Facultad de Ciencias Económicas y Empresariales, ICADE

OPTIMIZACIÓN DE LA VALORACIÓN DE OPCIONES AMERICANAS CON MACHINE LEARNING: MÁS ALLÁ DE LONGSTAFF-SCHWARTZ Y MODELOS HÍBRIDOS

Autora: María Vivas Redondo

Directora: María Coronado Vaca

MADRID | Abril 2024

RESUMEN

Este trabajo se centra en explorar el potencial de distintos algoritmos de Machine Learning de Aprendizaje Supervisado en la valoración de opciones americanas, contrastándolos con el modelo Longstaff-Schwartz. Para llevar a cabo esta investigación, se emplean en RStudio los algoritmos KNN (K-Nearest Neighbors), RF (Random Forest), MLP (Multi-Layer Perceptron) y CNN (Convolutional Neural Network). El estudio se enfoca específicamente en la predicción del precio de opciones americanas put de Apple mediante regresión, utilizando datos extraídos de Bloomberg como caso de estudio. Aunque los resultados muestran mejoras en la precisión de las predicciones, evidenciadas por un RMSE (Root Mean Squared Error) inferior al obtenido con el modelo Longstaff-Schwartz, aún persisten áreas de oportunidad para perfeccionar la precisión de las predicciones y abordar posibles sesgos o limitaciones en los modelos. Estos hallazgos subrayan la importancia de seguir investigando y perfeccionando los enfoques de valoración de opciones americanas para mejorar la toma de decisiones financieras y mitigar los riesgos en los mercados financieros.

Palabras clave: Opciones americanas, Machine Learning, Aprendizaje Supervisado, Regresión, Longstaff-Schwartz, KNN (K-Nearest Neighbors), RF (Random Forest), MLP (Multi-Layer Perceptron), CNN (Convolutional Neural Network), RMSE (Root Mean Squared Error).

ABSTRACT

This project focuses on exploring the potential of different Machine Learning Supervised Learning algorithms in the valuation of American options, contrasting them with the Longstaff-Schwartz model. To carry out this research, the algorithms KNN (K-Nearest Neighbors), RF (Random Forest), MLP (Multi-Layer Perceptron) and CNN (Convolutional Neural Network) are employed using RStudio. The project specifically targets the prediction of the price of Apple's put American options through regression, utilizing data extracted from Bloomberg as a case study. Although the results demonstrate improvements in prediction accuracy, evidenced by a lower RMSE (Root Mean Squared Error) compared to the Longstaff-Schwartz model, there are still areas for improvement to enhance predictions precision and address potential biases or limitations in the models. These findings underscore the importance of continuing researching and refining approaches to American options valuation in order to enhance financial decision-making and mitigate risks in financial markets.

Key words: American options, Machine Learning, Supervised Learning, Regression, Longstaff-Schwartz, KNN (K-Nearest Neighbors), RF (Random Forest), MLP (Multi-Layer Perceptron), CNN (Convolutional Neural Network), RMSE (Root Mean Squared Error).

ABREVIATURAS Y ACRÓNIMOS

CNN: Convolutional Neural Network

DNN: Deep Neural Network

GBM: Geometric Brownian Motion

ITM: In The Money

KNN: K-Nearest Neighbors

LGBM: Light Gradient-Boosting Machine

LP: Linear Perceptron

LSMC: Least Square Monte Carlo

MAE: Mean Absolute Error

ML: Machine Learning

MLP: Multi-Layer Perceptron

MSE: Mean Squared Error

PDE: Partial Differential Equation

ReLU: Rectified Linear Unit

RF: Random Forest

RMSE: Root Mean Squared Error

T: Tiempo

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1. Objetivo

El objetivo principal del presente Trabajo de Fin de Grado radica en analizar el potencial de distintos algoritmos de regresión de Machine Learning (ML, por sus siglas en inglés) en la compleja valoración de las opciones americanas. Esto se realizará contrastándolos entre sí y con el reconocido modelo Longstaff-Schwartz.

1.2. Justificación del tema objeto de estudio

El empleo de nuevas tecnologías, como el ML, en la industria financiera está transformando la manera en la que se abordan los problemas financieros. En este contexto, la aplicación de algoritmos de ML representa un hito significativo, promoviendo la capacidad de los modelos para adaptarse dinámicamente a los desafíos que se les presentan.

En el actual escenario financiero, la valoración de opciones americanas presenta considerables retos dada la posibilidad de su ejercicio en cualquier momento de su vida útil. Los métodos tradicionales evidencian ciertas limitaciones al abordar esta complejidad, dando paso a una gran oportunidad para la aplicación de algoritmos de ML en la determinación del precio de estos instrumentos financieros. Estos algoritmos se presentan como una alternativa para una valoración más precisa y eficaz, aspecto crucial para la toma de decisiones financieras, la gestión de riesgos y la eficiencia de los mercados financieros.

Recientes estudios se han centrado en demostrar el potencial del ML mediante el uso de modelos híbridos que combinan algoritmos de ML con técnicas tradicionales, como Ecuaciones Diferenciales Parciales (PDE, por sus siglas en inglés) y Longstaff-Schwartz. Sin embargo, aún queda por investigar el potencial del empleo de algoritmos de ML entrenados exclusivamente con observaciones históricas del mercado financiero, prescindiendo de complejos modelos matemáticos en su entrenamiento.

Teniendo esto en cuenta, este trabajo tiene como objetivo superar los límites que presentan los métodos convencionales en la valoración de opciones americanas. Además, se pretende demostrar si el empleo de distintos algoritmos de ML de Aprendizaje Supervisado, entrenados exclusivamente con datos históricos del mercado financiero, puede mejorar la modelización financiera.

1.3. Metodología

A lo largo del trabajo, la metodología usada se fundamenta en la revisión de literatura sobre valoración de opciones americanas, la extracción de datos de Bloomberg, la implementación de distintos algoritmos de ML y del modelo Longstaff-Schwartz en RStudio, y la posterior contrastación de los resultados obtenidos. En el análisis, se emplean opciones americanas put de Apple como caso de estudio.

Los algoritmos de ML que se van a desarrollar junto al modelo Longstaff-Schwartz son K-Nearest Neighbors (KNN, por sus siglas en inglés), Random Forest (RF, por sus siglas en inglés), Perceptrón Multicapa (MLP, por sus siglas en inglés) y Redes Neuronales Convolucionales (CNN, por sus siglas en inglés). La metodología específica del estudio empírico para cada uno de los algoritmos se detallará más adelante en el epígrafe 3.2. Metodología.

El método empleado para dar respuesta al interrogante planteado ha sido tanto cualitativo como cuantitativo, y deductivo.

Se establece como punto de partida la premisa de que el ML tiene el potencial de valorar las opciones americanas con más precisión y eficiencia que el modelo Longstaff-Schwartz. Para validar esta afirmación, se aplican distintos algoritmos a un caso específico, opciones americanas put de Apple, en RStudio. Si al llevar a cabo el análisis, los resultados no son concluyentes o van en contra de la premisa establecida, esto conllevaría a descartar la aplicabilidad del ML como un principio general en la valoración de las opciones americanas.

1.4. Estructura

La estructura del presente trabajo se articula en las siguientes tres secciones, cada una de las cuales aborda directamente el objeto de estudio.

En la primera sección, se realiza una revisión de la literatura con el propósito de comprender la situación actual en la que se encuentra la valoración de las opciones americanas. Este análisis permite investigar los avances más recientes identificados en diversos estudios, estableciendo así un fundamento que sirve de guía en las fases subsiguientes del trabajo.

En la segunda sección, se lleva a cabo un análisis empírico del caso práctico empleado. Aquí, se implementan distintos algoritmos de ML y el modelo Longstaff-Schwartz en la valoración de opciones americanas put de Apple, usando RStudio como interfaz de programación.

Por último, la tercera sección reúne las conclusiones derivadas de la investigación que se lleva a cabo en el trabajo. En esta última parte, se exponen los hallazgos y aprendizajes obtenidos tanto en la revisión de la literatura como en el análisis empírico.

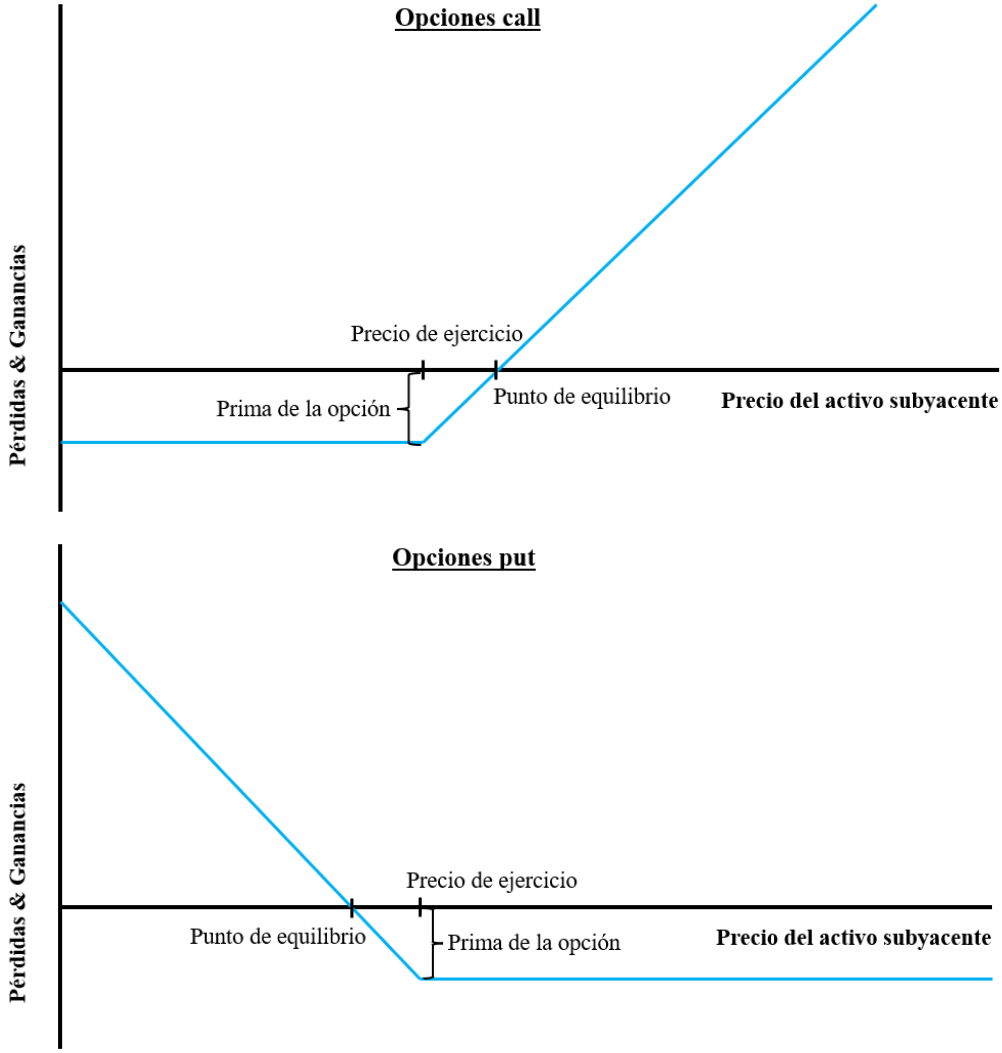
2. REVISIÓN DE LA LITERATURA

Las opciones americanas son instrumentos financieros derivados de naturaleza compleja que otorgan a su titular un derecho, no una obligación, a ejercerlas en cualquier momento durante su vigencia. Este rasgo las diferencia de las opciones europeas, cuyo ejercicio está limitado al vencimiento, y de las opciones bermudas, las cuales únicamente pueden ejercerse en unas fechas predefinidas. Además, dada la flexibilidad que se ofrece al titular de la opción americana, su precio va a ser más elevado en comparación con otros tipos de opciones. La valoración de las opciones americanas representa un desafío significativo para la industria financiera y ha sido objeto de gran investigación por parte de diversos autores.

En las opciones americanas, al haber múltiples fechas posibles para ser ejercitadas, es importante analizar estratégicamente el momento más beneficioso para su ejercicio. Este estudio permite determinar si en cada período conviene ejercerlas o conservarlas sin

ejercer, posibilitando así asignarles un precio preciso con el objetivo de maximizar los beneficios potenciales. En el caso de la adquisición de una opción americana de compra (call), el titular considerará el ejercicio en aquellos períodos en los que el precio de mercado del activo subyacente supere el precio de ejercicio y alcance el punto de equilibrio, habiéndose descontado la prima de la opción pagada. Por otro lado, en la adquisición de una opción americana de venta (put), el titular contemplará el ejercicio en aquellos períodos en los que el precio de mercado del activo subyacente esté por debajo del precio de ejercicio y alcance el punto de equilibrio, habiéndose descontado la prima de la opción pagada (figura 1).

Figura 1: Pérdidas & Ganancias de opciones call y put compradas.



Fuente: Elaboración propia.

El modelo Black-Scholes (1973), diseñado para valorar las opciones europeas, no puede capturar completamente la flexibilidad de ejercicio anticipado que caracteriza a las opciones americanas. Al asumir que el ejercicio solo se produce en el vencimiento, se corre el riesgo de subestimar su verdadero valor. Por ello, se necesitan modelos más sofisticados para evitar que la subestimación resulte en decisiones de inversión desfavorables y en una gestión ineficaz del riesgo financiero.

En este contexto, varios autores han subrayado la importancia de determinar el valor de continuación esperado en cada momento para optimizar la estrategia de ejercicio de las opciones americanas y maximizar los potenciales rendimientos en los mercados financieros. Destacando entre ellos, se encuentran Longstaff y Schwartz (2001), cuyo influyente modelo ha tenido un impacto significativo en la valoración de estos instrumentos financieros complejos. Su relevancia se evidencia al superar los límites a los que se enfrentaban los enfoques anteriores, como los árboles binomiales popularizados por Cox, Ross y Rubinstein (1979), y las PDE empleadas por Hull y White (1990), cuando había múltiples factores complicados que afectaban al valor de las opciones americanas.

El modelo desarrollado por Longstaff y Schwartz (2001) para valorar las opciones americanas se fundamenta en un enfoque numérico basado en la simulación con optimización. Emplea una estimación funcional basada en regresión, utilizando trayectorias de muestras del activo subyacente simuladas con Monte Carlo que se encuentran en el dinero (ITM, por sus siglas en inglés). En la industria financiera, este enfoque también es conocido como el método de Monte Carlo de Mínimos Cuadrados (LSMC, por sus siglas en inglés).

