Financial Regression: Predecir Costo del Oro

Benítez De La Portilla Rodrigo Antonio

A01771433@tec.mx

Abstract: Este estudio presenta la implementación y evaluación de dos modelos de Inteligencia Artificial para predecir el costo de cierre del oro, utilizando el dataset "financial_regression". Los modelos estiman los coeficientes asociados a variables clave como precio de apertura, máximo diario y precios históricos de cierre, con el objetivo de generar proyecciones precisas que faciliten decisiones de inversión más informadas, identificando momentos óptimos de compra y venta. El primer modelo corresponde a un algoritmo implementado manualmente, mientras que el segundo se desarrolló utilizando el framework "scikit-learn".

Los resultados muestran que ambos modelos alcanzan un coeficiente de determinación (R^2) cercano a 0.99 en el conjunto de prueba, lo que evidencia su capacidad para explicar alrededor del 99% de la variabilidad en los precios del oro.

En cuanto al error cuadrático medio (MSE) el modelo manual obtuvo un valor de 0.0006 mientras que el modelo con "scikit-learn" registró 0.0016. Aunque ambos presentan un desempeño sobresaliente, los resultados sugieren que el modelo implementado manualmente ofrece un menor margen de error en las predicciones.

Keywords: Oro, Inteligencia Artificial, predicción, Regresión Lineal Multivariable, Gradiente Descendiente.

1. INTRODUCCIÓN

En los últimos años las inversiones, trading y criptomonedas han evolucionado el mercado haciendo que día con día más personas inicien en este tipo de mercado a hacer inversiones, este término de inversión se remonta a la Antigüedad. Desde los contratos primitivos en Mesopotamia y Egipto hasta la consolidación de conceptos modernos en Europa entre los siglos XVI y XVII con el auge del comercio marítimo y la creación de fideicomisos y sociedades por comerciantes holandeses, la inversión ha evolucionado hacia enfoques cada vez más cuantitativos. En ese espíritu, este trabajo explora técnicas de aprendizaje automático para estimar el precio de cierre del oro a partir de variables de mercado.

El oro, a menudo considerado activo refugio, concentra ciclos de volatilidad asociados a expectativas macroeconómicas, tipos de interés y fortaleza del dólar [1]. Anticipar su

comportamiento facilita decisiones tácticas y de cobertura en portafolios. Para ello, se emplea un conjunto de datos estructurado "financial_regression.csv" con variables de apertura, máximo, mínimo y volumen, orientadas a predecir el precio de cierre.

El flujo de trabajo incluye un ETL básico con eliminación de nulos y normalización basada en estadísticas del conjunto de entrenamiento y una partición 70/15/15 en entrenamiento, validación y prueba. Se evalúan dos enfoques complementarios: un ensamble Bagging con regresión lineal como estimador base, y una regresión lineal multivariable implementada desde cero y optimizada vía descenso por gradiente. El desempeño se mide con **MSE y R^2**, y se visualizan tanto las predicciones frente a valores reales como comparativas de error entre conjuntos.

El objetivo es construir una línea base clara, reproducible y explicable para pronóstico de precios del oro, que sirva como punto de partida para iteraciones futuras. Entre las extensiones recomendadas se incluyen validación temporal (para evitar fuga de información), incorporación de rezagos y ventanas móviles, y variables exógenas (índice del dólar, tasas, inflación, volatilidad implícita). Asimismo, se sugiere contrastar regularización, modelos de árboles y enfoques específicos de series temporales.

En conjunto, este proyecto conecta la tradición histórica de la inversión con herramientas modernas de análisis, mostrando cómo métodos lineales y ensambles pueden apoyar la toma de decisiones en mercados financieros sensibles a la información y al tiempo.

2. DESCRIPCIÓN DEL DATASET

El data set *"financial_regression"* proporciona un registro detallado de los costos de diferentes metales preciosos, combustibles, aceites y otros tipos de químicos.
La parte que se analiza es la del oro, con los precios de apertura, máximos, cierre y volumen. Los datos abarcan desde el 14 de enero de 2010 hasta el 23 de octubre de 2024 y están organizados en registros de horarios, sumando un total de 3,905 instancias.

2a.- Features.

El data set incluye 5 columnas indispensables para la predicción del costo de cierre del oro. A continuación, se presentan las características disponibles:

- **Date**: La fecha en formato YYYY-MM-DD, representando el día específico de registro
- **Gold ope**n: El costo de apertura en USD y dos decimales.
- Gold High: El costo más alto en USD y dos decimales.
- Gold low: El costo más bajo en USD y dos decimales.
- Gold close: El costo de cierre en USD y dos decimales.

- Gold volume: La cantidad de oro intercambiada.

3. EXTRACCIÓN Y LIMPIEZA DE DATOS.

La extracción de datos implica recuperar información de fuentes para su análisis [8]. En este estudio se utilizó el archivo "financial_regression.csv" y se aplicaron pasos de preprocesamiento para entrenar y evaluar modelos de predicción del precio de cierre del oro.

3a.- Carga y filtrado de datos.

Se cargó el dataset y se eliminaron las filas con valores faltantes mediante dropna, conservando únicamente observaciones completas.
La columna "date" no se empleó como característica predictora; solo conserva el orden del archivo. No se aplicó tratamiento de atípicos en esta etapa.

3b.- Análisis de correlación y selección de características.

Para identificar las variables que tienen una relación lineal más fuerte con la variable objetivo "gold_close", se calculó una matriz de correlación. Esta matriz muestra los coeficientes de correlación de Pearson entre cada una de las variables, lo que indica el grado de asociación lineal entre ellas.

El coeficiente de correlación de Pearson varía entre -1 y 1. Un valor de 1 indica una correlación positiva perfecta, es decir, cuando una variable aumenta, la otra también lo hace de manera proporcional.
Un valor de-1 indica una correlación negativa perfecta, donde un aumento en una variable corresponde a una disminución en la otra. Un valor cercano a 0 sugiere que no existe una relación lineal significativa entre las variables [9].

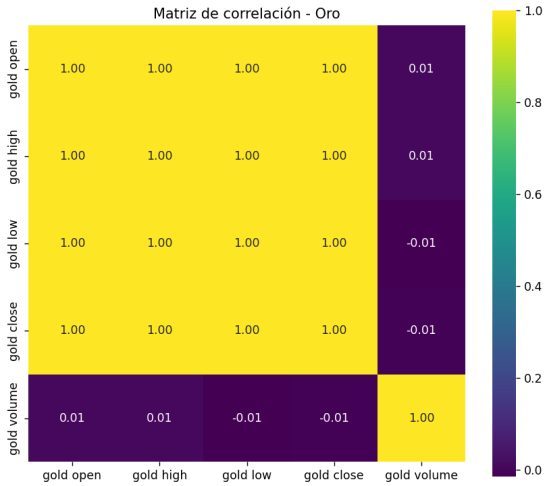


Figura 1. *Matriz de correlación entre las variables.*

Cómo se puede observar en la matriz de correlación, las relaciones respecto al cierre del oro son:

- Gold open: Ofrece una correlación fuerte, es decir que tiene un impacto significativo y aporta para este modelo.

