UNIVERSIDADE DA CORUÑA

Facultade de Economía e Empresa

Trabajo de fin de máster

Construcción de modelos de predicción de activos financieros

Carbonell Tato, Eugenia Anahí

Martínez Filgueira, Xosé Manuel

Máster Universitario en Banca y Finanzas

Curso académico 2022/23

Trabajo de Fin de Máster presentado en la Facultad de Economía y Empresa de la Universidade da Coruña para la obtención del Máster Universitario en Banca y Finanzas (MUBF).

Resumen

En las últimas décadas, el aprendizaje automático ha tomado gran protagonismo, capturando la atención del público en general. El objeto de este trabajo radica en comparar diversos modelos de predicción de activos financieros, tales como las redes neuronales recurrentes (LSTM), *random forest*, regresión lineal y SVR. El caso de estudio será la cotización semanal al cierre de la acción SAN.MC, mediante este se buscará dar con un modelo que sea considerado óptimo para guiar a un individuo que desea conocer la posible evolución del precio de una acción que posee en cartera, sin contar con conocimientos de análisis técnico y fundamental.

Palabras clave: precio, cotización, modelo de regresión, Python, SVM, SVR, LSTM, redes neuronales recurrentes, random forest, modelo de ensamble, MAE, MSE, R^2, MAPE, RMSE, aprendizaje automático.

El número de palabras del presente documento es de: 14.927.

Abstract

In recent decades, machine learning has gained significant importance, capturing the attention of the public. The purpose of this project is to compare various models for predicting financial assets, such as recurrent neural networks (LSTM), random forest, linear regression, and SVR. The case study will focus on the weekly closing price of the SAN.MC stock. The aim is to find an optimal model that can guide individuals who wish to understand the potential evolution of a stock's price in their portfolio, without requiring knowledge of technical and fundamental analysis.

Keywords: price, regression model, Python, SVM, SVR, LSTM, recurrent neural networks, random forest, ensemble model, MAE, MSE, R^2, MAPE, RMSE, machine learning.

The number of words in this document is: 14.927.

Índice de contenido

Índice de Figuras

Índice de Tablas

1. Introducción

El ámbito financiero ha comprendido las múltiples ventajas que puede obtener a partir de la aplicación de algoritmos de aprendizaje automático para resolver sus inquietudes. Esta tecnología debe su relevancia a la capacidad para generar aportes de valor en diversos campos, donde mediante su aplicación a tareas de clasificación y predicción, permite entre otras cosas, que los individuos cuenten con una mejor gestión de sus finanzas personales sin realizar esfuerzos adicionales.

En el ámbito de la gestión de inversiones, el aprendizaje automático dio lugar a la posibilidad de predicción del valor de diversos activos financieros, entre ellos las acciones. La implementación de un modelo de predicción permitiría aprovechar ventajas competitivas y mejorar el rendimiento en las estrategias de inversión individuales al ser capaz de identificar patrones subyacentes que pasan desapercibidos para el ojo humano.

En primer lugar, se describen diversos modelos de aprendizaje automático que podrían se aplicar a la hora de predecir el precio de cierre de las acciones.

En segundo lugar, se opta por desarrollar cuatro modelos que permitan predecir el precio semanal al cierre de la acción SAN.MC. Los modelos mencionados incluirán un modelo de regresión lineal, un modelo de *random forest*, un modelo de redes neuronales recurrentes (LSTM) y un modelo de SVR.

En tercer lugar, se analiza el rendimiento y precisión de cada modelo y se efectúa una comparativa con el objeto de decidir cuál de ellos cuenta con el mejor desempeño.

En cuarto lugar, se desarrolla un modelo de ensamble que busca eliminar ciertas debilidades encontradas en los modelos anteriores y potenciar mediante sus fortalezas mediante la combinación de las mejores características de los dos modelos con mejor desempeño en las métricas individuales.

Por último, se desarrolla una breve conclusión derivada del análisis de los modelos presentados.

2. Machine Learning

En 1959, Arthur Samuel, pionero americano en inteligencia artificial definió el aprendizaje automático (machine learning) como “un campo de estudio que le brinda a los ordenadores la capacidad de aprender sin ser programados explícitamente (Samuel, 1959)” [1]

Si bien hoy en día escuchamos hablar de forma novedosa acerca del aprendizaje automático en casi todos los ámbitos, desde el ocio como pueden ser las sugerencias de títulos recomendados que realizan las plataformas de streaming hasta estudios científicos donde se aplica a efectos de diseñar nuevos medicamentos (Jang, 2020) [12], este no es un nuevo descubrimiento, sino que ha ido evolucionando de manera paulatina desde la década de 1950. Hoy en día, se ha visto poderosamente beneficiado de la disponibilidad de datos existentes, puesto que la mayoría de los procesos y transacciones tanto monetarias como no monetarias que se llevan a cabo en la vida diaria dejan un rastro digital.

El simple hecho de disponer de datos almacenados en la nube o en un ordenador no implica que el mismo pueda capitalizar automáticamente tales datos en conocimiento. El aprendizaje automático consta de tomar el conjunto de datos (*dataset*), donde cada dato representa una muestra que se aplicará para el entrenamiento del modelo de aprendizaje. Dicho modelo estudia los casos para posteriormente poder aplicar ese aprendizaje a la predicción de respuestas vinculadas a casos futuros.

En este sentido, el aprendizaje automático no depende de que un humano detecte y provea la vinculación entre un dato y el resultado que le ha de ser asignado, sino que utiliza los datos provistos para encontrar relaciones en las palabras o valores, a efectos de realizar las predicciones directamente. De esta forma, detecta también patrones no usuales y se actualiza de manera automática en caso de identificar algún patrón adicional.

[1] Samuel, A. (1959). Some studies in machine learning using the game of checkers. IBM Journal of Research and Development.

[2] Jang, S. K. (2020). Machine learning identifies inhibitors of the SARS-CoV-2 main protease. Nature, 586(7827), . Nature, 113-119.

Resulta evidente lo ventajosa que es la aplicación de aprendizaje automático. En primer lugar, debido a que pueden aprender de forma autónoma sin necesidad de realizar actualizaciones manuales ante la incorporación de nueva información, lo que brinda simplicidad y eficiencia al modelo. En segundo lugar, por su habilidad de trabajar con grandes volúmenes de datos e identificar patrones complejos que escaparían al análisis humano. (Géron, 2019) [3]

1.1. Tipos de aprendizaje automático

Existen diversos tipos de aprendizaje automático, abordaremos brevemente algunos de ellos a efectos de determinar cuál sería el más adecuado para la predicción del precio de acciones.

Hemos de mencionar que el aprendizaje puede ser tanto por lotes (*batch learning*) como en línea (*online learning*). La diferencia principal es que, en el primer caso, el algoritmo debe ser entrenado nuevamente incorporando todos los datos cada vez que deseamos agregar una actualización. Por otro lado, en el aprendizaje en línea, el algoritmo recibe los nuevos datos y los incorpora para su aprendizaje instantáneamente (o bien en pequeños lotes), sin necesidad de la intervención de un ser humano. En este sentido, el aprendizaje en línea resulta ventajoso para aquellos algoritmos que requieran actualizarse con velocidad ante los cambios, o bien en caso de contar con recursos informáticos limitados. (Géron, 2019) [3]

Aprendizaje supervisado

En el aprendizaje supervisado se provee un dataset que contiene diversas entradas (*input*) y sus correspondientes salidas o respuestas (*output*). De esta forma el algoritmo entrena y crea diversas vinculaciones entre tales *inputs* y *outputs* con el objeto de reconocerlas y aplicarlas a las siguientes entradas que no posean una salida preasignada. Este tipo de aprendizaje es muy utilizado a la hora de realizar predicciones sobre nuevos datos, siendo alguno de los algoritmos más utilizados el modelo de regresión lineal, redes neuronales, random forest y arboles de decisión, etc. (Géron, 2019) [3]

Si bien requiere de un gran esfuerzo en términos de tiempo disponible para categorizar la información que se utilizará al entrenar el modelo, se obtienen resultados de gran

[3] Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2da Edición. O'Reilly Media, Inc.

precisión a la hora de categorizar los nuevos datos, por lo que resulta un tipo de aprendizaje viable en referencia a nuestro caso de estudio, que consiste en predecir la cotización tanto de acciones como del IBEX35.

Aprendizaje no supervisado

A diferencia del caso anterior, el aprendizaje no supervisado no requiere que le sea provisto un conjunto de datos cuyas entradas(inputs) cuentan con salidas (outputs) ya definidos, puesto que este tipo de aprendizaje automático encuentra las vinculaciones directamente mediante los inputs que le fueron dados.

Es de gran utilidad a la hora de descubrir patrones que no son fácilmente detectables, sin embargo, suelen ser difíciles de explicar puesto que los patrones encontrados pueden carecer de un significado o explicación evidente. (Géron, 2019) [3]

Este tipo de aprendizaje es muy utilizado a la hora de generar grupos o clústeres compuestos de acuerdo con las similitudes encontradas, por lo que resultan de gran utilidad por ejemplo al momento de segmentar el público que visita un determinado sitio web. Asimismo, también son utilizados para detectar anomalías, tales como puede ser una transacción inusual realizada con la tarjeta de crédito, etc.

Aprendizaje semi supervisado

El aprendizaje semi supervisado busca combinar las ventajas de los dos tipos de aprendizaje mencionados anteriormente. Consiste en proveer al algoritmo con una pequeña cantidad de datos que poseen un *output* asignado, el cual aporta precisión, y una gran cantidad de datos sin *output* alguno, de modo que pueda capitalizar esa gran variedad de datos para reconocer los patrones que los vinculan, de esta forma el modelo es capaz de aplicar lo aprendido sobre nuevos datos. (Géron, 2019) [3]

Si bien este tipo de aprendizaje podría aplicarse a la hora de predecir el precio de las acciones, consideramos que no es el más efectivo puesto que los datos históricos de cotizaciones y volúmenes negociados son datos categorizados (con un *output* definido) y los mismos son mucho más efectivos a la hora de realizar una predicción que los datos no categorizados, como pueden ser las noticias o los factores psicológicos que incentivan la negociación de dichos activos.

2.1. Desafíos a los que se enfrenta el Machine Learning

Existen diferentes desafíos a los que nos enfrentaremos al momento de decidir qué datos utilizar y que algoritmo emplear para la predicción de valores.

Géron (2019), indica que podemos afrontar diversos problemas a la hora de seleccionar los datos, ya que una baja calidad en los datos o una cantidad insuficiente de ellos puede generar que el modelo arribe a conclusiones inadecuadas, generando predicciones o respuestas lejanas a la realidad. En relación con el modelo, al momento de seleccionar el algoritmo y la técnica a utilizar, es importante considerar el tiempo, el rendimiento requerido y los recursos disponibles. El inconveniente principal que podría presentarse es que tal modelo presente una "sobre adaptación" o una "subadaptación" en función de los datos recibidos. En el primero de los casos, el modelo se ajusta excesivamente a los datos provistos, por lo que fallaría a la hora de reconocer los patrones y relaciones vigentes entre los datos, ya que se ha centrado principalmente en memorizarlos. En el caso de la "subadaptación", por ser demasiado simple, el modelo no capturaría la complejidad de los patrones existentes entre los datos, arribando también a resultados poco precisos. (págs.32-36)

En relación con lo anterior, podemos indicar que los siguientes puntos son de vital importancia a la hora de construir y entrenar nuestro modelo:

- Los datos utilizados a la hora de entrenar el modelo deben ser lo suficientemente representativos de los nuevos datos sobre los que deseamos que sean aplicadas las vinculaciones que ha encontrado el algoritmo.

- Los datos deben limpiarse para despojarlos al máximo posible de valores extremos, errores o ruido, de esta forma más fácil para el modelo encontrar los patrones subyacentes.

- El modelo seleccionado para cada caso en particular debe ser aquel que posee un balance entre la capacidad del modelo para ajustarse a los datos de entrenamiento y la simplicidad, que le permitirá generalizar los patrones encontrados a nuevos datos. Para ello podremos utilizar un hiperparámetro de regularización, que ayuda a controlar la complejidad del modelo evitando que se ajuste en exceso a los datos y mejorando sus capacidades de generalización.

