INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE OCCIDENTE

SISTEMAS Y MICROESTRUCTURA DE TRADING

Advanced Trading Strategies: Deep Learning

Profesor: Mtro. Luis Felipe Gómez Estrada

PRESENTAN

Ana Luisa Espinoza López

Karimme Anahi Tejeda Olmos

Objetivos

El objetivo de este proyecto es desarrollar un sistema robusto de modelado predictivo aplicado a series de tiempo financieras, mediante la ingeniería de *features* derivadas de

indicadores técnicos en múltiples marcos temporales. Se busca diseñar y entrenar diversas arquitecturas de aprendizaje profundo, mediante la implementación de MLflow para el seguimiento de experimentos. Finalmente, se incorporará un monitoreo continuo del *data drift* en condiciones similares a producción, junto con el desarrollo de un sistema de *backtesting* que integre las predicciones del modelo considerando costos de transacción realistas.

Introducción

Descripción General de la Estrategia

Este informe documenta el diseño, la implementación y la evaluación de una estrategia cuantitativa de *trading* que utiliza Deep Learning (DL) para predecir el impulso de precios a corto plazo (compra, venta y mantener) en la acción Amazon (AMZN). Para la implementación de la estrategia, se construyeron dos arquitecturas de Aprendizaje Profundo: perceptrón multicapa (MLP) y red neuronal convolucional 1D (CNN), con el fin de capturar relaciones complejas y no lineales entre varios indicadores técnicos y el movimiento futuro del precio para clasificar el estado de mercado en tres acciones distintas: comprar, vender o mantener. Para el modelado y validación de los modelos de deep learning se realizó un data split cronológico para evitar leakage '*look ahead*' con 60% set de entrenamiento, 20% test y el 20% restante de validación.

Para capturar distintos aspectos de la información temporal y transversal de los datos, se optó por emplear múltiples arquitecturas de *aprendizaje profundo*. En primer lugar, se utilizó una MLP (Perceptrón Multicapa) como modelo base, enfocada en identificar relaciones transversales entre *features* (de momentum, volatilidad y volumen) correspondientes a un solo día. En segundo lugar, se implementó una CNN (Red Neuronal Convolucional) con el objetivo de detectar patrones locales y dependencias temporales dentro de una ventana de observación (*lookback window*), reconociendo que los movimientos del mercado tienden a seguir estructuras recurrentes en el tiempo.

Entre las ventajas esperadas destaca la capacidad del aprendizaje profundo para modelar interacciones no lineales con mayor precisión que los métodos tradicionales. No obstante, se identifican posibles limitaciones, como la sensibilidad al *data drift*, donde el rendimiento del modelo puede degradarse por cambios en la dinámica del mercado y la susceptibilidad al *overfitting* debido al ruido inherente de los datos históricos. Además, la rentabilidad de la

estrategia está condicionada por los costos de transacción y el *slippage*[1], factores que pueden reducir significativamente la efectividad de la predicción de las redes.

Metodología

Ingeniería de Características

Los datos de entrada se ampliaron con la introducción de 30 indicadores técnicos de momentum, volatilidad, análisis técnico y volumen para reflejar ampliamente las condiciones de mercado. Se utilizó un enfoque de normalización y escalado de los datos utilizando el método de Z-score, calculado con la media y desviación estándar del set de entrenamiento para prevenir *data leakage*. El cálculo de normalización con Z-score se realizó de la siguiente manera:

$$X_{norm} = (X - \mu_{train}) / \sigma_{train}$$

Ecuación 1. Normalización Z-score

Los indicadores seleccionados son los siguientes:

Tabla 1. Indicadores para ingeniería de características

Clasificación	Indicador	Descripción (Tipo Original)
Tendencia	MA_20	Media Móvil Simple (SMA)
	EMA_20	Media Móvil Exponencial (EMA)
	MACD	Convergencia/Divergencia del Promedio Móvil
	MACD_signal	Línea de Señal MACD
	ADX	Índice Direccional Promedio
	Parabolic_SAR	Stop and Reverse Parabólico
	Ichimoku_base	Línea Base de Ichimoku (Kijun-sen)
	Ichimoku_conversion	Línea de Conversión de Ichimoku (Tenkan-sen)
	SuperTrend	SuperTrend (Indicador de tendencia/volatilidad)

[1] *slippage*: diferencia entre el precio esperado de una operación y el precio real al que se ejecuta.

	Pivot_Point	Punto Pivote (Precio promedio)
Momentum	RSI	Índice de Fuerza Relativa
	ROC	Tasa de Cambio (Rate of Change)
	Stoch	Oscilador Estocástico
	CCI	Índice de Canal de Mercancías (CCI)
	Momentum	Cambio de precio en N períodos
	Williams_%R	Rango Porcentual de Williams
	AO	Oscilador Asombroso (Awesome Oscillator)
Volatilidad	ATR	Rango Verdadero Promedio (Average True Range)
	BB_high/low	Bandas de Bollinger (Alto/Bajo)
	KC_high/low	Canal Keltner (Alto/Bajo)
	Donchian_high/low	Canal Donchian (Alto/Bajo)
	Chaikin_Volatility	Volatilidad Chaikin
	Ulcer_Index	Índice de Úlcera (Medida de Drawdown)
Volumen	OBV	Balance en Volumen (On-Balance Volume)
	VROC	Tasa de Cambio de Volumen
	MFI	Índice de Flujo de Dinero (Money Flow Index)
	CMF	Flujo de Dinero Chaikin (Chaikin Money Flow)
	AD	Índice de Acumulación/Distribución (A/D)
	EOM	Facilidad de Movimiento (Ease of Movement)
	VWAP	Precio Promedio Ponderado por Volumen

Definición de Variable Objetivo

La estrategia está basada en clasificar la acción entre comprar, vender o mantener (1, -1, 0). Para el set de entrenamiento, se definió la variable objetivo (*y*) basado en el movimiento del precio respecto al día anterior. Se determinaron umbrales alcistas y bajistas para generar las etiquetas de señales de trading. La lógica para la asignación de clases para el set de train es la siguiente:

- **Umbral alcista:** 1.4%
- **Umbral bajista:** -1.4%
- **Retorno:** retorno de periodicidad diaria

Asignación de clases:

➔ Long (1): retorno ≥ umbral alcista

➔ Hold (0): retorno $\in [-1.4\%, 1.4\%]$

➔ Short (-1): retorno ≤ umbral bajista

La variable objetivo (signal) está desequilibrada (88.0% 'Hold'). Esto se aborda aplicando pesos de clase (class_weight="balanced") durante el entrenamiento del modelo para asegurar que las clases minoritarias ('Long', 'Short') no sean ignoradas.