- Gold high: Muestra una correlación con un coeficiente de 1.0, que nos indica que tomemos esa variable, pues nos dará información para nuestro modelo.

- Gold low: Indica que, como las variables anteriores, tiene un impacto importante para nuestro modelo, su coeficiente de 1.0 comparte información valiosa.

- Gold Close: Al tener un coeficiente de 1.0, que hace referencia a una buena participación para el buen funcionamiento del algoritmo.

- Gold volumen: Esta es la única variable que tiene un coeficiente diferente a las demás variables siendo de -0.01. Realmente podríamos ignorar esta variable, pero se utilizo para ejercicio de práctica y normalización de los datos.

4. TRANSFORMACIÓN DE DATOS

La transformación de datos prácticamente es cambiar los valores de las columnas numéricas del conjunto de datos para usar una escala común, sin distorsionar las diferencias en los intervalos de valores ni perder información [5].
Para ello existe una fórmula de normalización de datos como se muestra en la Ecuación 1.

$$z = \frac{X - mean(x)}{stdev(x)}$$

Ecuación 1. *Normalización de datos.*

Esta fórmula nos ayuda a hacer comparables las escalas de las variables, evitando que una con valores grandes domine el gradiente para acelerar y estabilizar el descenso de gradiente con una sola tasa de aprendizaje razonable, también mejora la condición numérica y evita coeficientes inestables.

4a.- Normalización de datos.

Los features normalizados son: gold open, gold high, gold low, gold volume.

Target normalizado: gold close.

Los promedios y desviaciones se calcularon solo con el set de entrenamiento y se aplicaron a train, validation y test. Esto evita fuga de información.

4b.- División del dataset en conjuntos.

Tras la limpieza y transformación de los datos, el conjunto se segmentó en tres particiones mediante slicing secuencial en una proporción 70%, 15%, 15%, preservando el orden cronológico original. Esta decisión evita la mezcla de observaciones futuras con pasadas y previene la fuga de información, algo crítico en datos con dependencia temporal como los precios del oro.

El 70% inicial se usa para ajustar los parámetros del modelo; el 15% siguiente se reserva para validación, donde se monitorea el desempeño fuera de entrenamiento y se calibran hiperparámetros (por ejemplo, tasa de aprendizaje y número de épocas) sin contaminar la evaluación final, el 15% restante se mantiene como conjunto de prueba totalmente independiente para

estimar el rendimiento out-of-sample. Esta partición ofrece un equilibrio adecuado entre cantidad de datos para aprender, capacidad de ajuste fino y evaluación honesta.

Este procedimiento beneficia al modelo puesto que entrega una medición realista de generalización en datos no vistos.

Reduce el riesgo de sobreajuste al disponer de un canal de validación para detener o ajustar el entrenamiento y mantiene la coherencia temporal del problema, crucial para tareas predictivas sobre series financieras.

5. CONSTRUCCIÓN Y EVALUACIÓN DEL MODELO

5a.- Selección de variables predictoras y objetivo.

Las variables predictoras para entrenar al modelo son: “gold open”, “gold high”, “gold low”, “gold volume”.

Mientras que el valor que queremos predecir o la variable objetivo es: “gold close”.

El resto de las columnas no se utilizaron para esta línea base.

“gold volumen” se incluyó como predictor y se estandarizó junto con las demás características, la unidad exacta depende de la fuente original, pero en el CSV aparece como conteo diario de negociaciones a gran escala.

5b.- Inicialización de parámetros.

Los parámetros del modelo se inicializan de manera aleatoria, esto para que nuestro modelo no tenga “patrones” que puedan llevar al modelo a un mal funcionamiento.

5c.- Cálculo de la hipótesis.

La función de hipótesis puede describirse como el deseo de utilizar los datos disponibles para aprender una función que asigne mejor las entradas a las salidas [2].

La fórmula para la hipótesis es la siguiente:

$$Y = \beta_0 + \beta_1 X1$$

Ecuación 2. *Función de hipótesis para regresión lineal.*

En este proyecto se aborda la regresión lineal múltiple para predecir el precio de cierre del oro a partir de varias características de mercado. La hipótesis del modelo se formula como una combinación lineal de todas las entradas más un término de sesgo, y se implementa explícitamente en la función hyp(x, theta, b), donde el producto punto entre las características normalizadas y los parámetros theta, sumado al bias b, produce la predicción.

5d.- Cálculo del error.

El error del modelo cuantifica qué tan lejos están las predicciones de la realidad y guía todo el proceso de aprendizaje [6].

La métrica elegida es el Mean Squared Error (MSE), calculado sobre los datos normalizados: para cada muestra se obtiene el residuo como la diferencia entre la predicción hyp(x[i], theta, b) y el valor real y[i], se eleva al cuadrado, se suman todos los residuos y se promedian dividiendo entre el número de ejemplos la fórmula es la siguiente:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2$$

Ecuación 3. *Fórmula del error cuadrático medio.*

En el código, esto se implementa en la función mse (x, theta, b, y), donde x son las características de entrada estandarizadas, theta y b representan los parámetros actuales del modelo, y "y" es la variable objetivo también normalizada. Minimizar este MSE es el objetivo del entrenamiento.

Durante el proceso se registran errores de entrenamiento y validación por época, lo que permite monitorear convergencia y detectar sobreajuste, al final, se reporta el error, para estimar el desempeño en datos no vistos. El MSE mostrado se mantiene en la escala normalizada, lo que hace comparables las curvas de aprendizaje; si se requiere, podría calcularse también un MSE en la escala original.

5e.- Descenso de Gradiente.

El descenso de gradiente en este contexto se usa para ajustar un modelo de regresión lineal múltiple que predice el precio de cierre del oro a partir de otros precios históricos, minimizando el MSE sobre datos estandarizados.

En cada época, primero se evalúa la pérdida en el entrenamiento y validación, luego, la función *update* recorre todo el conjunto de entrenamiento y calcula los gradientes a partir de los residuos con la siguiente fórmula:

$$J(\theta_0 - \theta_1) = \frac{1}{2m}\sum_{i=0}^{m}\left(h\theta\left(x^i\right) - y^i\right)^2$$

Ecuación 4. *Fórmula de actualización de parámetros para gradiente descendente.*

Con esos gradientes, los parámetros se actualizan en la dirección opuesta escalándolos por la tasa de aprendizaje La hipótesis hyp implementa la combinación lineal de entradas y el término b, mse mide la discrepancia promedio al cuadrado, update aplica el paso de optimización, y el bucle de entrenamiento repite este proceso época veces.

La normalización previa de x e y estabiliza la magnitud de los gradientes y favorece la convergencia, mientras que el registro de errores permite monitorear el progreso y detectar posibles desalineaciones entre ajuste y generalización.

5f.- Modelado y evaluación.