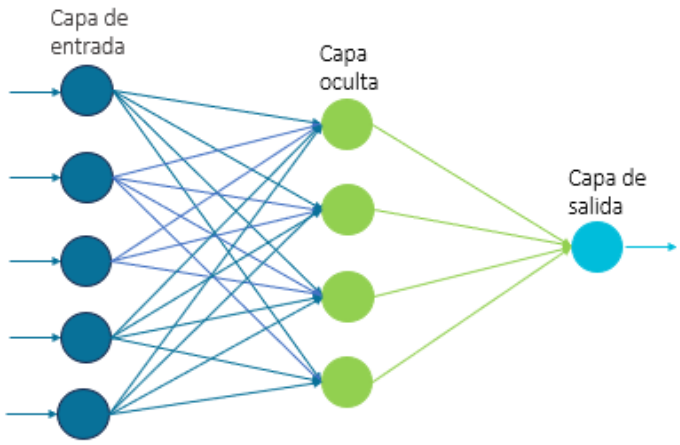
2.2 Técnicas de machine learning

2.1.1. Redes neuronales simples

Según lo expuesto por Isasi P. y Galván I. en "Redes de neuronas artificiales, un enfoque práctico" (2004)[12], las redes neuronales artificiales son conjuntos de neuronas que procesan y elaboran cierta información de entrada recibida con el objeto de obtener una salida.

Cada neurona posee la capacidad de presentar un estado activo o inactivo, dependiendo dicho nivel de activación de las diversas entradas recibidas por tal célula, ponderadas por el peso asignado a la fuerza de cada conexión sináptica presente entre las neuronas. Por su parte, el cuerpo de la neurona suma las entradas ponderadas algebraicamente por su peso y produce una única salida.

Figura 1: Estructura de una red neuronal artificial de dos capas.



Fuente: Elaboración propia a partir de Isasi Viñuela, P., & Galván León, I. (2004). Redes neuronales artificiales, un enfoque práctico. Pearson.

En este sentido, una red neuronal esta compuesta por un conjunto de neuronas interconectadas por una red multicapa, donde el primer nivel está compuesto por cédulas de entrada que reciben la información que ingresa a la red. A continuación, existen una o varias capas intermedias u ocultas de neuronas que conectan las cedulas de entrada con el último nivel, compuesto por las células de salida que sirven como salida de toda la red. Las interconexiones entre las células de la red transmiten datos y los evalúan en función de los pesos de las conexiones, ajustando estos durante la fase de aprendizaje.

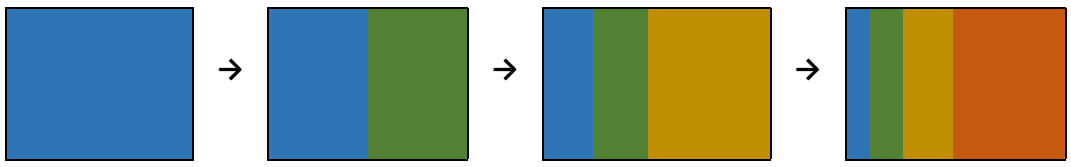
Puesto que el aprendizaje de una red neuronal consiste en determinar los pesos ponderados de sus conexiones a efectos de brindar una respuesta eficiente a un problema, es de gran importancia que cuente con datos representativos y en cantidad suficiente con el objetivo de que al introducir los mismos durante esta etapa, la red sea capaz de modificar los pesos de las conexiones en caso de que aún no se produzca la salida adecuada. (Isasi y Galván, 2004, pág.6-15)

2.1.2. Deep learning – Redes neuronales recurrentes

En función de lo expuesto por Goodfellow, I., Bengio, Y., y Courville, A. en "Deep Learning" (2016) [5] , las redes neuronales recurrentes son una familia de redes neuronales que se utilizan para procesar datos recurrentes teniendo en cuenta la información previa de cada elemento de la secuencia. Esto es posible gracias al "estado oculto", que consiste en que la memoria se actualiza no solo con la información de la entrada actual, sino también con la información que la red neuronal ha almacenado en instancia previa. (Goodfellow, Bengio, & Courville, 2016)

Puesto que las redes neuronales recurrentes (RNN) cuentan con una memoria de corto plazo, los datos que se procesan al final tendrán mayor peso que aquellos procesados en un inicio, lo que se conoce como desvanecimiento de gradiente. Esto genera que pueda recibirse poca información realmente útil a efectos del aprendizaje.

Figura 2: Estructura de memoria en las redes neuronales recurrentes



Fuente: elaboración propia a partir de *Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Network* de Staudemeyer, R., y Rothstein Morris, E. (2019).

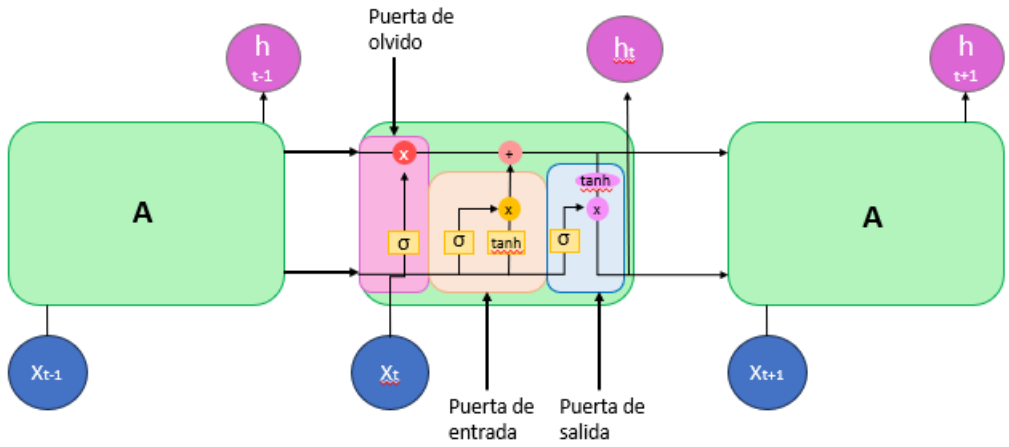
En este sentido, a efectos de compensar la carencia presentada por las RNN, se emplea arquitectura conocida como "Long short-term memory" (LSTM). La LSTM añade operaciones complementarias a efectos del algoritmo conozca que información ha de

[5] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.
[12] Isasi Viñuela, P., & Galván León, I. (2004). Redes neuronales artificiales, un enfoque práctico. Pearson.

olvidar y como ha de modificar el estado oculto, ayudando a crear una memoria selectiva para nuestra red neuronal y evitando que dicha memoria se componga parcialmente por datos que no resultan productivos para el algoritmo. (Hochreiter & Schmidhuber, 1997) [6]

La LSTM posee tres tipos de compuertas, la de entrada, la de olvido y la de salida. Si bien las tres ayudan a almacenar información útil a largo plazo, controlando el flujo de esta; cada compuerta posee una función característica.

Figura 3: Estructura de compuertas dentro de las redes neuronales recurrentes



Fuente: Elaboración propia a partir de *An Overview on Long Short Term Memory (LSTM). Data Science Blogathon* de Debasish, K. (2022).

La compuerta de entrada se encarga de determinar la nueva información que ha de ser almacenada en la memoria de la celda. Por el contrario, la compuerta de olvido indica cual es la información antigua que debe ser eliminada de la memoria. Por último, la compuerta de salida es aquella que establece la información que se transmitirá a la siguiente capa de la red, evitando que la información útil se desvanezca en el entrenamiento del modelo.

En función del análisis realizado por Meshram y Kulai (2022), donde se estudió la precisión de modelos de regresión linear, Support Vector Marchines (SVM) y LSTM a la hora de predecir el precio de las acciones, tomando como parámetros el precio de apertura, cierre, ajuste y volumen. Se arribó a la conclusión de que la LSTM tuvo un mayor grado de precisión gracias a la capacidad de procesar la información histórica de

[6] Hochreiter,S., Schmidhuber, J. (1997). Long short-term memory. Neural Computation

manera eficiente en series de tiempo complejas, olvidando aquellos datos que no son relevantes para su propósito. (Meshram & Kulal, 2022).

Entendemos que la prevalencia del modelo LSTM se funda en su capacidad de manejo de series temporales con dependencias a largo plazo. Sin embargo, a la hora de optar por este modelo, hay que considerar también que requiere un mayor tiempo de entrenamiento y una mayor cantidad de datos en comparación con SVM y con modelos de regresión lineal.

2.1.3. Modelos de regresión - Support Vector Machines

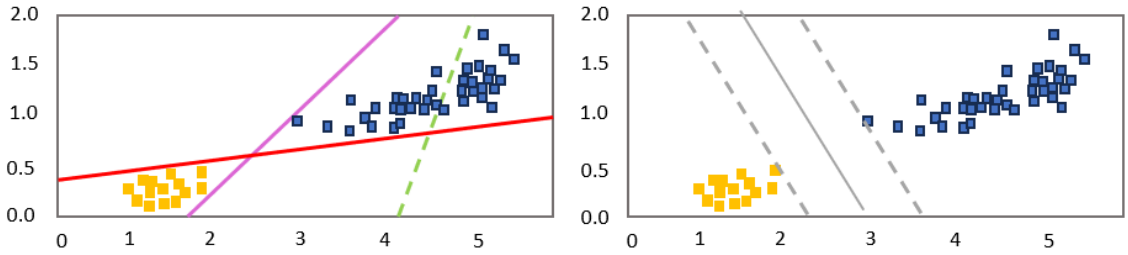
Géron (2019) indica que estos algoritmos de aprendizaje supervisado muy utilizado para clasificaciones y regresiones, que consiste en separar mediante un vector las distintas clases de datos. Si bien en muchos casos los vectores pueden trazarse en diversos sentidos y siguiendo diversas funciones a la hora de dividir las clases de datos, la clave está en que el vector óptimo será aquel que brinde la máxima separación entre los puntos que se encuentren más cercanos al vector divisorio. (pag.157-159)

De acuerdo con Géron (2019), el vector que mencionamos puede ser lineal o no lineal, dentro de los cuales encontramos al kernel polinómico y al Gaussiano, entre otros. (pag.157-165)

Función Kernel Lineal

Según Géron (2019) puede visualizarse en la figura de la derecha como la línea en color gris separa las dos clases de datos, maximizando la distancia entre los datos más cercanos de ambas clases (margen). La máxima distancia respecto de los datos de ambas clases en relación con el vector indica que ese es el vector que mejor separa ambas clases y que funcionaria adecuadamente a la hora de clasificar nuevos datos. Por el contrario, en la figura de la izquierda se visualiza como los datos pueden ser divididos mediante una línea recta pero no queda margen alguno entre los datos más cercanos al vector y el mencionado vector. Esto implica que si bien el modelo funcionaria para los datos expuestos, no sería eficiente a la hora de clasificar nuevos datos desconocidos al momento. (pág. 157)

Figura 4: División de un *dataset* aplicando función kernel lineal



Fuente: elaboración propia a partir de Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2da Edición.* O'Reilly Media, Inc.(Pág 158)

En nuestro caso particular, trabajar con un modelo SMV con kernel lineal es una opción viable ya que permitiría trabajar con grandes volúmenes de datos históricos sin incurrir en un gran riesgo de "sobre adaptar" el modelo, como sucedería en caso de emplear redes neuronales.