Diseño y Arquitectura del Modelo

Perceptrón Multicapa (MLP)

El modelo MLP implementado constituye una arquitectura de red neuronal feedforward densamente conectada, diseñada específicamente para capturar relaciones no lineales complejas entre los 30 indicadores técnicos y la señal de trading objetivo. La arquitectura se construye mediante una secuencia de capas completamente conectadas (Dense layers), donde cada neurona en una capa está conectada a todas las neuronas de la capa siguiente, permitiendo al modelo aprender representaciones jerárquicas de los patrones de mercado.

La configuración óptima encontrada durante la experimentación consta de dos capas ocultas con 256 y 128 neuronas respectivamente. Esta estructura permite una compresión progresiva

de la información, donde la primera capa aprende representaciones de bajo nivel de los indicadores técnicos individuales, mientras que la segunda capa sintetiza estas representaciones en patrones de mercado de mayor abstracción. Cada capa oculta utiliza la función de activación ReLU (Rectified Linear Unit), que introduce no linealidad al modelo y permite capturar relaciones complejas entre variables, además de mitigar el problema del gradiente desvaneciente que afecta a funciones de activación tradicionales como la sigmoide.

Para prevenir el overfitting, un desafío crítico en el modelado de series financieras debido al alto ruido inherente, se implementaron múltiples mecanismos de regularización. Después de cada capa oculta se aplica Batch Normalization, que normaliza las activaciones y estabiliza el proceso de entrenamiento, permitiendo tasas de aprendizaje más altas y acelerando la convergencia. Inmediatamente después, se aplica Dropout con una probabilidad de 0.3, lo que significa que durante el entrenamiento se desactivan aleatoriamente el 30% de las neuronas en cada iteración, forzando al modelo a aprender representaciones redundantes y más robustas. Adicionalmente, se aplica regularización L2 con un coeficiente de 5e-4 sobre los pesos de las capas Dense, penalizando pesos grandes en la función de pérdida y promoviendo soluciones más generalizables.

La capa de salida consta de 3 neuronas con activación softmax, que transforma los logits de la red en una distribución de probabilidad sobre las tres clases (Short, Hold, Long), asegurando que las probabilidades sumen 1 y permitiendo interpretar la confianza del modelo en cada predicción. El modelo se entrena mediante el optimizador Adam con una tasa de aprendizaje de 1e-3, que combina las ventajas de momentum y RMSprop para lograr una convergencia eficiente. La función de pérdida utilizada es Sparse Categorical Crossentropy con label smoothing de 0.05, que suaviza las etiquetas one-hot hacia una distribución más uniforme, reduciendo la sobre-confianza del modelo y mejorando su calibración probabilística.

Red Neuronal Convolucional 1D (CNN)

La arquitectura CNN se diseñó para explotar la estructura temporal inherente en las series financieras, reconociendo que los movimientos de precios no son eventos aislados sino que forman patrones secuenciales. A diferencia del MLP que procesa observaciones individuales, la CNN opera sobre ventanas temporales (lookback windows) de 100 días consecutivos, permitiendo detectar patrones locales y dependencias temporales que pueden ser predictivas del movimiento futuro del precio.

El input del modelo consiste en secuencias tridimensionales de forma (batch_size, lookback, n_features), donde cada muestra contiene 100 días de historia con indicadores técnicos por día. Esta representación permite que los filtros convolucionales actúen como detectores de patrones temporales, deslizándose a lo largo de la dimensión temporal para identificar estructuras recurrentes independientemente de su posición exacta en la secuencia.

La primera capa convolucional aplica 128 filtros de tamaño kernel 5, lo que significa que cada filtro examina simultáneamente 5 días consecutivos de los 30 indicadores. Esta operación genera 128 mapas de características (feature maps), cada uno capturando un patrón temporal específico, como tendencias alcistas graduales, reversiones bruscas, o formaciones de consolidación. El uso de padding='same' asegura que la longitud de la secuencia se preserve después de la convolución. La activación ReLU se aplica elemento por elemento, introduciendo no linealidad, seguida por Batch Normalization para estabilizar las activaciones. Posteriormente, una capa de MaxPooling1D con tamaño 2 reduce la resolución temporal a la mitad, extrayendo las características más salientes y reduciendo la carga computacional.

La segunda capa convolucional aplica 256 filtros de kernel 5 sobre los feature maps producidos por la primera capa, permitiendo al modelo aprender representaciones jerárquicas más abstractas que combinan los patrones detectados previamente. Por ejemplo, mientras la primera capa podría detectar cruces de medias móviles individuales, la segunda capa podría identificar patrones multi-indicador como divergencias MACD acompañadas de sobrecompra en RSI. Esta capa también incluye Batch Normalization para mantener la estabilidad del entrenamiento.

En lugar de flatten las características espaciales, se utiliza Global Average Pooling 1D, que calcula el promedio de cada feature map a lo largo de la dimensión temporal, produciendo un vector de 256 características. Este enfoque es más robusto que flatten porque reduce drásticamente el número de parámetros en las capas subsecuentes y actúa como una forma de regularización estructural, forzando a cada filtro convolucional a especializarse en detectar un patrón específico presente en cualquier parte de la secuencia.

Las características extraídas pasan por una capa de Dropout (0.3) y luego a una capa Dense de 256 neuronas con activación ReLU, que integra la información temporal para formar representaciones de alto nivel del estado del mercado. Finalmente, la capa de salida con 3 neuronas y activación softmax produce las probabilidades de clase. El modelo se entrena con el optimizador Adam usando una tasa de aprendizaje más conservadora de 5e-4, dado que las CNNs pueden ser más sensibles a tasas de aprendizaje altas debido al mayor número de parámetros y la complejidad de las representaciones aprendidas.

Estrategia de Entrenamiento

El entrenamiento de ambos modelos se realizó durante un máximo de 50 épocas, aunque en la práctica convergieron significativamente antes gracias a los mecanismos de control implementados. Se utilizó un tamaño de batch de 256 muestras, que ofrece un balance entre estabilidad del gradiente y eficiencia computacional, permitiendo actualizaciones de pesos más frecuentes que batch sizes más grandes mientras mantiene suficientes muestras para estimaciones robustas del gradiente.

