

## **Entrega Final. Deep Learning**

Andrea Sánchez Castrillón  
Alejandro Vargas Ocampo



Universidad de Antioquia  
Facultad de Ingeniería  
2025

## **1. Estructura de los notebooks entregados.**

### **01\_exploración\_de\_datos:**

- Tipos de datos
- Tamaño en disco
- Distribución de clases
- Distribución de volumen
- Evolución y autocorrelación del precio de cierre

### **02 - preprocesado:**

- Enfoque binario
- Enfoque cálculo precio de la acción

### **03 - arquitectura de linea de base**

Enfoque binario:

- CNN

Enfoque en el precio de la acción:

- Time Series Transformer
- CNN 1D
- CNN-LSTM (Arquitectura Híbrida)

## **2. Descripción de la solución (arquitectura, preprocesado, etc).**

### **2.1 Preprocesado**

El primer bloque de preprocesamiento (orientado a clasificación binaria) comenzó con la estandarización del formato temporal y la ordenación cronológica del dataset. Posteriormente, se construyó la variable objetivo mediante un desplazamiento de un día en la serie de precios, definiendo el target como:

$$\text{target}_t = \mathbf{1}\{\text{close}_{t+1} > \text{close}_t\}$$

lo cual genera una etiqueta binaria que indica si el precio de cierre del día siguiente presenta un incremento respecto al día actual.

A continuación, se realizó una depuración de variables altamente redundantes. Las columnas open, high, low y adj\_close fueron descartadas debido a su correlación casi perfecta con close, lo que introduce multicolinealidad sin aportar información adicional útil para el modelo.

Finalmente, las variables retenidas fueron sometidas a un proceso de normalización con el fin de homogeneizar sus escalas numéricas y facilitar la convergencia de los modelos de aprendizaje profundo empleados posteriormente.

El segundo bloque de preprocesamiento corresponde al cálculo del precio de la acción se diseñó específicamente para transformar datos financieros crudos en representaciones adecuadas para modelos de deep learning. Se implementó la clase `TimeSeriesPreprocessor` que ejecuta una pipeline secuencial de transformaciones. Primero, se extrajeron características temporales fundamentales del campo de fecha: año, mes, día, día de la semana, trimestre, día del año y semana del año.

Posteriormente, se aplicaron transformaciones cíclicas mediante funciones seno y coseno para los meses y días de la semana. Esta técnica es crucial porque convierte variables circulares (donde diciembre y enero están cercanos) en representaciones continuas que los modelos pueden procesar correctamente. Sin esta transformación, el modelo trataría el mes 12 (diciembre) y el mes 1 (enero) como valores distantes, cuando en realidad representan períodos contiguos en el año.

Se generaron características de retardo (lag features) para 1, 2, 3, 5, 7, 10, 14 y 20 días anteriores. Estos lags son fundamentales porque capturan la inercia del mercado, es decir, el hecho de que los precios de hoy están correlacionados con los precios recientes. Se calcularon medias móviles para ventanas de 5, 10, 20 y 30 días, que suavizan el ruido y revelan tendencias subyacentes. Paralelamente, se computaron medidas de volatilidad para las mismas ventanas, proporcionando información sobre la estabilidad o turbulencia del mercado en períodos recientes.

El preprocesamiento incluyó el cálculo de retornos diarios porcentuales, que transforman los precios absolutos en cambios relativos. Finalmente, se eliminaron filas con valores nulos generados por las operaciones de desplazamiento, resultando en un dataset final con 35 características. Esta pipeline de preprocesamiento convierte una serie temporal univariada simple en un espacio de características de varias dimensiones.

## **2.2 Arquitecturas**

### **2.2.1 Time Series Transformer**

Se implementó el `TimeSeries Transformer` de Hugging Face, una arquitectura encoder-decoder que utiliza mecanismos de atención multi-head. Esta elección se basó en que los transformers permiten modelar dependencias de largo alcance en secuencias. La configuración específica incluyó 2 capas en el encoder y decoder, dimensión del modelo de 64, y heads de atención de 4. Se aplicó dropout del 10% para regularización y función de activación GELU.