En la evaluación del rendimiento para el modelo, se ocupó la métrica de coeficiente de determinación (R^2), este coeficiente nos explica cuanto porcentaje llega a poder explicar nuestro modelo, si tenemos un coeficiente de 0.99, nuestro modelo es

capaz de explicar el 99% de las predicciones con alta seguridad.

$$R^2 = 1 - \frac{RSS}{TSS}$$

Ecuación 5. *Cálculo del coeficiente de determinación R^2.*

Donde:

- R^2: coeficiente de determinación
- RSS : suma de cuadrados de los residuos
- TSS : suma total de cuadrados

5g.- Entrenamiento, validación y Prueba.

Para evaluar y potencialmente mejorar la capacidad predictiva de nuestro modelo de regresión con Gradient Descent en datos financieros, dividimos el conjunto de datos en tres segmentos: entrenamiento (70%), validación (15%) y prueba (15%). Esta división es fundamental en el aprendizaje automático, ya que utilizar los mismos datos para entrenar y evaluar el modelo puede provocar un sesgo significativo, donde el algoritmo memoriza los ejemplos en lugar de aprender patrones generalizables, resultando en predicciones pobres para datos nuevos.

El conjunto de entrenamiento se utiliza para optimizar los parámetros theta y el término de sesgo (b) de nuestro modelo lineal. Durante este proceso, aplicamos el algoritmo de Gradient Descent para minimizar iterativamente el error cuadrático medio (MSE) entre los precios de cierre del oro predichos y los valores reales, ajustando los parámetros en cada época.

El conjunto de validación cumple un papel crítico durante el entrenamiento, permitiéndonos monitorear el rendimiento del modelo con datos no vistos previamente. En nuestro caso, calculamos y registramos el error en este conjunto después de cada época de entrenamiento, lo que nos ayuda a detectar posibles problemas de sobreajuste y a evaluar la capacidad de generalización del modelo mientras se entrena.

El conjunto de prueba permanece completamente aislado hasta que el entrenamiento finaliza. Solo entonces lo utilizamos para obtener una evaluación imparcial y definitiva del rendimiento del modelo, calculando tanto el MSE como el coeficiente de determinación (R^2) para determinar la precisión y exactitud de nuestras predicciones del precio de cierre del oro.

Nuestro modelo calcula sistemáticamente dos métricas fundamentales MSE (error cuadrático medio) y R^2 (coeficiente de determinación) para los conjuntos de entrenamiento, validación y prueba. El MSE cuantifica la diferencia promedio al cuadrado entre los precios de cierre del oro predichos y los valores reales, mientras que el R^2 indica la proporción de varianza en la variable dependiente que el modelo explica. Ambas métricas se implementan como funciones independientes en nuestro código y se visualizan mediante gráficos de barras para comparar el rendimiento entre los tres conjuntos.

Nuestro enfoque de división del dataset, busca mantener un equilibrio óptimo entre los conjuntos para detectar y prevenir el sobreajuste. En cada época del entrenamiento, registramos tanto el error de entrenamiento como el de validación, permitiéndonos monitorizar visualmente la evolución de ambos errores en una gráfica temporal. Si el modelo comenzara a memorizar los datos de entrenamiento en lugar de aprender patrones generalizables, observaríamos una divergencia entre estas curvas, el error de entrenamiento continuaría disminuyendo mientras el error de validación aumentaría o se estancaría.

El monitoreo constante de estas métricas en los tres conjuntos es crucial para nuestro modelo financiero. Un rendimiento desequilibrado (por ejemplo, R^2 alto en entrenamiento, pero bajo en validación y prueba) indicaría que nuestro modelo podría fallar al predecir precios futuros del oro en escenarios reales de mercado. Las visualizaciones que implementamos, especialmente la comparación de errores finales y scores R^2 entre los tres conjuntos, nos proporcionan una herramienta efectiva para evaluar si estamos logrando un equilibrio adecuado entre aprender de los datos históricos y mantener la capacidad de generalizar a nuevos datos del mercado de oro.

5h.- Fitting, overfitting y underfitting.

Al entrenar nuestro modelo de regresión para predecir el precio de cierre del oro, identificamos tres escenarios posibles que afectan su capacidad predictiva: subajuste, ajuste adecuado y sobreajuste.

El subajuste (underfitting) ocurre cuando nuestro modelo lineal es demasiado simple para capturar las relaciones complejas entre las características del mercado del oro. En este escenario, observaríamos valores de R^2 bajos y errores MSE elevados tanto en entrenamiento como en validación y prueba en nuestras gráficas comparativas. Esto indicaría un alto sesgo (bias) y baja varianza en el modelo.

En nuestro contexto, el subajuste podría manifestarse cuando:

Las cuatro características seleccionadas ('gold open', 'gold high', 'gold low', 'gold volume') son insuficientes para modelar adecuadamente el precio de cierre.
Nuestra implementación lineal no captura relaciones no lineales en los datos financieros.
El conjunto de entrenamiento (70% de nuestros datos) es insuficiente para detectar patrones.
La normalización aplicada no resuelve problemas de calidad en los datos.
Podríamos mitigar el subajuste:

Aumentando la complejidad con términos polinómicos en nuestras características.
Incorporando características adicionales como indicadores económicos.
Incrementando el número de épocas de entrenamiento más allá de 10,000.
Refinando nuestro proceso de limpieza y normalización.

El sobreajuste (overfitting) ocurre cuando nuestro modelo se ajusta excesivamente a los datos de entrenamiento. En nuestras visualizaciones, esto se detectaría cuando las curvas de error muestran divergencia, con errores MSE bajos en entrenamiento, pero altos en validación, y valores de R^2 altos en entrenamiento, pero bajos en validación y prueba. Esto refleja alta varianza y bajo sesgo.

En nuestro modelo, el sobreajuste podría surgir cuando:

Las características capturan ruido específico de los datos históricos del oro.
La tasa de aprendizaje (alfa = 0.001) es demasiado alta, permitiendo cambios bruscos en parámetros.
Podemos prevenirlo mediante:

Aumentar la representatividad del conjunto de entrenamiento.
Implementar técnicas de regularización.
Ajustar nuestra tasa de aprendizaje.
Considerar implementar early stopping basado en la divergencia entre errores de entrenamiento y validación.
El ajuste adecuado (fitting) se evidencia cuando nuestro modelo logra un equilibrio, mostrando valores similares de MSE y R^2 en los tres conjuntos. Esto sugiere que ha captado los patrones fundamentales del mercado del oro sin memorizar peculiaridades. Nuestras gráficas comparativas de barras para MSE y R^2 permitirían identificar visualmente si estamos logrando este equilibrio ideal entre sesgo y varianza, indicando que el modelo podría funcionar adecuadamente en condiciones reales de mercado [7].

5i.- Bias.

El bias representa la diferencia sistemática entre las predicciones del modelo y los valores reales. Es un indicador de la capacidad del modelo para capturar la complejidad subyacente en los datos [3]:

Bias bajo: Indica que el modelo hace pocas suposiciones simplificadoras, permitiéndole adaptarse estrechamente al conjunto de entrenamiento. Esto facilita la captura de patrones complejos y relaciones no lineales entre variables.

