incremental-refresh*.

El primero provee un pantallazo del libro de órdenes entero a ese momento; un ejemplo se muestra a continuación:

```
"2019/12/18 15:13:46.103164 8=FIXT.1.1 9=697 35=W 34=29 49=STUN"
"52=20191218-15:13:47.005 56=XXXX 115=FGW 48=AY24D-0003-C-CT-USD"
"55=AY24D 106=0500-R 167=GO 207=XMEV 262=-LwOZbED_R-E3dVsfS 1021=2"
"268=14 269=0 270=39.95 271=47022 346=1 290=1 63=3 269=0 270=39.94"
"271=222464 346=1 290=2 63=3 269=0 270=39.92 271=42000 346=2 290=3"
"63=3 269=0 270=39.9 271=69531 346=4 290=4 63=3 269=0 270=39.86"
"271=501 346=1 290=5 63=3 269=1 270=39.96 271=2031 346=1 290=1 63=3"
"269=1 270=39.98 271=207 346=1 290=2 63=3 269=1 270=39.99 271=15409"
"346=3 290=3 63=3 269=1 270=40 271=799865 346=16 290=4 63=3 269=1"
"270=40.01 271=996 346=1 290=5 63=3 269=2 270=39.96 271=200"
"273=15:13:46 288=- 289=- 63=3 269=4 270=37.73 63=3 269=8 270=37.5"
"63=3 269=7 270=40.2 63=3 10=135"
```

Aquí se puede notar la longitud del mensaje, relativamente extensa en comparación al segundo tipo de mensajes, los de refresco del libro, que se muestra a continuación:

```
"2019/12/18 15:13:52.297170 8=FIXT.1.1 9=214 35=X 34=159 49=STUN"
"52=20191218-15:13:53.203 56=XXXX 115=FGW 262=HUB083_MD_1576670713142"
"1021=2 268=1 279=0 269=0 55=AY24D 48=AY24D-0003-C-CT-USD 167=GO"
"207=XMEV 106=0500-R 270=39.95 271=43821 346=1 290=1 63=3 10=141"
```

Para analizar estos mensajes se desarrolla un módulo que utiliza el paquete de *Python* llamado *Simplefix*. Se procesan los registros del protocolo FIX con los libros de órdenes y se convierten a una estructura de datos legible por los agentes, en forma de tabla con precios y cantidades del libro de órdenes. Una muestra de los datos de salida, guardados en formato *.csv*, se exhibe a continuación:

| B1P | B2P | B3P | B4P | B5P | A1P | A2P | A3P | A4P | A5P |
|---|---|---|---|---|---|---|---|---|---|
| 39.95 | 39.94 | 39.92 | 39.90 | 39.86 | 39.96 | 39.98 | 39.99 | 40.00 | 40.01 |
| 39.95 | 39.94 | 39.92 | 39.90 | 39.86 | 39.96 | 39.98 | 39.99 | 40.00 | 40.01 |
| 39.94 | 39.92 | 39.90 | 39.86 | 39.85 | 39.96 | 39.98 | 39.99 | 40.00 | 40.01 |

Aquí B1P se refiere a *Bid 1 Price* y representa el precio de la oferta de compra más competitiva, mientras que A5P se refiere a *Ask 5 Price* y representa el precio de venta menos competitivo del libro.

Las siguientes 10 columnas acompañan a estos datos, mostrando las cantidades de estas ofertas de compra y venta:

| B1S | B2S | B3S | B4S | B5S | A1S | A2S | A3S | A4S | A5S |
|---|---|---|---|---|---|---|---|---|---|
| 47022 | 222464 | 42000 | 69531 | 501 | 2031 | 207 | 15409 | 799865 | 996 |
| 44422 | 222464 | 42000 | 69531 | 501 | 2031 | 207 | 15409 | 799865 | 996 |
| 222464 | 42000 | 69531 | 501 | 60354 | 2031 | 207 | 15409 | 799865 | 996 |

Aquí B1S se refiere a *Bid 1 Size* y representa la cantidad de la oferta de compra más competitiva, mientras que A5S se refiere a *Ask 5 Size* y representa la cantidad de la oferta de venta menos competitiva del libro.



# B. Módulo de entrenamiento de los agentes de aprendizaje reforzado

En este módulo del programa se entrena y se prueba el modelo sobre los datos procesados en el módulo precedente. En una primera etapa se importan los datos de entrada y los paquetes necesarios. El paso siguiente es definir el ambiente en el que se entrenará al agente, con los hiperparámetros propios de este ambiente, como es la longitud del episodio de entrenamiento. Luego se define el modelo a utilizar junto con su política y se crean las instancias de *logging* para analizar el modelo elegido. Una vez que el modelo está creado se lo entrena con el conjunto de datos de entrenamiento, donde se define un nuevo hiperparámetro, que definirá los *timesteps* totales a entrenar. Finalmente, una vez que el modelo ya culminó su entrenamiento, se busca medir su rendimiento sobre el set de datos de prueba.

Para la creación y entrenamiento de los agentes se utilizó el paquete de *Python* denominado *Stable-Baselines v2*, el cual está basado en los paquetes de código abierto *Tensorflow* y *OpenAI Gym*. El primero brinda herramientas

para la construcción de sistemas de inteligencia artificial en general, mientras que el segundo está específicamente orientado al entrenamiento de agentes de aprendizaje reforzado.

El proceso iterativo de ajuste del modelo incluyó en un inicio la definición de un ambiente con ciertas características y la utilización de 3 algoritmos de aprendizaje reforzado: DQN, PPO2 y ACER. Ante la imposibilidad de obtener resultados deseados con todos ellos, se fue modificando el ambiente de entrenamiento incorporando, quitando o modificando hiperparámetros. La cronología de los experimentos, ampliando lo detallado en la sección 7, se plasma en el apéndice D.

# C. Código fuente de los programas

El repositorio con el código en el lenguaje de programación Python del proyecto completo, junto con un archivo de Excel con el listado de todos los experimentos y un archivo .zip que contiene todos los gráficos recogidos de los experimentos relevantes se encuentra en el siguiente link: https://github.com/Mansurte/Tesis-UdeSA-Quant-Reinforcement-Learning