La aplicación de esta arquitectura se justificó por su capacidad para capturar relaciones no locales en la serie temporal. La salida se configuró con distribución t-Student para modelar incertidumbre predictiva, reconociendo la naturaleza probabilística de las predicciones financieras.

### **2.2.2 CNN 1D (Red Neuronal Convolucional Unidimensional)**

La arquitectura CNN 1D se diseñó específicamente para la naturaleza secuencial de los datos financieros. Se implementó con 4 capas convolucionales progresivas con kernels de tamaño 3 y padding para preservar la longitud. Cada capa incluye ReLU, Batch Normalization, y capas selectivas incluyen MaxPooling para reducción dimensional. Esta arquitectura permite detectar patrones locales en ventanas temporales, como formaciones de precios, soportes y resistencias que ocurren en períodos específicos.

La aplicación de CNN 1D se justificó en que las convoluciones capturan invarianza a pequeñas traslaciones temporales (patrones similares que ocurren en momentos ligeramente diferentes); por otra parte, la arquitectura jerárquica permite aprender características de bajo nivel (patrones diarios) que se combinan en características de alto nivel (tendencias semanales/mensuales). El diseño incluyó dropout estratégico del 30% después de las capas intermedias para mejorar la generalización.

### **2.2.3 CNN-LSTM (Arquitectura Híbrida)**

Se implementó una arquitectura híbrida que combina CNN para extracción de características locales con LSTM bidireccional para modelar dependencias temporales largas. La CNN procesa primero la serie temporal, extrayendo patrones locales y reduciendo la dimensionalidad mediante MaxPooling. La salida de la CNN se reordena y alimenta a un LSTM bidireccional de 2 capas con 128 unidades ocultas por dirección.

Esta combinación se justificó por la hipótesis de que los datos financieros requieren tanto el reconocimiento de patrones locales (capturados por CNN) como la memoria de eventos pasados importantes (capturada por LSTM). El LSTM bidireccional procesa la secuencia en ambas direcciones, capturando tanto dependencias hacia adelante como hacia atrás.

### **2.2.4 CNN (binaria)**

Se empleó una CNN porque este tipo de modelo permite identificar patrones locales en series de tiempo de forma eficiente. La arquitectura incluye capas convolucionales, normalización y Dropout para mejorar la estabilidad y reducir el sobreajuste, mientras que el uso de padding causal preserva el orden temporal sin incorporar información futura. Además, se establecen parámetros controlados (semilla, ventana de 20-60 días, tasa de aprendizaje y número de épocas) para garantizar reproducibilidad y un entrenamiento equilibrado.

Tras realizar múltiples experimentaciones con el enfoque de clasificación binaria (predicción de dirección), se observó que este esquema no ofrecía un desempeño suficientemente robusto ni estable. En consecuencia, se optó por descartar dicho enfoque y orientar el desarrollo del modelo exclusivamente hacia la estimación directa del precio de la acción.

## **3. Descripción de las iteraciones**

### **3.1 Iteración 1: Establecimiento de baseline y preprocesamiento básico.**

Se inició estableciendo un pipeline de preprocesamiento de datos financieros. Se diseñó la clase `TimeSeriesPreprocessor` que transforma datos crudos de precios en un conjunto de características multidimensional. Esta etapa generó 35 features incluyendo características temporales básicas, transformaciones cíclicas para periodicidad mensual y semanal, lags temporales estratégicos (1, 2, 3, 5, 7, 10, 14, 20 días), medias móviles y medidas de volatilidad. Se implementó split temporal estricto (1999-2019 para entrenamiento, 2020-2025 para prueba) para evaluar capacidad de generalización a datos futuros no vistos.

### **3.2 Iteración 2: Desarrollo del transformer**

Se implementó el `TimeSeries Transformer` de Hugging Face con configuración personalizada para series financieras. Se definieron parámetros específicos: contexto de 80 días, horizonte de predicción de 20 días, dimensión del modelo de 64, 2 capas en encoder/decoder, y distribución t-Student para salidas probabilísticas.