En los últimos años, la aparición de nuevas tecnologías ha desempeñado un papel crucial en abordar el desafío de valorar las opciones americanas. Entre estas innovaciones destaca el ML, que es una rama de la inteligencia artificial y se puede definir como la capacidad de una máquina para imitar el comportamiento humano inteligente (Brown, 2021). Este enfoque ha proporcionado unas herramientas avanzadas que permiten modelar y prever de manera más precisa los movimientos que se producen en el mercado financiero. Además, el ML fomenta la optimización y el procesamiento de los datos del mercado

financiero que provienen de distintas plataformas prestigiosas como Bloomberg o Reuters (Arévalo de Pablos et al., 2024). De esta manera, se va a garantizar un acceso más rápido a información fiable y actualizada, aspecto de gran importancia en la toma de decisiones estratégicas, especialmente teniendo en cuenta la elevada volatilidad asociada a las opciones americanas.

Recientemente, diversos autores han recurrido al empleo del ML para valorar las opciones americanas y otros instrumentos financieros. Los algoritmos de ML, capaces de realizar tareas como clasificación, regresión, clustering o reducción de dimensiones, destacan por su eficacia en el manejo de grandes volúmenes de datos. Su versatilidad y capacidad para manejar conjuntos de datos complejos los convierten en unas herramientas útiles para valorar las opciones americanas en los mercados financieros. Además, su capacidad para identificar los patrones y las relaciones ocultas en los datos facilita la comprensión de la dinámica de la industria financiera, lo cual permite realizar predicciones con más precisión.

Ante esta situación, varios estudios han combinado distintos algoritmos de ML con técnicas tradicionales, como por ejemplo PDE y LSMC, para llevar a cabo la valoración de las opciones americanas. Entre ellos, Anderson y Ulrych (2023) junto a Becker, Cheridito y Jentzen (2019) optaron por Redes Neuronales Profundas (DNN, por sus siglas en inglés); Hoshisashi y Yamada (2023) emplearon MLP; Kanashiro Felizardo, Matsumoto y Del Moral Hernández (2022) usaron CNN; Dubrov (2015) junto con Mohamed, Mehdi y Boubker (2022) se inclinaron por RF; Feng, Liu y Sun (2013) prefirieron KNN; mientras que Malpica y Frias (2019) aplicaron RF, KNN, Light Gradient-Boosting Machine (LGBM, por sus siglas en inglés) y una combinación de estos algoritmos mediante la técnica de stacking.

Los estudios de estos sujetos demuestran la diversidad de enfoques y la tendencia hacia el desarrollo de modelos híbridos, que combinan técnicas tradicionales con las capacidades predictivas de los algoritmos de ML, en la valoración de las opciones americanas. Las conclusiones de estas investigaciones se resumen a continuación en la tabla 1.

Tabla 1: Modelos híbridos para estimar el precio de opciones americanas.

Autor - Año	Algoritmo ML híbrido	Conclusiones
Anderson y Ulrych (2023)	DNN	• Entrenando DNN con PDE y el modelo de volatilidad estocástica de Heston (1993) mejora la predicción del precio en términos del equilibrio entre velocidad y precisión, en comparación con los métodos estándares de valoración de las opciones americanas.
Becker, Cheridito y Jentzen (2019)	DNN	• La adaptación de LSMC empleando DNN proporciona una estimación del precio de la opción americana con sesgo bajo. • El entrenamiento preciso de DNN exige mayor tiempo computacional.
Dubrov (2015)	RF	• RF entrenado con LSMC siempre obtiene mejores resultados que el modelo Longstaff-Schwartz. • Su sencillez y precisión lo convierten en un algoritmo idóneo.
Feng, Liu y Sun (2013)	KNN	• El Error Cuadrático Medio de la Raíz (RMSE, por sus siglas en inglés) de los experimentos numéricos muestran que KNN entrenado con LSMC es prometedor para valorar las opciones americanas. • Se necesita más investigación para determinar cuál es el umbral de dimensión necesario para que los estimadores de KNN sean viables.
Hoshisashi y Yamada (2023)	MLP	• Entrenar MLP con LSMC no es mucho mejor que el modelo Longstaff-Schwartz cuando hay pocas fechas de ejercicio. Sin embargo, su eficiencia y precisión en términos de RMSE es mejor cuando hay gran cantidad de fechas. • Se requieren altos esfuerzos y recursos computacionales. • Es necesario entrenar frecuentemente MLP en respuesta a las condiciones del mercado financiero.
Kanashiro Felizardo, Matsumoto y Del Moral Hernández (2022)	CNN	• El entrenamiento de CNN con LSMC permite mejorar el rendimiento del punto óptimo de parada. • La mejora se logra mediante la transformación de la información histórica en un estado de Markov, junto con la extracción de las características de la capa de CNN. • Los resultados muestran que esta metodología mejora el valor esperado en comparación con el modelo Longstaff-Schwartz.
Malpica y Frias (2019)	RF, KNN, LGBM y Stacking	• Los modelos RF, KNN y LGBM entrenados con LSMC obtienen una precisión aproximada en términos de Error Absoluto Medio (MAE, por sus siglas en inglés), sin diferencias significativas entre ellos. • Su combinación mediante stacking aumenta la precisión en términos de MAE. • En promedio, las estimaciones del precio se acercan al valor real.
Mohamed, Mehdi y Boubker (2022)	RF	• El precio estimado por RF entrenado con LSMC es similar al obtenido mediante el modelo Longstaff-Schwartz, pero ligeramente superior en términos de Error Cuadrático Medio (MSE, por sus siglas en inglés). • RF generalmente funciona mejor en el contexto de modelos multidimensionales no lineales y altamente correlacionados debido a su estructura de árbol aleatoria.

Fuente: Elaboración propia.

Dichas conclusiones muestran que el empleo de modelos híbridos, centrados en identificar el valor esperado condicional de continuación y la regla de parada óptima para valorar las opciones americanas, es más eficiente y preciso que el modelo Longstaff-Schwartz. Sin embargo, aún queda por resolver si el entrenamiento supervisado de algoritmos de ML exclusivamente con datos históricos del mercado financiero, sin recurrir a métodos tradicionales, logra mejores resultados que el modelo Longstaff-Schwartz.

Por ello, el objetivo de este trabajo es desarrollar un enfoque para predecir el precio de las opciones americanas utilizando únicamente algoritmos de ML y datos históricos del mercado financiero. Se busca prescindir de modelos híbridos, como los implementados por los autores mencionados anteriormente, y centrarse en entrenar los algoritmos KNN, RF, MLP y CNN empleando los factores del mercado financiero más influyentes en las opciones americanas. Este método permite aumentar la adaptabilidad y simplificar el proceso de valoración, evitando la complejidad asociada con las fórmulas matemáticas utilizadas en los modelos híbridos. Además, se incluyen nuevas variables que no se han tenido en cuenta en los estudios anteriores, como las sensibilidades de las opciones americanas, que son de gran relevancia en la predicción del precio y pueden mejorar significativamente la precisión.

3. ANÁLISIS EMPÍRICO

3.1. Datos

3.1.1. Origen de los datos

La base de datos en la que se basa este trabajo contiene un histórico diario con información de distintas opciones americanas de Apple, con un período temporal que abarca entre el 18 de enero de 2018 y el 19 de enero de 2024. Los datos han sido recopilados de Bloomberg, plataforma de software líder en la industria financiera, que ofrece datos en tiempo real, herramientas de análisis, noticias y distintas funciones de negociación.

Durante el proceso, se ha presentado el siguiente desafío: mientras que “OMON” de la terminal de Bloomberg ofrece información sobre las opciones vigentes, no incluye datos de opciones que ya han vencido. Al necesitar extraer datos históricos de opciones ya vencidas, como solución se ha empleado en Excel la función “=BDS("AAPL US Equity"; "OPT_CHAIN";"SINGLE_DATE_OVERRIDE=YYYYMMDD")”, la cual devuelve los tickers de todas las opciones que se encuentran en el rango temporal especificado. Posteriormente, mediante el uso de la tabla de datos históricos del Add-In de Bloomberg en Excel, se han extraído los datos históricos de las opciones americanas de Apple, así como de las acciones de Apple y de los bonos del Tesoro de Estados Unidos.

Debido a la diversidad de formatos y al gran volumen de información, se han creado macros en Visual Basic de Excel para estandarizar los datos y adaptarlos para su posterior implementación en RStudio. Los datos de las opciones americanas se han dividido en dos archivos Excel, denominados "Appleoptions1" y "Appleoptions2", debido a las limitaciones de Excel en cuanto al número máximo de filas (1.048.576). Por otro lado, las observaciones sobre las acciones de Apple y los bonos del Tesoro de Estados Unidos se han guardado en otros dos archivos Excel, denominados “Applestocks” y “TNOTES”.

Tras importar los archivos Excel en RStudio, se ha construido la tabla final mediante su fusión usando “DATES” como elemento común. Como resultado, se han obtenido 1.661.772 observaciones y 23 variables, con un peso total de en torno a 0,3 gigabytes de memoria.

3.1.2. Preprocesamiento de los datos

En la tabla final hay valores perdidos, y se han valorado distintas alternativas para manejarlos, como tomar la media, el máximo, el mínimo, cero o eliminar todos los datos de ese día para la opción correspondiente. Finalmente, se ha optado por la última alternativa al darse en algunos casos la ausencia de los datos durante varios días consecutivos. Dada la elevada volatilidad y la variabilidad en la valoración de las opciones americanas, esta situación puede afectar significativamente los resultados del estudio. Como consecuencia, las observaciones finales se reducen a un total de 988.384.

Además, al contener la tabla información tanto de opciones americanas call como put, se han extraído las opciones americanas put con las que se va a trabajar. Como resultado, se reduce el conjunto de datos a 476.395 observaciones correspondiendo a 1.267 opciones americanas put.

3.1.3. Muestra y división de los datos

Para facilitar el manejo de los datos se ha realizado un muestreo aleatorio para obtener un cinco por ciento del número total de observaciones (23.819). Esto permite realizar un análisis y construir modelos con eficiencia sin comprometer significativamente la precisión de los resultados.

Por otro lado, se ha procedido a dividir los datos en dos conjuntos: entrenamiento y prueba. Esta partición es esencial en el contexto de los modelos predictivos de ML que se van a emplear, ya que permite evaluar el rendimiento del modelo en datos no utilizados durante el entrenamiento. Si se utilizaran todos los datos para estimar cada modelo, no se podría obtener una medida precisa del grado de error en las predicciones pues el error en el conjunto de entrenamiento es siempre cero. Además, si se predijese la variable target para un individuo que no estuviera en la base de datos, no se conocerían los auténticos valores de la variable target para ese individuo y tampoco el error de predicción.

En el conjunto de entrenamiento, que contiene el 70% de los datos (16.673 observaciones), se ha ajustado y estimado el modelo para que pueda aprender de los datos y encontrar las relaciones existentes entre las variables. Mientras tanto, en el conjunto de prueba, que contiene el 30% de los datos (7.146 observaciones), se ha reservado una parte de los datos originales para realizar predicciones sobre ellos y comparar los valores predichos de la variable objetivo con los valores reales, lo que permite evaluar el error de predicción y las medidas de rendimiento predictivo.

Para evitar sesgos, la asignación de datos a los conjuntos de entrenamiento y prueba se ha realizado de forma aleatoria. De esta manera, se asegura una representación adecuada de los datos en ambos conjuntos, lo que garantiza una evaluación precisa del modelo.

3.1.4. Variables

A partir de las 23 variables extraídas de Bloomberg, se han creado cuatro nuevas que se consideran relevantes para predecir el precio de las opciones americanas put: “DAYS_UNTIL_MATURITY”, “PAYOFF”, “MAX_PAYOFF_PER_OPTION” y “MAXIMUM_DATE_PAYOFF_OPTION”.

En primer lugar, “DAYS_UNTIL_MATURITY” indica los días restantes hasta el vencimiento de cada opción americana put. La proximidad al vencimiento es un factor crucial en la determinación del precio. A medida que el vencimiento se acerca, la incertidumbre sobre el comportamiento futuro de las acciones de Apple se reduce, lo que puede influir en la volatilidad y, por lo tanto, en el precio de las opciones americanas put de Apple. Además, cuantos menos días queden, menor será el valor temporal de las opciones americanas put.

En segundo lugar, “PAYOFF” es la diferencia entre el precio actual de la acción de Apple y el precio de ejercicio de la opción americana put de Apple. Un rendimiento positivo implica beneficios, mientras que un rendimiento negativo resulta en pérdidas. Conocer el rendimiento es clave para decidir sobre el ejercicio de la opción americana put.

En tercer lugar, “MAX_PAYOFF_PER_OPTION” permite registrar el rendimiento máximo histórico de cada opción americana put. En este trabajo al estar utilizando una muestra aleatoria de observaciones diarias de distintas opciones americanas put de Apple, esta variable ayuda a identificar patrones o tendencias en el comportamiento del precio y proporciona una referencia para evaluar su desempeño futuro.

Por último, “MAXIMUM_DATE_PAYOFF_OPTION” indica cuántos días quedaban para el vencimiento de cada opción americana put cuando se produjo el rendimiento máximo. Comprender en qué momento de la vida de la opción americana put se alcanzó el mejor rendimiento histórico puede proporcionar información sobre cómo la proximidad al vencimiento puede influir en la rentabilidad.