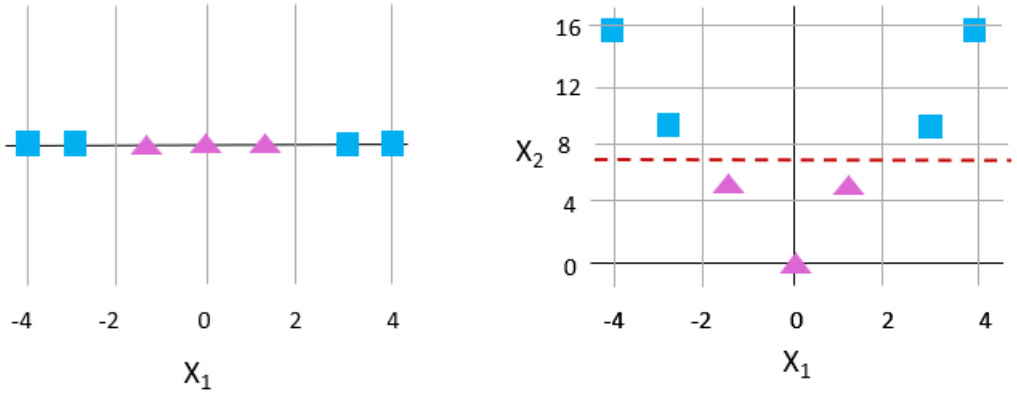
Función Kernel Polinómica

Existen *datasets* cuyas diferentes clasificaciones de datos no pueden ser separados mediante una línea recta, debe trabajarse sobre sus características, elevándolas a un exponente para obtener características polinómicas que si puedan ser separadas de dicha manera. (Géron, 2019. Págs. 160-161)

Una característica es un atributo o propiedad que describe los datos. Por ejemplo, en el caso de las acciones, una característica podría ser el volumen de operaciones, su precio al cierre, etc. Para obtener nuevas características polinómicas, se elevaría a un exponente tales características, lo que nos permitiría contar con nuevas características polinómicas que facilitaran el modelado de una posible relación entre el volumen de operaciones y el precio al cierre de las acciones, separándolas mediante una línea recta.

En la ***figura 5*** se visualiza como el *dataset* no puede ser subdividido mediante una línea recta considerando sus características originales, es por ello por lo que se eleva las mismas al cuadrado, obteniendo nuevas características calculadas en función de las originales pero representadas en bidimensionalmente, lo que permite clasificar los datos mediante la utilización del vector.

Figura 5: División de un *dataset* aplicando función kernel polinómica.



Fuente: elaboración propia a partir de Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2da Edición.* O'Reilly Media, Inc.(Pág 161)

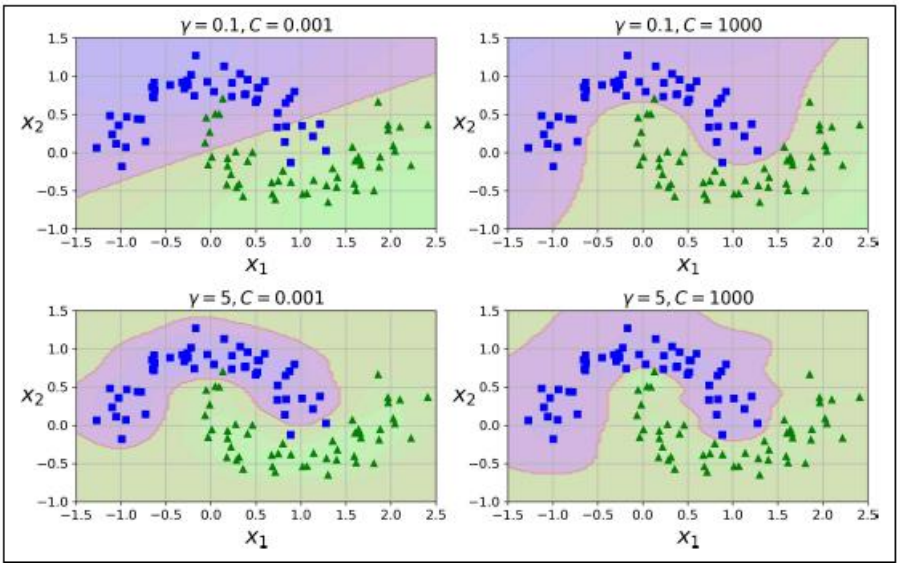
Según Géron (2019), al momento de implementar un kernel polinomial, debe considerarse que un polinomio de grado bajo no será suficiente para trabajar con *datasets* complejos, pero uno de grado muy elevado ralentizará demasiado el modelo puesto que trabajará con múltiples dimensiones. (pág. 162)

En este sentido, a efectos de predecir el precio de las acciones, la aplicación de un kernel polinómico sería ventajoso puesto a que el modelo sería capaz de trabajar con datos no lineales y reconocer patrones y relaciones más complejas entre los datos. La desventaja de lo anterior radica en la complejidad de comprender tales relaciones, puesto que el modelo puede ser difícil de interpretar.

Función Kernel Gaussiana (RBF)

En línea con la perspectiva de Géron (2019), la función kernel Gaussiana consiste en una función que nos brinda el mismo valor que si proyectásemos todos y cada uno de nuestros datos en un espacio dimensional superior, reduciendo tanto el tiempo de procesamiento y la necesidad de poder de procesamiento. Al momento de obtener dicha función, busca maximizar el margen entre los vectores de soporte y el hiperplano. (pág.164)

Figura 6: División de un *dataset* aplicando función kernel RBF.



Fuente: Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2da Edición.* O'Reilly Media, Inc.(Pág 165)

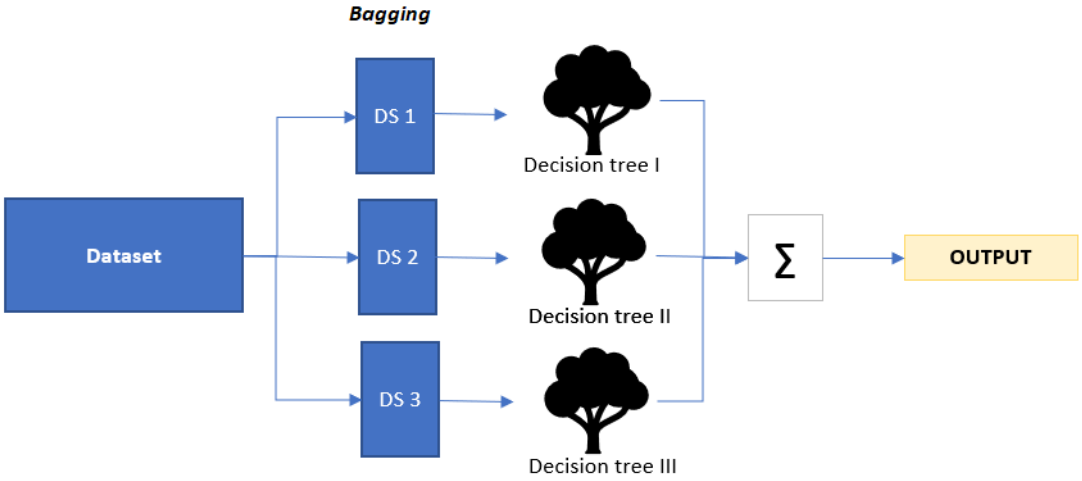
De acuerdo con un estudio realizado por Miranda Henrique, B., Amorim Sobreiro, V. y Kimura, H. (2018) [4] donde se emplea la técnica llamada *Support Vector Regression (SVR)* a efectos de predecir el precio de las acciones con grande y pequeñas capitalizaciones, utilizando precios diarios y frecuencias minuto a minuto. Los resultados obtenidos indican que la técnica *SVR* cuenta con poder de predicción, especialmente si se actualiza el modelo de forma periódica. Se detectó que el modelo incrementa su precisión en periodos de menor volatilidad y que las predicciones realizadas sobre valores diarios contaban con mayor precisión que aquellas efectuadas en frecuencia minuto a minuto. En los modelos se aplicaron tanto kernel lineal, como gaussiano y polinómico, siendo el lineal aquel que mostro mayor poder predictivo. Asimismo, el modelo obtuvo un mejor rendimiento en el *testing set* que en el *training set*, lo que indica que pudo generalizar correctamente a partir de los datos de entrenamiento y no ha incurrido en una “sobre adaptación” a ellos.

2.1.4. Modelos de regresión - Random Forest

De acuerdo con Breiman (2001), Random Forest es un algoritmo de machine learning supervisado, que se basa en la creación de diversos arboles de decisión que, combinados, mejorarían su poder de predicción. Entre las alternativas utilizadas para combinar los árboles de decisión, entre ellas se encuentra el *bagging*, que consiste subdividir el *dataset* y entrenar varios modelos independientes con una de dichas partes

del *dataset*, para posteriormente combinar las predicciones de los modelos entrenados, obteniendo una predicción más robusta al reducir la varianza de sus predicciones.(pág. 1 a 7)

Figura 7: Estructura de un modelo de regresión Random Forest.



Fuente: elaboración propia a partir de Random Forest de Breiman,L.(2001).

En función del estudio publicado por Sijian,T. (2023), donde se comparó la precisión de un modelo de regresión lineal múltiple, LSTM y una regresión random forest para predecir el precio de las acciones de BMW, partiendo de sus cotizaciones históricas de los últimos 5 años. De acuerdo con los resultados de este estudio, el modelo de regresión lineal múltiple presento el nivel más alto de precisión con el menor error cuadrático, que mide el promedio de los errores elevados al cuadrado comparando los resultados arrojados por el modelo versus los valores reales. (Sijian, 2023)

Asimismo, el modelo *random forest* requiere de más recursos computacionales que el modelo de regresión lineal múltiple a la hora de entrenar el modelo, así como también una mayor cantidad de tiempo. Adicionalmente, debido a su estructura puede resultar complejo de interpretar como el modelo *random forest* arriba a sus predicciones, lo que dificulta también la detección de posibles errores.

2.1.5. Modelos de regresión – Regresión lineal

En línea con lo expuesto por Rencher & Schaalje (2007), un modelo de regresión lineal simple es aquel que modela la relación que presentan dos variables, mientras que para modelo de regresión lineal múltiple existe más de una variable predictora, que ocupa un rol importante a la hora de determinar el valor que es predicho por el modelo.(pag.17) [5]

Tal como allí se expone, un modelo de regresión lineal simple busca explicar o predecir la variable “Y”, basándose en los valores que toma la variable “X” (variable independiente).

$$Y = \beta_0 + \beta_1 X + N(0,\sigma^2)$$

Siendo:

- β_0 el valor que tomará Y en caso de que X sea cero.
- β_1 el impacto que posee el cambio de valor de X sobre el valor de Y.
- $N(0,\sigma^2)$ el error o cambio de valor de Y que no puede ser explicado por el modelo.

Por otro lado, un modelo de regresión lineal múltiple basa su explicación o predicción sobre el valor de “Y” en la influencia que posee más de una variable sobre el mismo.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + N(0,\sigma^2)$$

Siendo:

- β_0 el valor que tomará Y en caso de que X sea cero.
- $\beta_{1\ldots n}$ el impacto que posee el cambio de valor de $X_1, X_2, \ldots, X_n$ sobre el valor de Y.
- $N(0,\sigma^2)$ el error o cambio de valor de Y que no puede ser explicado por el modelo.

Por lo expuesto anteriormente, los modelos de regresión lineal pueden ser utilizados para la predicción de valores, considerando la asunción de que las correlaciones existentes al momento en que se colectaron los datos prevalecen al tiempo que realizamos tales predicciones. (Rencher & Schaalje, 2007, pág. 18)

Asimismo, en caso de optar por un modelo de regresión lineal múltiple, es importante considerar que las variables predictoras empleadas deben ser independientes entre sí, es decir, no contar con colinealidad entre ellas.

3. Creación de modelos de predicción de activos

3.1. Datos y metodología

En el presente apartado exploraremos el mundo de las predicciones del precio futuro de acciones, tomando como caso de estudio las acciones de Banco Santander S.A. (SAN.MC), una sólida entidad financiera fundada en 1857, que comenzó a cotizar en la Bolsa de Madrid (MCE) en junio de 1988.

Las predicciones realizadas por los modelos que expondremos a continuación se basan en la información histórica semanal disponible en Yahoo! Finanzas desde el año 2000 hasta mayo 2023, inclusive. Se ha optado por considerar los datos semanales en lugar de diarios, puesto que de esta forma se logra reducir el efecto del ruido y la volatilidad diaria a la que se encuentran sujetas las cotizaciones por causa de eventos aleatorios, logrando una visión más clara de la tendencia subyacente de la cotización. Asimismo, trabajar con datos semanales nos permite evitar datos faltantes, como sucedería a la hora de trabajar con valores diarios.