### **3.3 Iteración 3: Implementación de CNN 1D**

Se desarrolló una CNN 1D de 4 capas convolucionales progresivas (1→32→64→128→256 canales) con kernel size 3, BatchNorm y Dropout estratégico. Esta arquitectura se diseñó específicamente para la naturaleza secuencial de series financieras, donde las convoluciones capturan patrones locales en ventanas temporales. Se implementaron capas de MaxPooling para reducción dimensional progresiva y se conectaron a capas totalmente conectadas para generar predicciones de 20 pasos futuros.

### **3.4 Iteración 4: Creación de arquitectura híbrida CNN-LSTM**

Se construyó un modelo híbrido que combina CNN para extracción de características locales con LSTM bidireccional para modelado de dependencias temporales largas. La CNN procesa patrones locales, seguida por un LSTM de 2 capas que procesa la secuencia en ambas direcciones, capturando contextos pasado y futuro. Las salidas del LSTM se alimentan a capas totalmente conectadas para generar predicciones finales.

### **3.5 Iteración 5: Optimización sistemática de hiperparámetros**

Se estableció un proceso sistemático de optimización para los tres modelos. Para cada arquitectura se configuró: optimizador AdamW con learning rate 1e-3, scheduler CosineAnnealingLR para Transformer y ReduceLROnPlateau para CNN y CNN-LSTM, función de pérdida MSE, batch size de 16-32 dependiendo del modelo, y entrenamiento por 10-15 épocas. Se implementó gradient clipping (norm=0.5) y weight decay (1E-5) para mejorar estabilidad del entrenamiento.

### **3.6 Iteración 6: Implementación de evaluación comparativa**

Se desarrolló un sistema completo de evaluación con base en la métrica MSE. Se crearon visualizaciones comparativas que muestran predicciones vs valores reales.

## **4. Descripción de los resultados**

### **4.1 Preprocesamiento y Preparación de Datos**

El proceso comenzó con un dataset histórico de NVDA que contenía 6,684 observaciones diarias desde enero de 1999 hasta agosto de 2025, abarcando 26 años de datos de mercado. Tras la limpieza inicial, que incluyó la eliminación de filas mal formateadas y conversión de tipos de datos, se trabajó con 6,683 registros válidos. El preprocesamiento transformó la serie temporal univariada original en un conjunto de 35 características, incluyendo transformaciones cíclicas para patrones mensuales y semanales, 8 lags temporales estratégicos (1, 2, 3, 5, 7, 10, 14, 20 días), y múltiples medidas de volatilidad y medias móviles. La división temporal asignó los primeros 5,323 puntos (1999-2019) para entrenamiento y los 1,331 puntos restantes (2020-2025) para prueba, preservando la secuencialidad temporal y simulando condiciones reales de predicción.

#### **4.2 Entrenamiento del Time Series Transformer**

El Transformer, con 182,723 parámetros configurables, mostró un comportamiento de entrenamiento particular durante 10 épocas. La pérdida de entrenamiento se volvió negativa desde la primera época (-0.9233), continuando su descenso hasta -2.3910, mientras que la pérdida de validación disminuyó consistentemente de 2.6916 a 1.7813. Este patrón de pérdidas negativas en entrenamiento sugiere posibles problemas de escala o implementación, aunque la mejora constante en validación indica cierta capacidad de generalización. Las predicciones generadas mostraron una media de 180.49 con desviación estándar de solo 0.69, indicando predicciones muy concentradas pero posiblemente suavizadas.