## C.1. Procesador de mensajes FIX

```
# Alonso, Ariel - MFE Thesis - FIX Books processing Module

import pandas as pd
import simplefix as sf
import time
import numpy as np
import matplotlib.pyplot as plt
import talib

## Defines

''' FIX FIELDS CONSTANTS '''

REFRESH_TYPE = '35'
FULL_REFRESH = b'W'
INCREMENTAL_REFRESH = b'X'

TICKER = '55'
```

```
AY24D = b'AY24D'
AY24C = b'AY24C'
AY24 = b'AY24'

PRICE = '270'
SIZE = '271'

NUM_MESSAGES = '268'

MESSAGE_TYPE = '269'
BID = b'0'
OFFER = b'1'

ACTION = '279'
NEW = b'0'
CHANGE = b'1'
DELETE = b'2'

POSITION = '290'

# Set the path AND specific file to process
DATA_PATH = 'C:/Users/Ariel/Documents/Maestria/TESIS/FIX/Logs
    ↪  FIX/MD-2019-12-18-price-depth-T2.log'

BOOK_COLUMNS = ['Bid1',
                'Bid2',
                'Bid3',
                'Bid4',
                'Bid5',
                'Ask1',
                'Ask2',
                'Ask3',
                'Ask4',
                'Ask5',
                ]
ID_COLUMNS = ['Message_type', 'Ticker']
DF_COLUMNS = ID_COLUMNS + BOOK_COLUMNS

## FIX Data loading

raw_df = pd.read_csv(DATA_PATH,
```

```
59                      names=['date', 'time', 'raw_data'],
60                      delimiter=' ')
61
62  raw_df['Datetime'] = pd.to_datetime(raw_df['date'] + ' ' +
        ↪ raw_df['time'])
63  raw_df = raw_df.drop(['date', 'time'], axis=1)
64
65  ## FIX Messages Parsing & Processing
66
67  # Methods for processing the FIX messages
68
69  def refresh_type(message):
70      ''' Returns the type of market data refresh type Fix
            ↪ message'''
71      if message.get(REFRESH_TYPE) == FULL_REFRESH:
72          return 'FULL'
73      elif message.get(REFRESH_TYPE) == INCREMENTAL_REFRESH:
74          return 'INCREMENTAL'
75      else:
76          print('Unknown refresh message type')
77
78
79  def get_message_from_df(df):
80      ''' Returns a Fix Message object from the df raw data'''
81      parser = sf.parser.FixParser()
82      parser.append_buffer(df.raw_data)
83      return parser.get_message()
84
85
86  def parse_full_refresh_message(message):
87      '''
88      Parses full refresh market data messages and returns a
            ↪ dict
89      with actual order book data
90      '''
91      data = [message.get(REFRESH_TYPE).decode('utf-8'),
92              message.get(TICKER).decode('utf-8')]
93      empty_book_entries = 0
94      for num_entry in range(1, 6):
95          if message.get(MESSAGE_TYPE, num_entry) == BID:
96              data.append((float(message.get(PRICE, num_entry)),
```

```
97                              float(message.get(SIZE, num_entry))))
98          else:
99              empty_book_entries += 1
100             data.append((0,0)) # empty bids in the book
101     for num_entry in range(6, 11):
102         if message.get(MESSAGE_TYPE, num_entry -
                ↪  empty_book_entries) == OFFER:
103             data.append((float(message.get(PRICE, num_entry -
                    ↪  empty_book_entries)),
104                              float(message.get(SIZE, num_entry -
                                    ↪  empty_book_entries))))
105         else:
106             data.append((0,0)) # empty offers in the book
107     return dict(zip(DF_COLUMNS, data))
108
109
110 def parse_incremental_refresh_message(message):
111     '''
112     Parses incremental refresh market data messages and returns
          ↪  a dict
113     containing info about the incremental message information
114     '''
115     data_dict = {}
116     data_dict['Message_type'] =
            ↪  message.get(REFRESH_TYPE).decode('utf-8')
117     data_dict['Ticker'] = message.get(TICKER).decode('utf-8')
118     for num_updates in range(1, int(message.get(NUM_MESSAGES))
            ↪  + 1):
119         if message.get(MESSAGE_TYPE, num_updates) == BID:
120             if message.get(ACTION, num_updates) == NEW:
121                 data_dict['Action{}'.format(num_updates)] =
                        ↪  ('Bid', 'New',
                        ↪  float(message.get(PRICE,
                        ↪  num_updates)), float(message.get(SIZE,
                        ↪  num_updates)),
                        ↪  int(message.get(POSITION,
                        ↪  num_updates)),)
122             elif message.get(ACTION, num_updates) == CHANGE:
```

```
123                 data_dict['Action{}'.format(num_updates)] =
                        ↪ ('Bid', 'Change',
                        ↪ float(message.get(PRICE,
                        ↪ num_updates)), float(message.get(SIZE,
                        ↪ num_updates)),
                        ↪ int(message.get(POSITION,
                        ↪ num_updates)),)
124             elif message.get(ACTION, num_updates) == DELETE:
125                 data_dict['Action{}'.format(num_updates)] =
                        ↪ ('Bid', 'Delete',
                        ↪ float(message.get(PRICE,
                        ↪ num_updates)), float(message.get(SIZE,
                        ↪ num_updates)),
                        ↪ int(message.get(POSITION,
                        ↪ num_updates)),)
126         elif message.get(MESSAGE_TYPE, num_updates) == OFFER:
127             if message.get(ACTION, num_updates) == NEW:
128                 data_dict['Action{}'.format(num_updates)] =
                        ↪ ('Ask', 'New',
                        ↪ float(message.get(PRICE,
                        ↪ num_updates)), float(message.get(SIZE,
                        ↪ num_updates)),
                        ↪ int(message.get(POSITION,
                        ↪ num_updates)),)
129             elif message.get(ACTION, num_updates) == CHANGE:
130                 data_dict['Action{}'.format(num_updates)] =
                        ↪ ('Ask', 'Change',
                        ↪ float(message.get(PRICE,
                        ↪ num_updates)), float(message.get(SIZE,
                        ↪ num_updates)),
                        ↪ int(message.get(POSITION,
                        ↪ num_updates)),)
131             elif message.get(ACTION, num_updates) == DELETE:
132                 data_dict['Action{}'.format(num_updates)] =
                        ↪ ('Ask', 'Delete',
                        ↪ float(message.get(PRICE,
                        ↪ num_updates)), float(message.get(SIZE,
                        ↪ num_updates)),
                        ↪ int(message.get(POSITION,
                        ↪ num_updates)),)
133     return data_dict
```

```
134
135
136 def parse_fix_raw_data(df):
137     ''' Parses a generic Fix message and returns a dict'''
138     message = get_message_from_df(df)
139     if refresh_type(message) == 'FULL':  # Book full refresh
140         full_refresh_data = parse_full_refresh_message(message)
141         return full_refresh_data
142     elif refresh_type(message) == 'INCREMENTAL':  # Book
          ↪ incremental refresh
143         incremental_refresh_data =
              ↪ parse_incremental_refresh_message(message)
144         return incremental_refresh_data
145     else:
146         print('Unknown message type')
147         return {}
148
149 #  We now pass the dataframe to the parsing method
150 aux_df = raw_df.apply(parse_fix_raw_data, axis=1,
      ↪ result_type='expand')
151
152 #  Here we combine both dataframes to keep all data together in
      ↪ one df
153 df_data = pd.concat([raw_df, aux_df], axis=1)
154
155 #  Now we need to split the dataframe in 3 by subsetting:
      ↪ AY24D, AY24C and AY24
156 df_AY24D = df_data[df_data['Ticker'] == 'AY24D']
157 df_AY24C = df_data[df_data['Ticker'] == 'AY24C']
158 df_AY24 = df_data[df_data['Ticker'] == 'AY24']
159
160 # At this point we have the dataframes with the full refresh
      ↪ fields but
161 # the incremental messages are not yet incorporated in the
      ↪ book
162
163 # First we remove all the incremental messages that don't
      ↪ modify
164 # the order book and reset the indexes
165 df_AY24D = df_AY24D.dropna(thresh=5)
166 df_AY24D.reset_index(drop=True, inplace=True)
```

```
167
168 df_AY24C = df_AY24C.dropna(thresh=5)
169 df_AY24C.reset_index(drop=True, inplace=True)
170
171 df_AY24 = df_AY24.dropna(thresh=5)
172 df_AY24.reset_index(drop=True, inplace=True)
173
174 # We need to parse the incremental messages and fill the order
      ↪  book
175 # for all the rows
176
177
178 def new_order_book(row, last_row):
179     row[BOOK_COLUMNS] = last_row
180     if row['Action1'][1] == 'Change':
181         position = row['Action1'][0] + str(row['Action1'][4])
182         row[position] = (row['Action1'][2], row['Action1'][3])
183     elif row['Action1'][1] == 'Delete':
184         for pos in range(row['Action1'][4], 5):
185             aux = pos + 1
186             position_original = row['Action1'][0] + str(aux)
187             position_destination = row['Action1'][0] + str(pos)
188             row[position_destination] = row[position_original]
189         aux2 = row['Action1'][0] + '5'
190         if not pd.isnull(row['Action2']):
191             row[aux2] = (row['Action2'][2], row['Action2'][3])
192     elif row['Action1'][1] == 'New':
193         for pos in range(5, row['Action1'][4], -1):
194             aux = pos - 1
195             position_original = row['Action1'][0] + str(aux)
196             position_destination = row['Action1'][0] + str(pos)
197             row[position_destination] = row[position_original]
198         aux2 = row['Action1'][0] + str(row['Action1'][4])
199         row[aux2] = (row['Action1'][2], row['Action1'][3])
200     return row[BOOK_COLUMNS]
201
202
203 def process_incremental_messages(df):
204     ''' Incorporates the incremental messages info to the df
          ↪  order book'''
205     for index, row in df.iterrows():
```

```
206             if row['Message_type'] == 'X':
207                 last_row_data = df.loc[index - 1, BOOK_COLUMNS]
208                 new_data = new_order_book(row, last_row_data)
209                 df.at[index, BOOK_COLUMNS] = new_data.values
210 
211 
212 # Now we apply the processing of the incremental messages to
        ↪ the 3 dataframes
213 process_incremental_messages(df_AY24D)
214 process_incremental_messages(df_AY24C)
215 process_incremental_messages(df_AY24)
216 
217 # Now we want to modify these dataframes to separate the Bid
        ↪ and Ask tuples
218 # We create one new column for price and size for each Bid and
        ↪ Ask
219 
220 
221 def separate_bid_and_ask(df):
222     ''' Takes a dataframe with a tuple of price and size and
            ↪ returns
223     a dataframe with that data separated in two different
            ↪ columns'''
224     for column in BOOK_COLUMNS:
225         df[[column+'_Price', column+'_Size']] =
                ↪ pd.DataFrame(df[column].tolist(),
                ↪ index=df.index)
226     df.drop(BOOK_COLUMNS, axis=1)
227 
228 
229 separate_bid_and_ask(df_AY24D)
230 separate_bid_and_ask(df_AY24C)
231 separate_bid_and_ask(df_AY24)
232 
233 '''Set datetime column as index. df must have a valid
        ↪ 'Datetime' column'''
234 df_AY24D = df_AY24D.set_index('Datetime',drop=False)
235 df_AY24C = df_AY24C.set_index('Datetime',drop=False)
236 df_AY24 = df_AY24.set_index('Datetime',drop=False)
237 
238 
```

```
239 # We now remove the duplicate indexes keeping only the last
        ↪ state (this caused problems)
240 df_AY24D = df_AY24D[~df_AY24D.index.duplicated(keep='last')]
241 df_AY24C = df_AY24C[~df_AY24C.index.duplicated(keep='last')]
242 df_AY24 = df_AY24[~df_AY24.index.duplicated(keep='last')]
243
244 ## Book saving as .csv
245
246 # Definition of columns to save in the .csv files
247 USEFUL_COLUMNS = ['Bid1_Price',
248                   'Bid2_Price',
249                   'Bid3_Price',
250                   'Bid4_Price',
251                   'Bid5_Price',
252                   'Ask1_Price',
253                   'Ask2_Price',
254                   'Ask3_Price',
255                   'Ask4_Price',
256                   'Ask5_Price',
257                   'Bid1_Size',
258                   'Bid2_Size',
259                   'Bid3_Size',
260                   'Bid4_Size',
261                   'Bid5_Size',
262                   'Ask1_Size',
263                   'Ask2_Size',
264                   'Ask3_Size',
265                   'Ask4_Size',
266                   'Ask5_Size',
267                   ]
268
269 # Actual saving of the files
270 df_AY24D[USEFUL_COLUMNS].to_csv('C:/Users/Ariel/Documents/Maestria/TESIS/Mi
        ↪ Tesis/AY24D_18-12-19.csv')
271 df_AY24C[USEFUL_COLUMNS].to_csv('C:/Users/Ariel/Documents/Maestria/TESIS/Mi
        ↪ Tesis/AY24C_18-12-19.csv')
272 df_AY24[USEFUL_COLUMNS].to_csv('C:/Users/Ariel/Documents/Maestria/TESIS/Mi
        ↪ Tesis/AY24_18-12-19.csv')
```