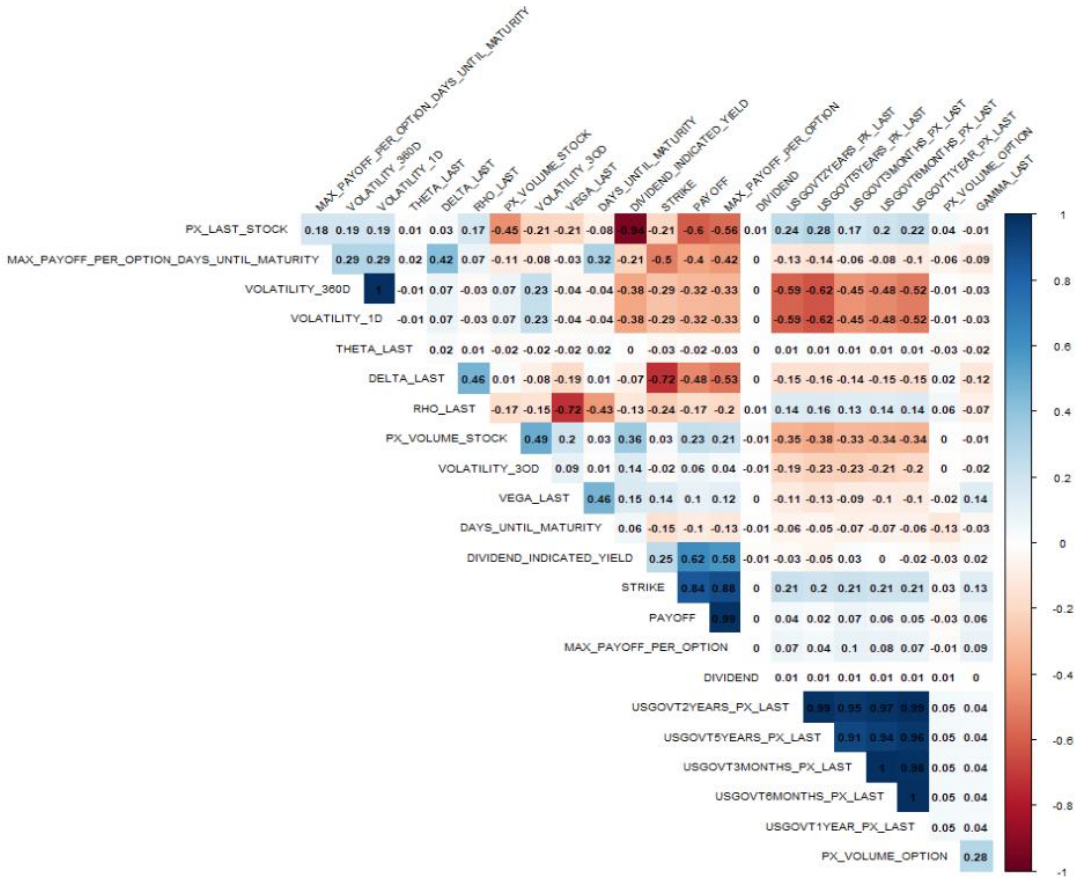
Además, dado que Bloomberg proporciona la volatilidad anual de las acciones, pero no la diaria, ésta se ha calculado mediante la volatilidad anual extraída de Bloomberg. Esta

variable es importante y de mayor relevancia al estar trabajando con un histórico diario. La volatilidad diaria permite comprender más detalladamente las fluctuaciones diarias de los precios en el mercado financiero. Asimismo, proporciona una evaluación y gestión más efectiva del riesgo.

Como consecuencia, el número total de variables asciende a 28, en las cuales hay contenidas variables de tipo cualitativas, cuantitativas y temporales.

Para simplificar el análisis y evitar la multicolinealidad, se han eliminado las variables categóricas, temporales y aquellas numéricas que presentan gran correlación y se considera que se puede prescindir de ellas. Se ha utilizado una matriz de correlación (figura 2) para evaluar la relación entre las variables e identificar aquellas que están altamente correlacionadas. Se ha excluido específicamente la variable target "PX_LAST_OPTION".

Figura 2: Matriz de Correlación.



Fuente: Elaboración propia.

Analizando el resultado obtenido, se ha procedido a eliminar las siguientes variables: “VOLATILITY_360D”, “VOLATILITY_3OD”, “DIVIDEND”, “DELTA_LAST”, “PAYOFF”, “USGOVT3MONTHS_PX_LAST”, “USGOVT6MONTHS_PX_LAST”, “USGOVT2YEARS_PX_LAST” y “USGOVT5YEARS_PX_LAST”. Estas variables tienen una alta correlación con otras y son redundantes en el análisis al estar representadas de manera más efectiva por otras en el conjunto de datos. En lo que respecta a los bonos del Tesoro de Estados Unidos, los cuales se usan como aproximación a la tasa libre de riesgo, se han seleccionado aquellos que tienen una frecuencia anual en los pagos de intereses al estar trabajando con opciones americanas put de Apple que tienen un vencimiento anual.

Aunque algunas variables como “STRIKE”, “DIVIDEND_INDICATED_YIELD”, “PX_LAST_STOCK”, “RHO_LAST” y “VEGA_LAST” presentan una correlación elevada con otras, se ha considerado necesario incluirlas en el análisis para valorar las opciones americanas put de Apple adecuadamente. Además, todas estas variables menos las sensibilidades “RHO_LAST” y “VEGA_LAST” son necesarias para poder aplicar posteriormente el modelo Longstaff-Schwartz en la predicción del precio de las opciones americanas put.

Como consecuencia, las variables numéricas se reducen a 15 (tabla 2). De todas estas, 14 son predictivas formando parte de la X y “PX_LAST_OPTION” es la variable target o Y.

Es importante destacar que en el contexto del modelo tradicional Longstaff-Schwartz, se han seleccionado específicamente las siguientes variables: “STRIKE”, “VOLATILITY_1D”, “PX_LAST_STOCK”, “USGOVT1YEAR_PX_LAST”, “DIVIDEND_INDICATED_YIELD” y “YEARS_UNTIL_MATURITY”. La variable “YEARS_UNTIL_MATURITY” ha sido creada a partir de la variable “DAYS_UNTIL_MATURITY” y muestra el tiempo restante expresado en términos anualizados hasta el vencimiento de las opciones americanas put. La necesidad de esta variable radica en su inclusión en el modelo Longstaff-Schwartz, donde se utiliza para calcular el valor presente esperado de la opción americana put en los distintos períodos.

Tabla 2: Listado de las variables incluidas en los algoritmos de ML junto al tipo de variable y su descripción.

Variable	Tipo	Descripción
STRIKE	Cuantitativa Continua	Precio de ejercicio diario de las opciones americanas put.
PX_LAST_OPTION	Cuantitativa Continua	Precio diario de las opciones americanas put.
PX_VOLUME_OPTION	Cuantitativa Discreta	Volumen de transacciones diarias de las opciones americanas put.
GAMMA_LAST	Cuantitativa Continua	Gamma diaria de las opciones americanas put, que mide la sensibilidad de delta ante cambios en el precio del activo subyacente.
VEGA_LAST	Cuantitativa Continua	Vega diaria de las opciones americanas put, que mide la sensibilidad del precio ante cambios en la volatilidad implícita del activo subyacente.
RHO_LAST	Cuantitativa Continua	Rho diaria de las opciones americanas put, que mide la sensibilidad del precio ante cambios en la tasa de interés libre de riesgo.
PX_LAST_STOCK	Cuantitativa Continua	Precio diario de las acciones.
PX_VOLUME_STOCK	Cuantitativa Discreta	Volumen de transacciones diarias de las acciones.
DIVIDEND_INDICATED_YIELD	Cuantitativa Continua	Rendimiento indicado por dividendo de una acción.
USGOVT1YEAR_PX_LAST	Cuantitativa Continua	Precio diario de los bonos del Tesoro de Estados Unidos con vencimiento anual, que se emplea como aproximación a la tasa libre de riesgo.
DAYS_UNTIL_MATURITY	Cuantitativa Discreta	Días restantes hasta el vencimiento de las opciones americanas put.
MAX_PAYOFF_PER_OPTION	Cuantitativa Continua	Máxima rentabilidad alcanzada por las opciones americanas put.
MAX_PAYOFF_PER_OPTION_DAYS_UNTIL_MATURITY	Cuantitativa Discreta	Días restantes hasta el vencimiento de las opciones americanas put cuando se alcanzó la máxima rentabilidad.
VOLATILITY_1D	Cuantitativa Continua	Volatilidad diaria de las acciones.

Fuente: Elaboración propia.

3.2. Metodología

A continuación, se procede a explicar, implementar y analizar cada uno de los algoritmos de ML seleccionados para resolver el problema de regresión planteado en este trabajo junto con el modelo Longstaff-Schwartz. Los algoritmos de ML son KNN, RF, MLP y CNN.

3.2.1. KNN

El algoritmo KNN es una técnica de Aprendizaje Supervisado ampliamente utilizada en los problemas de clasificación y regresión.

KNN es un algoritmo no paramétrico, lo que significa que no requiere un proceso de entrenamiento específico. Dado que KNN no tiene una etapa específica para el aprendizaje, se considera un algoritmo de aprendizaje perezoso. El mecanismo de este algoritmo para determinar el resultado se basa en calcular la distancia entre la nueva muestra y las otras muestras existentes en el conjunto de entrenamiento (Klidbary y Arabameri, 2023). Es decir, al recibir un nuevo individuo X, busca en todo el conjunto de entrenamiento quiénes son los K casos más similares a él (los K vecinos más cercanos). Dependiendo del problema, los vecinos se utilizan para clasificar la muestra en una categoría específica (clasificación) o para predecir su valor resumen (regresión). Como en este trabajo estamos ante un caso de regresión, el algoritmo procederá a predecir la media de los K vecinos más cercanos al nuevo individuo.

La elección adecuada del hiperparámetro K es crucial, ya que un valor muy pequeño puede llevar a un sobreajuste al ajustar demasiado el ruido en los datos, mientras que un valor demasiado grande puede introducir un sesgo excesivo en el modelo. Por ello, es importante encontrar un equilibrio para obtener resultados precisos y generalizables.

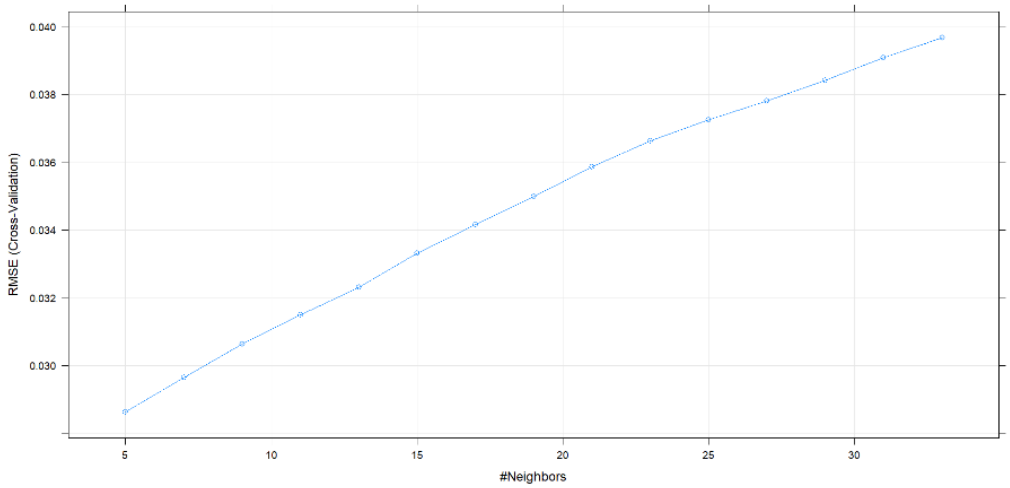
La simplicidad, fácil implementación e interpretabilidad de los resultados, alta precisión, adecuación para datos no lineales y una amplia gama de aplicaciones pueden considerarse como las ventajas del algoritmo KNN (Klidbary y Arabameri, 2023).

A continuación, se procede a aplicar el algoritmo KNN a los datos de las opciones americanas put de Apple para predecir su precio.

El modelo KNN se ha construido empleando la función “train()” del paquete “Caret” de RStudio. Para su entrenamiento, se han utilizado los datos del conjunto de entrenamiento normalizados para que se encuentren en la misma escala. “PX_LAST_OPTION” es la variable target y todas las demás variables son empleadas para hacer la predicción. Específicamente, se ha indicado en el argumento "method" que el algoritmo a utilizar es KNN. Además, se ha aplicado la validación cruzada 10 veces para evaluar el rendimiento del modelo. Esto implica dividir los datos en 10 partes iguales, entrenar el modelo en 9 partes y evaluar su rendimiento en la parte restante. Este proceso se ha repetido 10 veces para obtener una estimación más precisa del rendimiento del modelo en nuevos datos. Para la selección de los hiperparámetros, se ha empleado “tuneLength”, donde se especifica que se realizan 15 iteraciones para encontrar el valor óptimo de K.

En el siguiente gráfico (figura 3), se puede observar cómo, en función del valor que se asigne a K, el modelo tendrá un valor diferente de RMSE. RMSE es una medida que proporciona la diferencia promedio entre los valores predichos y los valores reales. El número óptimo de K a emplear es cinco al minimizar el valor de RMSE.

Figura 3: Gráfico RMSE en función de K.

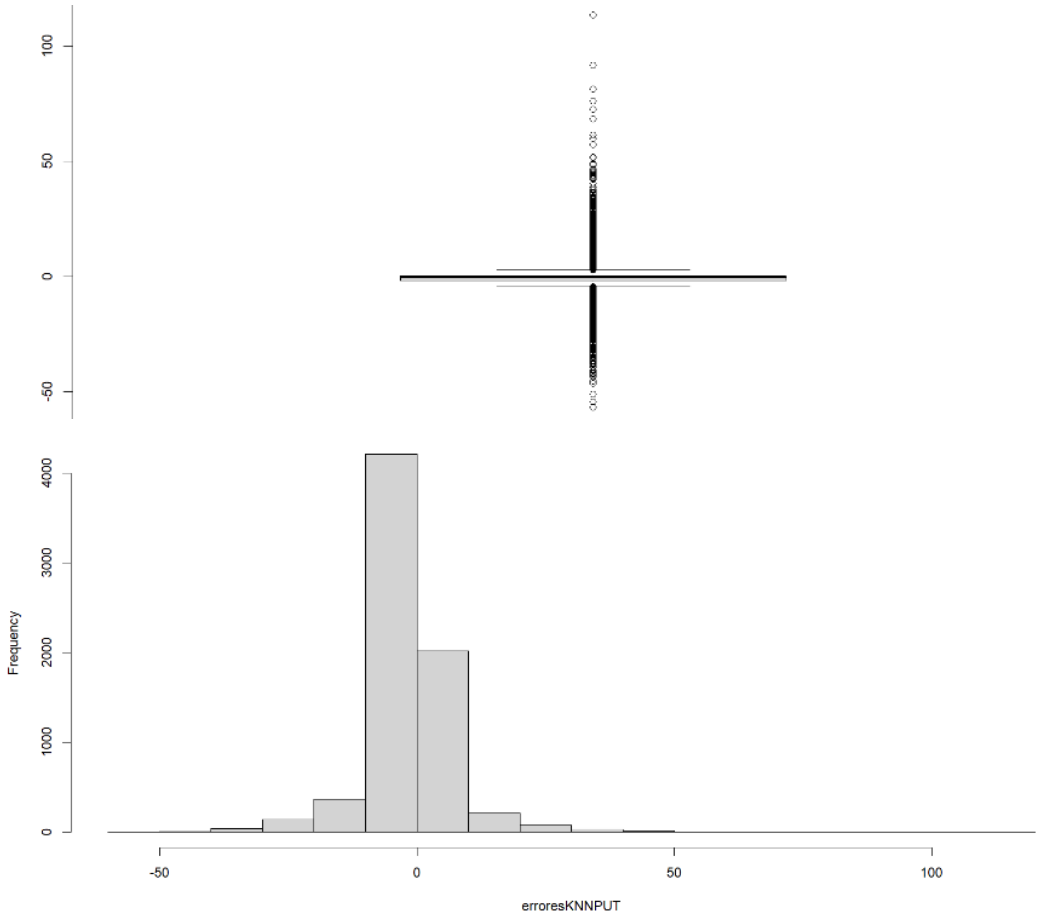


Fuente: Elaboración propia.