Se ha trabajado con cuatro modelos de aprendizaje automático supervisado, que se describirán brevemente a continuación. Si bien se ha descargado información semanal relativa al precio de apertura, precio de cierre y volumen negociado, todos los modelos descriptos se han basado en el análisis histórico de la evolución del precio al cierre, con el objeto de predecir el posible precio futuro.

El primer modelo consiste en redes neuronales recurrentes con *long short-term memory,* esto implica que es capaz de capturar patrones secuenciales en los precios de cierre históricos y aprender dependencias temporales presentes. En particular, este modelo trabaja con 50 celdas que buscan capturar y procesar tal información secuencial. Cada unidad posee su propia celda de memoria y tres puertas, una de olvido, una de entrada y otra de salida, recordando solo la información útil de cada a futuras predicciones. Asimismo, luego de las capas LSTM, existe una capa densa que genera la salida final del modelo al tomar los datos producidos por las unidades LSTM y transformarlos en un único valor que representa la predicción del precio de la acción. Estas características lo transforman en un buen candidato para conseguir el objetivo establecido.

El segundo modelo es el de regresión lineal múltiple. Este busca establecer una relación lineal entre los precios históricos en diferentes intervalos de tiempo, y el precio cierre futuro, capturando la dependencia temporal y los patrones históricos presentes en dichos datos a efectos de aprovechar la relación existente entre los precios históricos y el precio futuro para elaborar las predicciones.

El tercer modelo se presenta aplicando *support vector machines*, un algoritmo capaz de manejar problemas de regresión encontrando un hiperplano óptimo que se ajuste a los precios al cierre involucrados en la serie temporal, maximizando la separación entre ellos, y logrando predecir valores continuos que en este caso serán el precio futuro de tales acciones. En este sentido, el modelo cuenta con la capacidad de encontrar patrones subyacentes entre los datos históricos a efectos de extrapolarlos posteriormente al realizar predicciones.

El último modelo es el de bosque aleatorio, más conocido como *random forest,* este algoritmo cuenta con la capacidad de capturar relaciones no lineales, así como también características complejas que se presentan en los datos históricos. Asimismo, este modelo combina múltiples árboles de decisión independientes, lo que no solo hace posible captar una amplia gama de relaciones, sino que también lo convierte en una buena alternativa a la hora de reducir posibles riesgos de sobreajuste y generar predicciones más precisas y solidas.

3.2. Modelo de redes neuronales recurrentes (LSTM)

Como mencionamos anteriormente, las redes neuronales recurrentes constituyen un modelo muy atractivo a la hora de predecir los precios futuros de acciones. En este caso, para la predicción de precios semanales de SAN.MC optamos por una arquitectura de *Long Short-Term Memory* por las ventajas que posee a la hora de almacenar información relevante.

Este modelo también trabajará con las cotizaciones al cierre históricas semanales que fueron obtenidas de Yahoo! Finanzas desde el 1 de enero de 2000 hasta 31 de mayo de 2023 inclusive.

Al igual que en los otros modelos que se presentan en este trabajo, se ha utilizado el precio al cierre para trabajar con el presente algoritmo. Puesto que se trata de un modelo LSTM, es particularmente importante que todas las características contribuyan equitativamente al proceso de aprendizaje del modelo, permitiendo que este encuentre

patrones y relaciones de forma más efectiva. Es por ello por lo que se reajustaron los precios al cierre asignándoles un valor entre 0 y 1 mediante el objeto “MinMaxScaler” disponible en la librería scikit-learn. Esta transformación tiene por objeto que aquellas cotizaciones al cierre de gran magnitud dominen el modelo, generando un desajuste.

Asimismo, dicha normalización tendría un rol aún más relevante en caso de que el modelo trabaje con más de una característica, como por ejemplo si se desease considerar no solo el precio al cierre de las acciones, sino también su volumen negociado para el entrenamiento del modelo, puesto que el peso que se le atribuiría a los datos de cada característica se encontraría dentro de un mismo rango.

Por otro lado, el tamaño de la ventana (*“window_size”*) que en este caso particular fue fijado en 30 días, también es un dato fundamental que hemos de indicarle al modelo, puesto que determina la cantidad de días históricos que serán considerados por el modelo a la hora de predecir el precio de SAN.MC.

Los datos correspondientes a las cotizaciones al cierre normalizadas se dividen en dos conjuntos, uno de entrenamiento, que representa el 80% del *dataset* y uno de prueba, que se constituye por el 20% restante.

A la hora de construir el modelo LSTM se emplea la clase “Sequential”, presente en TensorFlow, que proporciona una interfaz para apilar capas en un orden determinado.

En este caso existen dos capas, la capa LSTM y la capa de salida. La primera de ellas es la encargada de aprender las relaciones secuenciales presentes en los datos, a mayor sea el número de neuronas en una capa, mayor será la posibilidad de que el modelo se sobreajuste a los datos de entrenamiento y presente dificultades a la hora de generalizar a datos nuevos. Para este caso particular, tras experimentar con diferentes configuraciones, se escogió trabajar con 50 unidades de memoria o neuronas, puesto que brindaron un equilibrio entre el ajuste del modelo a los datos proporcionados y su generalización a datos nuevos. Por otro lado, la capa de salida es aquella que toma como entrada la información producida por la capa LSTM y se encarga de realizar la predicción del precio de cierre.

En este caso, el conjunto de entrenamiento también se compone del 80% del *dataset*, siendo el conjunto de prueba el 20% restante.

A la hora de compilar el modelo, es decir, configurar los aspectos necesarios para que este se encuentre preparado para el proceso de entrenamiento, se seleccionó el

optimizador de Adam. Este algoritmo permite que los pesos y los sesgos de las neuronas se ajusten de manera tal que minimicen el error cuadrático medio (función de perdida aplicada en este modelo). De esta manera, al entrenar el modelo con datos históricos de precios de acciones, el optimizador de Adam calculará los gradientes de la función de pérdida en referencia a los pesos y los sesgos del modelo en cada iteración del entrenamiento, utilizando la información para actualizar los parámetros del modelo para acercarse al mínimo error cuadrático medio posible. Los mencionados gradientes, representan la pendiente de la función de pérdida y brindan información acerca de la dirección y la magnitud del cambio necesario para minimizar dicha métrica.

En referencia al entrenamiento del modelo, se estableció que el mismo trabajará con 100 épocas, lo que indica que el algoritmo recorrerá 100 veces el conjunto de entrenamiento durante su proceso de aprendizaje. La importancia del número de épocas radica en la posibilidad que se le brinda al modelo de rever en varias oportunidades los mismos datos, a efectos de que logre capturar patrones mas complejos y mejorar así su rendimiento.

Asimismo, durante cada iteración de entrenamiento, este modelo trabajará con un lote de 32 muestras correspondientes al conjunto de entrenamiento, que son utilizadas para actualizar los pesos del modelo. Dicho tamaño de lote permite actualizaciones frecuentes gracias a su tamaño, a la vez que requiere de menos memoria para almacenar los gradientes.

Recordando que las redes neuronales tienen celdas de memoria y puertas que controlan el flujo de información. Existen dos parámetros numéricos que se encuentran asociados a tales celdas de memoria y puertas, controlando de esta forma el procesamiento de la información. Estos parámetros son los pesos y sesgos. Los pesos son aquellos valores numéricos asociados a las conexiones entre las neuronas del modelo, es decir, son los valores que determinan la contribución relativa de las neuronas de entrada en la salida de la neurona actual. Por su parte, los sesgos, son valores constantes que permiten que el modelo pueda ajustar la salida y aprender patrones más complejos y no lineales. (Staudemeyer & Rothstein Morris, 2019, pág 6)[7]

[7] Staudemeyer, R., & Rothstein Morris, E. (2019). Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks. *Reaserch Gate*.

En resumen, el modelo utiliza el error cuadrático medio como medida para ajustar los pesos y sesgos obtenidos con la predicción inicial mediante la aplicación del algoritmo de optimización Adam, de forma tal que las predicciones se acerquen lo máximo posible a las cotizaciones al cierre actuales, repitiendo dicho proceso de ajuste iterativamente durante el entrenamiento hasta finalmente lograr minimizar ese error.

Una vez entrenado el modelo, se aplican las relaciones encontradas al conjunto de prueba, dando por resultado las predicciones del modelo.

Figura 8: Comparativa de la cotización real y la predicción del precio de SAN.MC realizada por el modelo LSTM por los años 2018 a 2023.

Fuente: elaboración propia a partir de las predicciones del modelo LSTM.

3.3. Modelo de regresión lineal

Se aborda la predicción del precio semanal futuro de la acción SAN.MC aplicando un modelo de regresión lineal, donde se utilizará la información histórica semanal de las cotizaciones al cierre descargadas de Yahoo! Finanzas desde enero 2000 hasta mayo 2023.

Luego de contar con la descarga de los datos, se eliminan aquellos valores nulos o faltantes para que los datos se encuentren preparados para ser procesados por el modelo.

Al igual que en los casos anteriores, también se considerará el 80% del *dataset* como parte del conjunto de entrenamiento y el 20% restante comprenderá el conjunto de prueba.

El modelo de regresión lineal es utilizado mediante la clase “LinearRegression”, incluida en la biblioteca sklearn.linear_model, aplicando para el modelo los intervalos de retardo (*lag*) [1, 2, 4, 8]. Esto implica que, a la hora de predecir el precio semanal al cierre de las acciones, se considerarán las semanas anteriores indicadas en el periodo de retardo para capturar la relación entre los precios históricos y los futuros.

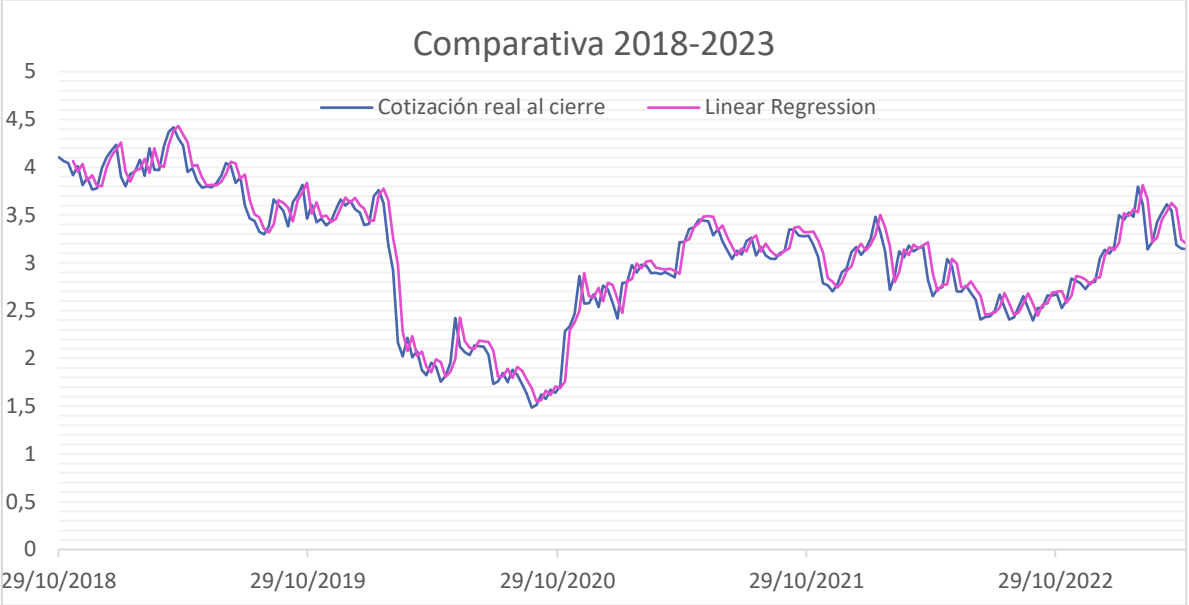
Las características de retardo correspondientes tanto al conjunto de prueba como al conjunto de entrenamiento son almacenadas en los *dataframe* “X_train_lagged” y “X_test_lagged”, lo que permite almacenar las relaciones capturadas entre los precios de semanas anteriores y el precio actual.