Bias alto: Sugiere que el modelo opera bajo múltiples suposiciones simplificadoras, resultando en un modelo demasiado simple que no logra capturar la complejidad inherente de los datos. Consecuentemente, el modelo tendrá un pobre desempeño incluso con los datos de entrenamiento.

5j.- Varianza.

La varianza mide la sensibilidad del modelo a fluctuaciones en los datos de entrenamiento. Refleja cuánto cambian las predicciones cuando se entrena el modelo con diferentes subconjuntos de datos [3]:

Varianza baja: El modelo mantiene un comportamiento estable cuando se entrena con diferentes conjuntos de datos, produciendo predicciones consistentes independientemente de las

particularidades del conjunto de entrenamiento.

Varianza alta: El modelo es extremadamente sensible a las características específicas de los datos de entrenamiento, lo que resulta en predicciones inconsistentes cuando se expone a diferentes conjuntos de datos. Esto suele indicar un sobreajuste.

Un modelo óptimo logra un balance adecuado entre bias y varianza. Demasiado bias resulta en underfitting (subajuste), mientras que excesiva varianza lleva al overfitting (sobreajuste). El modelo ideal tiene niveles moderados de ambos componentes, permitiéndole aprender patrones significativos de los datos de entrenamiento sin memorizar sus peculiaridades o ruido.

6. VISUALIZACIÓN DE RESULTADOS DEL CASO BASE

Después de pasar por el entrenamiento, validación y prueba y realizar las pruebas pertinentes, los resultados muestran que el modelo alcanzó una exactitud para explicar las predicciones (R^2) con los siguientes valores:

- Entrenamiento R^2: 0.998
- Validación R^2: 0.998
- Prueba R^2: 0.999

Estos resultados son muy buenos, pues sugieren que el modelo es capaz de capturar aproximadamente el 99% de la variabilidad de los datos.

Los valores del coeficiente R^2 en los conjuntos de entrenamiento, validación y prueba son extremadamente similares, lo que indica que el modelo no está experimentando overfitting.

Esta consistencia en los datos entre los tres conjuntos de datos sugiere que el modelo es estable y no está reaccionando de manera excesiva a las particularidades del conjunto entrenamiento, lo que indica una varianza baja.

Por otro lado, el modelo, tampoco está presentando un caso de underfitting, pues el modelo está no está haciendo suposiciones simplificadas sobre las relaciones entre los datos, lo que lleva a un ajuste eficiente.

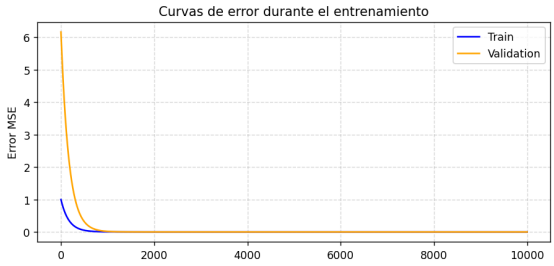


Figura 1. *Curvas de error durante entrenamiento y validación.*

La gráfica nos indica como el error empieza alto y conforme pasa el entrenamiento y las épocas este disminuye considerablemente, hasta que llega un momento en dónde el modelo a encontrado un buen balance para los datos.

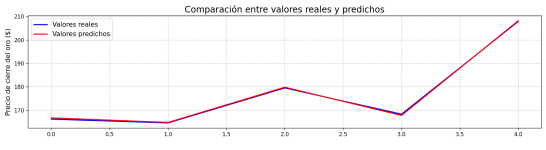


Figura 3*. Valores reales contra valores predichos Regresión y gradiente.*

Cómo podemos observar la gráfica nos muestra que los valores que el modelo predijo van muy similares a los valores reales, si hay una muy pequeña y casi insignificante diferencia entre los valores.

7.IMPLEMENTACIÓN DE FRAMEWORK

Dentro de esta parte se presenta la implementación del modelo de Bagging con Regresión lineal, utilizando frameworks (librerías), pandas, numpy, scikit-learn y matplotlib. Para poder predecir el costo de cierre del oro, aplicando técnicas de preprocesamiento y entrenamiento de modelos, además de analizar los resultados obtenidos.

7a.- Carga y preprocesamiento de datos.

El proceso comenzó cargando el dataset “financial_regression” con la librería pandas, seguido de una limpieza mediante la función *dropna* para eliminar las instancias con valores nulos y garantizar la integridad del conjunto de datos. Posteriormente, se seleccionaron las características más relevantes para el análisis

La etapa crucial del preprocesamiento consistió en la aplicación de una normalización z-score a todas las variables, calculando primero la media y desviación estándar de cada característica.

Este método estandariza los datos restando la media y dividiendo por la desviación estándar, transformando las distribuciones a una escala comparable con media 0 y desviación estándar 1. Este paso es fundamental cuando se trabaja con variables financieras que pueden operar en rangos muy distintos, como precios y volúmenes de negociación del oro, asegurando que todas las características contribuyan equitativamente al proceso predictivo y evitando que variables con magnitudes mayores dominen el comportamiento del modelo.

7b.- División del dataset en conjuntos.

Tras la limpieza y transformación de los datos, el conjunto se segmentó en tres particiones mediante slicing secuencial en una proporción 70%, 15%, 15%, preservando el orden cronológico original. Esta decisión evita la mezcla de observaciones futuras con pasadas y previene la fuga de información, algo crítico en datos con dependencia temporal como los precios del oro.

Tras la limpieza y transformación de los datos, el conjunto se segmentó en tres particiones mediante slicing secuencial en una proporción 70%, 15%, 15%, preservando el orden cronológico original. Esta decisión evita la mezcla de observaciones futuras con pasadas y previene la fuga de información, algo

crítico en datos con dependencia temporal como los precios del oro.

7c.- Selección de features para el modelo.

Las variables predictoras para entrenar al modelo son: “gold open”, “gold high”, “gold low”, “gold volume”.

Mientras que el valor que queremos predecir o la variable objetivo es: “gold close”.
El resto de las columnas no se utilizaron para esta línea base.
“gold volumen” se incluyó como predictor y se estandarizó junto con las demás características, la unidad exacta depende de la fuente original, pero en el CSV aparece como conteo diario de negociaciones a gran escala.

7d.- Creación del modelo.

Se implementó un modelo de BaggingRegressor utilizando como estimador base LinearRegression, configurado con 50 estimadores y una semilla aleatoria fija para garantizar la reproducibilidad de los resultados. El Bagging es una técnica de ensamble que mejora la estabilidad y precisión de los algoritmos de aprendizaje automático.

El BaggingRegressor funciona creando múltiples instancias del estimador entrenando cada una con diferentes subconjuntos aleatorios del conjunto de entrenamiento original. Este proceso sigue estos pasos:

- Genera múltiples muestras bootstrap del conjunto de entrenamiento mediante muestreo con reemplazo.
- Entrena una instancia de LinearRegression en cada muestra Bootstrap.
- Combina las predicciones de todos los modelos tomando el promedio de sus salidas.