Dado que los datos están normalizados, se devuelven a su estado original para poder comparar las predicciones del precio realizadas por el modelo KNN con los datos originales de las opciones americanas put de Apple.

Tanto en el boxplot como en el histograma (figura 4), se puede observar que la mayoría de los errores del modelo KNN están concentrados alrededor del valor 0 y no hay errores superiores a 50 en términos absolutos. La frecuencia más alta corresponde a valores negativos cercanos a 0. Esto sugiere que, en general, el modelo KNN tiende a producir resultados cercanos al valor esperado y se desempeña bien. Sin embargo, el hecho de que haya un mayor número de errores negativos que positivos indica una tendencia a sobrevalorar las opciones americanas put de Apple.

Figura 4: Boxplot & Histograma Errores de predicción de KNN.



Fuente: Elaboración propia.

Finalmente, para ver la precisión del modelo KNN se ha calculado su RMSE, obteniendo un valor de 8,538767. Esto significa que, en promedio, las predicciones del modelo tienen una diferencia de aproximadamente 8,54 unidades en relación con los valores reales.

3.2.2. RF

RF es uno de los algoritmos de ML más populares para resolver los problemas de clasificación y regresión. RF se adapta fácilmente a las relaciones no lineales presentes entre los datos, lo que le hace propenso a predecir con mayor precisión que la regresión lineal.

RF es un ensemble de modelos mediante bagging destinado a mejorar la precisión de la predicción y reducir la varianza a través de un conjunto de árboles de decisión, especialmente si las predicciones tienen correlación negativa o no están correlacionadas. En un RF, las observaciones se muestrean aleatoriamente con reemplazo (bootstrap) para crear una muestra de arranque del mismo tamaño que el conjunto de datos original. Luego, las observaciones se dividen repetidamente utilizando reglas de decisión binarias, que se caracterizan por un punto de corte en un predictor específico en el conjunto de datos. El predictor y el punto de corte del predictor se eligen para dividir las observaciones en dos grupos (He et al., 2018). Este proceso se repite para crear múltiples árboles de decisión dando lugar a un "bosque". Después, las predicciones de cada árbol se combinan para obtener una predicción final más precisa y robusta. En el supuesto de clasificación, se emplea el voto de la mayoría, mientras que para regresión se utiliza el promedio.

A continuación, se procede a aplicar el algoritmo RF a los datos de las opciones americanas put de Apple para predecir su precio.

El modelo RF se ha creado empleando la función "train()" del paquete "Caret" de RStudio.

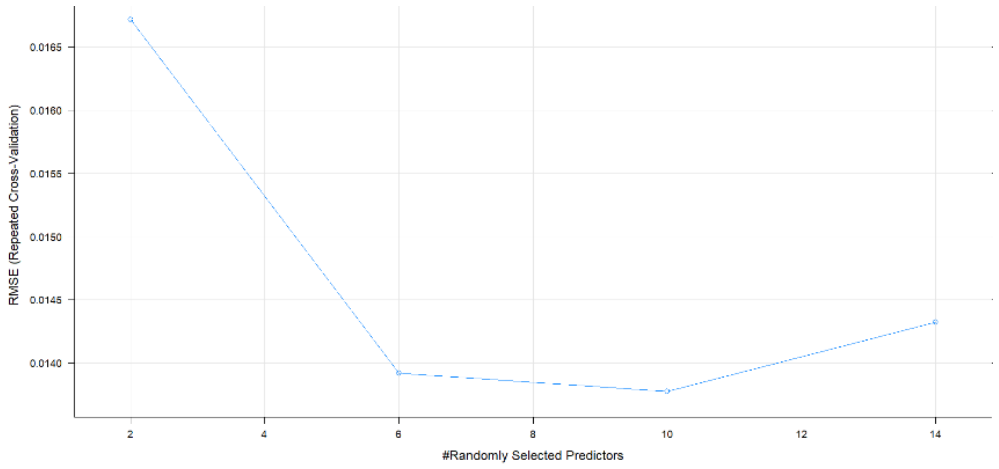
Para la construcción del algoritmo, en primer lugar, se ha creado la función "trainControl" para establecer los parámetros de control del entrenamiento del modelo. En esta función, se especifica el método de validación cruzada repetida, con 10 folds y tres repeticiones. Además, se muestra la impresión de mensajes durante la iteración ("verboseIter"), se

desactivan las probabilidades de clase (“classProbs”) al tratarse de un problema de regresión, y se define la función de resumen de los resultados (“defaultSummary”).

Posteriormente, se ha entrenado el modelo predictivo de regresión especificando que el método a emplear es RF sobre el conjunto de datos de entrenamiento normalizados. En él se utiliza la función de control previamente creada y con “tuneLength” se especifica que se van a probar cuatro modelos diferentes durante el proceso de ajuste de los hiperparámetros. El objetivo es obtener el mejor rendimiento del modelo RF. Los hiperparámetros probados son 2, 6, 10 y 14.

En el siguiente gráfico (figura 5), se puede observar cómo, en función de la estructura que se adopte, el modelo RF tendrá un valor diferente de RMSE. El hiperparámetro óptimo a emplear por el modelo es 10, ya que corresponde al valor mínimo de RMSE. Esto indica que el modelo entrenado con un valor 10 de hiperparámetro tiene el mejor rendimiento en términos de precisión en comparación con los otros valores de hiperparámetros probados.

Figura 5: Gráfico RMSE de RF en función del número de hiperparámetros.

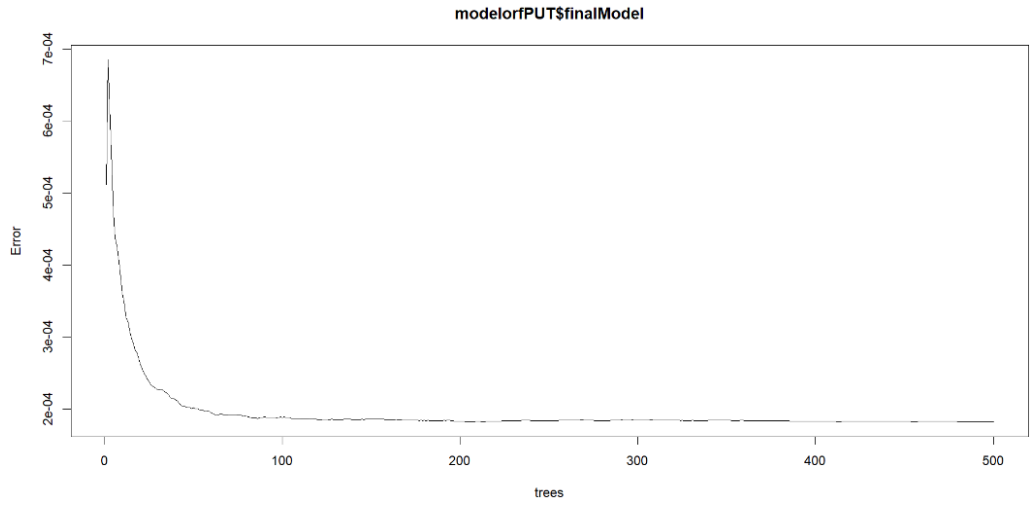


mtry	RMSE	Rsquared	MAE
2	0.01671634	0.9780883	0.006737356
6	0.01391632	0.9835039	0.005062166
10	0.01377315	0.9837533	0.004930576
14	0.01431913	0.9823263	0.005122383

Fuente: Elaboración propia.

Por otro lado, se puede ver cómo varía el error del modelo RF en función del número de árboles utilizados en el siguiente gráfico (figura 6). Se puede observar cómo, a partir de 100 árboles, la disminución del error es mínima. Esto sugiere que agregar más árboles al modelo no supone una mejora significativa en la precisión de la predicción del precio de las opciones americanas put de Apple, lo cual permite mejorar el rendimiento del modelo.

Figura 6: Gráfico Error del modelo RF en función del número de árboles.



Fuente: Elaboración propia.

Dado que el modelo RF no se puede visualizar como los árboles de decisión, se ha analizado la importancia de las variables a través de la función "varImp" (figura 7).

Figura 7: Importancia de las variables en el modelo RF.

	Overall
STRIKE	100.0000
MAX_PAYOFF_PER_OPTION	29.9976
GAMMA_LAST	26.3520
RHO_LAST	12.0151
PX_LAST_STOCK	10.2003
THETA_LAST	9.8956
DIVIDEND_INDICATED_YIELD	7.5129
VEGA_LAST	7.3286
USGOVT1YEAR_PX_LAST	5.8527
PX_VOLUME_OPTION	5.6126
VOLATILITY_1D	1.4298
MAX_PAYOFF_PER_OPTION_DAYS_UNTIL_MATURITY	0.8779
DAYS_UNTIL_MATURITY	0.4600
PX_VOLUME_STOCK	0.0000

Fuente: Elaboración propia.

En el análisis, se puede observar que las variables “STRIKE”, “MAX_PAYOFF_PER_OPTION” y “GAMMA_LAST” son las tres más influyentes en la predicción del precio de las opciones americanas put. Es importante destacar la variable “STRIKE”, la cual tiene asignado el valor 100. Esta cifra tan elevada se debe a la alta correlación que tiene con la variable target “PX_LAST_OPTION”. Esto implica que pequeños cambios en el valor de esta variable van a tener un impacto significativo en las predicciones que realice el modelo RF sobre el precio de las opciones americanas put.

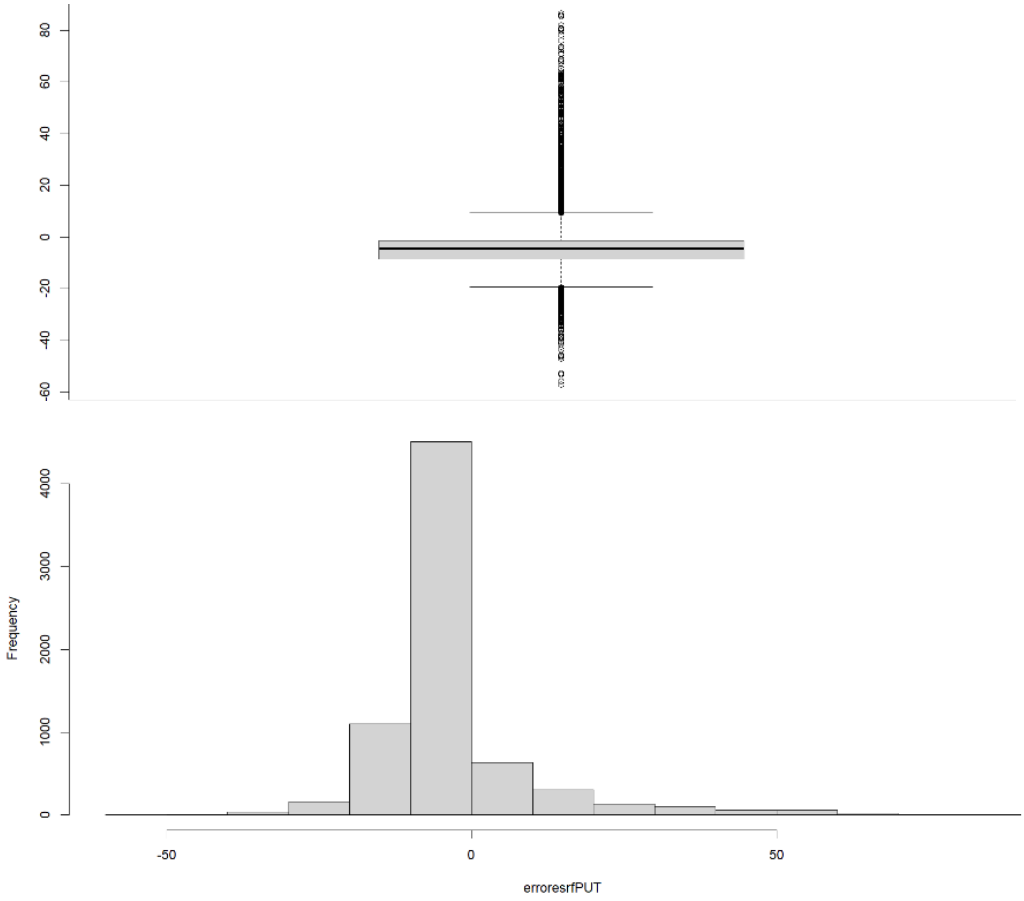
Por otro lado, las variables de menor importancia para el modelo son “PX_VOLUME_STOCK”, “DAYS_UNTIL_MATURITY”, “VOLATILITY_1D” y “MAX_PAYOFF_PER_OPTION_DAYS_UNTIL_MATURITY”. Los valores de estas variables son cercanos a 0, teniendo incluso “PX_VOLUME_STOCK” asignado el valor 0. Esto sugiere que las variables anteriores van a tener un impacto mínimo o insignificante en la predicción del precio de las opciones americanas put, y, por lo tanto, el modelo RF puede prescindir de ellas sin afectar significativamente a los resultados. Además, la eliminación de estas variables va a permitir simplificar la estructura del modelo RF y reducir la complejidad del mismo, sin afectar a la precisión de las predicciones que se lleven a cabo.

Dado que los datos están normalizados, se devuelven a su estado original para poder comparar las predicciones del precio realizadas por el modelo RF con los datos originales de las opciones americanas put de Apple.

Tanto en el boxplot como en el histograma (figura 8), se puede observar que la mayoría de los errores del modelo RF están concentrados alrededor del valor 0. La frecuencia de errores es más elevada en los valores negativos que en los positivos, alcanzando el máximo en los valores negativos cercanos a 0. Esta mayor frecuencia de errores negativos sugiere que el modelo RF tiende a sobrevalorar el precio de las opciones americanas put de Apple. Sin embargo, a pesar de que los errores superiores a 50 en términos absolutos son poco frecuentes, se puede ver que en esa cifra hay más valores positivos que negativos. Estos errores positivos grandes pueden ocurrir en aquellas situaciones en las que el precio real de la opción americana put es mucho más alto de lo que el modelo RF

predice. Esto es posiblemente debido a movimientos abruptos e inesperados en el mercado financiero o cambios repentinos en la volatilidad.