En otras palabras, a efectos de predecir el precio de la acción SAN.MC, el modelo no utiliza directamente los precios al cierre como variable de entrada, sino que crea una matriz de características de retardo, que constituyen variables independientes en el modelo, utilizando los diferentes intervalos de retardo de los precios anteriores. Es decir, en lugar de tener una sola variable de entrada correspondiente al precio de cierre de la semana anterior, utilizan múltiples variables de retardo que corresponden al precio de cierre de la semana anterior, el precio de cierre de hace dos semanas, el precio de cierre de hace 4 semanas y el de hace 8 semanas, para predecir el precio de cierre futuro.

Puesto que se cuenta con mas de una variable de entrada, durante el entrenamiento, el modelo ajusta los coeficientes que determinan la contribución relativa de cada variable de retardo para dar con la mejor combinación a la hora de la predicción del precio futuro.

Una vez capturadas las relaciones y patrones por parte del modelo, este aplica la combinación lineal de las variables de retardo ponderadas por los coeficientes asignados durante el entrenamiento a los nuevos datos que se encuentran almacenados en el conjunto de prueba, a efectos de estimar los precios futuros.

Figura 9: Comparativa de la cotización real y la predicción del precio de SAN.MC realizada por el modelo de regresión lineal por los años 2018 a 2023.



Fuente: elaboración propia a partir de las predicciones del modelo de regresión lineal.

3.4. Modelo de support vector regression

En esta sección se analiza la capacidad de modelo de *support vector regression* al ser aplicado a la predicción de precios semanales de la acción SAN.MC.

Este modelo también trabajará con las cotizaciones al cierre históricas semanales que fueron obtenidas de Yahoo! Finanzas desde el 1 de enero de 2000 hasta 31 de mayo de 2023 inclusive.

Luego de efectuar una transformación de los datos, tales como la fecha, a un formato que facilite su manipulación a la hora de utilizarla para el modelo, se dividen los datos en dos conjuntos. El conjunto de entrenamiento se compone del 80% de los datos, mientras que el conjunto de prueba toma el 20% restante.

El modelo *support vector regression* consiste en una variante de las *support vector machines*, que trabaja con regresiones utilizando en este caso particular, la función de kernel RBF (*Radial basis function*). El kernel RBF fue seleccionado porque posee una complejidad superior a la de un kernel lineal, permitiendo proyectar en un hiperplano datos que en un inicio no eran separables linealmente, sin modificar realmente el *dataset*.

Si bien no es el objeto del presente trabajo ahondar en los componentes de la función estadística del kernel RBF, consideramos importante destacar uno de sus componentes, gamma.

$$K(x,xi) = \exp(-gamma * sum(x-xi)^2)$$

Siendo:

- Gamma un parámetro que va de 0 a 1.

En el presente modelo, gamma representa la influencia que posee cada dato de entrenamiento en la construcción del modelo, tomando un valor de 0.22, lo que indica una influencia moderada. Es importante destacar que, en caso de contar con un valor muy elevado para gamma, siendo 1 el máximo, estaríamos exponiendo al modelo a un sobreajuste, causando que se replicaran los datos de entrenamiento puesto que el modelo asignaría un peso significativo extremadamente elevado a los datos cercanos, perdiendo completamente la capacidad de generalizar al momento de aplicarlo a los datos de prueba.

En primera instancia, se entrena el modelo con la función ".fit()" sobre el conjunto de entrenamiento, generando que este establezca relaciones entre las variables predictoras, representadas por los precios históricos al cierre y la variable objetivo, representada por el precio futuro al cierre.

Una vez entrenado el modelo, se aplica la función ".predict" utilizando el conjunto de prueba, con el objeto de conocer el comportamiento del modelo a la hora de predecir precios futuros en función del hiperplano que ha creado previamente, pero aplicado a nuevos datos.

Figura 10: Comparativa de la cotización real y la predicción del precio de SAN.MC realizada por el modelo support vector regressor por los años 2018 a 2023.

Fuente: elaboración propia a partir de las predicciones del modelo SVR.

3.5. Modelo de random forest

Con el objeto de estimar la evolución futura de los precios al cierre de SAN.MC, se optó por evaluar el desempeño de un modelo que utiliza el algoritmo de *random forest.*

Este modelo también trabajará con las cotizaciones al cierre históricas semanales que fueron obtenidas de Yahoo! Finanzas desde el 1 de enero de 2000 hasta 31 de mayo de 2023 inclusive.

Luego de preparar los datos mediante conversiones de formato y eliminar posibles valores faltantes existentes, se define el modelo utilizando la librería scikit-learn de Python, particularmente el módulo “RandomForestRegressor”.

Asimismo, se establecen los parámetros con los cuales tal modelo habrá de operar, entre los que se indica que el número de árboles a emplear es de 100 y el *random state* es de 42.

En un modelo de *random forest*, la cantidad de árboles utilizados determina la precisión con la que cuenta el modelo, siendo necesaria una cantidad lo suficientemente alta para reducir los sesgos y mejor el rendimiento; pero no excesivamente alta, dado que ello conduciría a un sobreajuste, además de incrementar considerablemente el tiempo de entrenamiento del modelo y los recursos computacionales necesarios para ello. Es por

dicho motivo que, basándonos en que la cantidad de datos utilizados asciende a 1200, se ha optado por emplear 100 árboles de decisión, cantidad con la que resultaba viable operar dados los recursos materiales y temporales disponibles.

Por su parte, los datos son divididos en dos conjuntos a la hora de trabajar con el modelo, un conjunto de entrenamiento y otro conjunto de prueba. El *random state,* es un parámetro que influye en dicha división, ya que de no definirlo (random_state = None), se obtendrán automáticamente diferentes conjuntos de entrenamiento y de prueba cada vez que se ejecute el código, obteniendo diversos resultados. En este caso particular, puesto que buscamos realizar una comparativa entre modelos, hemos decidido priorizar la reproducibilidad del código, estableciendo un *random state* de 42, puesto que es el número más utilizado comúnmente. Sin embargo, los resultados serían los mismos si se hubiese optado por establecer cualquier otro número entero.

El siguiente paso consiste en dividir el conjunto de datos o *dataset* en dos subconjuntos, uno de entrenamiento (*training set*), que se compone del 80% de los datos, y uno de prueba (*test set*) que contiene el 20% restante del *dataset*.

El entrenamiento del modelo se realiza mediante la aplicación del método ".fit()" sobre el *training set* , permitiendo en esta instancia que cada árbol de decisión entrene con un determinado *sub-training set*. Esto permitirá que luego pueda arribarse a una conclusión conjunta del modelo, que en este caso se tratará de una predicción del precio al cierre de la acción SAN.MC.

Por último, con el objetivo de obtener las predicciones que el modelo es capaz de efectuar, se utilizará el método ".predict()" sobre el *test set*. En este sentido, el método ".predict()" es utilizado para realizar predicciones sobre precios futuros valiéndose de información histórica nueva, es decir, sobre la que no fue entrenado. Para realizar tal predicción, el modelo se nutre de las relaciones entre los datos que cada árbol ha creado durante su entrenamiento y luego obtiene un precio final, que surge del promedio de los resultados predichos por cada árbol individual.

Es importante destacar que, a diferencia de los modelos anteriores, el modelo de random forest se han efectuado dos variantes.

La primera variante contempla que el entrenamiento se realice con información histórica disponible hasta septiembre de 2018, como sucede en los modelos anteriores, y posteriormente, con los patrones aprendidos hasta tal fecha, prediga las cotizaciones al cierre correspondientes a octubre 2018 en adelante.

En la figura 10 puede visualizarse como el modelo cuenta con buena capacidad de predicción durante el primer año, pero luego no es capaz de seguir la tendencia bajista que se da durante el año 2020, al ocurrir la pandemia del COVID-19. En este sentido, la mera visualización de las predicciones nos indica que este modelo no es lo suficientemente flexible para adaptarse a situaciones inusuales dado que no captura el cambio drástico que sufre el mercado durante tal época.

Figura 11: Comparativa predicción random forest con split en 2018

Fuente: elaboración propia a partir de las predicciones del modelo.

Es por ello por lo que generamos una segunda variante del modelo con la que se trabajó contemplado dentro de su conjunto de prueba diversas predicciones de cotizaciones al cierre que abarcan desde enero del 2000 hasta mayo 2023, distribuyendo de forma aleatoria las fechas sobre las cuales aplica la predicción, que es realizada siempre en función de los datos históricos disponibles al momento. Es por dicho motivo que en la ***figura 11*** pueden visualizarse líneas rectas que conectan dos cotizaciones al cierre predichas por el modelo con gran grado de precisión, pero tal precisión escasea en los puntos intermedios. En este caso, el modelo logra captar la tendencia bajista de la acción durante la pandemia del COVID-19, denotando una mejor adaptación al contexto económico y financiero, pero carece de la entrega de predicciones semanales continuas.

Figura 12: Comparativa de la cotización real y la predicción del precio de SAN.MC realizada por el modelo de random forest por los años 2018 a 2023.

Fuente: elaboración propia a partir de las predicciones del modelo Random Forest.

4. Metodología aplicada a la evaluación y validación

Géron (2019) sugiere que la mejor forma de evaluar si el modelo funciona correctamente es dividir el *dataset* en dos, asignando aproximadamente un 80% de los datos al entrenamiento del modelo (*training set*) y el 80% restante a la posterior evaluación (*testing set*). Al procesar el *testing set*, existirá una tasa de acierto y una tasa de error en los que ha incurrido el modelo, lo que se conoce como error de

[4] Miranda Henrique, B., Amorim Sobreiro, V., & Kiruma, H. (2018). Stock price prediction using support vector regression on daily and. *Science Direct* .

[5] Rencher, A., & Schaalje, G. (2007). *Linear models in statistics.* Wiley-interscience.

generalización, e indica cuan bueno ha sido el desempeño del modelo ante casos desconocidos. (pag.37)

En el presente trabajo se evaluará el desempeño de los cuatro modelos bajo análisis aplicando las métricas que se describirán a continuación, las cuales serán calculadas respecto del conjunto de prueba.

El error absoluto medio (MAE) calcula la dispersión promedio entre los valores predichos y los actuales, en términos de módulo, evitando que se neteen los errores expresados en valores positivos con los negativos. Es deseable que el MAE obtenido sea el menor valor posible. (Géron, 2019, p.47)

$$MAE = \frac{\sum_{i=1}^{n} |\text{Valores actuales} - \text{Valores predichos}|}{N}$$

Siendo:

- N: el total de datos predichos.

El error cuadrático medio (MSE) calcula la dispersión promedio presente en los errores de los valores predichos en comparación con los actuales. Por tal motivo, al utilizar el MSE para comparar la precisión de los modelos, buscaremos aquel que brinde el mínimo valor posible. (Mood,Graybill y Boes, 1974, p. 291) [8]

$$MSE = \frac{\sum_{i=1}^{n} (\text{Valores actuales} - \text{Valores predichos})^2}{N}$$

Siendo:

- N: el total de datos predichos.

[8] Mood, A., Graybill, F., & Boes, D. (1974). Introduction to the theory of statistics. McGraw-Hill International book company.

A diferencia del MAE, esta métrica penaliza los errores más significativos dado que le asigna un peso porque las diferencias se encuentran elevadas al cuadrado.

Por su parte, R al cuadrado (R^2) es una métrica estadística que brinda información acerca de cuan preciso es un modelo, puesto que indica la proporción de la variable dependiente que puede ser explicada por la variable independiente del modelo. En este sentido R^2 siempre toma valores entre 0 y 1, siendo 1 el indicador de que la variabilidad de la variable dependiente es explicada por el modelo, y 0 no es explicada en absoluto. (James, Witten, Hastie, y Tibshirani, 2017,p.69)[9]

$$R^2 = \frac{\sum_{i=1}^{n} (y_i - f(x_i))^2}{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

Siendo:

- $\sum_{i=1}^{n} (y_i - f(x_i))^2$: la suma de los errores cuadráticos calculados en función de los valores actuales (y_i) y los predichos por el modelo ($f(x_i)$). A menor sea esta sumatoria, mejor será el ajuste del modelo.
- $\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$: la suma de los cuadrados de los residuos, que indica la variabilidad no explicada por el modelo.