## C.2. Entrenamiento y prueba de los agentes

```
# Alonso, Ariel - MFE Thesis - Reinforcement Learning Module

import pandas as pd
import numpy as np

# Dataset

## Data configuration definitions

DATA_PATH = 'C:/Users/Ariel/Documents/Maestria/TESIS/Mi Tesis/'

# Definition for data dates to use
DATA_DATES = ['18-12-19', '19-12-19', '20-12-19', '06-01-20',
    ↪ '07-01-20']

# Definition of order book data for the agents
TOP_OF_BOOK = ['Bid1_Price', 'Bid1_Size',
               'Ask1_Price', 'Ask1_Size']
FULL_BOOK = ['Bid1_Price', 'Bid1_Size',
             'Bid2_Price', 'Bid2_Size',
             'Bid3_Price', 'Bid3_Size',
             'Bid4_Price', 'Bid4_Size',
             'Bid5_Price', 'Bid5_Size',
             'Ask1_Price', 'Ask1_Size',
             'Ask2_Price', 'Ask2_Size',
             'Ask3_Price', 'Ask3_Size',
             'Ask4_Price', 'Ask4_Size',
             'Ask5_Price', 'Ask5_Size',]

# DATASET CHARACTERISTICS

MARKET_IMBALANCE = True # Defines if we summarize the entire
    ↪ book into one number
SIMPLIFY_INDICATORS = True # Defines if simplify the MACD
    ↪ signal and RSI indicators
BID_ASK_SPREAD = True # Defines if we add the bid_ask_spread
    ↪ column to the dataframe
QUANTIZE = False # Defines if we quantize the size_imbalance
    ↪ and the bid_ask_spread columns
```

```
REWARD_HORIZON = 100

## Data loading

# Load dataframes in lists
df_data_list = []
for date in DATA_DATES:
    df_data_list.append(pd.read_csv('{}AY24D_{}.csv'
    .format(DATA_PATH, date)))

## Dataset enrichment

# Cut first and last entries of the dataframes to remove
    ↪ inaccurate data
ENTRIES_REMOVED = 1000
ENTRIES_RETAINED = 1000
START = 3000

for i, df in enumerate (df_data_list):
    df = df[ENTRIES_REMOVED:-ENTRIES_REMOVED]
    if i < 5:
        df = df[START:START + ENTRIES_RETAINED]
    df_data_list[i] = df.reset_index(drop = True)
    df_data_list[i].drop(columns=['Datetime'], inplace = True)
        ↪ # We drop unwanted columns

# Technical indicators for enriching the dataset

import talib
INDICATORS_EMA_SPAN_MULTIPLIER = 10 # We augment the period of
    ↪ the MACD EMAs to filter short term noise better
RSI_THRESHOLD = 40 # Defines oversold and overbought
    ↪ percentages, ranges from 0 to 50

def MACD(df):
    '''Calculates the MACD and MACD SIGNAL indicators'''
    EMA_12 =
        ↪ df.Mid_Price.ewm(span=12*INDICATORS_EMA_SPAN_MULTIPLIER,
        ↪ adjust=False).mean()
```

```
68      EMA_26 =
            ↪ df.Mid_Price.ewm(span=26*INDICATORS_EMA_SPAN_MULTIPLIER,
            ↪ adjust=False).mean()
69      MACD = EMA_12 - EMA_26
70      MACD_SIGNAL = MACD.ewm(span=9, adjust=False).mean()
71      return MACD, MACD_SIGNAL
72
73  def RSI(df):
74      '''Calculates the RSI indicator'''
75      RSI = talib.RSI(df['Mid_Price'], timeperiod =
            ↪ 14*INDICATORS_EMA_SPAN_MULTIPLIER)
76      return RSI
77
78  def indicators_enrichment(df):
79      '''Incorporates indicators to the dataframe'''
80      df['Mid_Price'] = (df['Bid1_Price'] + df['Ask1_Price'])/2
81
82      # MACD & SIGNAL
83      df['MACD'], df['MACD_SIGNAL'] = MACD(df)
84      if SIMPLIFY_INDICATORS:
85          # Simplifying the indicator
86          df['MACD_SIGNAL_CROSS'] = np.sign(np.sign(df['MACD'] -
                ↪ df['MACD_SIGNAL']).diff())
87          df.fillna({'MACD_SIGNAL_CROSS':0}, inplace=True)
88          df.drop(columns=['MACD', 'MACD_SIGNAL'], inplace=True)
89
90      # RSI
91      df['RSI'] = RSI(df)
92      df.fillna({'RSI':50}, inplace=True)# We fill the NaN with
            ↪ an indicator neutral value
93      if SIMPLIFY_INDICATORS:
94          # Simplifying the indicator
95          overbought =  df['RSI'] > (100 - RSI_THRESHOLD)
96          oversold   =  df['RSI'] < RSI_THRESHOLD
97          neither    = (df['RSI'] > RSI_THRESHOLD) & (df['RSI'] <
                ↪ (100 - RSI_THRESHOLD))
98          df['RSI'][overbought] = -1
99          df['RSI'][oversold]   =  1
100         df['RSI'][neither]    =  0
101
102     df.drop(columns=['Mid_Price'], inplace=True)
```

```
103     return df
104
105 # We apply the new indicators and incorporate them into the
        ↪ dataframes
106 for i, df in enumerate (df_data_list):
107     df_data_list[i] = indicators_enrichment(df)
108
109 # Bid-Ask spread feature
110 def bid_ask_spread(df):
111     df['Bid_Ask_spread'] = df['Ask1_Price'] - df['Bid1_Price']
112     return df
113
114 # We apply the new indicators and incorporate them into the
        ↪ dataframes
115 if BID_ASK_SPREAD:
116     for i, df in enumerate (df_data_list):
117         df_data_list[i] = bid_ask_spread(df)
118
119 # Defining agents, discarding irrelevant data for TOB agent
120 df_FB = df_data_list
121 df_TOB = []
122 for i, df in enumerate (df_data_list):
123     df_TOB.append(df.drop(columns=[e for e in FULL_BOOK if e
          ↪ not in TOP_OF_BOOK]))
124
125 # Market Imbalance calculations
126 tau = 1/20 # Penalizes bids and offers at distant prices
127
128 def book_imbalance(df, FULL_BOOK_AGENT):
129     if FULL_BOOK_AGENT:
130         Bid2_equiv = np.exp( - (df['Bid1_Price'] -
              ↪ df['Bid2_Price']) / tau ) * df['Bid2_Size']
131         Bid3_equiv = np.exp( - (df['Bid1_Price'] -
              ↪ df['Bid3_Price']) / tau ) * df['Bid3_Size']
132         Bid4_equiv = np.exp( - (df['Bid1_Price'] -
              ↪ df['Bid4_Price']) / tau ) * df['Bid4_Size']
133         Bid5_equiv = np.exp( - (df['Bid1_Price'] -
              ↪ df['Bid5_Price']) / tau ) * df['Bid5_Size']
134         Ask2_equiv = np.exp( - (df['Ask2_Price'] -
              ↪ df['Ask1_Price']) / tau ) * df['Ask2_Size']
```

```
135         Ask3_equiv = np.exp( - (df['Ask3_Price'] -
                ↪  df['Ask1_Price']) / tau ) * df['Ask3_Size']
136         Ask4_equiv = np.exp( - (df['Ask4_Price'] -
                ↪  df['Ask1_Price']) / tau ) * df['Ask4_Size']
137         Ask5_equiv = np.exp( - (df['Ask5_Price'] -
                ↪  df['Ask1_Price']) / tau ) * df['Ask5_Size']
138
139         df['Size_Imbalance'] = (df['Bid1_Size'] -
                ↪  df['Ask1_Size'])  + \
140                                 (Bid2_equiv - Ask2_equiv)    +
                                        ↪  \
141                                 (Bid3_equiv - Ask3_equiv)    +
                                        ↪  \
142                                 (Bid4_equiv - Ask4_equiv)    +
                                        ↪  \
143                                 (Bid5_equiv - Ask5_equiv)
144
145         DROP_COLUMNS = [e for e in FULL_BOOK if e not in
                ↪  ('Bid1_Price','Ask1_Price')]
146         df.drop(columns=DROP_COLUMNS, inplace=True)
147         return df
148
149     else:
150         df['Size_Imbalance'] = df['Bid1_Size'] -
                ↪  df['Ask1_Size']
151         df.drop(columns=['Bid1_Size', 'Ask1_Size'],
                ↪  inplace=True)
152         return df
153
154 if MARKET_IMBALANCE:
155     for i, df in enumerate (df_TOB):
156         df_TOB[i] = book_imbalance(df, False)
157     for i, df in enumerate (df_FB):
158         df_FB[i] = book_imbalance(df, True)
159
160 # Series quantizing
161 def quantize(df):
162     QUANTILES_QTY = 4
163     df['Size_Imbalance'] = pd.qcut(df['Size_Imbalance'],
            ↪  QUANTILES_QTY, labels=False, duplicates = 'drop')
```

```
164     df['Bid_Ask_spread']    = pd.qcut(df['Bid_Ask_spread'],
            ↪  QUANTILES_QTY, labels=False, duplicates = 'drop')
165     return df
166 
167 # We apply the new quantizing method to the dataframes
168 if QUANTIZE:
169     for i, df in enumerate (df_TOB):
170         df_TOB[i] = quantize(df)
171     for i, df in enumerate (df_FB):
172         df_FB[i] = quantize(df)
173 
174 # Date selection of the training, validation and testing
        ↪  datasets
175 TRAIN_DATA = ['18-12-19', '19-12-19', '20-12-19']
176 VALIDATION_DATA  = '06-01-20'
177 TEST_DATA = '07-01-20'
178 
179 # We obtain the final dataframes of training and testing
180 df_validation_TOB_agent =
        ↪  df_TOB[DATA_DATES.index(VALIDATION_DATA)]
181 df_validation_FB_agent  =
        ↪  df_FB[DATA_DATES.index(VALIDATION_DATA)]
182 
183 df_test_TOB_agent = df_TOB[DATA_DATES.index(TEST_DATA)]
184 df_test_FB_agent  = df_FB[DATA_DATES.index(TEST_DATA)]
185 
186 df_TOB_aux = df_TOB.copy()
187 df_FB_aux = df_FB.copy()
188 
189 del df_TOB_aux[DATA_DATES.index(TEST_DATA)]
190 del df_TOB_aux[DATA_DATES.index(VALIDATION_DATA)]
191 del df_FB_aux[DATA_DATES.index(TEST_DATA)]
192 del df_FB_aux[DATA_DATES.index(VALIDATION_DATA)]
193 
194 # Now we keep the daily cut indexes between days to prevent day
        ↪  hopping in training
195 daily_cuts = np.array([len(item) for item in df_TOB_aux])
196 daily_cuts = np.insert(daily_cuts, 0, 0) # We insert a 0 in the
        ↪  first position as this indicates the first day start
197 
```

```
# We now concatenate all the traning dataframes into one,
    ↪ resetting the indexes, we already have the daily cuts
df_train_TOB_agent = pd.concat(df_TOB_aux, ignore_index = True)
df_train_FB_agent  = pd.concat(df_FB_aux, ignore_index = True)

# RL Agents Training Environment

# TRAINING ENVIRONMENT

import gym
from gym import spaces
import tensorflow
import numpy as np
import random
from sklearn import preprocessing

TRAINING_DAY_COUNT = len(TRAIN_DATA)
EPISODE_LENGTH = len(df_train_TOB_agent)

POSITION_LIMITS = False
MAX_CASH = 10000
MAX_BOND_QTY = 5

POSITION_PENALTY = False
POSITION_PENALTY_AMOUNT = 10

TRADING_PENALTY = False
NON_TRADING_PENALTY_AMOUNT = 1

POSITIVE_REWARD_BOOSTER = False
REWARD_BOOST_FACTOR = 10

REWARD_IMMEDIATE = True
REWARD_END_OF_EPISODE = False

HIGH_FREQ_PENALTY = False
HIGH_FREQ_TICKS = REWARD_HORIZON + 1
HIGH_FREQ_PENALTY_AMOUNT = 10


class AY24BooksTrainEnv(gym.Env):
```

```
238         BUY  = 0
239         SELL = 1
240         HOLD = 2
241 
242         def __init__(self, df, daily_cuts = np.array([]),
                ↪  agent_bond_qty = 0, agent_cash = 0, tick = 0):
243             self.df = df
244             self.daily_cuts = daily_cuts
245             self.daily_cuts_cumsum = daily_cuts.cumsum().tolist()
246             self.traning_day = 0
247             self.tick = tick
248             self.terminal = False
249             self.reward = 0
250             self.cum_reward = 0
251             self.hf_ticks = 0
252             n_actions = 3
253             self.action_space = spaces.Discrete(n_actions)
254             self.observation_space = spaces.Box(low=-1, high=1,
                    ↪  shape=[self.df.shape[1]  - 2],
                    ↪  dtype=np.double)
255             # The observation space includes bid-ask spread, MACD
                    ↪  signal, RSI signal and market imbalance
                    ↪  features
256             # The - 2 results from substracting the bid and ask
                    ↪  prices from the feature set
257             self.agent_bond_qty = agent_bond_qty
258             self.agent_cash = agent_cash
259             self.data = self.df.loc[self.tick,:]
260             self.future_prices = self.df.loc[self.tick +
                    ↪  REWARD_HORIZON,:]
261             self.df_normalized = self.normalize_input_data()
262             self.df_normalized.drop(columns=
263             ['Bid1_Price','Ask1_Price'], inplace=True)
264             self.data_normalized =
                    ↪  self.df_normalized.loc[self.tick,:]
265             self.state = self.data_normalized.tolist()
266             self.pending_buy  = []
267             self.pending_sell = []
268 
269         def normalize_input_data(self):
270             x = self.df.values #returns a numpy array
```

```
271         min_max_scaler = preprocessing.MinMaxScaler()
272         x_scaled = min_max_scaler.fit_transform(x)
273         df_normalized = pd.DataFrame(x_scaled,
                ↪  columns=self.df.columns)
274         return df_normalized
275
276     def reset(self):
277         self.day = 0
278         self.tick = 0
279         self.initial_tick = self.tick
280         self.agent_cash = 0
281         self.agent_bond_qty = 0
282         self.passive = True
283         self.reward = 0
284         self.cum_reward = 0
285         self.hf_ticks = 0
286         self.data = self.df.loc[self.tick,:]
287         self.future_prices = self.df.loc[self.tick +
                ↪  REWARD_HORIZON,:]
288         self.data_normalized =
                ↪  self.df_normalized.loc[self.tick,:]
289         self.state = self.data_normalized.tolist()
290         observation = np.array(self.state)
291         self.pending_buy  = []
292         self.pending_sell = []
293         return observation
294
295     def buy_bond(self):
296         self.agent_cash -= self.data.Ask1_Price
297         self.agent_bond_qty += 1
298         if POSITION_LIMITS:
299             if self.agent_bond_qty < MAX_BOND_QTY and
                    ↪  self.agent_cash > -MAX_CASH:
300                 self.agent_cash -= self.data.Ask1_Price
301                 self.agent_bond_qty += 1
302             else:
303                 print("Can't buy, maximum bond quantity reached
                        ↪  or maximum negative cash limit
                        ↪  reached")
304
305     def sell_bond(self):
```

```
306             self.agent_cash += self.data.Bid1_Price
307             self.agent_bond_qty -= 1
308             if POSITION_LIMITS:
309                 if self.agent_bond_qty >= -MAX_BOND_QTY and
                        ↪  self.agent_cash < MAX_CASH:
310                     self.agent_cash += self.data.Bid1_Price
311                     self.agent_bond_qty -= 1
312                 else:
313                     print("Can't sell, maximum negative bond
                            ↪  quantity reached or maximum cash limit
                            ↪  reached")