Gracias a esta técnica podemos reducir la varianza, lo que disminuye la probabilidad de sobreajuste, mejorar la estabilidad de las predicciones.

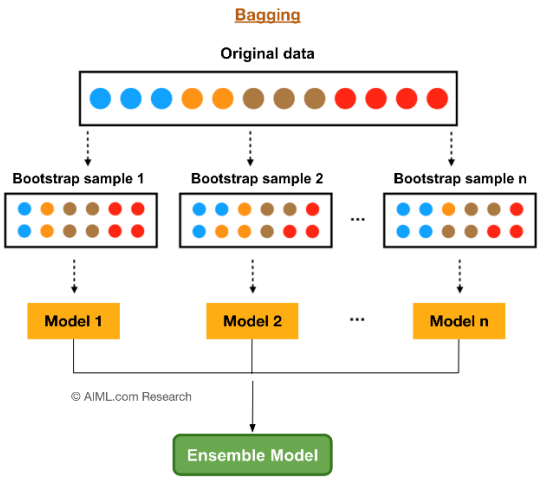


Figura 4. *Funcionamiento de Bagging.*

El modelo fue evaluado utilizando métricas estándar (MSE y R^2) en los conjuntos de entrenamiento, validación y prueba, permitiendo verificar su capacidad de generalización y evitar el sobreajuste en la predicción de los precios de cierre del oro.

7e.- Resultados para el modelo de Bagging.

Después de entrenar, validar y probar el modelo, los resultados obtenidos fueron los siguientes:

- Entrenamiento:
    - MSE: 0.0004
    - R^2: 0.99
- Validación:
    - MSE: 0.001
    - R^2: 0.99
- Prueba:
    - MSE: 0.001
    - R^2:0.99

Estos resultados reflejan que el modelo no está sufriendo de sobreajuste (overfitting), ya que no hay una muestra una consistencia notable entre los conjuntos de entrenamiento, validación y prueba. Los valores de R^2 relativamente cercanos entre estos tres conjuntos sugieren que el modelo ha logrado un equilibrio adecuado entre bias y varianza, lo que le permite generalizar eficazmente a datos no vistos durante el entrenamiento. Este comportamiento indica que el modelo no está simplemente "memorizando" los datos de entrenamiento, sino que ha capturado patrones subyacentes en las dinámicas del precio del oro.

La proximidad de los errores MSE entre los conjuntos de validación y prueba refuerza la robustez del modelo, demostrando su capacidad para mantener un rendimiento estable cuando se enfrenta a nuevos datos. Esta estabilidad es particularmente valiosa en contextos financieros donde las condiciones del mercado pueden fluctuar.

8. VISUALIZACIÓN DE RESULTADOS DEL CASO FRAMEWORK

Para entender mejor los resultados se hicieron gráficos del rendimiento y comparaciones de valores predichos y valores reales.

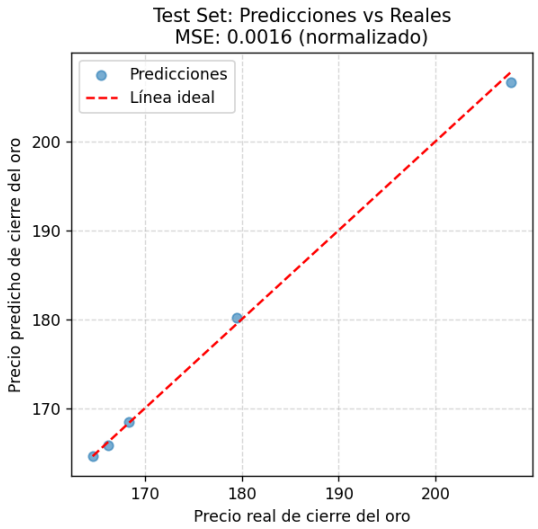


Figura 5. *Predicciones contra valores reales.*

En este gráfico podemos observar cómo nuestras predicciones, no se alejan mucho de los valores reales, esto es muy buen resultado pues el rendimiento es muy bueno.

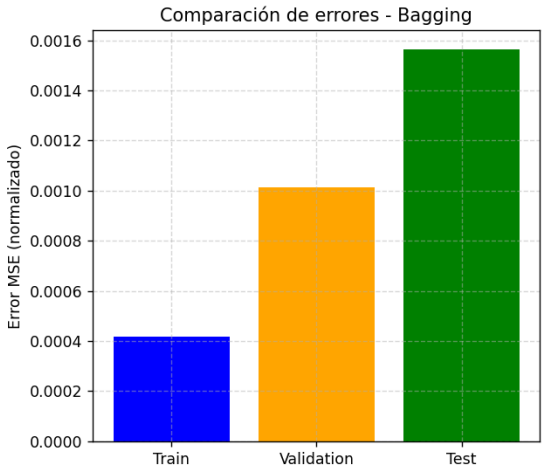


Figura 6. *Comparación de errores bagging.*

Esta diferencia entre el error de entrenamiento y los errores de validación/prueba sugiere la presencia de cierto grado de sobreajuste. Sin embargo, es importante notar que la magnitud general de los errores sigue siendo bastante baja (todos por debajo de 0.0016 en escala normalizada), lo que indica que el modelo mantiene un rendimiento aceptable incluso en datos no vistos.

La diferencia entre el conjunto de validación y prueba podría indicar que el conjunto de prueba contiene patrones más complejos o diferentes a los observados durante el entrenamiento. Esta disparidad, aunque presente, no es excesivamente grande, sugiriendo que el modelo BaggingRegressor ha logrado un balance razonable entre ajuste y generalización para la predicción del precio del oro.

9. COMPARACIÓN DE LOS MODELOS

A continuación, se presenta una comparación entre los dos modelos utilizados: Regresión lineal con gradiente descendente y el modelo de Bagging con estimador "LinearRegressor". Esta comparación se centra en dos métricas clave: el coeficiente de determinación R^2 y el Error cuadrático medio (MSE) por sus siglas en inglés, para los conjuntos de datos de entrenamiento, validación y prueba.

Tabla 1. *Comparación de resultados de modelos.*

Modelo	Train MSE	Val MSE	Test MSE	Train R^2	Val R^2	Test R^2
Gradiente descendente	0.00164	0.00056	0.00065	0.99836	0.99876	0.99939
Bagging Lineal	0.00042	0.00101	0.00160	0.99958	0.99775	0.99850

9a.- Análisis del Modelo de Gradient Descent.

Underfitting: El modelo de regresión lineal implementado con gradient descent muestra un comportamiento estable entre los conjuntos de entrenamiento, validación y prueba. Los valores de R^2 son consistentes entre los tres conjuntos, lo que sugiere que el modelo tiene una capacidad limitada pero estable para capturar las relaciones entre las variables. Esta consistencia indica que no está sobreajustando los datos de entrenamiento.