El callback de Early Stopping monitorea la accuracy de validación con una patience de 5 épocas, deteniendo el entrenamiento si no se observa mejora durante ese período y restaurando los pesos del mejor checkpoint. Este mecanismo previene el overfitting al evitar que el modelo continúe especializándose en el conjunto de entrenamiento a costa de la capacidad de generalización. Complementariamente, el ReduceLROnPlateau reduce la tasa de aprendizaje en un factor de 0.5 cuando el val_loss se estanca durante 3 épocas, permitiendo al optimizador realizar ajustes más finos cerca de los mínimos locales y potencialmente escapar de mesetas durante el entrenamiento.

Para abordar el severo desbalance de clases en la variable objetivo (88% Hold, 6% Long, 6% Short), se calculan pesos de clase inversamente proporcionales a su frecuencia usando class_weight='balanced'. Esto escala la contribución de cada muestra a la función de pérdida, asegurando que el modelo no aprenda simplemente a predecir siempre la clase mayoritaria (Hold) sino que preste atención a los eventos más raros pero financieramente más significativos (Long y Short). Matemáticamente, el peso para cada clase c se calcula como: w_c = n_samples / (n_classes × n_samples_c), amplificando la señal de las clases minoritarias durante el backpropagation.

Seguimiento de Experimentos con MLflow

La experimentación sistemática y el tracking riguroso de hiperparámetros son fundamentales para el desarrollo de modelos de machine learning reproducibles y para comprender qué configuraciones producen mejores resultados. MLflow se implementó como plataforma de gestión del ciclo de vida de los modelos, permitiendo registrar automáticamente cada experimento con todos sus artefactos asociados.

Se definió un espacio de búsqueda de hiperparámetros explorando múltiples configuraciones arquitectónicas. Para el MLP, se experimentó con tres configuraciones: arquitecturas con [256, 128] neuronas y dropout 0.2, [512, 256] con dropout 0.3 y label smoothing 0.05, y una

configuración más profunda [256, 256, 128] con mayor regularización L2. Para las CNNs, se variaron tres lookback windows diferentes (30, 60, 100 días), permitiendo evaluar cuánta historia temporal necesita el modelo para realizar predicciones efectivas, junto con diferentes combinaciones de filtros y kernel sizes.

Cada ejecución de entrenamiento se registra automáticamente en MLflow mediante mlflow.autolog(), capturando no solo los hiper parámetros sino también las métricas de entrenamiento y validación en cada época, los pesos del modelo final, y los artefactos generados como matrices de confusión y classification reports. Esto crea un registro completo y auditable de todos los experimentos, permitiendo comparaciones directas entre configuraciones y facilitando la reproducibilidad de resultados.

La selección del modelo final se basó en el F1-score macro en el conjunto de validación, métrica que balancea precisión y recall considerando todas las clases por igual, siendo más apropiada que accuracy para problemas con desbalance de clases. El modelo MLP con configuración [512, 256], dropout 0.3 y label smoothing 0.05 logró el mejor F1-score de validación (0.59), demostrando capacidad superior para distinguir entre las tres clases de trading sin mostrar signos de overfitting severo.

Metodología de Backtesting

El sistema de backtesting implementado simula la ejecución real de la estrategia de trading sobre datos históricos, permitiendo evaluar el desempeño económico del modelo más allá de métricas estadísticas tradicionales. A diferencia de la evaluación puramente clasificatoria que mide F1-score o accuracy, el backtesting considera la rentabilidad real después de costos de transacción, slippage y la lógica específica de entrada y salida de posiciones.

La estrategia opera con una posición fija de 100 acciones por trade, una simplificación que facilita el análisis inicial y permite aislar el efecto de las señales del modelo sin la complejidad adicional del position sizing dinámico. En un sistema de producción, el tamaño de posición idealmente se ajustaría según la volatilidad actual y el capital disponible, pero para propósitos de evaluación este approach constante proporciona comparabilidad directa entre diferentes modelos y períodos.

Generación de Señales

El proceso de traducción de las predicciones del modelo en señales ejecutables es crítico para el desempeño de la estrategia. El modelo genera tres probabilidades por cada día: p_short, p_hold y p_long, representando la confianza del modelo en cada acción. La clase con mayor probabilidad se selecciona como la predicción base, pero crucialmente, esta predicción solo se ejecuta si la probabilidad máxima supera un umbral de confianza (threshold) de 0.5. Este mecanismo de filtrado previene que el modelo opere en situaciones de alta incertidumbre, donde la diferencia entre las probabilidades de las tres clases es pequeña y la decisión correcta es ambigua.

Cuando max(p_short, p_hold, p_long) ≤ 0.5, el sistema automáticamente genera una señal de HOLD independientemente de cuál clase tenía la probabilidad mayor, evitando trades de baja convicción que históricamente resultan en win rates pobres. Este threshold actúa como un control de calidad sobre las señales, priorizando precision sobre recall en el contexto de trading donde trades innecesarios incurren en costos tangibles.

Las predicciones numéricas del modelo (0, 1, 2) se mapean inversamente a las señales de trading originales: clase 0 corresponde a Short (-1), clase 1 a Hold (0), y clase 2 a Long (+1). Este mapeo preserva la interpretación económica de las señales durante todo el pipeline de predicción y ejecución.

Tabla 2. Parámetros de Backtest

Parámetro	Valor	Justificación
Posición Fija	100 shares	Simplifica análisis
Stop Loss	2.0%	Limita pérdidas
Take Profit	4.0%	Risk:Reward 1:2
Comisiones	0.125% RT	Comisión realista
Borrow Rate	2.5% anual	Costo shorts

La implementación de stop-loss y take-profit automáticos es esencial para controlar el riesgo de cada operación individual. El stop-loss del 2.0% establece la pérdida máxima aceptable por trade, cerrando automáticamente la posición si el precio se mueve en contra de la dirección anticipada más allá de este umbral. Este mecanismo limita la exposición al riesgo de cola y previene que trades perdedores individuales erosionen significativamente el capital.

El take-profit del 4.0% captura ganancias cuando el precio alcanza el objetivo establecido, implementando efectivamente un ratio risk-reward de 1:2. Esta asimetría es fundamental en sistemas de trading algorítmico porque permite mantener rentabilidad positiva incluso con win rates modestos alrededor del 40-45%. Matemáticamente, si el sistema gana 4% en trades exitosos y pierde 2% en trades fallidos, solo necesita una tasa de acierto superior al 33.3% para ser rentable antes de costos.

Ambos niveles (stop y take-profit) se calculan en base al precio de entrada y se evalúan durante la jornada de trading usando los precios High y Low del día. Si el Low del día alcanza el stop-loss, se asume ejecución a ese nivel; si el High toca el take-profit, se asume salida en ese precio. Esta simulación es conservadora porque asume que el stop siempre se ejecuta al precio especificado, ignorando gap downs potenciales o slippage adverso que ocurrirían en mercados reales.