314
315         def step(self, action):
316             if action == self.BUY:
317                 print('Action: Buy')
318                 self.buy_bond()
319                 self.pending_sell.append(self.tick +
                        ↪  REWARD_HORIZON)
320                 print('I am buying at tick', self.tick, ' and will
                        ↪  sell at tick', self.tick + REWARD_HORIZON)
321                 self.passive = False
322                 if HIGH_FREQ_PENALTY and self.hf_ticks <
                        ↪  HIGH_FREQ_TICKS:
323                     self.reward -= HIGH_FREQ_PENALTY_AMOUNT
324                 elif REWARD_IMMEDIATE:
325                     self.reward = self.future_prices.Bid1_Price -
                            ↪  self.data.Ask1_Price
326                 else:
327                     self.reward = 0
328                 self.hf_ticks = 0
329                 if POSITIVE_REWARD_BOOSTER:
330                     if self.reward > 0:
331                         self.reward *= REWARD_BOOST_FACTOR
332
333             elif action == self.SELL:
334                 print('Action: Sell')
335                 self.sell_bond()
336                 self.pending_buy.append(self.tick + REWARD_HORIZON)
337                 print('I am selling at tick', self.tick, ' and will
                        ↪  buy at tick', self.tick + REWARD_HORIZON)
338                 self.passive = False
```

```
339             if HIGH_FREQ_PENALTY and self.hf_ticks <
                    ↪  HIGH_FREQ_TICKS:
340                 self.reward =- HIGH_FREQ_PENALTY_AMOUNT
341             elif REWARD_IMMEDIATE:
342                 self.reward = self.data.Bid1_Price -
                        ↪  self.future_prices.Ask1_Price
343             else:
344                 self.reward = 0
345             self.hf_ticks = 0
346             if POSITIVE_REWARD_BOOSTER:
347                 if self.reward > 0:
348                     self.reward *= REWARD_BOOST_FACTOR
349                     print('Reward boosteada=', self.reward)
350         elif action == self.HOLD:
351             print('Action: Hold')
352             self.reward = 0
353             self.hf_ticks += 1
354         else:
355             print('Error: Invalid action')
356 
357         # We close previously opened position
358         if self.tick in self.pending_buy:
359             print('Buying to close position...')
360             self.buy_bond()
361             self.pending_buy.remove(self.tick)
362         elif self.tick in self.pending_sell:
363             print('Selling to close position...')
364             self.sell_bond()
365             self.pending_sell.remove(self.tick)
366 
367         # When the end of the day approaches, we stop and close
                ↪  the opened position at the corresponding tick
368         if self.tick >= (self.daily_cuts[self.day+1] -
                ↪  REWARD_HORIZON):
369             for i in self.pending_buy:
370                 self.tick = i
371                 self.data = self.df.loc[self.tick,:]
372                 self.buy_bond()
373             self.pending_buy = []
374             for i in self.pending_sell:
375                 self.tick = i
```

```
376                 self.data = self.df.loc[self.tick,:]
377                 self.sell_bond()
378             self.pending_sell = []
379 
380             self.day += 1
381             self.tick = self.daily_cuts_cumsum[self.day]
382 
383         # When the end of the whole df is reached, the terminal
                ↪ state needs to be checked:
384         ptf_value = self.agent_bond_qty * self.data.Bid1_Price
                ↪ + self.agent_cash if self.agent_bond_qty > 0 \
385                     else self.agent_bond_qty *
                            ↪ self.data.Ask1_Price +
                            ↪ self.agent_cash
386         self.terminal = self.tick >= self.initial_tick +
                ↪ EPISODE_LENGTH
387         if self.terminal:
388             print('---------END OF EPISODE----------')
389             print('self.agent_cash=', round(self.agent_cash,2))
390             print('self.agent_bond_qty=', self.agent_bond_qty)
391             print('ptf_value=', ptf_value)
392             if REWARD_END_OF_EPISODE:
393                 self.reward += ptf_value  # CASO DE REWARD AL
                        ↪ FINAL
394             if self.passive and TRADING_PENALTY:
395                 self.reward -= NON_TRADING_PENALTY_AMOUNT
396                 print('Test reward after NTP =', self.reward)
397             if POSITIVE_REWARD_BOOSTER:
398                 if self.reward > 0:
399                     self.reward *= REWARD_BOOST_FACTOR # NO
                            ↪ BOOST AT THE END
400                     print('reward =',self.reward)
401             print('---------------------------------')
402             return self.state, self.reward, self.terminal, {}
403 
404 
405         # If no terminal state is reached, the training
                ↪ continues
406 
407         # We return the info of the next tick and the cash and
                ↪ position state in the observation state
```

```
408             if POSITION_PENALTY:
409                 if abs(self.agent_bond_qty) > MAX_BOND_QTY:
410                     print("Bond Quantity penalty reached")
411                     self.reward =- POSITION_PENALTY_AMOUNT
412             self.data = self.df.loc[self.tick,:]
413             self.future_prices = self.df.loc[self.tick +
                    ↪  REWARD_HORIZON,:]
414             self.data_normalized =
                    ↪  self.df_normalized.loc[self.tick,:]
415             self.state = self.data_normalized.tolist()
416             observation = np.array(self.state)
417
418             return observation, self.reward, self.terminal, {} # 1.
                    ↪  proximo book,
419
420     # Environment checks
421
422     from stable_baselines.common.env_checker import check_env
423
424     env = AY24BooksTrainEnv(df_train_FB_agent, daily_cuts)
425
426     # This will check the custom environment and output additional
            ↪  warnings if needed
427     check_env(env)
428
429     # RL Model definitions and selection
430
431     from stable_baselines import DQN
432     from stable_baselines import PPO2
433     from stable_baselines import ACER
434     from stable_baselines.common import make_vec_env
435     from stable_baselines.common.vec_env import DummyVecEnv,
            ↪  SubprocVecEnv, VecNormalize
436
437     # Model selection
438
439     MODEL = 'DQN' # Can be DQN, PPO2 or ACER
440
441     # Logging path definition
442     import datetime
```

```
443 log_dir = "logs\\" + MODEL + "_" + "EP" + str(EPISODE_LENGTH) +
        ↪  "_" +
        ↪  datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
444
445 # Model creation
446
447 # Case 1: DQN model
448 if MODEL == 'DQN':
449     from stable_baselines.deepq.policies import MlpPolicy,
            ↪  LnMlpPolicy, CnnPolicy
450     env_TOB = AY24BooksTrainEnv(df_train_TOB_agent, daily_cuts)
451     env_FB  = AY24BooksTrainEnv(df_train_FB_agent, daily_cuts)
452     model_TOB = DQN(LnMlpPolicy, env_TOB, verbose=1,
            ↪  tensorboard_log=".\\{}_TOB".format(log_dir), seed
            ↪  = 1, n_cpu_tf_sess = 1)
453     model_FB  = DQN(LnMlpPolicy, env_FB, verbose=1,
            ↪  tensorboard_log=".\\{}_FB".format(log_dir), seed =
            ↪  1, n_cpu_tf_sess = 1)
454
455 # Case 2: PPO2 model
456 elif MODEL == 'PPO2':
457     from stable_baselines.common.policies import MlpPolicy,
            ↪  CnnLstmPolicy, CnnPolicy, MlpLnLstmPolicy
458     env_TOB = DummyVecEnv([lambda:
            ↪  AY24BooksTrainEnv(df_train_TOB_agent,
            ↪  daily_cuts)])
459     env_FB  = DummyVecEnv([lambda:
            ↪  AY24BooksTrainEnv(df_train_FB_agent, daily_cuts)])
460     # Additional normalization layer
461     env_TOB = VecNormalize(env_TOB)
462     env_FB  = VecNormalize(env_FB)
463     model_TOB = PPO2(MlpPolicy, env_TOB, verbose=1,
            ↪  tensorboard_log=".\\{}_TOB".format(log_dir),
            ↪  seed=1, n_cpu_tf_sess=1)
464     model_FB  = PPO2(MlpPolicy, env_FB, verbose=1,
            ↪  tensorboard_log=".\\{}_FB".format(log_dir),
            ↪  seed=1, n_cpu_tf_sess=1)
465
466 # Case 3: ACER model
467 elif MODEL == 'ACER':
```

```
468     from stable_baselines.common.policies import MlpPolicy,
            ↪ CnnLstmPolicy, CnnPolicy, MlpLnLstmPolicy
469     env_TOB = DummyVecEnv([lambda:
            ↪ AY24BooksTrainEnv(df_train_TOB_agent,
            ↪ daily_cuts)])
470     env_FB  = DummyVecEnv([lambda:
            ↪ AY24BooksTrainEnv(df_train_FB_agent, daily_cuts)])
471     # Additional normalization layer
472     env_TOB = VecNormalize(env_TOB)
473     env_FB  = VecNormalize(env_FB)
474     model_TOB = ACER(MlpLnLstmPolicy, env_TOB, verbose=1,
            ↪ tensorboard_log=".\\{}_TOB".format(log_dir),
            ↪ seed=1, n_cpu_tf_sess=1)
475     model_FB  = ACER(MlpLnLstmPolicy, env_FB, verbose=1,
            ↪ tensorboard_log=".\\{}_FB".format(log_dir),
            ↪ seed=1, n_cpu_tf_sess=1)
476
477 else:
478     print('Invalid model')
479
480 # NOW OPEN TENSORBOARD
481 # In Anaconda shell, change to env: "conda activate tf-1.15".
482 # Type "tensorboard --logdir logs"
483 # Open Tensorboard on browser at: http://localhost:6006/
484
485 # RL Agents Testing Environment
486
487 # TESTING ENVIRONMENT
488
489 import matplotlib.pyplot as plt
490
491 class AY24BooksTestEnv(AY24BooksTrainEnv):
492     def reset(self):
493         self.tick = 0 # We set the testing data always at the
                ↪ beginning of each DF
494         self.agent_cash = 0
495         self.agent_bond_qty = 0
496         self.agent_bond_qty_graph = []
497         self.agent_cash_graph = []
498         self.agent_ptf_value = []
499         self.data = self.df.loc[self.tick,:]
```

```
500             self.data_normalized =
                    ↪  self.df_normalized.loc[self.tick,:]
501             self.state = self.data_normalized.tolist()
502             observation = np.array(self.state)
503             self.pending_buy  = []
504             self.pending_sell = []
505             return observation
506
507         def step(self, action):
508             ptf_value = self.agent_bond_qty * self.data.Bid1_Price
                    ↪  + self.agent_cash if self.agent_bond_qty > 0 \
509                         else self.agent_bond_qty *
                                ↪  self.data.Ask1_Price +
                                ↪  self.agent_cash
510             # We check if the end of the dataset has been reached
511             self.terminal = self.tick > len(self.df.index.unique())
                    ↪  - REWARD_HORIZON - 1
512             if self.terminal:
513                 print('Closing position in tick ', self.tick)
514                 for i in self.pending_buy:
515                     self.tick = i
516                     self.data = self.df.loc[self.tick,:]
517                     self.buy_bond()
518                 self.pending_buy = []
519                 for i in self.pending_sell:
520                     self.tick = i
521                     self.data = self.df.loc[self.tick,:]
522                     self.sell_bond()
523                 self.pending_sell = []
524
525                 print('-----END OF DATAFRAME-----')
526                 print('Tick number is:', self.tick)
527                 print('self.agent_cash=', round(self.agent_cash,2))
528                 print('self.agent_bond_qty=', self.agent_bond_qty)
529                 print('reward=',self.reward)
530                 info = {'bid1_price': self.data.Bid1_Price,
531                         'agent_bond_qty': self.agent_bond_qty,
532                         'agent_cash': self.agent_cash,
533                         'agent_ptf_value': ptf_value}
534                 return self.state, self.reward, self.terminal, info
535
```

```
536             if action == self.BUY:
537                 print('Action: Buy')
538                 self.buy_bond()
539                 self.pending_sell.append(self.tick +
                        ↪  REWARD_HORIZON)
540                 print('I am buying at tick', self.tick, ' and will
                        ↪  sell at tick', self.tick + REWARD_HORIZON)
541
542             elif action == self.SELL:
543                 print('Action: Sell')
544                 self.sell_bond()
545                 self.pending_buy.append(self.tick + REWARD_HORIZON)
546                 print('I am selling at tick', self.tick, ' and will
                        ↪  buy at tick', self.tick + REWARD_HORIZON)
547
548             elif action == self.HOLD:
549                 print('Action: Hold')
550                 pass
551             else:
552                 print('Error: Invalid action')
553
554             # We close previously opened position
555             if self.tick in self.pending_buy:
556                 print('Buying to close position...')
557                 self.buy_bond()
558                 self.pending_buy.remove(self.tick)
559             elif self.tick in self.pending_sell:
560                 print('Selling to close position...')
561                 self.sell_bond()
562                 self.pending_sell.remove(self.tick)
563
564             self.tick += 1
565
566             # Devolvemos en observation la info del proximo tick
567             self.data = self.df.loc[self.tick,:]
568             self.data_normalized =
                    ↪  self.df_normalized.loc[self.tick,:]
569             self.cum_reward += self.reward
570             self.state = self.data_normalized.tolist()
571             observation = np.array(self.state)
```

```
572             ptf_value = self.agent_bond_qty * self.data.Bid1_Price