Bias: El modelo presenta un bias considerable debido a su naturaleza lineal, lo que limita su capacidad para capturar relaciones más complejas en los datos financieros. Esta limitación se refleja en los valores de MSE que, aunque similares entre conjuntos, no logran reducirse

significativamente durante el entrenamiento después de cierto punto, evidenciando las limitaciones inherentes del modelo lineal para representar dinámicas no lineales del mercado del oro.

Varianza: La varianza es baja, como se evidencia en la proximidad de los errores entre los conjuntos de entrenamiento, validación y prueba. La implementación desde cero del algoritmo gradient descent mantiene un rendimiento estable, aunque a costa de una capacidad predictiva limitada por su estructura lineal.

9b.- Análisis del Modelo BaggingRegressor.

Underfitting/Overfitting: El BaggingRegressor muestra una diferencia más marcada entre el error de entrenamiento y los errores de validación/prueba, como se visualiza en la gráfica de comparación de errores. Sin embargo, los valores de R^2 mantienen cierta consistencia entre conjuntos, lo que sugiere que el modelo ha logrado un balance razonable entre ajuste y generalización, aunque con tendencia hacia cierto sobreajuste.

Bias: El modelo presenta un bias moderado. La técnica de ensemble con múltiples estimadores lineales permite capturar relaciones más complejas que un simple modelo lineal, reduciendo el bias en comparación con el modelo de gradient descent. Esto se refleja en un mejor rendimiento general en términos de métricas, aunque sigue limitado por la naturaleza lineal de sus estimadores bases.

Varianza: La varianza es mayor que en el modelo de gradient descent, como muestra la diferencia más amplia entre el error de entrenamiento y los errores de validación/prueba. Sin embargo, el bagging como técnica está específicamente diseñado para reducir la varianza a través de la combinación de múltiples estimadores, logrando un equilibrio que permite mejor generalización que un único modelo lineal.

REFERENCIAS

1. A Brief History of Investments | Indian River Financial Group, Inc. (2016). Paulmilleradvisor.com. https://www.paulmilleradvisor.com/blog/brief-history-investments#:~:text=La%20inversi%C3%B3n%20comenz%C3%B3%20en%20el,largo%20plazo%20en%20el%20extranjero

2. Brownlee, J. (2019, March 3). What is a Hypothesis in Machine Learning? - MachineLearningMastery.com. https://machinelearningmastery.com/what-is-a-hypothesis-in-machine-learning/

3. Difference Between Bias and Variance. (2024, April 15). Mastersindatascience.org. https://www.mastersindatascience.org/learning/difference-between-bias-and-variance/

4. Grosskelwing G. Análisis de Regresión Lineal Simple. (2021, August 23). Rpubs.com. https://rpubs.com/Gabo6381/801498

5. likebupt. (2024, September). Normalización de datos: referencia de componente - Azure Machine Learning. Microsoft.com. https://learn.microsoft.com/es-es/azure/machine-learning/component-reference/normalize-data?view=azureml-api-2

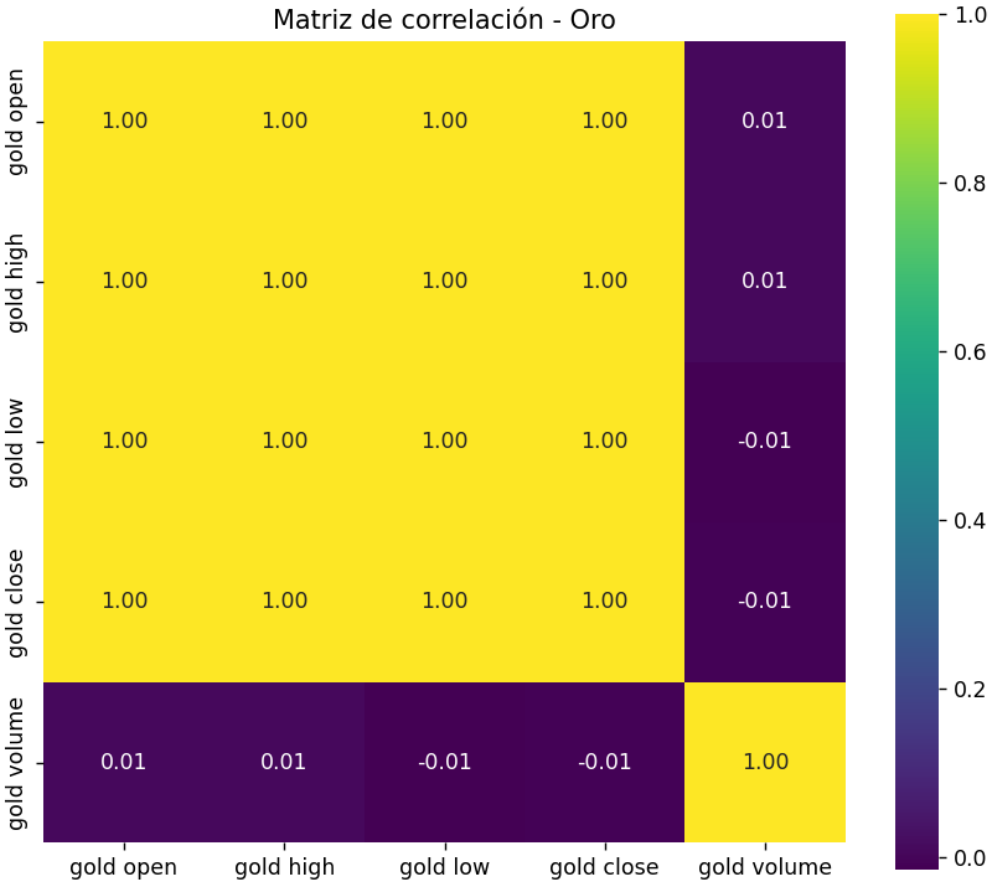
6. Mohamad. (2017, March 9). MSE – Error Cuadrático Medio. Centro de Ayuda. https://support.numxl.com/hc/es/articles/115001223423-MSE-Error-Cuadr%C3%A1tico-Medio