El error porcentual absoluto (MAPE) es una métrica que calcula en términos de promedio porcentual, la proporción que representa el error de la predicción en relación con el valor actual. En este sentido, el resultado expresado por la métrica hace referencia a la diferencia porcentual promedio entre las predicciones y los valores actuales, por lo que, a menor sea tal diferencia, mejor desempeño tendrá el modelo.(Kim y Kim, 2016, p.1) [10]

[9] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). An introduction to statistical learning with applications in R. Springer.

[10] Kim, S., & Kim, H. (2016). A new metric of absolute percentage error for intermittent demand forecasts. International journal of forecasting.

$$MAPE = \frac{\sum_{i=1}^{n} \left(\frac{|\text{Valores actuales} - \text{Valores predichos}|}{|\text{Valor actual}|}\right)}{N} * 100$$

Asimismo, luego de calcular las cuatro métricas expuestas anteriormente para cada uno de los modelos, se realizará una validación cruzada hacia delante de cada uno de ellos, puesto que al tratarse de series temporales es imprescindible considerar la dependencia temporal existente entre los datos a la hora de evaluar el desempeño del modelo. En este sentido, se genera una determinada cantidad de divisiones (*splits*) en el *dataset* que separa conjuntos de entrenamiento y conjuntos de prueba. Posteriormente, el modelo se entrena utilizando los datos de entrenamiento correspondientes a cada división y se evalúan los datos de prueba de cada uno de ellos. Por último, a efectos de calcular las métricas de desempeño del modelo, se toma en cuenta el promedio de las métricas obtenidas individuales calculadas para cada conjunto de prueba.(Hyndman y Athanasopoulos, 2021)[11]

La métrica seleccionada a la hora de comparar los resultados obtenidos por la validación cruzada de los modelos ha sido la raíz del error cuadrático medio (RMSE).

$$RMSE(x,h) = \sqrt{\frac{\sum_{i=1}^{n} \left(h\left(x^{(i)}\right) - y^{(i)}\right)^2}{N}}$$

Siendo:

- N: el número de muestras del *dataset*
- $h\left(x^{(i)}\right)$: las predicciones realizadas por el modelo
- $y^{(i)}$: los valores reales

[11] Rob J., H., & Athanasopoulos, G. (2021). *Forecasting: principles and practice.* Texts.

Esta métrica nos ayuda a conocer la desviación del modelo de predicción respecto de los valores reales, penalizando la existencia de errores significativos de predicción y generando un menor impacto por parte de aquellos que solo se desvían levemente del valor real. Será deseable obtener el menor valor de RMSE posible, puesto que ello implica que las predicciones del modelo no se desvían significativamente de los valores reales. (Géron, 2019,p.45)

4.1. Evaluación del modelo de redes neuronales recurrentes (LSTM)

En relación con el desempeño del modelo LSTM, se han calculado las métricas presentes en la siguiente tabla:

Tabla 1: Métricas de evaluación del modelo LSTM.

Estadísticas del modelo	
MSE	0,0497955
MAE	0,2460294
R^2	0,9187436
MAPE	0,0931705

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

El modelo cuenta con un error cuadrático medio (MSE) de 0,0498 entre los valores predichos por el modelo y los valores reales de la cotización al cierre de SAN.MC. Es importante recordar que esta métrica penaliza en mayor medida aquellas predicciones que son significativamente diferentes a los valores reales. Puesto que las cotizaciones oscilaron entre €1,48 y €4,41 durante el lapso por el cual se efectuó la predicción, se considera que un MSE de 0,0498 indica que el modelo cuenta con un buen ajuste puesto que las predicciones realizadas son cercanas a los valores reales.

El error medio absoluto (MAE) es de 0,246, lo que indica que las predicciones del modelo analizadas en términos de valor absoluto difieren en promedio en un 0,246 respecto de los valores reales. Basándonos en el rango de valores mencionado que han tomado las cotizaciones al cierre, un MAE de 0,246 es relativamente bajo, indicando nuevamente que el modelo cuenta con buena precisión.

En cuanto a R^2, recordamos que esta métrica representa el coeficiente de determinación, es decir, indica cuan bien se ajustan los valores predichos a los reales. En este caso, un R^2 de 0,9187 implica que este modelo es capaz de explicar aproximadamente un 91,87% de la varianza de los datos. En este caso, el modelo cuenta con un buen ajuste en términos de la variabilidad que es capaz de explicar, por lo que cuenta con una capacidad significativa de predicción.

El error porcentual absoluto (MAPE) del modelo asciende a 0,09317, lo que indica que en promedio las predicciones del modelo difieren en un 9,317% respecto de los valores reales, lo que es considerado relativamente bajo dado que el rango de cotizaciones variaba entre €1,48 y €4,41 durante el lapso por el cual se efectuó la predicción.

En relación con el proceso de validación cruzada hacia adelante, se utilizó la clase TimeSeriesSplit de la librería scikit-learn para dividir el *dataset* de manera adecuada, considerando que se trata de una serie temporal. En este caso se trabajó con 5 divisiones, siendo cada división un conjunto de prueba distinto para evaluar el rendimiento del modelo que entrenará con datos anteriores a la división, permitiendo posteriormente analizar el desempeño promedio del modelo a lo largo de la serie temporal. Asimismo, se le asigna un índice a cada dato, en función de la división a la que pertenece y de si se trata de un dato de entrenamiento o uno del conjunto de prueba.

En este caso, se ha definido una ventana de 30 observaciones, lo que implica que a efectos de predecir el precio al cierre semanal de SAN.MC, el modelo tomará en cuenta las 30 cotizaciones consecutivas anteriores. Esto permite evaluar el desempeño del modelo en diferentes ventanas temporales y analizar su capacidad para generalizar en diferentes momentos de la serie temporal los patrones que ha encontrado.

En el marco de cada división se crea un modelo secuencial que cuenta con una capa LSTM de 50 neuronas y una capa densa con una neurona cuya salida será el valor predicho por el modelo. El modelo se compila mediante el optimizador Adam, que ajusta automáticamente el algoritmo con cada iteración, a efectos de minimizar el error cuadrático medio (MSE) y a efectos de su entrenamiento, recorre todos los datos de entrenamiento 100 veces, para comprender las relaciones existentes entre los datos, que serán posteriormente generalizadas para efectuar la predicción.

Una vez entrenado el modelo, las predicciones correspondientes a cada división, así como también el MSE de cada una es almacenado en una lista de datos, que son

posteriormente utilizados para obtener las siguientes métricas promedio derivadas de la validación cruzada:

Tabla 2: Métricas promedio derivadas de la validación cruzada del modelo LSTM

Estadísticas de la validación cruzada	
RMSE promedio	3,1254929

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

En este caso, considerando que los valores de cotización al cierre del periodo sometido a la validación cruzada, que abarca desde enero del 2000 hasta mayo 2023 fluctuaron entre €1,48 y €12,87. Un RMSE promedio de 3,125 representa que las predicciones realizadas por el modelo han diferido en promedio en 3,125 de los valores reales, lo que implica una discrepancia significativa.

Este modelo ha presentado un buen desempeño al ser entrenado sobre el 80% de los datos y evaluado sobre el 20% restante correspondiente al conjunto de prueba, sin embargo, al realizar la validación cruzada la capacidad de predicción del modelo denota un deterioro significativo, que podría deberse a que no logra capturar adecuadamente el patrón temporal, lo que podría estar ligado al tamaño de la ventana asignado para la validación cruzada. Sin embargo, es importante destacar que en este caso no es posible asignar un tamaño de ventana superior a 30 dado que resultaría en una falta de datos para completar la ventana en algunos casos.

4.2. Evaluación del modelo de Regresión Lineal

En cuanto al desempeño del modelo de regresión lineal en la predicción del precio al cierre semanal de SAN.MC, se analizarán ciertas métricas que nos permitirán conocer la precisión de este.

Tabla 3: Métricas de evaluación del modelo de regresión lineal.

Estadísticas del modelo	
MSE	0,0006940
MAE	0,1234485
R^2	0,9409031
MAPE	0,0434691

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

Considerando que las cotizaciones durante el lapso temporal abarcado por el conjunto de prueba han oscilado entre los €1,48 y €4,41, un error cuadrático medio (MSE) de 0,00069 es considerado bajo, puesto que implica que en promedio las predicciones del modelo difieren en € 0,00069 de los valores de cotización al cierre reales. Tal resultado en el error cuadrático medio se encuentra en línea con la no existencia de diferencias altamente significativas entre las predicciones y los valores reales, puesto que esta métrica penaliza en mayor medida a los errores mas significativos al elevarlos al cuadrado. La mayor diferencia entre la predicción y el valory real en este modelo ha ascendido a € 0,822.

Con relación al error absoluto medio (MAE), el promedio de los errores de predicción del modelo contemplados con relación a su valor absoluto ha ascendido a 0,1234, lo que demuestra un buen grado de precisión por parte del algoritmo en términos absolutos considerando que el rango total de valores oscila entre €1,48 y €4,41.

En cuanto a R^2, la variabilidad de los precios semanales al cierre de SAN.MC puede ser explicada en un 94,09% por el modelo, lo que indica que este cuenta con un buen ajuste a los datos y es capaz de capturar la tendencia subyacente que presentan los precios y generalizarla para realizar predicciones.

El error porcentual absoluto medio (MAPE) asciende a 0,0434 lo que implica que, en promedio, las predicciones del modelo poseen un error absoluto de 4,35% con relación a las cotizaciones al cierre reales, lo que se considera un grado de precisión razonable.

El análisis de validación cruzada hacia adelante comienza con la división del *dataset* utilizando la clase TimeSeriesSplit de la librería scikit-learn, que toma en cuenta las características de la serie temporal. Cada división contará con su conjunto de prueba, permitiendo evaluar el desempeño del modelo a lo largo de toda la serie temporal.

Para este análisis se definió un tamaño de ventana equivalente a 30, lo que implica que se consideraran las 30 cotizaciones al cierre anteriores para predecir el precio al cierre semanal. Esto permitirá evaluar la capacidad del modelo para reconocer los patrones presentes en los precios de las acciones a través de diversos segmentos de la serie temporal.

Cada división del conjunto de prueba contará con un modelo de regresión lineal que una vez entrenado con los datos asignados a dicha división, generará las predicciones y las almacenará en una lista junto con el cálculo del MSE correspondiente a cada división.

Por último, una vez obtenidos los valores correspondientes a todas las divisiones, se calcula el RMSE promedio que proporciona una medida de rendimiento general del modelo en la validación cruzada hacia adelante.

Tabla 4: Métricas promedio derivadas de la validación cruzada del modelo de regresión lineal.

Estadísticas de la validación cruzada	
RMSE promedio	0,4014264

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

El rango de fluctuación de las cotizaciones al cierre desde enero de 2000 hasta mayo de 2023 ha ido desde €1,48 hasta €12,87. Un RMSE promedio de 0,4014 indica que, en promedio, las predicciones del modelo han diferido aproximadamente € 0,40 de los valores de cotización semanal al cierre reales de SAN.MC. Considerando el rango de fluctuación de los precios y las limitaciones temporales y de recursos presentes en esta tarea, entendemos que se trata de un buen desempeño.

En este caso, tanto las métricas calculadas sobre el conjunto de prueba que contenía el 20% del *dataset*, como aquella calculada en función de la validación cruzada han indicado que el modelo capturó las relaciones temporales subyacentes que se encuentran presentes en las cotizaciones al cierre y fue capaz de generalizarlas para efectuar predicciones con un buen grado de precisión.

4.3. Evaluación del modelo Support vector regression

En lo concerniente a la valoración del desempeño que presenta el modelo de *support vector regression*, se han calculado diversas métricas que se presentarán a continuación con el objeto de evaluar la precisión en la predicción de las cotizaciones al cierre de SAN.MC.

Tabla 5: Métricas de evaluación del modelo de SVR.

Estadísticas del modelo	
MSE	0,1579710
MAE	0,8065759
R^2	0,3490230
MAPE	0,3676619

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

En el presente modelo, el error cuadrático medio (MSE) es de 0,1579, lo que considerando el rango de variación de las cotizaciones que va de €1,48 y €4,41 resulta aceptable, aunque no particularmente prometedor. Esto se debe a que, si bien la gran mayoría de los errores son bajos, existen otros elevados que son penalizados en mayor medida por esta métrica, ascendiendo el error más significativo a € 3,9 en el momento en el cual se presenta una baja abrupta en el precio durante el año 2020.