                    ↪  + self.agent_cash if self.agent_bond_qty > 0 \
573                         else self.agent_bond_qty *
                                ↪  self.data.Ask1_Price +
                                ↪  self.agent_cash
574             self.reward = 0 # We are only interested in the reward
                    ↪  at end of episode at testing time
575             info = {'bid1_price': self.data.Bid1_Price,
576                     'agent_bond_qty': self.agent_bond_qty,
577                     'agent_cash': self.agent_cash,
578                     'agent_ptf_value': ptf_value}
579             return observation, self.reward, self.terminal, info
                    ↪  # 1. proximo book,
580
581     # Model Training and predictions
582
583     ## Training configuration
584
585     TRAINING_ITERATIONS = 20 # Number of models that will predict
            ↪  on the test data, each more trained than the previous
            ↪  one
586     INCREMENTAL_TIMESTEPS = 20000 # Ticks that each model will
            ↪  train (cumulative)
587     TRAINING_TIMESTEPS = np.arange(0,(TRAINING_ITERATIONS+1) *
            ↪  INCREMENTAL_TIMESTEPS, INCREMENTAL_TIMESTEPS).tolist()
588     RESET_MODEL = True
589     SEED = 1 # Seed setting
590
591     ## Training and predictions on the training and validation set
592
593     # Training and prediction loop on the training data, and
            ↪  information storage for future graphs
594
595     agent_cash_TOB_iterations = []
596     agent_cash_FB_iterations  = []
597
598     for iteration in range(len(TRAINING_TIMESTEPS)):
599         print ('Training iteration number ', iteration)
600         agent_cash_TOB = []
601         agent_cash_FB  = []
602
```

```
603     # Prediction loop
604     for i, date in enumerate(DATA_DATES[0:4]):
605         print('Day to predict is ', date)
606         # We need to vectorize the test environment for PPO2 or
                ↪  ACER models
607         if MODEL == 'PPO2' or 'ACER':
608             env_validation_TOB = DummyVecEnv([lambda:
                    ↪  AY24BooksTestEnv(df_TOB[i])])
609             env_validation_FB  = DummyVecEnv([lambda:
                    ↪  AY24BooksTestEnv(df_FB[i])])
610         else:
611             env_validation_TOB = AY24BooksTestEnv(df_TOB[i])
612             env_validation_FB  = AY24BooksTestEnv(df_FB[i])
613
614         obs_validation_TOB = env_validation_TOB.reset()
615         obs_validation_FB  = env_validation_FB.reset()
616
617         # Prediction of the TOB agent
618         while(True):
619             action, _states =
                    ↪  model_TOB.predict(obs_validation_TOB)
620             obs_validation_TOB, rewards, dones, info =
                    ↪  env_validation_TOB.step(action)
621             if dones:
622                 print('agent_cash_TOB_day is',
                        ↪  info[0]['agent_cash'])
623                 agent_cash_TOB_day = info[0]['agent_cash']
624                 break
625
626         # Prediction of the FB agent
627         while(True):
628             action, _states =
                    ↪  model_FB.predict(obs_validation_FB)
629             obs_validation_FB, rewards, dones, info =
                    ↪  env_validation_FB.step(action)
630             if dones:
631                 print('agent_cash_FB_day is',
                        ↪  info[0]['agent_cash'])
632                 agent_cash_FB_day = info[0]['agent_cash']
633                 break
634
```

```
635             agent_cash_TOB.append(agent_cash_TOB_day)
636             agent_cash_FB.append(agent_cash_FB_day)
637 
638         agent_cash_TOB_iterations.append(agent_cash_TOB)
639         agent_cash_FB_iterations.append(agent_cash_FB)
640 
641         if iteration < len(TRAINING_TIMESTEPS)-1:
642             # We train the model now and run the prediction in the
                    ↪  next iteration
643             if RESET_MODEL is True:
644                 del model_TOB
645                 del model_FB
646                 if MODEL == 'DQN':
647                     model_TOB = DQN(LnMlpPolicy, env_TOB,
                            ↪  verbose=1,
                            ↪  tensorboard_log=".\\{}_TOB".format(log_dir),
                            ↪  seed = SEED, n_cpu_tf_sess = 1)
648                     model_FB  = DQN(LnMlpPolicy, env_FB, verbose=1,
                            ↪  tensorboard_log=".\\{}_FB".format(log_dir),
                            ↪  seed = SEED, n_cpu_tf_sess = 1)
649                 elif MODEL == 'PPO2':
650                     model_TOB = PPO2(MlpPolicy, env_TOB, verbose=1,
                            ↪  tensorboard_log=".\\{}_TOB".format(log_dir),
                            ↪  seed=SEED, n_cpu_tf_sess=1)
651                     model_FB  = PPO2(MlpPolicy, env_FB, verbose=1,
                            ↪  tensorboard_log=".\\{}_FB".format(log_dir),
                            ↪  seed=SEED, n_cpu_tf_sess=1)
652                 elif MODEL == 'ACER':
653                     model_TOB = ACER(MlpLnLstmPolicy, env_TOB,
                            ↪  verbose=1,
                            ↪  tensorboard_log=".\\{}_TOB".format(log_dir),
                            ↪  seed=SEED, n_cpu_tf_sess=1)
654                     model_FB  = ACER(MlpLnLstmPolicy, env_FB,
                            ↪  verbose=1,
                            ↪  tensorboard_log=".\\{}_FB".format(log_dir),
                            ↪  seed=SEED, n_cpu_tf_sess=1)
655                 print('I will train for this amount of timesteps:',
                        ↪  TRAINING_TIMESTEPS[iteration+1])
656                 model_TOB.learn(total_timesteps=TRAINING_TIMESTEPS
657                 [iteration+1])
658                 model_FB.learn(total_timesteps=TRAINING_TIMESTEPS
```

```
659             [iteration+1])
660         else:
661             print('I will train for this amount of timesteps:',
                ↪  TRAINING_TIMESTEPS[iteration+1]-TRAINING_TIMESTEPS
662             [iteration])
663             model_TOB.learn(total_timesteps=TRAINING_TIMESTEPS
664             [iteration+1]-TRAINING_TIMESTEPS[iteration])
665             model_FB.learn(total_timesteps=TRAINING_TIMESTEPS
666             [iteration+1]-TRAINING_TIMESTEPS[iteration])
667
668 # print_profit() function for evaluating the performance of the
        ↪  TOB and FB agents
669 # predictions on the training and validation sets dates
670
671 def print_profit(TOB_series, FB_series):
672     df_TOB = pd.DataFrame(TOB_series)
673     df_TOB.columns = ['TRAIN DAY %d %s TOB'% (i, item) for
            ↪  i,item in enumerate(TRAIN_DATA)] + ['VALIDATION
            ↪  DAY %s TOB' % VALIDATION_DATA]
674     df_FB = pd.DataFrame(FB_series)
675     df_FB.columns  = ['TRAIN DAY %d %s FB'% (i, item) for
            ↪  i,item in enumerate(TRAIN_DATA)] + ['VALIDATION
            ↪  DAY %s FB' % VALIDATION_DATA]
676     df = pd.concat([df_TOB, df_FB], axis = 1)
677     ax = df.plot(figsize = (16,8),
678             title=('Agent Profit' + ' (' + MODEL + ', ENT=' +
                    ↪  str(ENTRIES_RETAINED) + ', RH=' +
                    ↪  str(REWARD_HORIZON) + ')'),
679             xlabel = ('Training Timesteps'),
680             rot = 45,
681             ylabel = ('Agent Profit ($)'),
682             sort_columns = True,
683             grid = True,
684             style = ['-','-','-','-','--','--','--','--'],
685             color = ['b', 'g', 'r', 'c', 'b', 'g', 'r', 'c'],
686             linewidth = 1,
687             )
688     ax.set_xticks(df.index)
689     ax.set_xticklabels(TRAINING_TIMESTEPS, rotation=45)
690
```

```
691     df_validation = df[['VALIDATION DAY %s TOB' %
            ↪ VALIDATION_DATA, 'VALIDATION DAY %s FB' %
            ↪ VALIDATION_DATA]]
692     ax_validation = df_validation.plot(figsize = (16,8),
693             title=('Agent Profit' + ' (' + MODEL + ', ENT=' +
                    ↪ str(ENTRIES_RETAINED) + ', RH=' +
                    ↪ str(REWARD_HORIZON) + ')'),
694             xlabel = ('Training Timesteps'),
695             rot = 45,
696             ylabel = ('Agent Profit ($)'),
697             sort_columns = True,
698             grid = True,
699             color = ['r', 'g'],#, 'r', 'c', 'b', 'g', 'r',
                    ↪ 'c'],
700             linewidth = 1,
701             )
702     ax_validation.set_xticks(df_validation.index)
703     ax_validation.set_xticklabels(TRAINING_TIMESTEPS,
            ↪ rotation=45)
704
705 # Produce graphs of training vs validation profit and isolated
        ↪ validation profit
706 print_profit(agent_cash_TOB_iterations,
        ↪ agent_cash_FB_iterations)
707
708 ## Predictions on the testing dataset
709
710 #We test now on the test day
711
712 # We need to vectorize the test environment for PPO2 or ACER
        ↪ models
713 if MODEL == 'PPO2' or 'ACER':
714     env_test_TOB = DummyVecEnv([lambda:
            ↪ AY24BooksTestEnv(df_test_TOB_agent)])
715     env_test_FB  = DummyVecEnv([lambda:
            ↪ AY24BooksTestEnv(df_test_FB_agent)])
716 else:
717     env_test_TOB = AY24BooksTestEnv(df_test_TOB_agent)
718     env_test_FB  = AY24BooksTestEnv(df_test_FB_agent)
719
720 obs_test_TOB = env_test_TOB.reset()
```

```
721 obs_test_FB  = env_test_FB.reset()
722
723 bid1_price_day = []
724 agent_bond_qty_TOB_day = []
725 agent_bond_qty_FB_day  = []
726 agent_cash_TOB_day = []
727 agent_cash_FB_day  = []
728 agent_ptf_value_TOB_day = []
729 agent_ptf_value_FB_day  = []
730
731 # Prediction of the TOB agent
732 while(True):
733     action, _states = model_TOB.predict(obs_test_TOB)
734     obs_test_TOB, rewards, dones, info =
          ↪  env_test_TOB.step(action)
735     bid1_price_day.append(info[0]['bid1_price'])
736     agent_bond_qty_TOB_day.append(info[0]['agent_bond_qty'])
737     agent_cash_TOB_day.append(info[0]['agent_cash'])
738     agent_ptf_value_TOB_day.append(info[0]['agent_ptf_value'])
739     if dones:
740         break
741
742 # Prediction of the FB agent
743 while(True):
744     action, _states = model_FB.predict(obs_test_FB)
745     obs_test_FB, rewards, dones, info =
          ↪  env_test_FB.step(action)
746     agent_bond_qty_FB_day.append(info[0]['agent_bond_qty'])
747     agent_cash_FB_day.append(info[0]['agent_cash'])
748     agent_ptf_value_FB_day.append(info[0]['agent_ptf_value'])
749     if dones:
750         break
751
752 # Definiton of another printing function to evaluate the bond
      ↪  positions and ptf value of the test day predictions
753 def print_graphs(benchmark_series, TOB_series, FB_series,
      ↪  title, ylabel, scaled = False):
754     if scaled:
755         print('Bond QTY final TOB: ', TOB_series[-1])
756         print('Bond QTY final FB: ', FB_series[-1])
757         plt.figure(figsize=(12,8))
```

```
758             plt.subplot(2, 1, 1)
759             plt.title(title + ' (' + MODEL + ', EP=' +
                    ↪  str(EPISODE_LENGTH) + ', RH=' +
                    ↪  str(REWARD_HORIZON) + ')')
760             #plt.xlabel('Timesteps')
761             plt.ylabel('Bid1_Price($)')
762             plt.plot(benchmark_series, label='Bid1_Price', alpha=1,
                    ↪  color = 'blue')
763             plt.subplot(2, 1, 2)
764             #plt.title('Bond Quantity')
765             plt.xlabel('Timesteps')
766             plt.ylabel(ylabel)
767             plt.plot(TOB_series, label="TOB", color = 'red')
768             plt.plot(FB_series, label="FB", color = 'green')
769             plt.legend(loc="upper left")
770         else:
771             print('Cash final TOB: $', TOB_series[-1])
772             print('Cash final FB: $', FB_series[-1])
773             benchmark_series = (benchmark_series -
                    ↪  benchmark_series[0])
774             plt.figure(figsize=(12,8))
775             plt.title(title + ' (' + MODEL + ', EP=' +
                    ↪  str(EPISODE_LENGTH) + ', RH=' +
                    ↪  str(REWARD_HORIZON) + ')')
776             plt.xlabel('Timesteps')
777             plt.ylabel(ylabel)
778             plt.plot(benchmark_series, label='Long 1 Bond',
                    ↪  alpha=0.5, color = 'blue')
779             plt.plot(TOB_series, label="TOB", color = 'red')
780             plt.plot(FB_series, label="FB", color = 'green')
781             plt.legend(loc="upper left")
782         plt.show()
783
784 # Produce test day graphs
785 date = TEST_DATA
786 print_graphs(bid1_price_day, agent_bond_qty_TOB_day,
        ↪  agent_bond_qty_FB_day, 'TEST ' + date + ' Agent Bond
        ↪  Quantity', 'Bond Quantity', scaled=True)
787 print_graphs(bid1_price_day, agent_ptf_value_TOB_day,
        ↪  agent_ptf_value_FB_day, 'TEST ' + date + ' Agent
        ↪  Portfolio Value', 'PTF Value ($)')
```

```
788
789 # Final graph to plot once all the seeds configuration values
        ↪ have been run.
790 # Values in d are the results of different previous
        ↪ experiments.
791 d = {'TEST DAY 07-01-20 TOB': [-4.13, -1.67, -3.21, -5.18,
        ↪ -3.79, -5.14, -2.71, -2.28], 'TEST DAY 07-01-20 FB':
        ↪ [-3.36, -6.58, -3.21, -3.56, -2.63, -2.91, -3.77,
        ↪ -3.18]}
792 df = pd.DataFrame(data=d)
793 df.index = [1,2,3,4,5,6,7,8]
794 ax = df.plot(figsize = (16,8),
795              title=('Agent Profit on Test Data' + ' (' + MODEL +
                     ↪ ', ENT=' + str(ENTRIES_RETAINED) + ', RH='
                     ↪ + str(REWARD_HORIZON) + ')'),
796              xlabel = ('DQN Model #Seed'),
797              rot = 0,
798              ylabel = ('Agent Profit ($)'),
799              sort_columns = True,
800              grid = True,
801              style = ['-','-'],
802              color = ['r', 'g'],
803              linewidth = 1,
804              )
805 ax.set_xticks(df.index)
```