Se implementa una comisión de 0.125% por lado (0.25% round-trip), que aunque puede parecer pequeña, se acumula significativamente en estrategias que generan múltiples trades. En un trade de $10,000, esto representa $25 en costos, que deben ser superados por el movimiento de precio favorable para que el trade sea rentable.

Para posiciones short, adicionalmente se modela un borrow rate anual de 2.5%, el costo de tomar prestadas las acciones para venderlas en corto. Este costo se prorratea diariamente según la duración de la posición, añadiendo una penalización económica realista que favorece ligeramente las posiciones long sobre short, todo lo demás siendo igual. En la práctica, este costo puede variar significativamente según la disponibilidad de acciones para préstamo y la demanda por shortear el título específico.

La estrategia opera con un paradigma de "flat overnight", significando que todas las posiciones se cierran al final del día de trading, evitando exposición al riesgo de gaps overnight. Las señales se generan al cierre del día t-1 basándose en la información disponible hasta ese momento, y la entrada efectiva ocurre al precio de cierre de ese día. La posición luego se monitorea durante el día t, evaluando continuamente los niveles de stop-loss y take-profit contra los precios intradiarios (aproximados por High/Low), y se cierra obligatoriamente al cierre del día t si no se activó ningún nivel de salida durante la sesión.

Esta aproximación simplifica la implementación del backtest porque evita la necesidad de datos de tick-by-tick, pero introduce ciertas idealizaciones. En particular, asume que podemos entrar exactamente al precio de cierre de t-1, lo cual en realidad requeriría ejecutar una orden market-on-close, y que podemos detectar y actuar sobre movimientos intradiarios usando solo los precios High y Low del día.

Walk-Forward Backtesting

El esquema walk-forward representa una metodología de validación más rigurosa que el simple train-test split, simulando más fielmente cómo se desplegaría el modelo en producción donde periódicamente se re entrena con datos frescos. El proceso comienza entrenando el modelo sobre 60% de los datos históricos más antiguos, validando en el 20% siguiente, y finalmente evaluando en el 20% más reciente como conjunto de test verdaderamente out-of-sample.

La innovación clave del walk-forward es el reentrenamiento trimestral con una ventana deslizante. Cada tres meses, se reentrena el modelo usando los últimos 18-24 meses de datos (la "ventana de entrenamiento"), se valida en el trimestre siguiente, y se despliega para predecir el trimestre subsecuente. Esta ventana se desplaza hacia adelante cronológicamente, asegurando que el modelo siempre opere con parámetros calibrados a la dinámica de mercado más reciente.

Este enfoque mitiga el data drift, el fenómeno donde la relación entre features y target cambia con el tiempo debido a cambios en régimen de mercado, estructura de

participantes, o condiciones macroeconómicas. Un modelo entrenado únicamente en 2020-2022 puede tener features cuya importancia predictiva ha cambiado para 2024-2025, pero el reentrenamiento regular permite al modelo adaptarse a estos shifts. Sin embargo, el walk-forward también introduce el riesgo de overfitting a los datos de reentrenamiento si los hiper parámetros se optimizan excesivamente en cada ventana, por lo cual es crítico mantener hiper parámetros estables y sólo re estimar los pesos de la red.

Análisis de Resultados

Curvas de Equity

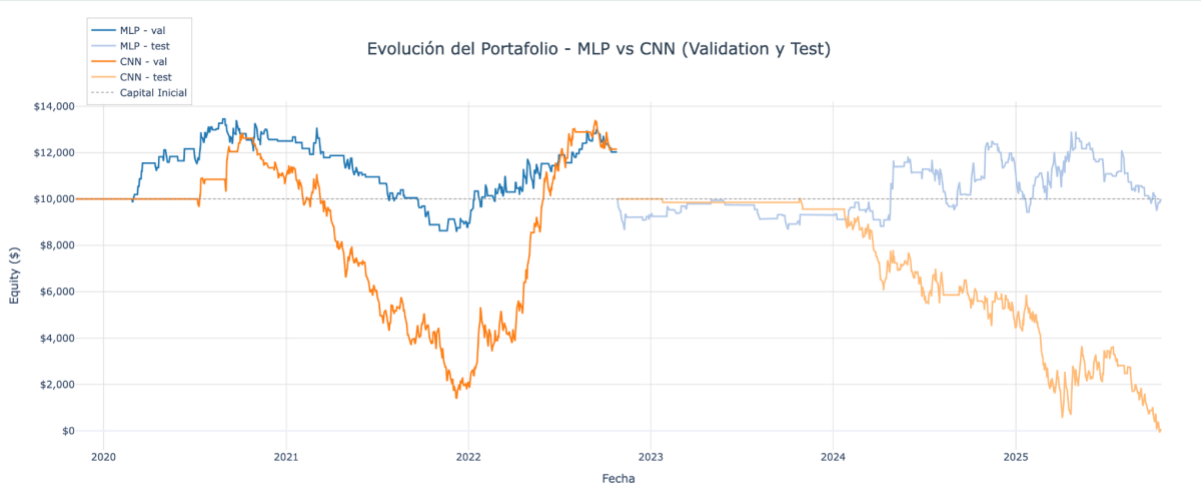

Figura 1. Curvas de Equity en Validation y Test

La figura 1 muestra los resultados detallados del *backtesting* para las arquitecturas MLP y CNN en los tres períodos: Train (datos de entrenamiento), Val (validación/optimización de parámetros) y Test (período no visto). A simple vista, se puede observar que en ambos periodos la arquitectura con MLP genera mejores resultados en el periodo de validación, los cuales se discutirán a continuación.