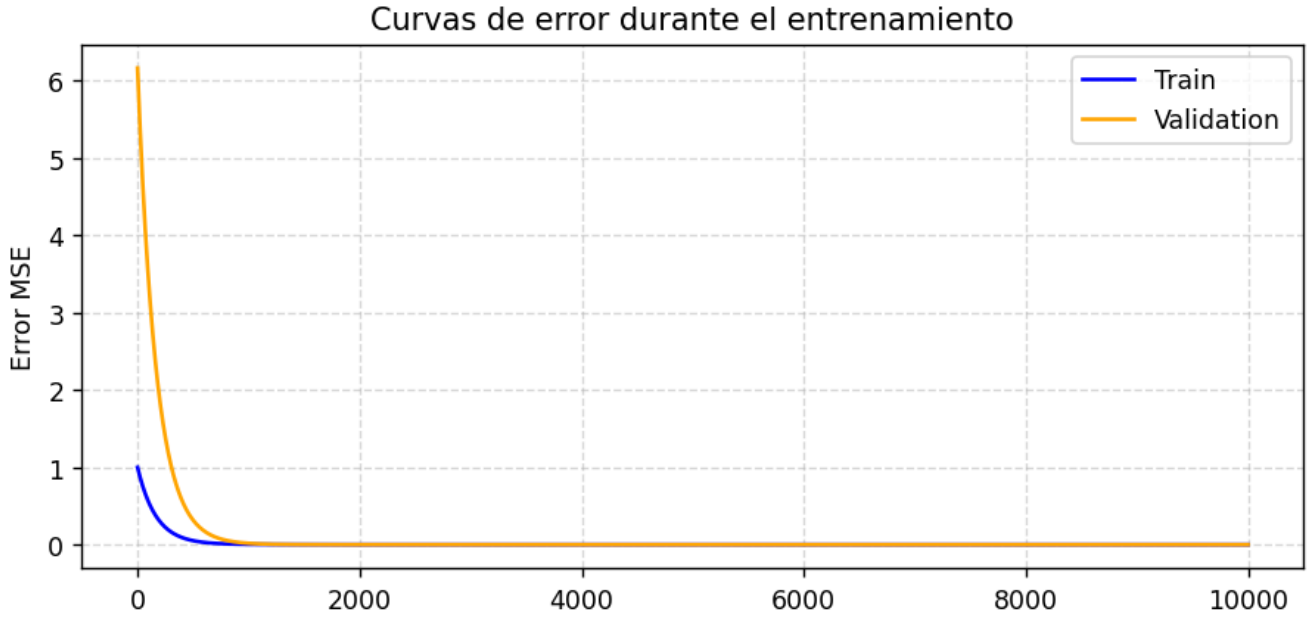
7. Na8. (2017, December 12). Qué es overfitting y underfitting y cómo solucionarlo | Aprende Machine Learning. Aprendemachinelearning.com. https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/#:~:text=Tal%20vez%20se%20pueda%20traducir,conocimiento%20que%20pretendemos%20que%20adquieran.

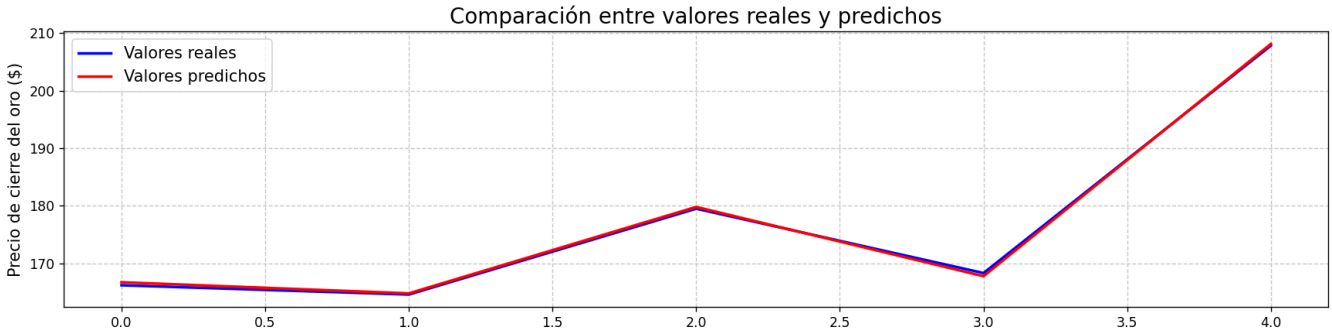
8. Team, A. A. (2023, April 27). ¿Qué es la extracción de datos? Tipos, usos y beneficios | Astera. Astera. https://www.astera.com/es/type/blog/what-is-data-extraction-a-brief-guide/

9. Turney, S. (2022, May 13). Pearson Correlation Coefficient (r) | Guide & Examples. Scribbr. https://www.scribbr.com/statistics/pearson-correlation-coefficient/