Figura 8: Boxplot & Histograma Errores de predicción de RF.



Fuente: Elaboración propia.

Finalmente, para ver la precisión del modelo RF se ha calculado su RMSE, obteniendo un valor de 13,54564. Esto significa que, en promedio, las predicciones del modelo RF tienen una diferencia de aproximadamente 13,55 unidades en relación con los valores reales.

3.2.3. MLP

Las redes neuronales artificiales son algoritmos de ML que se pueden emplear tanto para clasificación como para regresión.

Las MLP son redes neuronales artificiales feedforward con múltiples capas completamente conectadas entre sí que utilizan funciones de activación no lineales para entrenarse.

Un perceptrón simple (LP, por sus siglas en inglés) tiene limitaciones en cuanto a la capacidad de mapeo de entrada-salida deseada. Esto se debe a que solo contiene una neurona por pesos sinápticos adaptables y sesgo. La limitación anterior se puede abordar utilizando una red neuronal MLP con más nodos de entrada de datos y capas de salida intercaladas con nodos de capas ocultas (Isabona et al., 2022). En una MLP, cada neurona en las capas ocultas y de salida toma entradas ponderadas de las neuronas que se encuentran en la capa anterior. Esto va a permitir una mayor capacidad de procesamiento y modelado de la información. Además, la aplicación de funciones de activación no lineales, como la función sigmoide o la función de activación Rectified Linear Unit (ReLU, por sus siglas en inglés), permite capturar relaciones más complejas y no lineales en los datos.

A continuación, se procede a aplicar el algoritmo MLP a los datos de las opciones americanas put de Apple para predecir su precio.

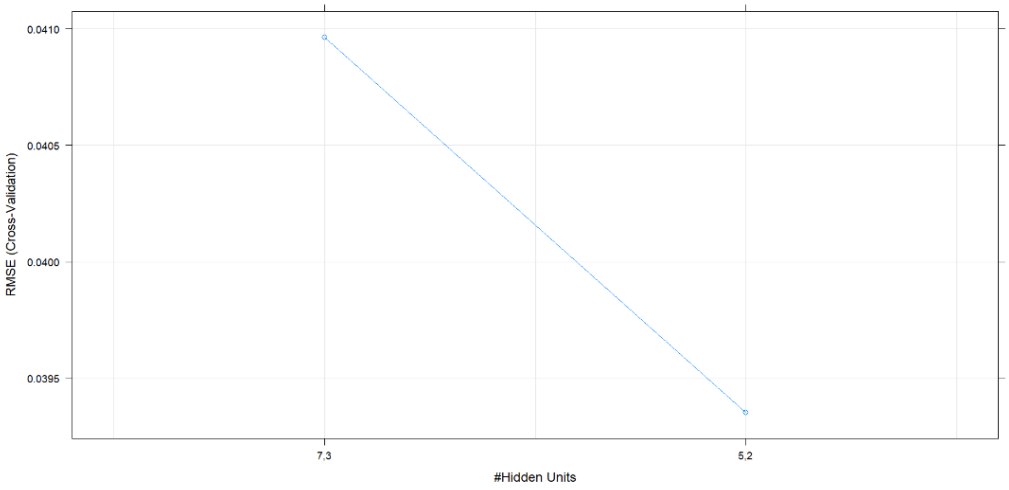
En primer lugar, a través de “expand.grid” se define un conjunto de cuadrícula para la búsqueda de los hiperparámetros del modelo MLP. En concreto, considerando la dimensión de las variables involucradas en el modelo, se han especificado dos posibles tamaños de dos capas ocultas: “7, 3” y “5, 2”.

El algoritmo MLP se ha construido empleando la función “train()” del paquete “Caret” de RStudio. Para su entrenamiento se han utilizado los datos del conjunto de entrenamiento normalizados para que todos estén en la misma escala. “PX_LAST_OPTION” es la variable target y todas las demás variables son empleadas para hacer la predicción del precio de las opciones americanas put de Apple. Específicamente, se indica en el argumento "method" que el algoritmo a utilizar es MLP. Además, se aplica la validación cruzada tres veces para evaluar el rendimiento del modelo. Esto implica dividir los datos en tres partes iguales, entrenar el modelo en dos partes y evaluar su rendimiento en la parte restante. Asimismo, se realiza un

preprocesamiento de los datos, centrando y escalando las características. Para la búsqueda de los hiperparámetros se utiliza la cuadrícula definida anteriormente en "tuneGrid = mlp_grid". Por otro lado, se establece "linOut = TRUE" para obtener una salida lineal en lugar de una logarítmica. Finalmente, el rendimiento del modelo MLP se evalúa utilizando RMSE y se emplea la función de activación ReLU, función no lineal definida como $f(x) = max(0, x)$.

En el siguiente gráfico (figura 9), se puede observar cómo, en función de la estructura que se adopte, el modelo MLP tendrá un valor diferente de RMSE. El modelo MLP va a tener una estructura con dos capas, donde la primera capa contiene cinco neuronas y la segunda dos neuronas, al minimizar el RMSE.

Figura 9: Gráfico RMSE de MLP en función de la estructura de la neurona.



size	RMSE	Rsquared	MAE
7,3	0.04096338	0.8571147	0.02442546
5,2	0.03935293	0.8793593	0.02385157

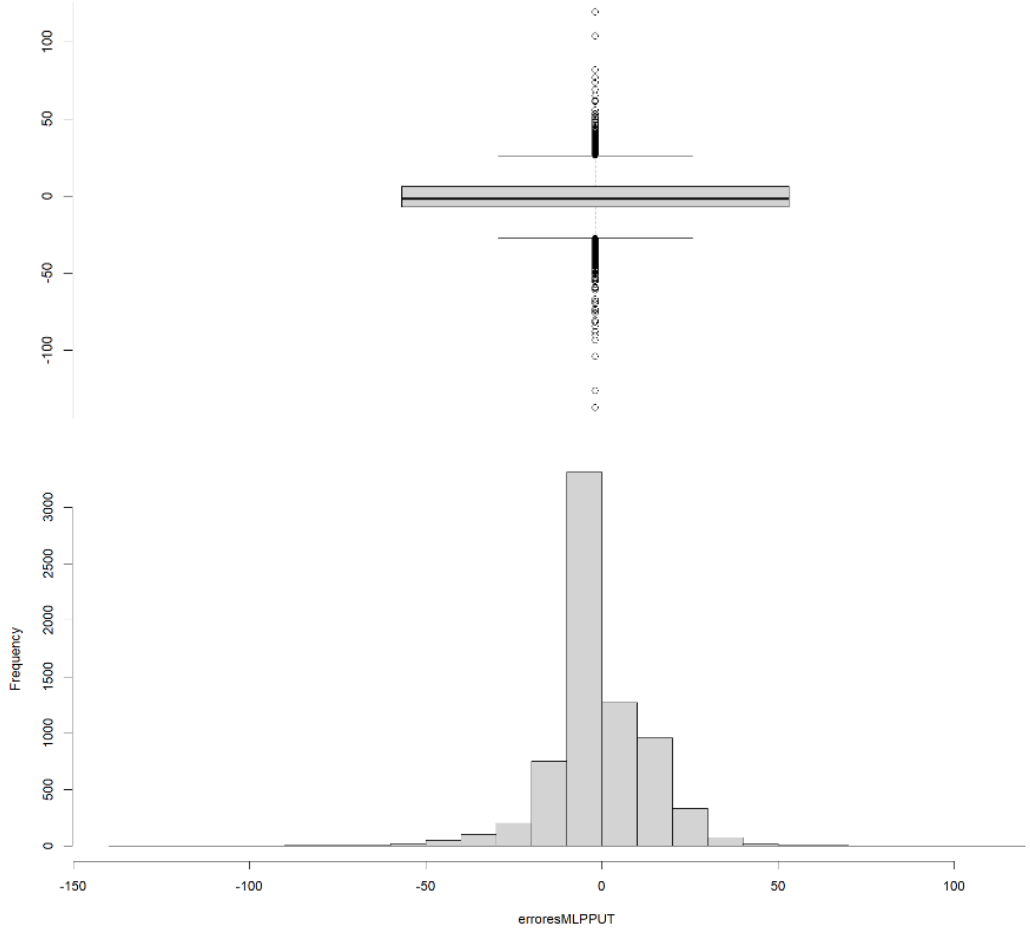
Fuente: Elaboración propia.

Dado que los datos están normalizados, se devuelven a su estado original para poder comparar las predicciones del precio realizadas por el modelo MLP con los datos originales de las opciones americanas put de Apple.

Tanto en el boxplot como en el histograma (figura 10), se puede observar que la mayoría de los errores del modelo MLP están concentrados alrededor del valor 0. La frecuencia

más alta corresponde a valores negativos cercanos a 0. Esto sugiere que, en general, el modelo MLP tiende a producir resultados cercanos al valor esperado y se desempeña bien. Sin embargo, el hecho de que haya un mayor número de errores positivos que negativos indica una tendencia del modelo MLP a infravalorar las opciones americanas put de Apple.

Figura 10: Boxplot & Histograma Errores de predicción de MLP.



Fuente: Elaboración propia.

Finalmente, para ver la precisión del modelo MLP se ha calculado su RMSE, obteniendo un valor de 14,30192. Esto significa que, en promedio, las predicciones del modelo MLP tienen una diferencia de aproximadamente 14,30 unidades en relación con los valores reales.

3.2.4. CNN

Las CNN son una forma especializada de redes neuronales profundas (deep learning) diseñadas para analizar datos de entrada que contienen alguna forma de estructura espacial. Estas redes neuronales profundas aprovechan de manera más eficiente tanto la extraordinaria potencia informática como los grandes conjuntos de datos disponibles en muchos campos en la actualidad. Las CNN se utilizan principalmente para resolver distintos problemas de visión por computadora utilizando imágenes como datos de entrada. En cuanto a su funcionamiento, primero derivan representaciones de bajo nivel, como bordes locales y puntos, y luego componen representaciones de nivel superior, como formas generales y contornos. El nombre de estas redes neuronales profundas se debe a la aplicación de convoluciones en al menos una de sus capas, que son un tipo de operación matemática lineal. El empleo de convoluciones sustituye a la multiplicación de matrices general empleada por las redes neuronales feedforward (Montesinos López et al., 2022).

Si bien, la clasificación es la aplicación predominante de CNN, su efectividad actual ha llevado a una expansión significativa en las tareas de regresión. En el ámbito financiero, se utilizan para predecir precios, tipos de interés, volatilidades y otros indicadores financieros clave, lo cual ayuda a los inversores y analistas a tomar decisiones más informadas en tiempo real.

A continuación, se procede a aplicar el algoritmo CNN a los datos de las opciones americanas put de Apple para predecir su precio.

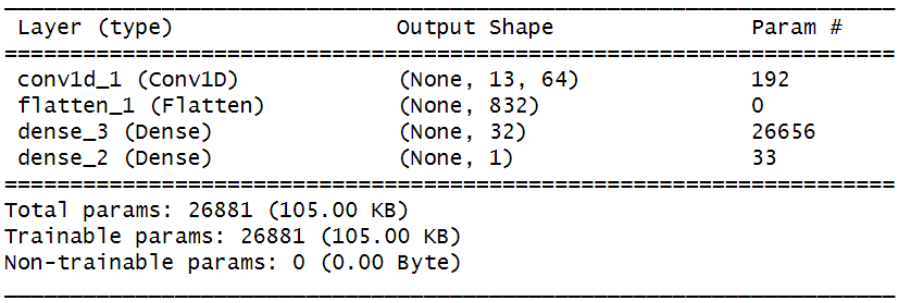
En primer lugar, teniendo en cuenta que las CNN esperan entradas tridimensionales, se ha procedido a agregar una dimensión adicional a los datos de entrada para adaptar el formato requerido por el modelo CNN.

El algoritmo CNN se ha construido empleando el paquete “Keras” de RStudio. El modelo secuencial se ha definido utilizando “keras_model_sequential()”, que permite agregar capas secuenciales al modelo CNN.

La arquitectura del modelo CNN comienza con una capa convolucional unidimensional que tiene 64 filtros y un tamaño dos de kernel. La entrada de la capa se define con el tamaño de los inputs y se utiliza la función de activación ReLU para mejorar su capacidad para aprender patrones complejos en los datos. A continuación, se introduce una capa aplanada para convertir los datos tridimensionales a bidimensionales, lo que permite su procesamiento por las capas densamente conectadas. La primera capa tiene 32 unidades y activación ReLU, permitiendo aprender representaciones más abstractas de los datos. La segunda capa tiene una unidad y activación lineal, lo que la convierte en una capa de salida que produce una predicción numérica continua. Para compilar el modelo se ha empleado MSE y el optimizador Adam, los cuales permiten minimizar los errores de predicción.

En la figura 11, se puede observar un resumen del modelo CNN empleado, que proporciona una visión general de cómo se estructura y conecta la red neuronal. En él se muestra la arquitectura de la red neuronal, incluyendo el tipo de capa empleada, el tamaño de la salida de las distintas capas y el número de parámetros, que ascienden a un total de 26.881.

Figura 11: Resumen del modelo CNN.



Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 13, 64)	192
flatten_1 (Flatten)	(None, 832)	0
dense_3 (Dense)	(None, 32)	26656
dense_2 (Dense)	(None, 1)	33

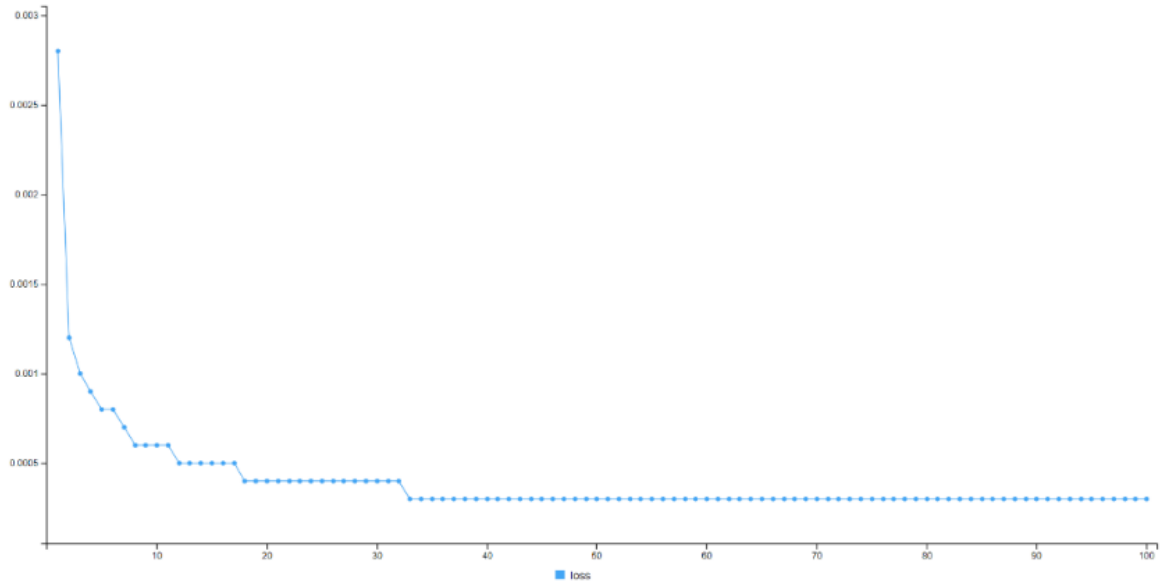
Total params: 26881 (105.00 KB)
Trainable params: 26881 (105.00 KB)
Non-trainable params: 0 (0.00 Byte)

Fuente: Elaboración propia.

Para ajustar el modelo CNN, se emplean los datos de entrenamiento y se utiliza la función "fit()" para entrenar el modelo CNN durante 100 épocas y con un tamaño 16 de lote. Durante el entrenamiento, el modelo CNN ajusta los parámetros para que las predicciones sean lo más cercanas posibles a las observaciones reales de las opciones americanas put de Apple, recogidas en los datos de entrenamiento. En el gráfico (figura 12), se puede

observar el error del modelo CNN en cada una de las épocas. Una vez pasada la época 33, el error del modelo CNN tiende a permanecer constante. Tras finalizar el entrenamiento, se evalúa con la función “evaluate()” y se obtiene un error de 0,000246571.

Figura 12: Gráfico Errores del modelo CNN por épocas.

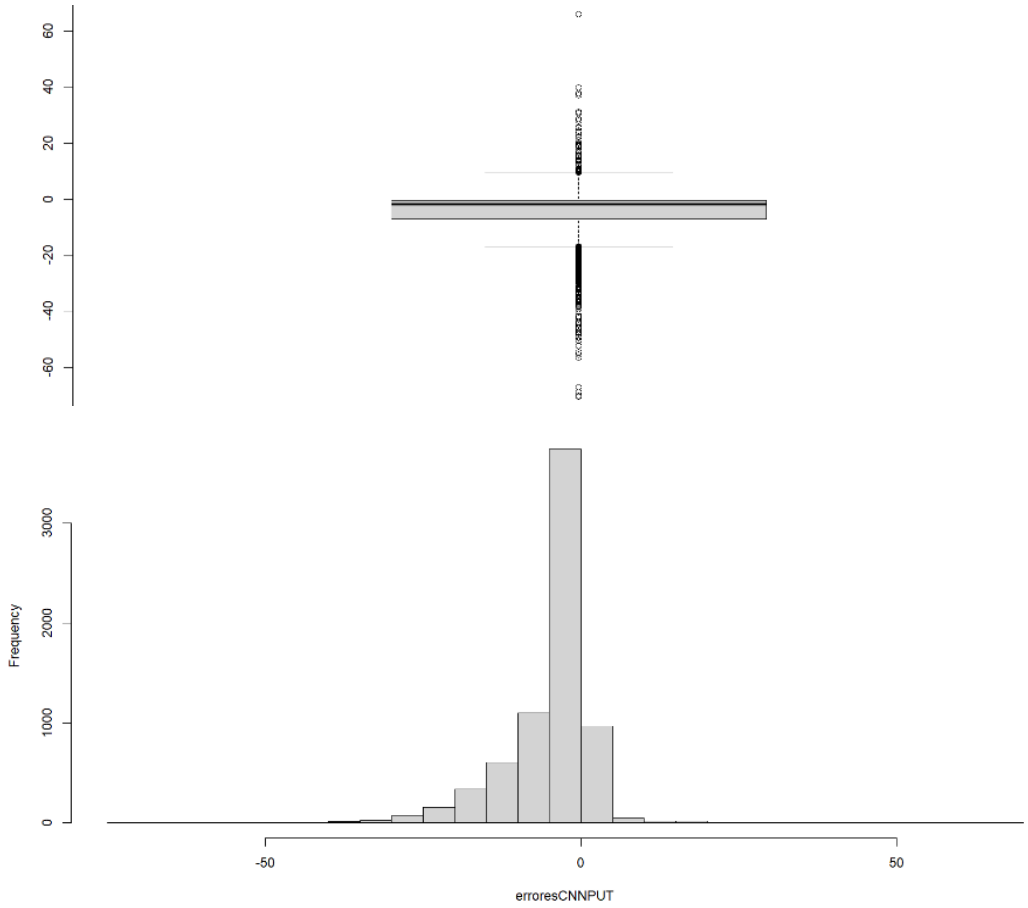


Fuente: Elaboración propia.

Dado que los datos están normalizados, se devuelven a su estado original para poder comparar las predicciones del precio realizadas por el modelo CNN con los datos originales de las opciones americanas put de Apple.

Tanto en el boxplot como en el histograma (figura 13), se puede observar que la mayoría de los errores del modelo CNN están concentrados alrededor del valor 0. La frecuencia más alta corresponde a valores negativos cercanos a 0, siendo mínimos los errores mayores a -25. Además, dado que la mayoría de los errores se localizan en los valores negativos, siendo escasos los errores situados en los valores positivos, se puede concluir que el modelo CNN tiende a sobrevalorar el precio de las opciones americanas put de Apple.

Figura 13: Boxplot & Histograma Errores de predicción de CNN.



Fuente: Elaboración propia.

Finalmente, para ver la precisión del modelo CNN se ha calculado su RMSE, obteniendo un valor de 8,751756. Esto significa que, en promedio, las predicciones del modelo CNN tienen una diferencia de aproximadamente 8,75 unidades en relación con los valores reales.

3.2.5. Longstaff-Schwartz

El modelo desarrollado por Longstaff y Schwartz (2001) es una herramienta clave en la valoración de opciones americanas, que proporciona un enfoque numérico basado en la simulación con optimización. El modelo compara el beneficio potencial de ejercer la opción americana en un momento dado (T) con el beneficio que se obtendría al mantenerla sin ejercitar.

El objetivo del modelo Longstaff-Schwartz es encontrar la regla de ejercicio que maximiza el valor de la opción americana en cada momento de su vida útil. Este proceso se realiza de manera recursiva y se concentra en los caminos que se encuentran en el dinero para mejorar la eficiencia del modelo y reducir la complejidad computacional.

Para determinar el valor esperado condicional de mantener la opción americana, Longstaff y Schwartz emplean LSMC. Este método implica la generación de múltiples caminos que representan los precios de los activos subyacentes a lo largo del tiempo.

A continuación, se realiza una regresión de los flujos de efectivo futuros en función de distintas variables relevantes para estimar el valor esperado condicional de continuar con la opción americana. La estimación resultante de la regresión proporciona una medida eficiente de la expectativa condicional, permitiendo así determinar la regla óptima para el ejercicio de la opción americana en cada punto a lo largo del tiempo.

Una vez calculado el valor de la opción americana en cada camino en el momento T, se procede a descontar los valores al momento inicial (T=0). Este proceso implica ajustar los flujos de efectivo futuros por el factor de descuento adecuado para llevarlos al valor presente. Después, mediante la media de los valores de cada camino simulado, se obtiene el precio final de la opción americana.

Para aplicar el modelo Longstaff-Schwartz sobre los datos del análisis empírico, se va a emplear el paquete “LSMonteCarlo” de RStudio (Beketov, 2013).

Como se ha expuesto previamente, las variables que se van a utilizar en este modelo son: “STRIKE”, “USGOVT1YEAR_PX_LAST”, “DIVIDEND_INDICATED_YIELD”, “PX_LAST_STOCK”, “VOLATILITY_1D” y “YEARS_UNTIL_MATURITY”.

Dado que la función “AmerPutLSM” del paquete “LSMonteCarlo” prevé que se introduzca únicamente un valor de cada variable, se ha procedido a crear distintas listas. En ellas, se recogen los datos de las observaciones de las opciones americanas put de Apple que integran el conjunto de prueba. Además, se ha creado una lista vacía para almacenar los resultados de la valoración de las opciones americanas put de Apple. Esta

lista se emplea para poder comparar posteriormente los resultados con los obtenidos por los modelos de ML empleados previamente en el estudio.

Para llevar a cabo una correcta valoración de las opciones americanas put con los datos disponibles, se han llevado a cabo dos modificaciones en la función “AmerPutLSM”.

En primer lugar, se ajusta para poder valorar las opciones americanas put el día de su vencimiento. Así, en este caso, el valor de la opción americana put va a ser el máximo entre 0 y la diferencia entre el precio de ejercicio y el precio del activo subyacente.

En segundo lugar, se crea un bucle en la función para que calcule el precio de las opciones americanas put para cada precio de las acciones de Apple contenidas en las observaciones del conjunto de prueba. Se generan trayectorias de precios del activo subyacente mediante el método Movimiento Browniano Geométrico (GBM, por sus siglas en inglés). Los resultados se introducen en la lista vacía creada anteriormente.

Posteriormente, se calculan los flujos de efectivo para cada período en cada trayectoria, teniendo en cuenta el valor máximo entre 0 y la diferencia entre el precio de ejercicio y el precio del activo subyacente. Se lleva a cabo una regresión lineal para estimar el valor de la opción americana put, utilizando los flujos de efectivo y los precios del activo subyacente en cada período.

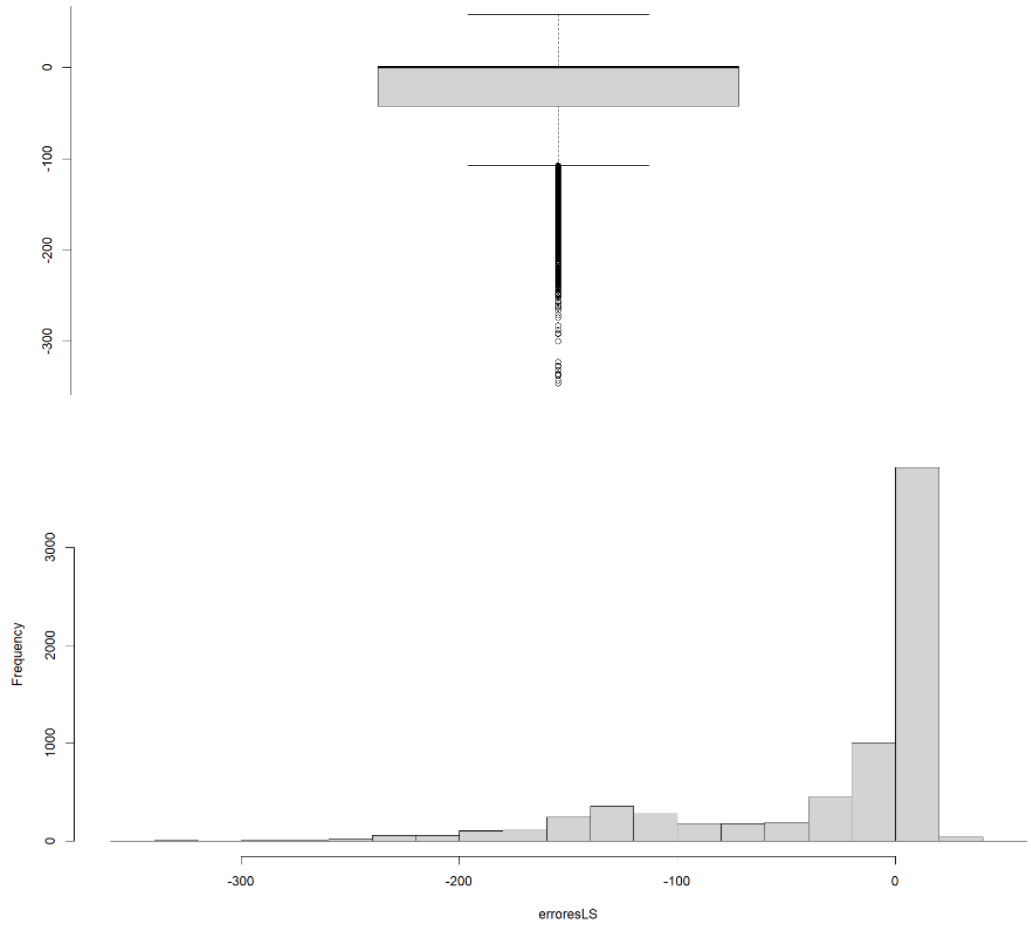
Con los resultados de la regresión, se calcula el valor de control que se utiliza para tomar las decisiones de ejercicio en cada período. Este valor de control se compara con el valor intrínseco de la opción americana put para determinar si es óptimo ejercerla o no en ese momento específico. Si el valor de control es mayor que el valor intrínseco, se decide no ejercer la opción americana put y se calcula su valor futuro. Esto se repite para todos los períodos hasta que se produzca el vencimiento.

Finalmente, el precio de la opción americana put se calcula como el promedio de los flujos de efectivo descontados al tiempo presente.

Una vez obtenidos los precios de las opciones americanas put, se ha procedido a su comparación con los precios reales. Tanto en el boxplot como en el histograma (figura

14), se puede observar que la máxima frecuencia de errores se encuentra en los valores positivos cercanos a 0. Sin embargo, los errores de mayor dimensión se sitúan en los valores negativos, llegando a alcanzar las 300 unidades. Por tanto, aunque el modelo Longstaff-Schwartz tiende a hacer predicciones precisas en la mayoría de los casos, puede tener problemas para predecir con precisión en ciertas situaciones extremas o inusuales. Esto se debe al empleo del método GBM, el cual no tiene en cuenta la heterocedasticidad y los retornos logarítmicos no normales. Estas limitaciones pueden conducir a predicciones menos precisas en escenarios donde la volatilidad es alta o la distribución de los retornos es significativamente diferente de una distribución normal, siendo este el caso durante eventos extremos o inusuales en los mercados financieros.

Figura 14: Boxplot & Histograma Errores de predicción de Longstaff-Schwartz.