Métricas de Desempeño

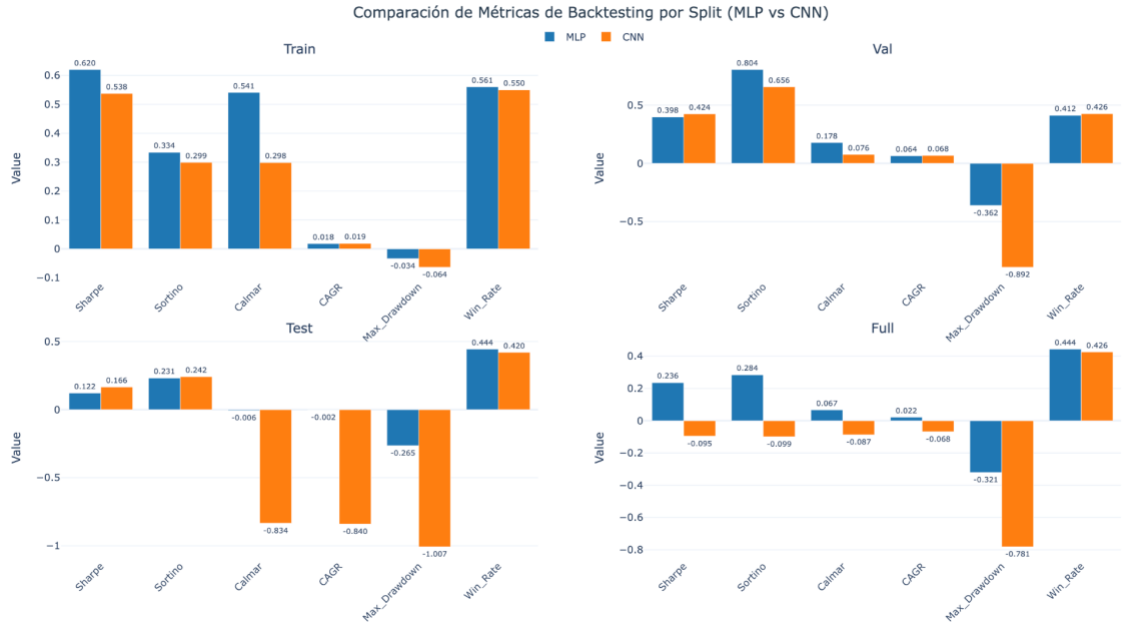

Figura 2. Métricas de desempeño para distintos splits temporales

Tabla 3. Resumen de Métricas de desempeño

Métrica	MLP (Train)	MLP (Val)	MLP (Test)	CNN (Train)	CNN (Val)	CNN (Test)
Sharpe Ratio	0.62	0.398	0.122	0.538	0.424	0.166
Calmar Ratio	0.541	0.178	-0.006	0.298	0.076	-0.834
Max Drawdown	-0.034	-0.362	-0.265	-0.064	-0.892	-1.007
CAGR (%)	0.018	0.064	-0.002	0.019	0.068	-0.84
Win Rate	0.561	0.412	0.444	0.55	0.426	0.42
Sortino	0.334	0.804	0.231	0.299	0.656	0.242

El análisis de rendimiento se fundamenta en la evaluación de los modelos tras la optimización de los parámetros de *backtesting*. Los resultados obtenidos son críticos y señalan problemas de fondo en la viabilidad de la estrategia. La tabla adjunta resume el rendimiento de las arquitecturas MLP y CNN en los tres períodos, destacando el colapso de las métricas de riesgo-retorno en los sets de Validación y Test, un comportamiento que es la principal evidencia de que los modelos sobre ajustaron el ruido del mercado durante el entrenamiento, sin lograr una generalización efectiva de las señales de alto valor.

El rendimiento de la estrategia en el set de Test es sumamente pobre y revela una falta de viabilidad operativa. El CAGR es negativo para ambas arquitecturas (MLP: -0.002, CNN: -0.84). Esto indica que la estrategia destruye valor anualmente en el período de prueba, ya que las pérdidas netas superan consistentemente las ganancias después de considerar los costos de transacción y de *borrow*. Los ratios

de riesgo ajustado refuerzan esta conclusión, con un Sharpe Ratio marginalmente positivo para ambos (MLP:0.122, CNN: 0.166), y un Calmar Ratio esencialmente nulo (MLP:-0.006) o claramente negativo (CNN: -0.834). Esto certifica que la estrategia no solo pierde dinero, sino que lo hace con alta inestabilidad.

La gestión de riesgo es la métrica más crítica y revela un fallo categórico en la estrategia de Stop Loss bajo condiciones de *Data Drift*. El Máximo Drawdown (Max DD) alcanza un valor de -1.007 para el CNN en Test. Un *Drawdown* superior al 100% implica un fallo total en el mecanismo de gestión de riesgo, lo que lleva a la destrucción del capital inicial durante la simulación de *backtesting*. Aunque el MLP es la arquitectura menos perjudicial (Max DD de -0.265), su Calmar Ratio de -0.006 indica que el retorno es insignificante en comparación con el riesgo asumido. La CNN es la única arquitectura que mostró una mejora en el Sharpe Ratio de Val a Test (0.424 a 0.166), pero esta ganancia de eficiencia se anula por su incapacidad de controlar el Max DD.

La Gráfica de Evolución del Portafolio valida visualmente estos fallos en el período Test. La curva de *equity* del CNN en Test se desploma abruptamente hasta cero, mientras que la del MLP en Test se estanca, reflejando sus respectivos CAGR's negativos y la volatilidad incontrolada. La estrategia no es viable en su estado actual, ya que ninguna arquitectura genera retornos ajustados al riesgo positivos en el período de Test.

Monitoreo de Data Drift

El data drift constituye una amenaza crítica para la estabilidad de los sistemas de *trading algorítmico*, ya que las relaciones estadísticas aprendidas por el modelo pueden deteriorarse o invertirse conforme cambian las condiciones del mercado. Para anticipar estos cambios, se implementa un monitoreo continuo que evalúa la estabilidad estadística de las variables a lo largo del tiempo.

El análisis utiliza el test de Kolmogorov–Smirnov (K-S), que compara la distribución actual de cada *feature* con la histórica del entrenamiento. Un p-value menor a 0.05 indica que la diferencia es significativa y, por tanto, existe *drift*. Por ejemplo, un

desplazamiento del RSI hacia valores más altos puede evidenciar un cambio de régimen de mercado.

Cuando se detecta *drift* en variables críticas, el sistema genera alertas que permiten decidir si se requiere reentrenar el modelo o ajustar parámetros. Además, se monitorean las distribuciones de las predicciones: un sesgo hacia la clase “Hold” puede señalar pérdida de poder discriminativo, mientras que una polarización extrema puede reflejar sobreconfianza del modelo.

El análisis también revisa la estabilidad de métricas de desempeño como el F1-score o el Sharpe ratio en ventanas móviles. Una degradación sostenida de estas métricas puede indicar que las condiciones del mercado —volatilidad, correlaciones o estructura de tendencias— han cambiado.

Finalmente, la estrategia de monitoreo busca un equilibrio entre sensibilidad y estabilidad: se utilizan ventanas de 30 días y umbrales de p-value que exigen que múltiples variables presenten *drift* antes de generar alertas críticas, evitando reentrenamientos innecesarios por ruido a corto plazo pero manteniendo la capacidad de detectar cambios estructurales.