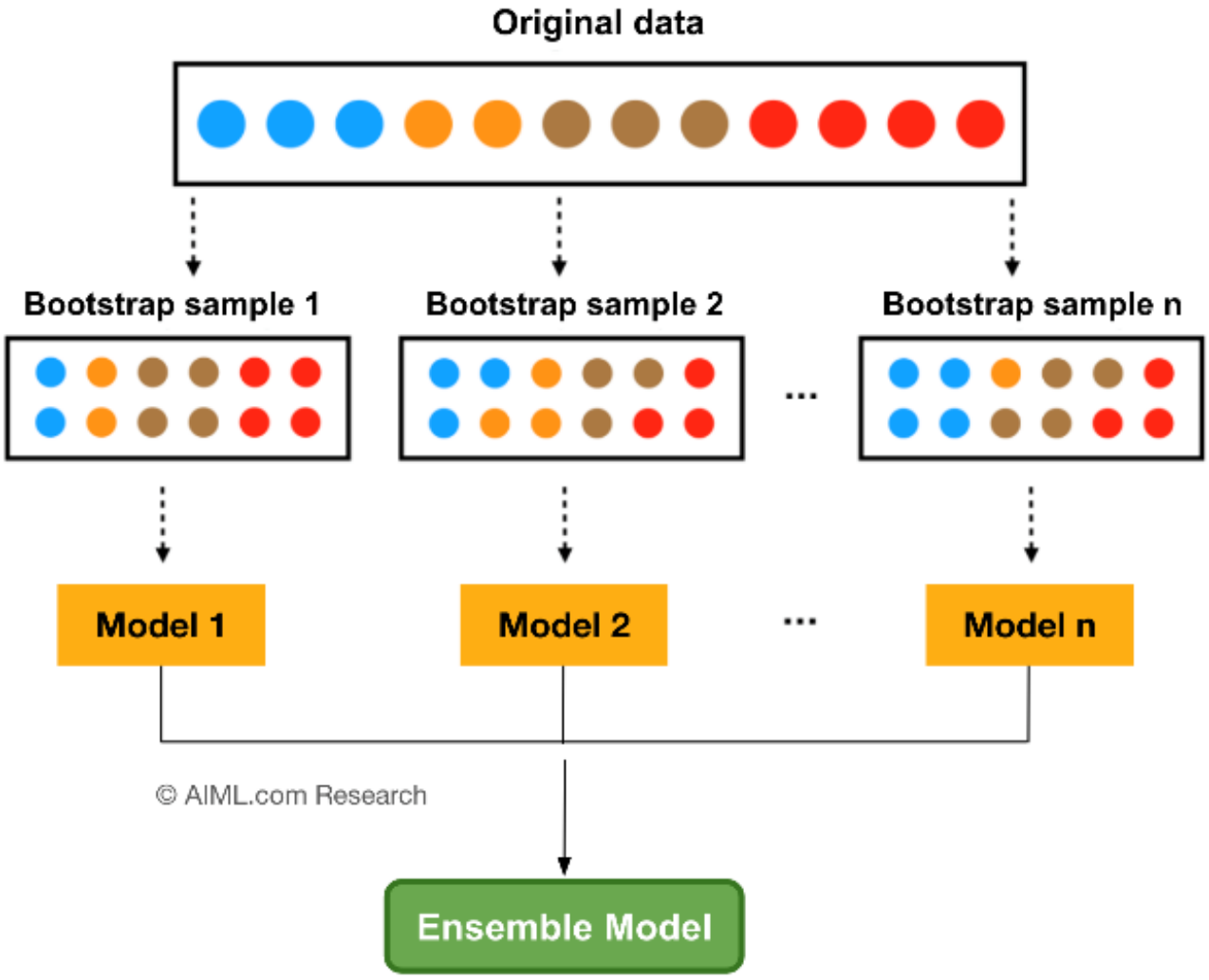
ANEXOS

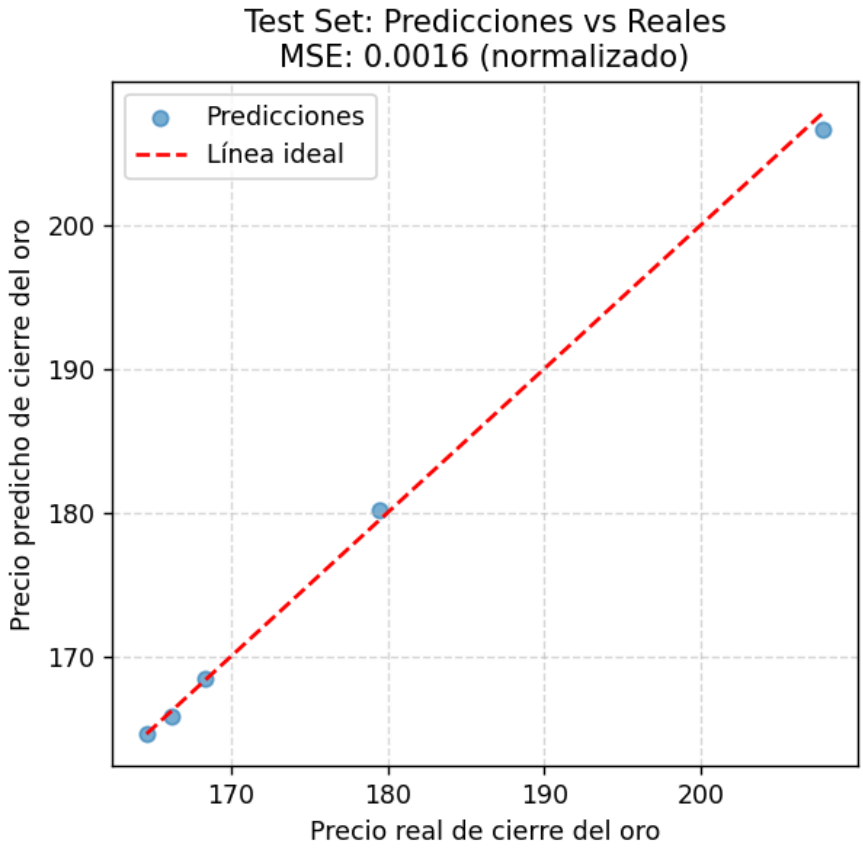

Matriz de correlación - Oro
gold open
gold high
gold low
gold close
gold volume
1.00 1.00 1.00 1.00 0.01
1.00 1.00 1.00 1.00 0.01
1.00 1.00 1.00 1.00 -0.01
1.00 1.00 1.00 1.00 -0.01
0.01 0.01 -0.01 -0.01 1.00
gold open
gold high
gold low
gold close
gold volume
1.0
0.8
0.6
0.4
0.2
0.0

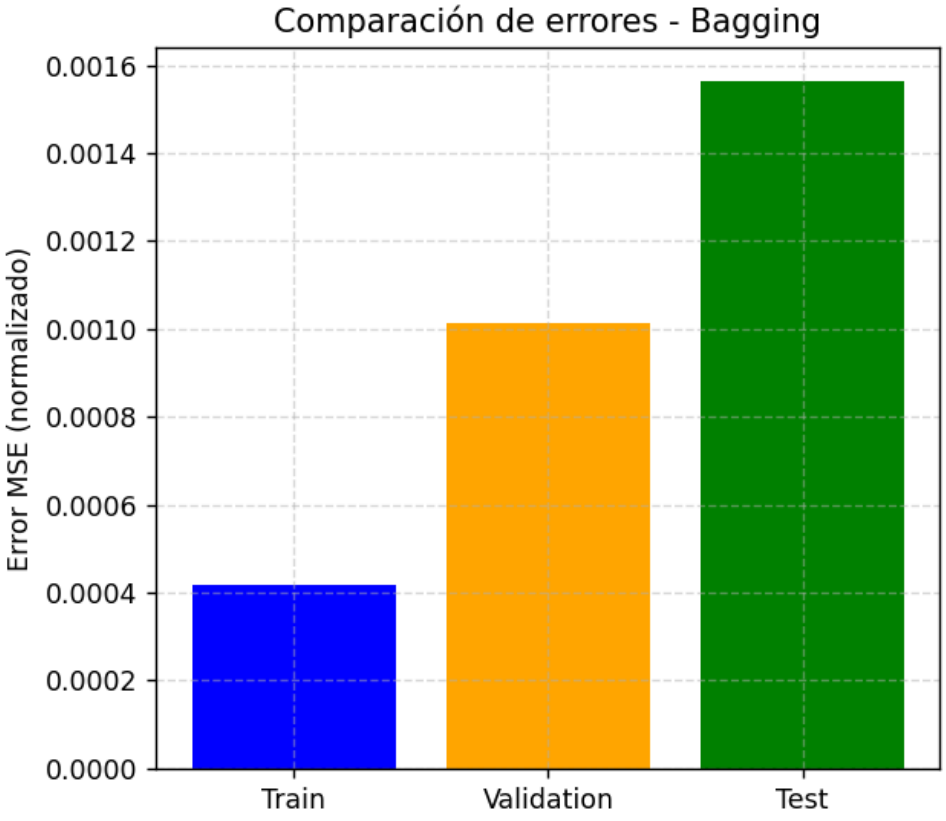

Curvas de error durante el entrenamiento
Train
Validation
Error MSE
0
1
2
3
4
5
6
0
2000
4000
6000
8000
10000


Comparación entre valores reales y predichos
Valores reales
Valores predichos
Precio de cierre del oro ($)
210
200
190
180
170
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

Bagging


Original data
Bootstrap sample 1
Bootstrap sample 2
Bootstrap sample n
Model 1
Model 2
Model n
© AIML.com Research
Ensemble Model


Test Set: Predicciones vs Reales
MSE: 0.0016 (normalizado)
Predicciones
Línea ideal
Precio predicho de cierre del oro
Precio real de cierre del oro
170
180
190
200


Comparación de errores - Bagging
Error MSE (normalizado)
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
0.0014
0.0016
Train
Validation
Test