Fuente: Elaboración propia

Finalmente, para ver la precisión del modelo Longstaff-Schwartz se ha calculado su RMSE, obteniendo un valor de 69,93252. Esto significa que, en promedio, las predicciones del modelo Longstaff-Schwartz tienen una diferencia de aproximadamente 69,93 unidades en relación con los valores reales.

3.3. Análisis de los resultados

Habiéndose explicado e implementado cada uno de los modelos en los datos de las opciones americanas put de Apple, se procede a analizar los resultados obtenidos.

En este análisis, se evalúa el desempeño de los modelos utilizando la métrica RMSE como indicador de precisión (tabla 3).

Tabla 3: Tabla RMSE de los modelos.

Modelo	RMSE
KNN	8,538767
RF	13,54564
MLP	14,30192
CNN	8,751756
Longstaff-Schwartz	69,93252

Fuente: Elaboración propia.

El modelo KNN tiene el mejor desempeño en términos de precisión al tener el RMSE inferior con un valor de 8,538767. Este resultado sugiere que este enfoque es altamente efectivo para predecir los precios de las opciones americanas put de Apple en este conjunto de datos. Aunque el algoritmo CNN tiene un RMSE competitivo de 8,751756, es preferible KNN dada su transparencia, simplicidad, capacidad interpretativa y eficiencia computacional.

Por otro lado, los modelos RF y MLP tienen un desempeño moderado en términos de precisión, con un RMSE de 13,54564 y 14,30192, respectivamente. Aunque estos algoritmos son capaces de manejar cierta complejidad de los datos, parece que no son tan adecuados para este conjunto de datos como lo son los enfoques de KNN y CNN.

El modelo Longstaff-Schwartz presenta un RMSE significativamente más alto que los algoritmos de ML con un valor de 69,93252. Esto implica que este modelo no se ajusta bien a los datos de las opciones americanas put de Apple en comparación con los enfoques de ML. Se podría deber a varias razones, como la simplificación excesiva de las suposiciones subyacentes del modelo y las limitaciones del método GBM. Esto resulta en una incapacidad para capturar la complejidad inherente de los datos del mercado financiero.

Por último, es importante destacar que incluso un RMSE con un valor en torno a 8,54, el cual puede considerarse bajo en comparación con los obtenidos por los otros modelos, sigue siendo significativo en el contexto de la valoración de las opciones americanas put. Una diferencia de 8,54 en la predicción de los precios puede tener importantes implicaciones financieras, especialmente en un mercado volátil como es el de las opciones americanas. Incluso pequeñas discrepancias pueden traducirse en considerables diferencias de valoración, que afectan a las estrategias de inversión y a las decisiones financieras.

4. CONCLUSIONES

Como conclusiones a este Trabajo de Fin de Grado, los algoritmos KNN, RF, MLP y CNN, entrenados exclusivamente con datos históricos del mercado financiero, consiguen una mejora notable en el rendimiento y la precisión de las predicciones del precio de las opciones americanas en comparación con el modelo Longstaff-Schwartz. Los resultados obtenidos evidencian el potencial de los algoritmos de ML para afrontar los desafíos más complejos a los que se enfrenta la industria financiera en la actualidad.

El modelo Longstaff-Schwartz, ampliamente utilizado en la valoración de opciones americanas, muestra limitaciones en su capacidad para capturar la complejidad y la dinámica del mercado financiero actual. Esto puede llevar a decisiones de inversión erróneas que no reflejan adecuadamente las condiciones del mercado financiero. En cambio, los algoritmos de ML, al ser entrenados con gran cantidad de datos históricos y utilizar técnicas avanzadas, pueden mejorar significativamente la precisión de las predicciones del valor de las opciones americanas.

A pesar de los notables avances logrados con estos modelos de ML, es importante reconocer que aún existen desafíos y limitaciones que deben abordarse. Aunque los valores predictivos de los algoritmos de ML empleados en este trabajo presentan un RMSE menor en comparación con el modelo Longstaff-Schwartz, se requiere la implementación de técnicas más sofisticadas para reducirlo aún más. Una correcta predicción es de gran relevancia en el contexto de las opciones americanas, donde los errores pueden acarrear consecuencias significativas, como pérdidas financieras, oportunidades de arbitraje, impacto en estrategias de inversión y aumento de la volatilidad del mercado financiero.

Finalmente, señalar que, si bien se ha demostrado que los algoritmos de ML empleados predicen mejor que el modelo Longstaff-Schwartz, aún queda pendiente de estudio determinar si obtienen resultados superiores que los modelos híbridos, los cuales han sido aplicados por distintos autores en los últimos años.

5. DECLARACIÓN DE USO DE HERRAMIENTAS DE INTELIGENCIA ARTIFICIAL GENERATIVA EN TRABAJOS FIN DE GRADO

ADVERTENCIA: Desde la Universidad consideramos que ChatGPT u otras herramientas similares son herramientas muy útiles en la vida académica, aunque su uso queda siempre bajo la responsabilidad del alumno, puesto que las respuestas que proporciona pueden no ser veraces. En este sentido, NO está permitido su uso en la elaboración del Trabajo fin de Grado para generar código porque estas herramientas no son fiables en esa tarea. Aunque el código funcione, no hay garantías de que metodológicamente sea correcto, y es altamente probable que no lo sea.

Por la presente, yo, María Vivas Redondo, estudiante de E3-Analytics de la Universidad Pontificia Comillas al presentar mi Trabajo Fin de Grado titulado "Optimización de la valoración de opciones americanas con Machine Learning: Más allá de Longstaff-Schwartz y modelos híbridos", declaro que he utilizado la herramienta de Inteligencia Artificial Generativa ChatGPT u otras similares de IAG de código sólo en el contexto de las actividades descritas a continuación:

1. **Corrector de estilo literario y de lenguaje:** Para mejorar la calidad lingüística y estilística del texto.

Afirmo que toda la información y contenido presentados en este trabajo son producto de mi investigación y esfuerzo individual, excepto donde se ha indicado lo contrario y se han dado los créditos correspondientes (he incluido las referencias adecuadas en el TFG y he explicitado para que se ha usado ChatGPT u otras herramientas similares). Soy consciente de las implicaciones académicas y éticas de presentar un trabajo no original y acepto las consecuencias de cualquier violación a esta declaración.

Fecha: 22/04/2024

Firma: *María Vivas*

6. REFERENCIAS BIBLIOGRÁFICAS

Anderson, D. y Ulrych, U. (2023). Accelerated American option pricing with deep neural networks. *Quantitative Finance and Economics*, 7(2), 207-228. https://doi.org/10.3934/QFE.2023011

Arévalo De Pablos, A., Camacho Miñano, M. y Pérez-Hernández, F. (2024). A hybrid model integrating artificial neural network with multiple GARCH-type models and EWMA for performing the optimal volatility forecasting of market risk factors. *Expert Systems with Applications*, 243, 122896. https://doi.org/10.1016/j.eswa.2023.122896

Becker, S., Cheridito, P. y Jentzen, A. (2020). Pricing and Hedging American-Style Options with Deep Learning. *Journal of Risk and Financial Management*, 13(7), 158. https://doi.org/10.3390/jrfm13070158

Beketov, M. A. (2013). *LSMonteCarlo: American options pricing with Least Squares Monte Carlo method* (Versión 1.0). R. https://cran.r-project.org/package=LSMonteCarlo

Black, F. y Scholes, M. (1973). The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81(3), 637–654. https://doi.org/10.1086/260062

Brown, S. (21 de abril, 2021). Machine learning, explained. *MIT Sloan*. https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained

Cox, J. C., Ross, S. A. y Rubinstein, M. (1979). Option pricing: A simplified approach. *Journal of Financial Economics*, 7(3), 229-263. https://doi.org/10.1016/0304-405X(79)90015-1

Dubrov, B. (2015). *Monte Carlo Simulation with Machine Learning for Pricing American Options and Convertible Bonds* [Tel Aviv University]. SSRN. http://dx.doi.org/10.2139/ssrn.2684523

Felizardo, L., Matsumoto, E. y Del Moral Hernández, E. (18 de julio, 2022). *Solving the optimal stopping problem with reinforcement learning: an application in financial option exercise* [Conferencia]. 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italia. https://doi.org/10.1109/IJCNN55064.2022.9892333

Feng, G., Liu, G. y Sun, L. (8-11 de diciembre, 2013). *A nonparametric method for pricing and hedging American options* [Conferencia]. Winter Simulations Conference (WSC), Washington D. C., Estados Unidos. http://dx.doi.org/10.1109/WSC.2013.6721462

He, L., Levine, R. A., Fan, J., Beemer, J. y Stronach, J. (2018). Random forest as a predictive analytics alternative to regression in institutional research. *Practical Assessment, Research, and Evaluation*, 23(1): 1. https://doi.org/10.7275/1wpr-m024

Heston, S. L. (1993). A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. *The Review Of Financial Studies*, 6(2), 327-343. https://doi.org/10.1093/rfs/6.2.327

Hoshisashi, K. y Yamada, Y. (2023). Pricing multi-asset Bermudan commodity options with stochastic volatility using neural networks. *Journal of Risk and Financial Management*, 16(3), 192. https://doi.org/10.3390/jrfm16030192

Isabona, J., Imoize, A. L., Ojo, S., Karunwi, O., Kim, Y., Lee, C.-C. y Li, C.-T. (2022). Development of a multilayer perceptron neural network for optimal predictive modeling in urban microcellular radio environments. *Applied Sciences*, 12(11), 5713. https://doi.org/10.3390/app12115713

Klidbary, S. H. y Arabameri, A. (1-2 de noviembre, 2023). *A Novel Density-Based KNN in Pattern Recognition* [Conferencia]. 2023 13th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Irán. https://doi.org/10.1109/ICCKE60553.2023.10326227

Longstaff, F. A. y Schwartz, E. S. (2001). Valuing American Options by Simulation: A Simple Least-Squares Approach. *The Review of Financial Studies*, 14(1), 113-147. http://dx.doi.org/10.1093/rfs/14.1.113

Maidoumi, M., Zahid, M. y Daafi, B. (2023). Pricing American option under exponential Levy Jump-diffusion model using Random Forest instead of least square regression. *Journal of Mathematical Modeling*, 11(2), 229-244. https://doi.org/10.22124/jmm.2022.21756.1909

Malpica, A. y Frias, P. (2019). Valuation of an American Option for the Spanish Secondary Reserve Market Using a Machine Learning Model. *IEEE Transactions on Power Systems*, 34(1), 544–554. https://doi.org/10.1109/TPWRS.2018.2859762

Montesinos López, O. A., Montesinos López, A. y Crossa, J. (2022). Convolutional neural networks. En *Multivariate Statistical Machine Learning Methods for Genomic Prediction* (533-577). Springer. https://doi.org/10.1007/978-3-030-89010-0_13

White, A. y Hull, J. (1990). Valuing Derivative Securities Using The Explicit Finite Difference Method. *The Journal of Financial and Quantitative Analysis*, 25(1), 87-100. https://doi.org/10.2307/2330889

7. ANEXOS

- **Código RStudio**

```
# Establecemos directorio

setwd("C:/Users/Usuario/OneDrive - Universidad Pontificia Comillas/Escritorio/TFG")

# Importamos las librerías que se van a emplear

library(readxl)

library(dplyr)

library(rsample)

library(corrplot)

library(tidyverse)

library(caret)

library(randomForest)

library(keras)

library(MASS)

library(LSMonteCarlo)

# Introducimos las tablas con los datos extraídos de Bloomberg

Appleoptions1 <- read_excel("Appleoptions.xlsx", col_types = c("date", "text", "text", "numeric","date", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric"))

Appleoptions2 <- read_excel("Appleoptions1.xlsx", col_types = c("date", "text", "text", "numeric","date", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric"))

Appleoptions <- rbind(Appleoptions1, Appleoptions2)

Applestocks <- read_excel("Applestocks.xlsx", col_types = c("date", "numeric", "numeric", "numeric", "numeric", "numeric", "numeric"))
```

```
TNOTES <- read_excel("TNOTES.xlsx", col_types = c("date", "numeric", "numeric", "numeric", "numeric", "numeric"))

# Fusionamos las tablas utilizando "DATES" como campo común

resultado_merge <- merge(Appleoptions,Applestocks, by = "DATES", all.x = TRUE)

resultado_merge1 <- merge(resultado_merge,TNOTES, by = "DATES", all.x = TRUE)

# Renombramos las columnas para una sencilla identificación

nuevos_nombres <- c("DATES", "TICKER", "SHORT_NAME", "STRIKE", "MATURITY_DATE", "PX_LAST_OPTION", "PX_VOLUME_OPTION", "DELTA_LAST", "GAMMA_LAST", "VEGA_LAST", "THETA_LAST", "RHO_LAST", "PX_LAST_STOCK", "PX_VOLUME_STOCK", "VOLATILITY_360D", "VOLATILITY_3OD", "DIVIDEND_INDICATED_YIELD", "DIVIDEND", "USGOVT3MONTHS_PX_LAST", "USGOVT6MONTHS_PX_LAST", "USGOVT1YEAR_PX_LAST", "USGOVT2YEARS_PX_LAST", "USGOVT5YEARS_PX_LAST")

names(resultado_merge1) <- nuevos_nombres

# Obtenemos el tamaño en memoria de “resultado_merge1”

tamaño_en_memoria <- object.size(resultado_merge1)

# Si algún campo de una opción americana contiene NA en un día, eliminamos todos sus datos de ese día

resultado_merge2<- resultado_merge1 [complete.cases(resultado_merge1 ),]

# Creamos “DAYS_UNTIL_MATURITY”, “PAYOFF”, “MAX_PAYOFF_PER_OPTION” y “MAXIMUM_DATE_PAYOFF_OPTION”

resultado_merge2$DAYS_UNTIL_MATURITY<-
as.numeric(difftime(resultado_merge2$MATURITY_DATE,
resultado_merge2$DATES, units = "days"))

resultado_merge2$PAYOFF<-pmax(resultado_merge2$STRIKE -
resultado_merge2$PX_LAST_STOCK, 0)

resultado_merge2 <- resultado_merge2 %>%

  group_by(TICKER) %>%
```

```
  mutate(MAX_PAYOFF_PER_OPTION = max(PAYOFF)) %>%
  ungroup()
resultado_merge2 <- resultado_merge2 %>%
  group_by(TICKER) %>%
  mutate(MAX_PAYOFF_PER_OPTION_DAYS_UNTIL_MATURITY =
DAYS_UNTIL_MATURITY[which.max(PAYOFF)]) %>%
  ungroup()
# Como Bloomberg no proporciona la volatilidad diaria de las acciones, procedemos a calcularla mediante la volatilidad anual extraída de Bloomberg
resultado_merge2$VOLATILITY_1D<-
resultado_merge2$VOLATILITY_360D/sqrt(360)
# Separamos las opciones americanas put
PUT1 <- subset(resultado_merge2, substr(SHORT_NAME, 8, 8) == "P")
PUT2 <- subset(resultado_merge2, substr(SHORT_NAME, 9, 9) == "P")
PUT <- rbind(PUT1, PUT2)
# Vemos el número total de opciones americanas put
OPCIONESPUT <- length(unique(PUT$SHORT_NAME))
# Obtenemos un resumen de los datos que tenemos
summary(PUT)
# Extraemos las variables categóricas
PUT<-PUT[-c(1,2,3,5)]
# Obtenemos la matriz de correlación de las variables numéricas (sin la variable "PX_LAST_OPTION" que es la target) y vemos si hay multicolinealidad entre ellas
correlacionPUT <- cor(PUT[-2])
pairs(correlacionPUT)
corrplot(correlacionPUT, method = "color", addCoef.col = "black", tl.col = "black",
```

```
tl.srt = 45, tl.cex = 0.4, cl.cex = 0.4, number.cex = 0.4,
order = "hclust", type = "upper", diag = FALSE,
mar = c(0, 0, 2, 0),
title = "Matriz de Correlación")
```