Hallazgos

El presente análisis de data drift revela cambios distribucionales significativos en el 92.5% de las características del modelo predictivo, indicando una degradación sustancial en su capacidad de generalización. Mediante la aplicación del test de Kolmogorov-Smirnov, se identificaron 37 de 40 características con desviaciones estadísticamente significativas, siendo los indicadores de volumen y volatilidad los más afectados. El análisis se realizó con un enfoque estadístico con KS, método no paramétrico para comparar distribuciones acumulativas con un nivel de significancia (α) de 5%, realizando comparaciones entre los conjuntos: entrenamiento vs validación y validación vs test.

Criterios de evaluación:

- ➔ Valor p < α: el drift es estadísticamente significativo

➔ Estadístico KS: Indica la máxima diferencia entre distribuciones (0: idénticas, 1: completamente diferentes)

La cantidad de datos analizada consta de 40 características financieras técnicas con un volumen de datos de 2242, 748 y 748 observaciones en los conjuntos de entrenamiento, validación y test, respectivamente. Utilizando los criterios de evaluación se encontró un drift significativo en 37 características y 3 estables. Las 5 características con mayor drift significativo son:

Tabla 4. Top 5 de características con drift significativo

Feature	KS	P_Value	Interpretación
OBV	1.0000	0.0	Cambio en volumen o liquidez del mercado.
AD	0.9969	0.0	Cambio en volumen o liquidez del mercado.
ATR	0.9710	0.0	Cambio de régimen de volatilidad.
Donchian_high	0.9416	0.0	Cambio de régimen de volatilidad.
Ichimoku_base	0.9412	0.0	Cambio en tendencia o estructura de precios.

Para el top 5 de características con mayor drift se realizaron objetos visuales para observar los cambios en la distribución así como la evolución temporal.

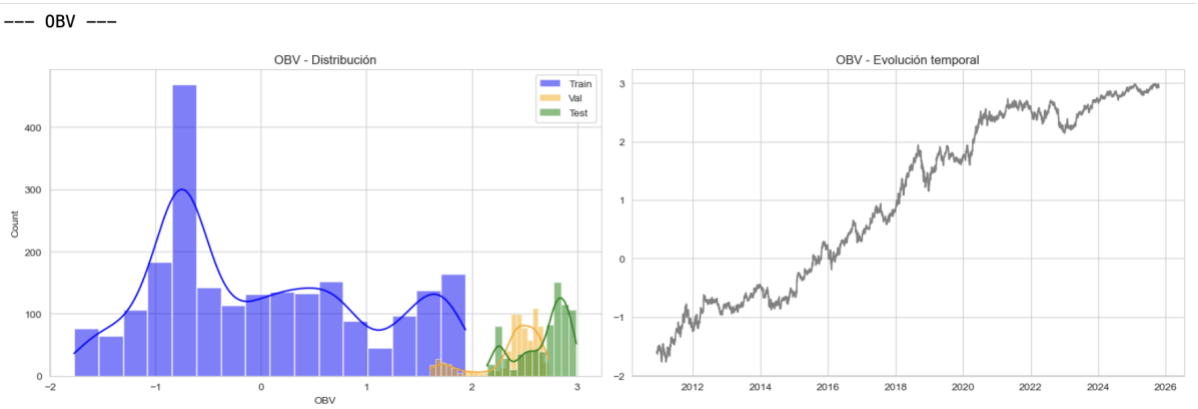

Figura 3. Distribución y evolución: OBV

El valor de KS = 1, indica una ruptura estructural en los patrones de volumen del mercado. La evolución temporal muestra cambios fundamentales en la participación y liquidez.

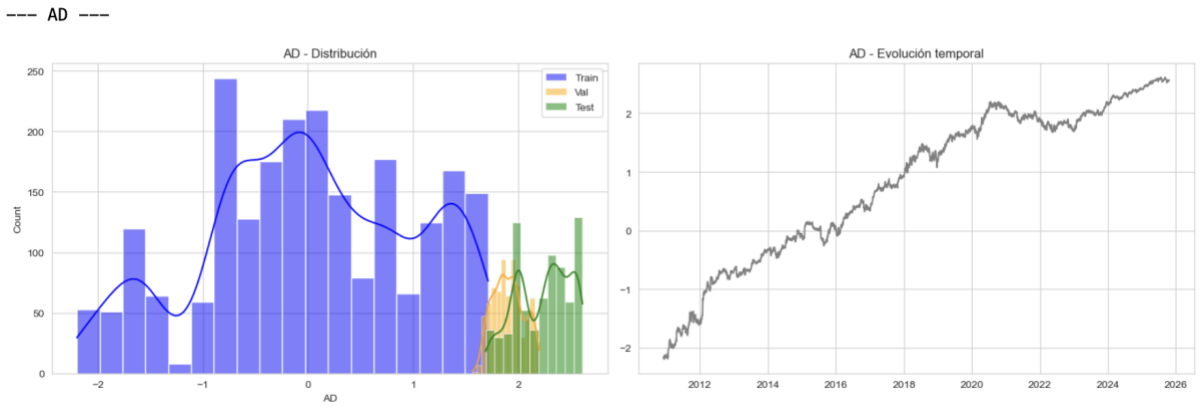

Figura 4. Distribución y evolución: AD

Para el feature AD (Accumulation-Distribution) se muestra Comportamiento alterado en los flujos de capital institucional (KS = 0.9969), sugiriendo modificaciones en las estrategias de inversión de grandes actores del mercado.

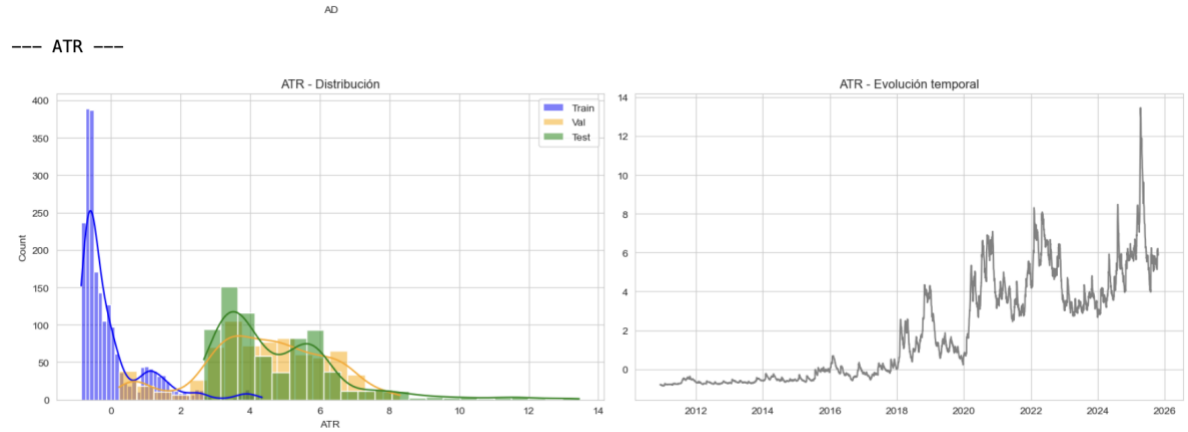

Figura 5. Distribución y evolución: ATR

El feature ATR (Average True Range) muestra un cambio significativo en el régimen de volatilidad (KS = 0.9710), posible transición entre periodos de alta y baja volatilidad que afecta estrategias basadas en rangos.