Por su parte, el error absoluto medio (MAE), que considera el error en términos de valor absoluto presente entre las predicciones y los precios al cierre reales, asciende a 0,8065, indicando que, en promedio, dichas predicciones han diferido en € 0,80 de las cotizaciones reales al cierre de SAN.MC.

Con relación a R^2, el modelo tan solo es capaz de explicar la variabilidad de las cotizaciones al cierre en un 34,9%, lo que no resulta óptimo ya que el modelo no capturo relaciones y patrones subyacentes suficientes, dejando un porcentual significativo de tales variaciones sin explicación alguna, lo que dificulta la tarea de predicción.

En cuando al error porcentual absoluto medio (MAPE), este indicador expone que las predicciones del modelo han diferido, en promedio, en un 36,8% respecto de los valores reales de cotización al cierre, lo que resulta altamente significativo, no posicionando a este modelo como una opción óptima frente a los anteriormente analizados.

Por otro lado, la validación cruzada hacia adelante del modelo *support vector regressor* se ha efectuado considerando que se trata de una serie temporal. En este caso, se ha definido una ventana con tamaño 30 por lo que el modelo considerará los 30 datos consecutivos anteriores al valor a predecir con el objeto de encontrar los patrones subyacentes y aplicarlos posteriormente a la predicción.

El algoritmo calcula la cantidad de divisiones en función del tamaño de los datos de prueba y el tamaño de la ventana, lo que da por resultado unas 216 divisiones. En este sentido, existirán 216 conjuntos de prueba diferentes sobre los cuales se obtendrán predicciones y métricas de desempeño que serán posteriormente promediadas para obtener una métrica que abarque la totalidad del modelo.

Tabla 6 :Métricas promedio derivadas de la validación cruzada del modelo SVR.

Estadísticas de la validación cruzada	
RMSE promedio	4,5223590

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

En este caso, el rango de fluctuación de las cotizaciones al cierre desde enero de 2000 hasta mayo de 2023 ha ido desde €1,48 hasta €12,87. El modelo SVR presenta un error cuadrático medio de 4,5223, lo que se considera significativo en comparación con el tango detallado, así como también con el desempeño de los modelos analizados con anterioridad.

En resumen, este modelo no ha logrado explicar un porcentual significativo de las variaciones de los precios, lo que se condice con los grados de error que arrojan sus métricas al momento de evaluar la diferencia entre los valores predichos y las cotizaciones al cierre reales de SAN.MC.

4.4. Evaluación del modelo Random Forest

En cuanto al modelo de *random forest* se han calculado las métricas expuestas a continuación en función de las predicciones del precio semanal de SAN.MC efectuadas por la segunda variante del modelo en el conjunto de prueba, que representa el 20% del *dataset.*

Es importante destacar que, a diferencia de los tres modelos anteriores, en el modelo *random forest* las predicciones se realizaron sobre valores de fechas aleatorias, no siendo todas ellas posteriores a 2018, sino comprendidas entre enero del 2000 y mayo de 2023. La evaluación presente en este apartado se realiza sobre dicha totalidad de valores predichos, que son equivalentes a los valores predichos por los tres modelos anteriores (240 cotizaciones al cierre).

Tabla 7: Métricas de evaluación del modelo de random forest.

Estadísticas del modelo	
MSE	0,0000259
MAE	0,0083283
R^2	0,9999722
MAPE	0,0013863

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

El error cuadrático medio (MSE) es de 0,0000259, ascendiendo el error más significativo a € 0,099, lo que indica que el modelo ha logrado un buen grado de precisión en sus predicciones.

En cuando al error absoluto medio (MAE), la diferencia promedio entre los valores predichos por el modelo y las cotizaciones al cierre reales en términos de valor absoluto, ascienden a 0,0083, lo que nuevamente se encuentra en línea con un muy buen grado de precisión.

En línea con lo anterior, el modelo de *random forest*, logra explicar en un 99,99% la variabilidad de las cotizaciones al cierre, de acuerdo con lo expuesto en la métrica R^2. Esta métrica se acerca a la medida máxima alcanzable por R^2, demostrando que el modelo logra explicar casi la totalidad de la variabilidad de los datos.

El error porcentual absoluto medio (MAPE) es de 0,138%, lo que indica que, en promedio, existe un grado de error muy bajo en las predicciones efectuadas por este modelo, denotando nuevamente un excelente grado de ajuste y una muy buena capacidad de predicción.

Al analizar el modelo mediante validación cruzada hacia adelante, definiendo una ventana de tamaño 30 y calculando el numero de divisiones en función de dicha ventana y de la cantidad de datos de prueba disponibles, lo que en este caso da origen a 212 divisiones.

En este caso, los datos son divididos en un conjunto de entrenamiento y un conjunto de prueba, que a su vez será subdividido en 212 divisiones diferentes. El modelo es entrenado solo una vez con los datos correspondientes al conjunto de entrenamiento. Posteriormente, las predicciones realizadas por cada una de las divisiones en función del modelo entrenado son analizadas y se calculan las métricas de evaluación que posteriormente se promediarán para obtener el desempeño del modelo en relación con la validación cruzada.

Tabla 8: Métricas promedio derivadas de la validación cruzada del modelo Random Forest.

Estadísticas de la validación cruzada	
RMSE promedio	0,0129035

Fuente: elaboración propia a partir del resultado de las predicciones del modelo.

Puesto que rango de fluctuación de las cotizaciones al cierre desde enero de 2000 hasta mayo de 2023 ha ido desde €1,48 hasta €12,87. Un RMSE promedio de 0,0129 indica que, en promedio, las predicciones del modelo han diferido aproximadamente € 0,01 de

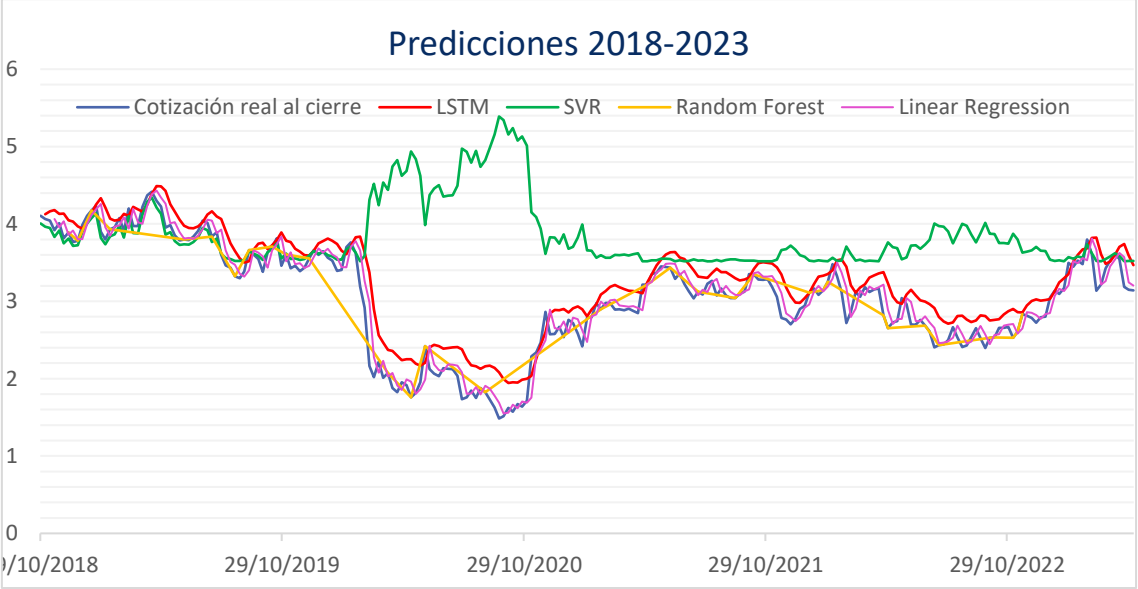
los valores de cotización semanal al cierre reales de SAN.MC, siendo el modelo más preciso en sus predicciones dentro de los cuatro analizados en el presente trabajo.

El modelo *random forest* ha demostrado tener un muy bajo grado de error tanto al analizar el conjunto de prueba correspondiente al 20% del *dataset*, como al implementar la validación cruzada. Asimismo, presenta una excelente capacidad de explicar las variaciones en los precios de la acción bajo estudio.

4.5. Comparación de Resultados

En el presente apartado se comparará la performance de los modelos analizados tanto mediante la representación gráfica de sus predicciones y contraste con los valores de cotización al cierre reales de SAN.MC, como mediante la comparación de las métricas de evaluación de cada modelo.

Figura 13: Comparativa de las predicciones sobre la cotización al cierre de SAN.MC

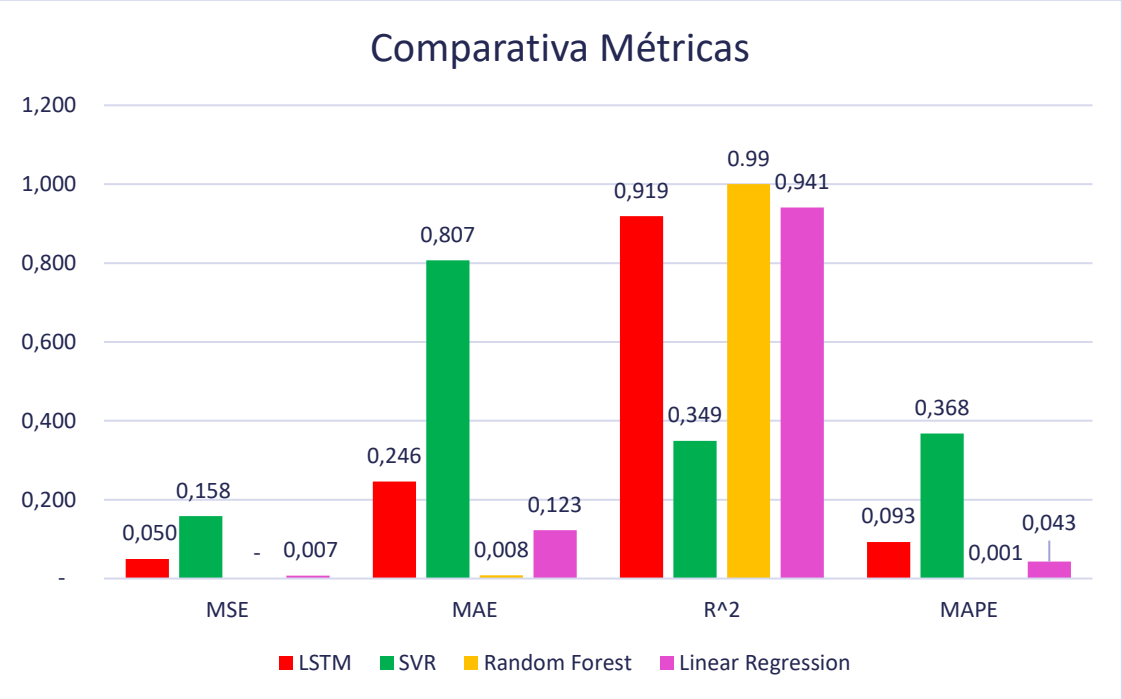


Fuente: elaboración propia a partir de las predicciones de cada modelo y las cotizaciones reales de SAN.MC disponibles en Yahoo! Finanzas.

En la gráfica anterior, podemos notar como los modelos LSTM, random forest8 y regresión lineal logran captar las tendencias de movimientos, acompañando a la cotización real de SAN.MC durante los últimos cinco años. En contraposición, el modelo de SVR comienza mostrando una buena adaptación a los datos, pero cuando cambia la tendencia durante 2020 y 2022, no logra capturar que se trata de un movimiento bajista y predice que el precio de cierre de las acciones irá al alza, lo que incrementa el error de predicción del modelo considerablemente.