# D. Listado de experimentos

| # | Platform | Model | Policy | Train data | Test Data | Normalizacion | Actions | Reward | Train Method | Train Cycles | Features de precios | Indicadores Técnicos | Market Imbalance Simplification | Max Bonds | Status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Google Colab | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 10k | | No | No | 100 | |
| 2 | Google Colab | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 3 | Google Colab | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 4 | Google Colab | DQN | LnMlpPolicy | FB 18 | FB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 5 | Google Colab | DQN | LnMlpPolicy | FB 18 | FB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 6 | Google Colab | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 7 | Google Colab | PPO2 | CnnLstmPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 8 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 9 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 10 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 11 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 12 | Jupyter | PPO2 | CnnLstmPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 100k | | No | No | 100 | |
| 13 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 14 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 15 | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 16 | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 17 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 18 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 19 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 20 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 21 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 22 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | DF entero | 1M | | No | No | 100 | |
| 23 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 100k | | No | No | 100 | |
| 24 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 100k | | No | No | 100 | |
| 25 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 100k | | No | No | 100 | |
| 26 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 100k | | No | No | 100 | |
| 27 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 100k | | No | No | 100 | |
| 28 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 100k | | No | No | 100 | |
| 29 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 1M | | No | No | 100 | |
| 30 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 1M | | No | No | 100 | |
| 31 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 1M | | No | No | 100 | |
| 32 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | A 1 paso (-spread) | EPISODE (1k) | 1M | | No | No | 100 | |
| 33 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 100k | | No | No | 100 | |
| 34 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 100k | | No | No | 100 | |
| 35 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 100k | | No | No | 100 | |
| 36 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 100k | | No | No | 100 | |
| 37 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 100k | | No | No | 100 | |
| 38 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 100k | | No | No | 100 | |
| 39 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 40 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 41 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 42 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 43 | Jupyter | DQN | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 44 | Jupyter | DQN | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 45 | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |

| # | Platform | Model | Policy | Train data | Test Data | Normalizacion | Actions | Reward | Train Method | Train Cycles | Features de precios | Indicadores Técnicos | Market Imbalance Simplification | Max Bonds | Status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 46 | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 47 | Jupyter | DQN | LnMlpPolicy | TOB 20 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 48 | Jupyter | DQN | LnMlpPolicy | TOB 20 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 49 | Google Colab | PPO2 | MlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 10k | | No | No | 100 | |
| 50 | Google Colab | PPO2 | MlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 10k | | No | No | 100 | |
| 51 | Google Colab | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 52 | Google Colab | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 53 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 18 | VecNormalize | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 54 | Jupyter | PPO2 | MlpPolicy | TOB 18 | TOB 20 | VecNormalize | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 55 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | No | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 56 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | No | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 57 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 18 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 58 | Jupyter | DQN | LnMlpPolicy | TOB 18 | TOB 20 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 59 | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 18 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 60 | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 20 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (1k) | 1M | | No | No | 100 | |
| 61 | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 18 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 62 | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 20 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 63 | Google Colab | ACER | MlpPolicy | TOB 18 | TOB 18 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 64 | Google Colab | ACER | MlpPolicy | TOB 18 | TOB 20 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 65 | Jupyter | ACER | MlpLnLstmPolicy | TOB 18 | TOB 18 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 66 | Jupyter | ACER | MlpLnLstmPolicy | TOB 18 | TOB 20 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 67 | Jupyter | ACER | MlpPolicy | TOB 18 | TOB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 68 | Jupyter | ACER | MlpPolicy | TOB 18 | TOB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 69 | Jupyter | ACER | MlpLnLstmPolicy | TOB 18 | TOB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 70 | Jupyter | ACER | MlpLnLstmPolicy | TOB 18 | TOB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | No | No | 100 | |
| 71 | Google Colab | ACER | MlpLnLstmPolicy | TOB 18 | TOB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | 100 | |
| 72 | Google Colab | ACER | MlpLnLstmPolicy | TOB 18 | TOB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | 100 | |
| 73 | Google Colab | ACER | MlpLnLstmPolicy | TOB 18 | TOB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 100k | | No | No | 100 | |
| 74 | Google Colab | ACER | MlpLnLstmPolicy | TOB 18 | TOB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 100k | | No | No | 100 | |
| 75 | Google Colab | ACER | MlpLnLstmPolicy | FB 18 | FB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 50k | | No | No | 100 | |
| 76 | Google Colab | ACER | MlpLnLstmPolicy | FB 18 | FB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 50k | | No | No | 100 | |
| 77 | Google Colab | PPO2 | MlpPolicy | TOB 18 | TOB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 50k | | No | No | 100 | |
| 78 | Google Colab | PPO2 | MlpPolicy | TOB 18 | TOB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 50k | | No | No | 100 | |
| 79 | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 50k | | No | No | 100 | |
| 80 | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 50k | | No | No | 100 | |
| 81 | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 500k | | No | No | 100 | |
| 82 | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 500k | | No | No | 100 | |
| 83 | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 18 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 500k | | No | No | 100 | |
| 84 | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 500k | | No | No | 100 | |
| 85 | Jupyter | ACER | MlpLnLstmPolicy | TOB 19 | TOB 19 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 100k | | No | No | 100 | |
| 86 | Jupyter | ACER | MlpLnLstmPolicy | TOB 19 | TOB 06 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 100k | | No | No | 100 | |
| 87 | Jupyter | ACER | MlpLnLstmPolicy | TOB 19 | TOB 19 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 1M | | No | No | 100 | |
| 88 | Jupyter | ACER | MlpLnLstmPolicy | TOB 20 | TOB 20 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 1M | | No | No | 100 | |
| 89 | Jupyter | ACER | MlpLnLstmPolicy | TOB 01 (mon. creciente) | TOB 01 (mon. creciente) | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 10k | | No | No | 100 | |
| 90 | Jupyter | ACER | MlpLnLstmPolicy | TOB 02 (triangulo) | TOB 02 (triangulo) | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 10k | | No | No | Infinito | |
| 91 | Jupyter | DQN | LnMlpPolicy | TOB 02 (triangulo) | TOB 02 (triangulo) | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | Infinito | |
| 92 | Jupyter | DQN | LnMlpPolicy | TOB 02 (triangulo) | TOB 02 (triangulo) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | Infinito | |
| 93 | Jupyter | DQN | LnMlpPolicy | TOB 03 (tri inv) | TOB 03 (tri inv) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | Infinito | |
| 94 | Jupyter | DQN | LnMlpPolicy | TOB 03 (tri inv) | TOB 02 (triangulo) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | Infinito | |
| 95 | Jupyter | DQN | LnMlpPolicy | TOB 03 (tri inv) | TOB 02 (triangulo) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | No | 10 | |

| # | Platform | Model | Policy | Train data | Test Data | Normalizacion | Actions | Reward | Train Method | Train Cycles | Features de precios | Indicadores Técnicos | Market Imbalance Simplification | Max Bonds | Status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **96** | Jupyter | DQN | LnMlpPolicy | FB 03 (tri inv) | FB 02 (triangulo) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | Si | Si | 10 | |
| **97** | Jupyter | DQN | LnMlpPolicy | FB 03 (tri inv) | FB 00 (sierra) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | Si | Si | 10 | |
| **98** | Jupyter | DQN | LnMlpPolicy | FB 03 (tri inv) | FB 02 (triangulo) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | No | Si | 10 | |
| **99** | Jupyter | DQN | LnMlpPolicy | FB 03 (tri inv) | FB 02 (triangulo) | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | Si | No | 10 | |
| **100** | Jupyter | ACER | MlpLnLstmPolicy | FB 03 (tri inv) | FB 02 (triangulo) | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | Si | Si | 10 | |
| **101** | Jupyter | ACER | MlpLnLstmPolicy | FB 03 (tri inv) | FB 00(sierra) | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | Si | Si | 10 | |
| **102** | Jupyter | PPO2 | MlpPolicy | FB 03 (tri inv) | FB 00(sierra) | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 100k | | Si | Si | 10 | |
| **103** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | Final del Episode | EPISODE (10) | 1M | | Si | Si | 10 | |
| **104** | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (10) | 1M | | Si | Si | 10 | |
| **105** | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 00 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (100) | 1M | | Si | Si | 10 | |
| **106** | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 00 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (20) | 1M | | Si | Si | 10 | |
| **107** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY única | EPISODE (100) | 1M | | Si | Si | 100 | |
| **108** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY única | EPISODE (10) | 1M | | Si | Si | 100 | |
| **109** | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY única | EPISODE (10) | 3M | | Si | Si | 100 | |
| **110** | Google Colab | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | PENALTY única | EPISODE (50) | 1M | | Si | Si (Improved) | 100 | |
| **111** | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode | EPISODE (50) | 100k | | Si (simplified) | Si (Improved) | 100 | |
| **112** | Jupyter | PPO2 | MlpPolicy | FB 06 | FB 07 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode c/booster | EPISODE (50) | 100k | | Si (simplified) | Si (Improved) | 100 | |
| **113** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 07 | Normalizacion manual + VecNormalize | 1, la elegida | Final del Episode c/booster | EPISODE (10) | 100k | | Si (simplified) | Si (Improved) | 100 | |
| **114** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | PENALTY + booster | EPISODE (50) | 1.1M | | Si (simplified) | Si (Improved) | 100 | |
| **115** | Jupyter | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY + booster | EPISODE (50) | 2M | | Si (simplified) | Si (Improved) | 100 | |
| **116** | Jupyter | ACER | MlpLnLstmPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY + booster | EPISODE (50) | 500k | | Si (simplified) | Si (Improved) | Infinito | |
| **117** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 07 | Normalizacion manual + VecNormalize | 1, la elegida | Solo booster, sin penalty | EPISODE (20) | 220k | Spread, sin precios ni portfolio | Si (simplified) | Si (Improved) | Infinito | |
| **118** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 07 | Normalizacion manual + VecNormalize | 1, la elegida | Solo booster, sin penalty | EPISODE (20) | 50k | Spread QUANTIZED 10 | Si (simplified) | Q 10 | Infinito | |
| **119** | Google Colab | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | No penalty, REWARD HORIZON = 20 | EPISODE (20) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **120** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | No penalty, REWARD HORIZON = 20 | EPISODE (20) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **121** | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No penalty, REWARD HORIZON = 20 | EPISODE (20) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **122** | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No penalty, REWARD HORIZON = 20 | EPISODE (20) | 200k | Spread NO Q | Si (simplified) (50 Mult) | NO Q | Infinito | |
| **123** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | No penalty, REWARD HORIZON = 20 | EPISODE (20) | 100k | Spread Q 3 | Si (simplified) (50 Mult) | Q 3 | Infinito | |
| **124** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | No penalty, REWARD HORIZON = 20 | EPISODE (20) | 300k | Spread Q 50 | Si (simplified) (50 Mult) | Q 50 | Infinito | |
| **125** | Google Colab | PPO2 | MlpPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY 0.1, REWARD HORIZON = 20 | EPISODE (20) | 500k | Spread Q 5 | Si (simplified) (50 Mult) | Q 5 | Infinito | |
| **126** | Jupyter | DQN | LnMlpPolicy | FB 18 | FB 19 | Normalizacion manual | 1, la elegida | No penalty, REWARD HORIZON = 100 | EPISODE (100) | 100k+400k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **127** | Jupyter | DQN | LnMlpPolicy | FB 06 | FB 19 | Normalizacion manual | 1, la elegida | No penalty, REWARD HORIZON = 100 | EPISODE (100) | 100k+400k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **128** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No penalty, REWARD HORIZON = 100 | EPISODE (100) | 200k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **129** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | PENALTY, REWARD HORIZON = 100 | EPISODE (100) | 1M | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **130** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No penalty, REWARD HORIZON = 200 | EPISODE (200) | 1M | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **131** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | BOOSTER INDIVIDUAL, PENALTY, REWARD HORIZON = 100 | EPISODE (1000) | 500k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **132** | Jupyter | ACER | MlpLnLstmPolicy | FB 06 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No boost, no penalty, REWARD HORIZON = 100 | EPISODE (1) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **133** | Jupyter | ACER | MlpLnLstmPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No boost, no penalty, REWARD HORIZON = 100 | EPISODE (1) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **134** | Jupyter | ACER | MlpLnLstmPolicy | FB 18 | FB 19 | Normalizacion manual + VecNormalize | 1, la elegida | No boost, no penalty, REWARD HORIZON = 100 | EPISODE (1) | 100k | Spread Q 10 | Sí (50 Mult) | Q 10 | Infinito | |
| **135** | Jupyter | ACER | MlpLnLstmPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual + VecNormalize | 1, la elegida | BOOSTER INDIVIDUAL, no penalty, REWARD HORIZON = 100 | EPISODE (1000) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **136** | Jupyter | DQN | LnMlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual | 1, la elegida | No booster, no penalty, REWARD HORIZON = 100 | EPISODE (1000) | 100k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **137** | Jupyter | DQN | LnMlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual | 1, la elegida | No booster, no penalty, REWARD HORIZON = 100 | EPISODE (1000) | 500k | Spread NO Q | Sí (50 Mult) | NO Q | Infinito | |
| **138** | Jupyter | DQN | LnMlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual | 1, la elegida | No booster, no penalty, REWARD HORIZON = 100 | EPISODE (1000) | 1M | Spread NO Q | Sí (50 Mult) | NO Q | Infinito | |
| **139** | Jupyter | DQN | LnMlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual | 1, la elegida | No Booster, no penalty, REWARD HORIZON = 20 | EPISODE (1000) | 1M | Spread NO Q | Sí (50 Mult) | NO Q | Infinito | |
| **140** | Jupyter | DQN | LnMlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual | 1, la elegida | No Booster, no penalty, REWARD HORIZON = 1000 | EPISODE (1000) | 1M | Spread NO Q | Sí (50 Mult) | NO Q | Infinito | |
| **141** | Jupyter | DQN | LnMlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual | 1, la elegida | No Booster, no penalty, REWARD HORIZON = 50 | EPISODE (1000) | 1M | Spread NO Q | Sí (50 Mult) | NO Q | Infinito | |
| **142** | Jupyter | PPO2 | MlpPolicy | TOB 18+19+20+06 | FB 07 | Normalizacion manual + VecNormalize | 1, la elegida | No Booster, no penalty, REWARD HORIZON = 50 | EPISODE (1000) | 1M | Spread NO Q | Sí (50 Mult) | NO Q | Infinito | |
| **143** | Jupyter | DQN | LnMlpPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | 1, la elegida | No Booster, no penalty, REWARD HORIZON = 50 | EPISODE (1000) | 500k | Spread Q 10 | Si (simplified) (50 Mult) | Q 10 | Infinito | |
| **144** | Jupyter | DQN | LnMlpPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | 1, la elegida | No Booster, no penalty, REWARD HORIZON = 20 | EPISODE (100) | 1M | Spread NO Q | Si (simplified) (50 Mult) | NO Q | Infinito | |
| **145** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 1M | Spread NO Q | Si (simplified) (50 Mult) | NO Q | Infinito | |