#### **4.4 Entrenamiento de la CNN 1D**

La CNN 1D, con 1,477,012 parámetros, demostró el mejor rendimiento entre todas las arquitecturas evaluadas. El entrenamiento mostró una rápida convergencia inicial, alcanzando su mejor pérdida de validación (82.025571) en la cuarta época, con una pérdida de entrenamiento correspondiente de 0.314449. Posteriormente, la pérdida de validación fluctuó significativamente (140.918061 en época 5, 355.342088 en época 15), mientras que la pérdida de entrenamiento continuó disminuyendo consistentemente hasta 0.168968. Este patrón indica cierto grado de sobreajuste, pero considerablemente menor que en otras arquitecturas. Las predicciones de la CNN mostraron valores en el rango de 125-141 dólares, consistentes con los niveles de precio histórico reciente de NVDA.

#### **4.5 Entrenamiento del CNN-LSTM Híbrido**

La arquitectura CNN-LSTM, con 642,900 parámetros, exhibió problemas fundamentales durante el entrenamiento. Aunque la pérdida de entrenamiento disminuyó adecuadamente de 2.116317 a 0.072610, la pérdida de validación se mantuvo extremadamente alta, fluctuando entre 4,436.36 y 4,706.91 sin mostrar mejora significativa. Esta discrepancia masiva entre pérdida de entrenamiento y validación indica un alto sobreajuste. Más preocupante aún, las predicciones generadas por el CNN-LSTM cayeron en el rango de 6-6.5 dólares, completamente fuera del rango real de precios de NVDA (que opera alrededor de 180 dólares), demostrando que el modelo falló en aprender la escala apropiada de los datos.

#### **4.6 Comparación final**

La evaluación comparativa rigurosa reveló que la CNN 1D obtuvo el mejor MSE (1,708.31), superando significativamente al Transformer (MSE 6,026.94) y al CNN-LSTM (MSE 29,267.93). En términos relativos, el Transformer fue 252.8% peor que la CNN, mientras que el CNN-LSTM fue 1,613.3% peor. Esta disparidad en rendimiento es notable considerando que el Transformer representa arquitecturas state-of-the-art y el CNN-LSTM combina dos enfoques teóricamente complementarios. La superioridad de la CNN 1D sugiere que para series financieras diarias, la capacidad de capturar patrones locales en ventanas temporales cercanas es más importante que modelar dependencias de muy largo plazo o combinar múltiples mecanismos de aprendizaje.

#### **4.7 Análisis de limitaciones Identificadas**

Los resultados exponen limitaciones fundamentales en todos los modelos evaluados. Aunque la CNN fue la mejor relativamente, su MSE de 1,708.31 se traduce en un RMSE de aproximadamente 41.33 dólares, representando un error del 22.6% respecto al precio promedio de NVDA (~183 dólares). Este nivel de error es inaceptable para aplicaciones reales de trading, donde se requieren precisiones significativamente mayores. Además, todos los modelos mostraron evidencias de sobreajuste en mayor o menor grado, siendo el CNN-LSTM particularmente problemático.

En síntesis, ninguno de los modelos es adecuado para predecir el valor de la acción NVDA con la precisión requerida para aplicaciones prácticas de trading. Aunque la CNN 1D demostró ser la arquitectura más efectiva entre las evaluadas, su error promedio del 22.6% y su evidente sobreajuste la sitúan muy por debajo de los estándares mínimos necesarios para decisiones de inversión confiables. Los resultados confirman que la complejidad arquitectural no compensa las limitaciones fundamentales impuestas por la naturaleza de los mercados financieros eficientes, donde el ruido domina sobre la señal en los datos históricos de precio y volumen.

#### **Referencias**

[1] S. Pessoa, "Conclusion (Mean Reversion & Cointegration study on MAG7 stock)," *Kaggle Notebook*, 2024.

[2] R. Yadav, "SLT-Final\_project," *Kaggle Notebook*, 2024.

[3] S. Waskar, "Stock Prediction LSTM, Linear regression, Xgboost," *Kaggle Notebook*, 2024.

[4] B. Risk, "NVIDIA Stock Trends and Predictions 1999–2024," *Kaggle Notebook*, 2024.

[5] M. DeLong, "NVDA price and volume EDA with scatter plots," *Kaggle Notebook*, 2024.