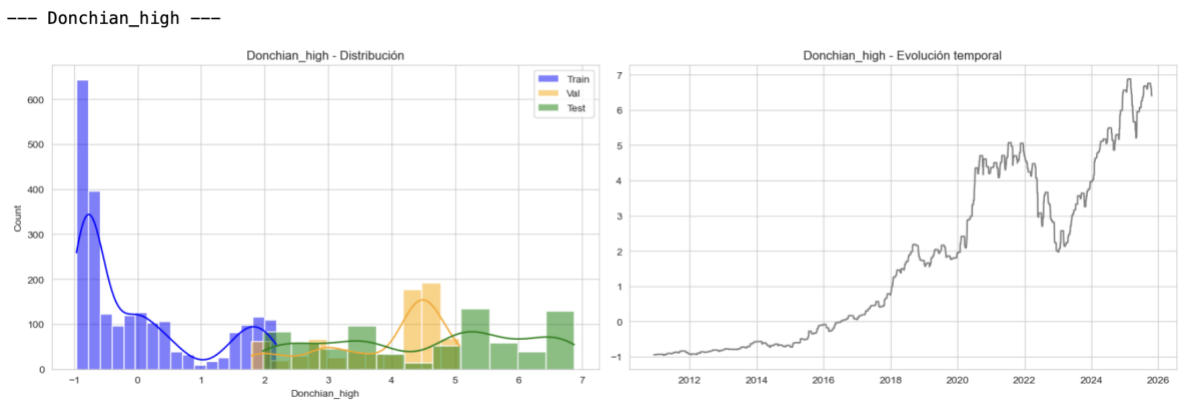

Figura 6. Distribución y evolución: Donchian high

El feature de Donchian high sugiere alteración en los patrones de ruptura de canales (KS = 0.9416), impactando estrategias de seguimiento de tendencia e identificación de máximos históricos.

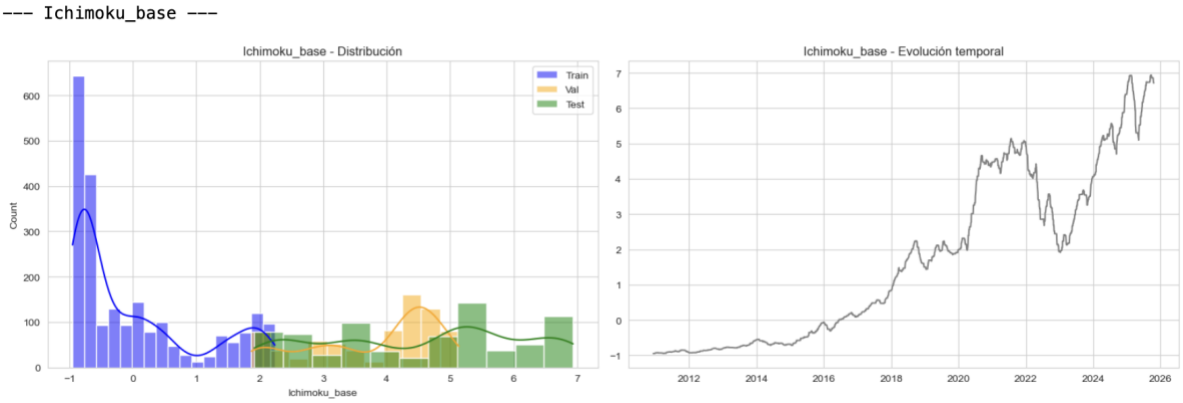

Figura 7. Distribución y evolución: Ichimoku base

Finalmente, el feature de Ichimoku base sugiere modificación en la estructura de tendencias subyacentes (KS = 0.9412), indicando cambios en la dinámica direccional del mercado.

De igual manera, en la siguiente figura se observan las 3 principales características por categoría con drift significativo:

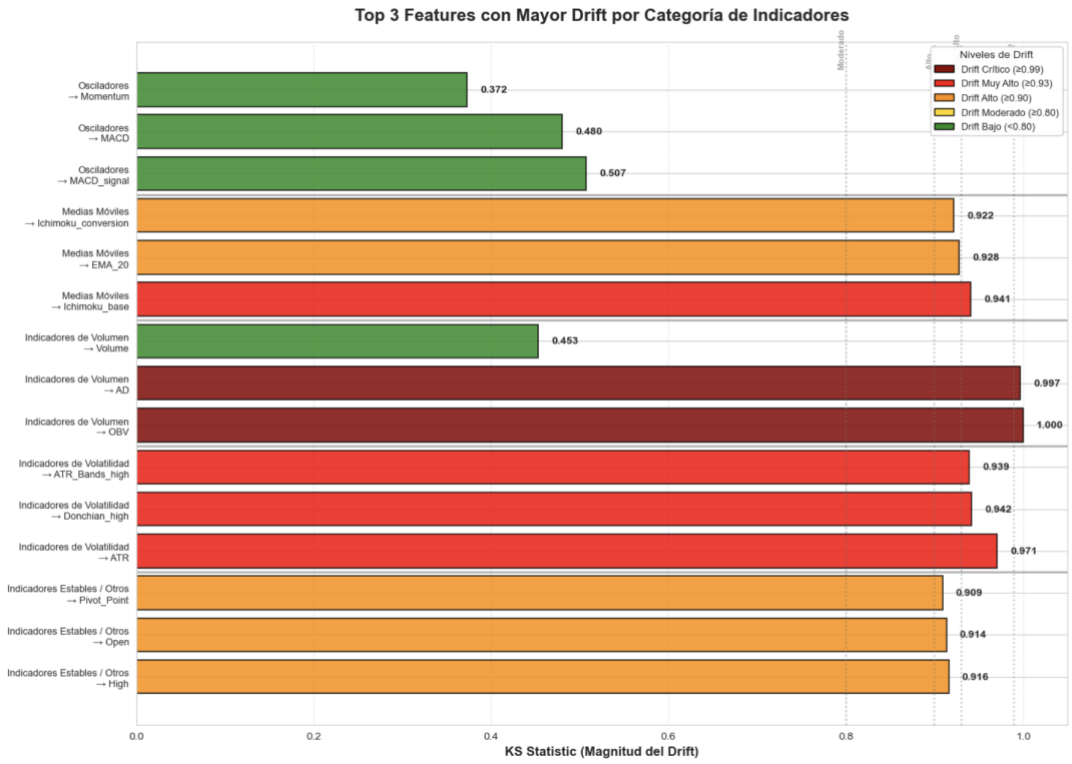

Figura 8. Features con mayor drift por categoría de indicadores.