Eliminamos las variables “DELTA_LAST”, “VOLATILITY_360D”, “VOLATILITY_3OD”, “DIVIDEND”, “USGOVT3MONTHS_PX_LAST”, “USGOVT6MONTHS_PX_LAST”, “USGOVT2YEARS_PX_LAST”, “USGOVT5YEARS_PX_LAST” y “PAYOFF”

```
PUT<-PUT[-c(4,11,12,14,15,16,18,19,21)]
```

Obtenemos una muestra aleatoria del 5% del total de observaciones y dividimos los datos en train (70%) y test (30%)

```
set.seed(123)
indicesPUT <- sample( 1:nrow(PUT), 0.05*nrow(PUT))
PUTMUESTRA<-PUT [indicesPUT, ]
trainIndexPUT<-sample(1:nrow(PUTMUESTRA),
nrow(PUTMUESTRA)*0.7,replace=FALSE)
trainPUT<-PUTMUESTRA[trainIndexPUT,]
testPUT<-PUTMUESTRA[-trainIndexPUT,]
```

Creamos una función para normalizar las variables para su aplicación en los algoritmos, aunque luego para comparar los resultados volverán a su estado original

```
maxstrainPUT <- apply(trainPUT, 2, max)
minstrainPUT <- apply(trainPUT, 2, min)
scaledtrainPUT <- as.data.frame(scale(trainPUT, center = minstrainPUT, scale = maxstrainPUT - minstrainPUT))
maxstestPUT <- apply(testPUT, 2, max)
minstestPUT <- apply(testPUT, 2, min)
```

```
scaledtestPUT <- as.data.frame(scale(testPUT, center = minstestPUT, scale = maxstestPUT - minstestPUT))

# Procedemos a aplicar los algoritmos de ML para predecir el precio de las opciones americanas put de Apple que es la variable target

# KNN

set.seed(244)

modeloKNNPUT <- train(PX_LAST_OPTION~., data = scaledtrainPUT, method = "knn", trControl = trainControl("cv", number = 10), tuneLength = 15)

# Vemos en un gráfico RMSE y los diferentes valores de K

plot(modeloKNNPUT)

# Obtenemos el valor de K que minimiza RMSE

modeloKNNPUT$bestTune

# Hacemos predicciones con los datos de test

prediccionesKNNPUT <- predict(modeloKNNPUT, newdata = scaledtestPUT)

head(prediccionesKNNPUT)

# Dado que los resultados están normalizados, los devolvemos a su estado original para poder comparar

prediccionesKNNPUTor <- prediccionesKNNPUT*(maxstestPUT[2]-minstestPUT[2])+minstestPUT[2]

# Calculamos los errores de predicción en test

erroresKNNPUT<-testPUT$PX_LAST_OPTION-prediccionesKNNPUTor

# Vemos el gráfico de los errores de predicción

boxplot(erroresKNNPUT)

hist(erroresKNNPUT, breaks=20)

# Calculamos RMSE en los datos de test

hPUT<-nrow(testPUT)
```

```
RMSE_KNNPUT<-sqrt(sum(erroresKNNPUT^2)/hPUT)

"RMSE_PUT";RMSE_KNNPUT

# RANDOM FOREST

# Establecemos el control sobre train

set.seed(244)

x<-trainControl(method = "repeatedcv",

                number = 10,

                repeats = 3,

                verboseIter = TRUE,

                classProbs = FALSE,

                summaryFunction = defaultSummary)

# Estimamos el modelo

modelorfPUT<-train(PX_LAST_OPTION~., data=scaledtrainPUT, method="rf", trControl=x, tuneLength=4)

modelorfPUT

plot(modelorfPUT)

plot(modelorfPUT$finalModel)

# Vemos la importancia de las variables

varImp(modelorfPUT)

# Hacemos predicciones con los datos de test

predrfPUT<-predict(modelorfPUT, newdata=scaledtestPUT)

head(predrfPUT)

# Dado que los resultados están normalizados, los devolvemos a su estado original para poder comparar

prediccionesrfPUTor <- predrfPUT*(maxstestPUT[2]-minstestPUT[2])+minstestPUT[2]
```

```
# Calculamos los errores de predicción en test
erroresrfPUT<-testPUT$PX_LAST_OPTION-prediccionesrfPUTor
# Vemos el gráfico de los errores de predicción
boxplot(erroresrfPUT)
hist(erroresrfPUT, breaks=20)
# Calculamos RMSE en los datos de test
hPUT<-nrow(testPUT)
RMSE_rfPUT<-sqrt(sum(erroresrfPUT^2)/hPUT)
"RMSE_PUT";RMSE_rfPUT
# MLP
# Entrenamos el modelo
mlp_grid <- expand.grid(size = c("7,3", "5,2"))
set.seed(244)
MLPPUT <- caret::train(PX_LAST_OPTION ~ .,
                       data = scaledtrainPUT,
                       method = "mlp",
                       trControl = trainControl(method = "cv",
                                                number = 3,
                                                savePredictions = "final"),
                       preProc = c("center", "scale"),
                       tuneGrid = mlp_grid,
                       linOut = TRUE,
                       actFunc = "ReLU",
                       metric = "RMSE")
# Visualizamos el modelo
```

```
print(MLPPUT)

plot(MLPPUT)

# Hacemos predicciones con los datos de test

prediccionesMLPPUT <- predict(MLPPUT, newdata = scaledtestPUT)

# Dado que los resultados están normalizados, los devolvemos a su estado original para poder comparar

prediccionesMLPPUTor <- prediccionesMLPPUT*(maxstestPUT[2]-minstestPUT[2])+minstestPUT[2]

# Calculamos los errores de predicción en test

erroresMLPPUT<-testPUT$PX_LAST_OPTION-prediccionesMLPPUTor

# Vemos el gráfico de los errores de predicción

boxplot(erroresMLPPUT)

hist(erroresMLPPUT, breaks=20)

# Calculamos RMSE en los datos de test

hPUT<-nrow(testPUT)

RMSE_MLPPUT<-sqrt(sum(erroresMLPPUT^2)/hPUT)

"RMSE_PUT";RMSE_MLPPUT

# CNN

# Separamos x input & y output tanto en train como en test

xtrainPUT = as.matrix(scaledtrainPUT[,-2])

ytrainPUT = as.matrix(scaledtrainPUT[,2])

xtestPUT = as.matrix(scaledtestPUT[,-2])

ytestPUT = as.matrix(scaledtestPUT[,2])

dim(xtrainPUT)

dim(ytrainPUT)
```

```
# Dado que CNN espera como inputs figuras tridimensionales, vamos a añadir una dimensión

xtrainPUT = array(xtrainPUT, dim = c(nrow(xtrainPUT), 14, 1))

xtestPUT = array(xtestPUT, dim = c(nrow(xtestPUT), 14, 1))

dim(xtrainPUT)

dim(xtestPUT)

# A continuación, extraemos las dimensiones de los inputs

input_dimensionPUT = c(dim(xtrainPUT)[2:3])

print(input_dimensionPUT)

# Establecemos el modelo

# Definimos el modelo secuencial de Keras y agregamos una capa convolucional unidimensional. Añadimos capas Flatten y Dense, y las compilamos con el optimizador Adam

set.seed(244)

modeloCNNPUT = keras_model_sequential() %>%

  layer_conv_1d(filters = 64, kernel_size = 2,

                input_shape = input_dimensionPUT, activation = "relu") %>%

  layer_flatten() %>%

  layer_dense(units = 32, activation = "relu") %>%

  layer_dense(units = 1, activation = "linear")

modeloCNNPUT%>% compile(

  loss = "mse",

  optimizer = "adam")

modeloCNNPUT%>% summary()

# Ajustamos el modelo con los datos de train
```

```
modeloCNNPUT%>% fit(xtrainPUT, ytrainPUT, epochs = 100, batch_size=16, verbose = 1)

scoresPUT = modeloCNNPUT%>% evaluate(xtrainPUT, ytrainPUT, verbose = 1)

print(scoresPUT)

# Predecimos con test y visualizamos los resultados

ypredPUT = modeloCNNPUT %>% predict(xtestPUT)

# Dado que los resultados están normalizados, los devolvemos a su estado original para poder comparar

prediccionesCNNPUTor <- ypredPUT*(maxstestPUT[2]-minstestPUT[2])+minstestPUT[2]

# Calculamos los errores de predicción en test

erroresCNNPUT<-testPUT$PX_LAST_OPTION-prediccionesCNNPUTor

# Vemos el gráfico de los errores de predicción

boxplot(erroresCNNPUT)

hist(erroresCNNPUT, breaks=20)

# Calculamos RMSE en los datos de test

hPUT<-nrow(testPUT)

RMSE_CNNPUT<-sqrt(sum(erroresCNNPUT^2)/hPUT)

"RMSE_PUT";RMSE_CNNPUT

# LONGSTAFF-SCHWARTZ

# Vemos el paquete LSMonteCarlo y la función AmerPutLSM que se van a emplear

?LSMonteCarlo

?AmerPutLSM

# Creamos distintas listas que contienen los datos de las opciones americanas put que se han empleado en los algoritmos de ML anteriores

preciotestPUT = unlist(testPUT[,8])
```

```
striketestPUT = unlist(testPUT[,1])
voltestPUT = unlist(testPUT[,15])
rftestPUT = unlist(testPUT[,11])
drtestPUT = unlist(testPUT[,10])
daysmattestPUT = unlist(testPUT[,12])
yearsmattestPUT = sapply(daysmattestPUT, function(dias) dias / 365)
# Creamos una lista vacía para introducir los resultados
resultados3 <- list()
# Ejecutamos y modificamos la función AmerPutLSM
 AmerPutLSM<- function(Spot, sigma, n, m, Strike, r, dr, mT)
 {
 # Ajuste para el vencimiento igual a 0 (modificación del código original)
 if (mT == 0| m == 0) {
  intrinsic_value <- pmax(Strike - Spot, 0)
  res <- list(price = intrinsic_value, Spot, Strike, sigma, n, m, r, dr, mT)
  class(res) <- "AmerPut"
  return(res)
 }
 GBM <- matrix(NA, nrow = n, ncol = m)
 for (i in 1:n) {
  GBM[i, ] <- Spot * exp(cumsum(((r - dr) * (mT/m) - 0.5 *
                          sigma * sigma * (mT/m)) + (sigma * (sqrt(mT/m)) *
                                                  rnorm(m, mean = 0, sd = 1))))
 }
 X <- ifelse(GBM < Strike, GBM, NA)
```

```
  CFL <- matrix(pmax(0, Strike - GBM), nrow = n, ncol = m)
  Xsh <- X[, -m]
  X2sh <- Xsh * Xsh
  Y1 <- CFL * exp(-1 * r * (mT/m))
  Y2 <- cbind((matrix(NA, nrow = n, ncol = m - 1)), Y1[, m])
  CV <- matrix(NA, nrow = n, ncol = m - 1)
  try(for (i in (m - 1):1) {
    reg1 <- lm(Y2[, i + 1] ~ Xsh[, i] + X2sh[, i])
    CV[, i] <- (matrix(reg1$coefficients)[1, 1]) + ((matrix(reg1$coefficients)[2,
                                                    1]) * Xsh[, i]) +
((matrix(reg1$coefficients)[3,
1]) * X2sh[, i])
    CV[, i] <- (ifelse(is.na(CV[, i]), 0, CV[, i]))
    Y2[, i] <- ifelse(CFL[, i] > CV[, i], Y1[, i], Y2[,
                                          i + 1] * exp(-1 * r * (mT/m)))
  }, silent = TRUE)
  CV <- ifelse(is.na(CV), 0, CV)
  CVp <- cbind(CV, (matrix(0, nrow = n, ncol = 1)))
  POF <- ifelse(CVp > CFL, 0, CFL)
  FPOF <- firstValueRow(POF)
  dFPOF <- matrix(NA, nrow = n, ncol = m)
  for (i in 1:m) {
    dFPOF[, i] <- FPOF[, i] * exp(-1 * mT/m * r * i)
  }
  PRICE <- mean(rowSums(dFPOF))
```

```
  res <- list(price = PRICE, Spot, Strike, sigma, n, m, r, dr, mT)
  class(res) <- "AmerPut"
  return(res)
}
set.seed(244)
for (p in seq_along(preciotestPUT)) {
  resultado <- AmerPutLSM(
    Spot = preciotestPUT[[p]],
    sigma = voltestPUT[[p]],
    n = 1000,
    m = daysmattestPUT[[p]],
    Strike = striketestPUT[[p]],
    r = rftestPUT[[p]],
    mT = yearsmattestPUT[[p]],
    dr = drtestPUT[[p]]
  )
  resultados3 <- c(resultados3, list(resultado))
}
# Comparamos precio real VS precio LS
preciosLS<- sapply(resultados3, function(x) as.numeric(gsub("[^0-9.]", "", x$price)))
preciosreal<-unlist(testPUT[,2])
erroresLS<-preciosreal-preciosLS
# Vemos el gráfico de los errores de predicción
boxplot(erroresLS)
hist(erroresLS, breaks=20)
```

```
# Calculamos RMSE en los datos de test

hLS<-length(preciosLS)

RMSE_LS<-sqrt(sum(erroresLS^2)/hLS)

"RMSE_PUT";RMSE_LS
```