| # | Platform | Model | Policy | Train data | Test Data | Normalizacion | Actions | Reward | Train Method | Train Cycles | Features de precios | Indicadores Técnicos | Market Imbalance Simplification | Max Bonds | Status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **146** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster,PENALTY 0.1, REWARD HORIZON = 20 | EPISODE (10) | 1M | Spread NO Q | Si (simplified) (50 Mult) | NO Q | Infinito | |
| **147** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (10) | 1M | Spread NO Q | Si (50 Mult) y SIN | NO Q | Infinito | |
| **148** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (10) | 600k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **149** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 100 | EPISODE (100) | 500k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **150** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD **HORIZON = 20** | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **151** | Jupyter | DQN | LnMlpPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **152** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+19+20+06 | TOB + FB 07 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 300k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **153** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+19+20+07 | TOB + FB 06 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **154** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+19+20+07 | TOB + FB 06 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **155** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+19+20+07 | TOB + FB 06 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD **HORIZON = 10** | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **156** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+20+07 | TOB + FB 06 | Normalizacion manual | 1, la elegida | No Booster,PENALTY, REWARD HORIZON = 10 | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **157** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+19+20+07 | TOB + FB 06 | Normalizacion manual | 1, la elegida | No Booster, **NO PENALTY**, REWARD HORIZON = 10 | EPISODE (100) | 100k | Spread NO Q + Bond QTY | Si (50 Mult) y SIN MACD_SIGN | NO Q | Infinito | |
| **158** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 1M | Spread NO Q + Bond QTY | Si (simplified) (50 Mult) | NO Q | Infinito | |
| **159** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 500k | Spread NO Q + Bond QTY | Si (simplified) (50 Mult) | NO Q | Infinito | |
| **160** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07+00 +01+02+03+04 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 300k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **161** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, PENALTY, REWARD HORIZON = 20 | EPISODE (100) | 300k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **162** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, PENALTY, REWARD HORIZON = 20 | EPISODE (1) | 300k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **163** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, **NO PENALTY**, REWARD HORIZON = 20 | EPISODE (5000) | 300k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **164** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, NO PENALTY, **REWARD HORIZON = 100** | EPISODE (5000) | 300k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **165** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, **REWARD HORIZON = 10** | EPISODE (5000) | 500k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **166** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 10 | EPISODE (5000) | 500k | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **167** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 10 | EPISODE (5000) | 3M | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **168** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | No Booster, NO PENALTY, REWARD HORIZON = 10, **NO** | EPISODE (100) | 1M | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **169** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 10 | EPISODE (5000) | 300k+1M | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **170** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 10 | EPISODE (100) | 1M | Spread Q 4 + Bond QTY | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **171** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 100 | EPISODE (1000) | 1M | **NO Spread** + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **172** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 100 | EPISODE (1000) | 1M | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **173** | Jupyter | PPO2 | MlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | 1, la elegida | Booster, PENALTY, REWARD HORIZON = 100 | EPISODE (1000) | 1M | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **174** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | **NO Booster,** PENALTY, REWARD HORIZON = 100, NO | EPISODE (1000) | 300k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **175** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | NO Booster, PENALTY, **REWARD HORIZON = 10** AND | EPISODE (1000) | 300k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **176** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | **SOLO AL FINAL DEL EPISODE**; NO Booster, PENALTY, REWARD | EPISODE (100) | 300k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **177** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | **INMEDIATA + AL FINAL DEL EPISODE**; NO Booster, PENALTY, | EPISODE (100) | 300k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **178** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | **AL FINAL DEL EPISODE**; NO Booster, PENALTY, REWARD HORIZON = | EPISODE (100) | 300k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **179** | Jupyter | PPO2 | MlpPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | **INMEDIATA**, NO Booster, **PENALTY = 1**, REWARD HORIZON = | EPISODE (1000) | 1M | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **180** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | **INMEDIATA**, NO Booster, **PENALTY = 1**, REWARD HORIZON = | EPISODE (1000) | 300k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **181** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 1M | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **182** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **183** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **184** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+20+06 | TOB + FB 07 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **185** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+20+07 | TOB + FB 06 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **186** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+06+07 | TOB + FB 20 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | NO Spread + Bond QTY + vs Future | Si (simplified) (50 Mult) | No Q | Infinito | |
| **187** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+06+07 | TOB + FB 20 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread Q 4 + Bond QTY + vs Future | Si (simplified) (50 Mult) | Q 4 | Infinito | |
| **188** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+06+07 | TOB + FB 20 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread No Q + Bond QTY + vs | Si (simplified) (50 Mult) | No Q | Infinito | |
| **189** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread No Q + Bond QTY + vs | Si (simplified) (50 Mult) | No Q | Infinito | |
| **190** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread No Q **- Bond** QTY + vs | Si (simplified) (50 Mult) | No Q | Infinito | |
| **191** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread No Q **- Bond** QTY + vs | Si (simplified) (50 Mult) | No Q | Infinito | |
| **192** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread No Q - Bond QTY + vs | Si (simplified) (50 Mult) | No Q | Infinito | |
| **193** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 19+20+06+07 | TOB + FB 18 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | Spread No Q - Bond QTY + vs | Si (simplified) (50 Mult) | No Q | Infinito | |
| **194** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | **NO SPREAD** + **Sí** (50 Mult) | No Q | Infinito | | |
| **195** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 | TOB + FB 19 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | **NO SPREAD** + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |

| # | Platform | Model | Policy | Train data | Test Data | Normalizacion | Actions | Reward | Train Method | Train Cycles | Features de precios | Indicadores Técnicos | Market Imbalance Simplification | Max Bonds | Status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **196** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+20+06+07 SUBSAMPLED | TOB + FB 19 SUBSAMPLED 100 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (1000) | 500k | SPREAD + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |
| **197** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18 reduced | TOB + FB 19 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 500k | SPREAD + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |
| **198** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18 reduced | TOB + FB 19 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 500k | SPREAD + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |
| **199** | Jupyter | DQN | LnMlpPolicy | TOB + FB 19 reduced | TOB + FB 18 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 500k | SPREAD + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |
| **200** | Jupyter | DQN | LnMlpPolicy | TOB + FB 19+20+06+07 reduced | TOB + FB 18 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 1.5M | SPREAD + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |
| **201** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 reduced | TOB + FB 19 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, **REWARD HORIZON =** | EPISODE (ENTRIES_RETAINED) | 1M | SPREAD + vs Future bid y ask | Sí (50 Mult) | No Q | Infinito | |
| **202** | Jupyter | DQN | LnMlpPolicy | TOB + FB 19 reduced | TOB + FB 18 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, **REWARD HORIZON =** | EPISODE (ENTRIES_RETAINED) | 1.5M | **SPREAD + vs Future bid y ask** | Sí (50 Mult) | No Q | Infinito | |
| **203** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18 reduced | TOB + FB 19 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 500k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **204** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18 reduced | TOB + FB 19 reduced | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, **REWARD HORIZON =** | EPISODE (ENTRIES_RETAINED) | 500k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **205** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 19 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | **1M (6h)** | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **206** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 19 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | **100k (25m)** | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **207** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 19 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | **10k (3m)** | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **208** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 reduced | TOB + FB 19 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | **50k (15m)** | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **209** | Jupyter | DQN | LnMlpPolicy | TOB + FB 18+20+06+07 reduced | TOB + FB 19 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | **500k (2h)** | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **210** | Jupyter | DQN | LnMlpPolicy | **TOB + FB 18+19+20+06 COMBINED** | **TOB + FB 07 reduced 1000** | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | **EPISODE (ENTRIES_RETAINED) FIXED DAYS** | 100k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **211** | Jupyter | ACER | MlpLnLstmPolicy | TOB + FB 18+19+20+06 COMBINED | TOB + FB 07 reduced 1000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 100k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **212** | Jupyter | **DQN Seed=1** | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 10k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **213** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18+19+06+07 COMBINED | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 50k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **214** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18+19 COMBINED reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 50k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **215** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18+19+06+07+01+02+03+04 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 100k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **216** | Jupyter | **DQN Seed=1 + Learning** | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 100k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **217** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, **REWARD HORIZON =** | EPISODE (ENTRIES_RETAINED) | 50k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **218** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 50k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **219** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18+19 COMBINED reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 200k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **220** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 **COMPLETO** | TOB + FB 20 **COMPLETO** | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE **(df completo)** | 500k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **221** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | **AL FINAL DEL EPISODE,** NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (ENTRIES_RETAINED) | 100k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **222** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 **SOLO 2000 muestras por** | TOB + FB 20 **COMPLETO** | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE **(df completo, pero 2000 por error)** | 200k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **223** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 **COMPLETO** | TOB + FB 20 **COMPLETO** | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, **REWARD HORIZON =** | EPISODE **(df completo)** | 500k | SPREAD | Sí (50 Mult) | No Q | Infinito | |
| **224** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 COMPLETO | TOB + FB 20 COMPLETO | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (df completo) | 500k | SPREAD + **BOND_QTY** | Sí (50 Mult) | No Q | Infinito | |
| **225** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 reduced 2000 | TOB + FB 20 reduced 2000 | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, **REWARD HORIZON =** | EPISODE (df completo) | 50k | SPREAD + BOND_QTY | Sí (50 Mult) | No Q | Infinito | |
| **226** | Jupyter | DQN Seed=1 | LnMlpPolicy | TOB + FB 18 **reduced 5000** | TOB + FB 20 **reduced 5000** | Normalizacion manual | Z, elegida + la closing action según PH | INMEDIATA, NO Booster, PENALTY = 1, REWARD HORIZON = | EPISODE (df completo) | 100k | SPREAD + BOND_QTY | Sí (50 Mult) | No Q | Infinito | |