Los hallazgos del análisis revelan implicaciones críticas para la validez operativa del modelo predictivo. La presencia de data drift significativo en el 92.5% de las características indica una degradación sustancial en la capacidad de generalización del modelo, traduciéndose en una probable reducción de su rendimiento predictivo y confiabilidad en escenarios reales. Esta situación genera riesgo operacional al basar decisiones estratégicas en relaciones históricas que han perdido validez en el contexto de mercado actual. Los patrones identificados sugieren un cambio de régimen estructural en el entorno financiero, caracterizado por alteraciones fundamentales en los patrones de volumen, liquidez y volatilidad, lo que hace que las inferencias del modelo basadas en comportamientos pasados sean potencialmente erróneas o misleading en el nuevo contexto de mercado.

Frente a estos resultados, la recomendación se centra en re entrenar el modelo con condiciones del mercado actuales. Dentro de las posibles áreas de mejora se recomienda establecer mejoras de proceso mediante la implementación de un sistema de monitoreo continuo de data drift con umbrales de alerta automáticos,

junto con el desarrollo de un pipeline de retraining que permita adaptaciones ágiles frente a cambios distribucionales.

Mejoras en la Arquitectura

La arquitectura MLP, aunque eficiente en términos computacionales, tiene la limitación de analizar cada día de trading de forma aislada, sin considerar las relaciones temporales entre días consecutivos. Para mejorarla, sería valioso incorporar mecanismos de atención que permitan al modelo identificar automáticamente qué indicadores técnicos son más relevantes en cada contexto de mercado. También se podría optimizar la estructura de capas mediante búsquedas automatizadas de arquitecturas, e implementar conexiones residuales que faciliten el entrenamiento de redes más profundas y estables.

En el caso de la CNN 1D, aunque captura patrones locales dentro de la ventana temporal, tiene dificultades para reconocer dependencias de largo plazo. Una dirección prometedora sería desarrollar arquitecturas híbridas que combinen las capas convolucionales con capas recurrentes o mecanismos de auto-atención, permitiendo integrar tanto patrones de corto plazo como tendencias extendidas. También se podría experimentar con convoluciones dilatadas para ampliar el horizonte temporal sin aumentar excesivamente la complejidad, y diseñar ventanas de lookback adaptativas que se ajusten a la volatilidad del mercado.

Ambos enfoques se beneficiarían de incorporar técnicas de regularización más avanzadas específicas para series financieras, así como funciones de pérdida personalizadas que incluyan métricas de riesgo financiero directamente en el proceso de optimización. Finalmente, implementar un sistema ensemble dinámico que combine estratégicamente las fortalezas de ambas arquitecturas (ponderando sus predicciones según las condiciones actuales de mercado) podría generar un sistema más robusto y adaptable ante cambios en la dinámica del mercado.

Conclusiones

En conclusión, este proyecto logró desarrollar un sistema integral de modelado predictivo que cumplió con los objetivos establecidos: se implementó con éxito un pipeline robusto de ingeniería de features utilizando más de 30 indicadores técnicos, se diseñaron y compararon sistemáticamente arquitecturas de deep learning (MLP y CNN) mediante MLflow para seguimiento experimental, y se estableció un framework de backtesting que incorporó costos de transacción realistas y gestión de riesgo. Sin embargo, los resultados operativos evidenciaron limitaciones significativas en la capacidad de generalización, principalmente atribuibles al data drift masivo detectado en el 92.5% de las features, lo que comprometió la viabilidad financiera de la estrategia.

El valor principal del proyecto reside no en el rendimiento financiero obtenido, sino en la creación de una infraestructura metodológica sólida con sistemas de monitoreo, validación walk-forward y gestión de experimentos, que constituye una base esencial para futuras iteraciones que incorporen mecanismos de adaptación en tiempo real y arquitecturas más avanzadas para enfrentar los desafíos de los mercados financieros dinámicos.

El proyecto demostró que, a pesar de implementar arquitecturas avanzadas de Deep Learning (MLP y CNN) junto con técnicas rigurosas de regularización y balanceo de clases, los modelos presentaron una capacidad de generalización limitada en datos fuera de muestra. Métricas de desempeño como el CAGR negativo y drawdowns significativos en el conjunto de test , especialmente crítico para la CNN, con un máximo drawdown que superó el 100%, evidenciaron un sobreajuste a patrones históricos no persistentes. Esto resalta la inherente dificultad de capturar relaciones estables en series financieras ruidosas, incluso con un diseño robusto de ingeniería de características y entrenamiento.

En conclusión, la viabilidad operativa de la estrategia se ve comprometida por la combinación de overfitting, data drift no mitigado y una gestión de riesgo insuficiente ante condiciones de mercado cambiantes. Futuras iteraciones deberían integrar un pipeline de reentrenamiento automático con ventanas deslizantes, explorar

arquitecturas temporales más sofisticadas (como LSTM) y reforzar el monitoreo proactivo de drift para mejorar la robustez y adaptabilidad del sistema en contextos financieros reales.

Notas:

Se tuvieron problemas al configurar un servidor de MLflow externo, el cual crea automáticamente una carpeta llamada **mlruns/** dentro del entorno local donde se ejecuta el código.

Por lo tanto, cada computadora crea su propio historial de ejecuciones y su propia copia del "mejor modelo", basada únicamente en lo que esa computadora registró al entrenar; y al correr puede no ejecutar el modelo que ase adjuntó en el reporte.

Como mejoras proponemos configurar el modelo para que no existan problemas de rutas ni de carpeta mlruns y pueda aparece el mismo modelo al correr el código en diferentes computadoras.