Si bien a partir de la gráfica resulta evidente que el modelo SVR no será el candidato seleccionado a la hora de escoger el modelo de predicción más optimo, se dificulta identificar cual de los tres modelos restantes es el más adecuado por su sola visualización.

Figura 14: Comparativa métricas de los modelos de predicción.



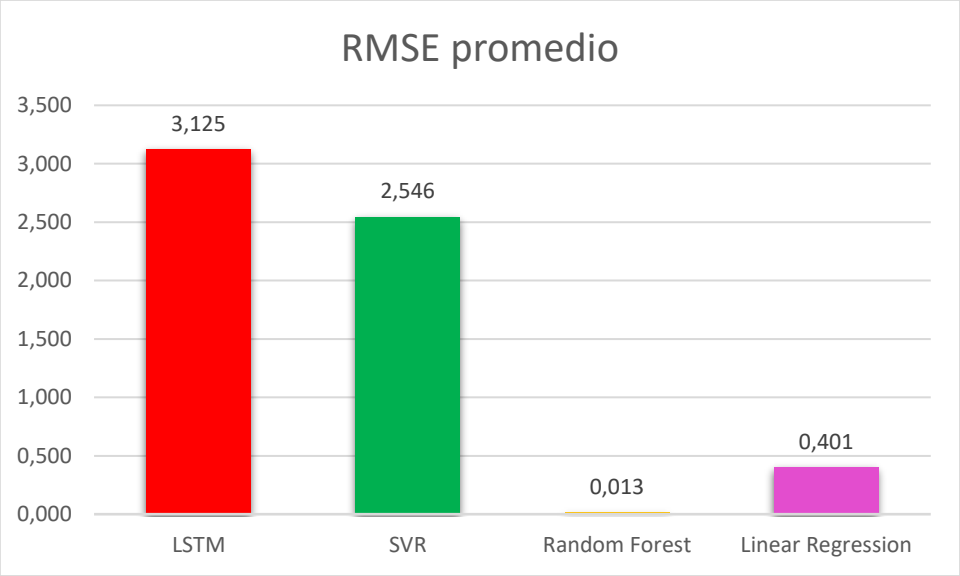
Fuente: elaboración propia a partir de las predicciones de cada modelo.

Al comparar las métricas obtenidas a partir del conjunto de prueba de cada modelo, destaca la performance de aquel que utiliza *random forest*, puesto que logra los valores mínimos tanto de MSE, como de MAE y MAPE. Es decir, es el modelo que incurre en un menor error de predicción promedio y también aquel con menores errores significativos. Asimismo, este modelo también logra el máximo valor de R^2, siendo el que mejor explica las variaciones en los precios al cierre a partir de la captación de patrones subyacentes presentes en la serie temporal bajo análisis. Sin embargo, es importante destacar que las predicciones del modelo de *random forest* no corresponden enteramente a semanas continuas, sino a fechas aleatorias comprendidas entre enero del 2000 y mayo 2023.

En un contexto de tenencia de acciones, donde se busca maximizar el beneficio al momento de vender la acción con que se cuenta en cartera, la habilidad de contar con

el posible precio futuro de cierre que tendrá un activo financiero es altamente valiosa para los inversionistas, es por ello por lo que el modelo de regresión lineal resulta una destacable, aunque sus métricas de desempeño no superen las del modelo *random forest*.

Figura 15: Comparativa de RMSE promedio derivado de la validación cruzada.



Fuente: elaboración propia a partir de los resultados de la validación cruzada de cada modelo.

En línea con lo anterior, los resultados derivados de la validación cruzada también indican que el modelo de *random forest* y el de regresión lineal cuentan con el mayor grado de precisión en las predicciones, ascendiendo el promedio de error a €0,01 y €0,40, respectivamente, en relación con las cotizaciones al cierre reales.

5. Creación de un modelo de ensamble

5.1 Descripción del modelo de ensamble

Tras haber evaluado las métricas de desempeño de cada modelo, se ha arribado a la conclusión de que el modelo de regresión lineal predice las cotizaciones al cierre de SAN.MC de forma semanal y continua, lo que resulta de gran valor para conocer la tendencia. Por otro lado, el modelo de *random forest*, logra el mejor desempeño con relación a precisión de las predicciones, sin embargo, estas son realizadas en fechas aleatorias no continuas.

Se propone la creación de un modelo de ensamble que permita potenciar las ventajas de cada uno de ellos, brindándole mayor estabilidad, precisión y capacidad de generalización al modelo.

Para ello se trabaja con ambos modelos predictivos individualmente, es decir, tanto el modelo de *random forest* como el de regresión lineal son entrenados sobre un mismo 80% de los datos correspondientes al conjunto de entrenamiento, donde habrán de captar los patrones subyacentes a los precios semanales al cierre de la acción SAN.MC.

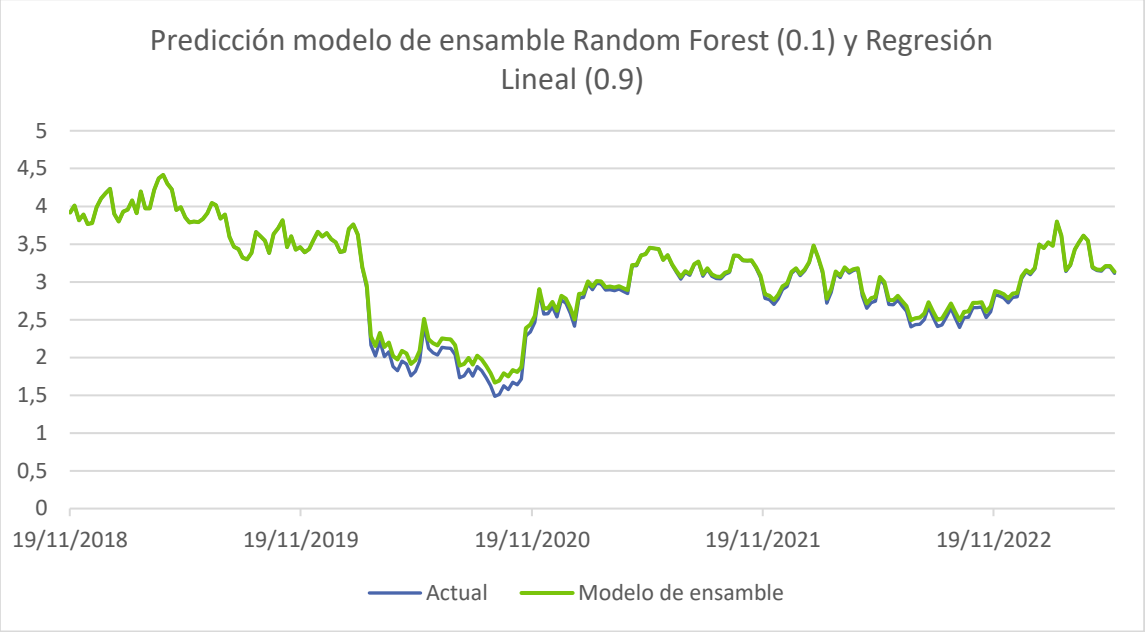
Posteriormente, cada modelo realiza sus predicciones sobre un mismo conjunto de entrenamiento, correspondiente al 20% restante del *dataset*. Es importante mencionar que el conjunto de prueba se compone por los datos de cotización al cierre semanales más recientes.

Tras contar con las predicciones individuales de cada modelo, se ensamblan los resultados aplicando una ponderación con peso equivalente a 0.1 para el modelo de *random forest* y 0.9 para el modelo de regresión lineal.

Si bien se esperaría que el modelo de *random forest* cuente con un mal desempeño a la hora de captar la tendencia bajista que se presenta en 2020 puesto que está siendo entrenado con datos previos a septiembre 2018, como ha sucedido en la primera versión del modelo donde no contó con la flexibilidad suficiente para capturar la pandemia del COVID-19. En este caso, la ponderación con el modelo de regresión lineal genera que no se incurra en el mismo error (véase figura 15), logrando a su vez mejorar el ajuste del modelo de regresión lineal individual gracias a la precisión que posee el modelo de *random forest* durante los periodos temporales cercanos a los valores de entrenamiento.

Esto representaría una gran ventaja en caso de optar por trabajar con aprendizaje en línea, es decir, un modelo que permita un entrenamiento continúo abasteciéndose automáticamente de nuevas muestras, como son los datos vinculados a cotizaciones al cierre disponibles en Yahoo! Finanzas, con el objeto de generar predicciones precisas para el corto plazo y capturar tendencias del mercado en el mediano plazo.

Figura 16: Predicción de las cotizaciones semanales al cierre de SAN.MC con un modelo de ensamble Random Forest (0,1) y Regresión Lineal (0.9)



Fuente: elaboración propia a partir de las predicciones del modelo.

5.2 Evaluación del modelo de ensamble

El modelo de ensamble expuesto logra entregar predicciones de la cotización semanal al cierre de SAN.MC de mera continua, sin omitir semana alguna dentro del periodo comprendido entre noviembre 2018 y mayo 2023, resolviendo de esta forma la carencia que presenta el modelo de *random forest* (segunda versión).

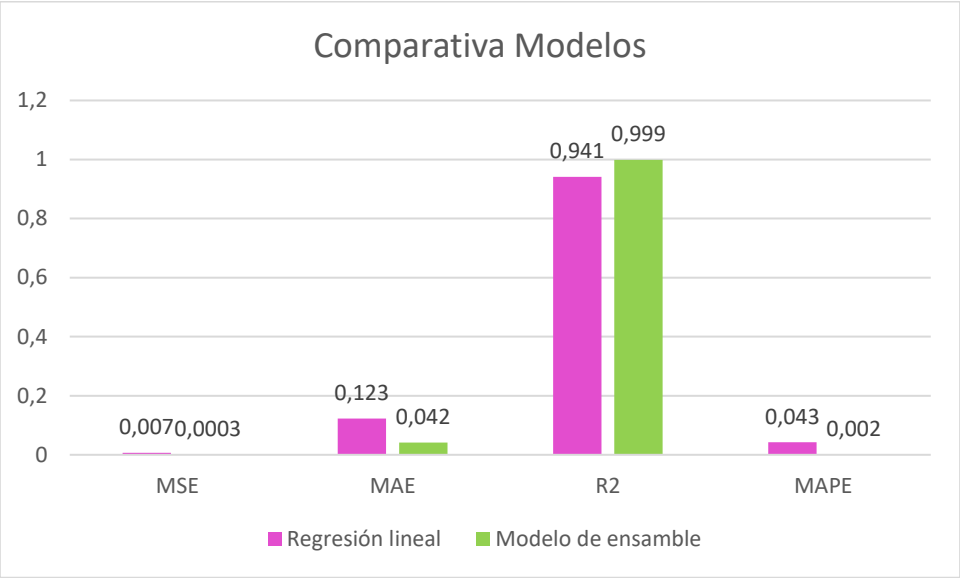
Asimismo, se analizan a continuación las métricas de rendimiento del presente modelo (véase tabla 9).

Tabla 9: Métricas de evaluación del modelo de ensamble

Estadísticas del modelo	
MSE	0,0003557
MAE	0,0424266
R^2	0,9992327
MAPE	0,0019253

Fuente: elaboración propia a partir de las predicciones del modelo.

Figura 17: Comparativa métricas de desempeño del modelo de regresión lineal y el modelo de ensamble.



Fuente: elaboración propia a partir de las predicciones obtenidas de los modelos.

A efectos de analizar las métricas, debe considerarse que las cotizaciones al cierre semanales de SAN.MC durante el periodo comprendido en los datos del conjunto de prueba, que abarca desde noviembre 2018 hasta mayo 2023, oscilaba entre los €1,48 y €4,41.

El modelo de ensamble presenta un error cuadrático medio (MSE) y un error absoluto medio (MAE) significativamente inferior al modelo de regresión lineal (véase figura 16), gracias a la acción de la ponderación del modelo de *random forest*. De esta forma, se logra mejorar la precisión de las predicciones del modelo base, que en este caso es el de regresión lineal, siendo el máximo error en el que incurre el modelo de ensamble equivalente a €0,18, mientras que el modelo de regresión lineal alcanzó a incurrir en un error máximo de €0,82.

Asimismo, en el modelo de ensamble también se potencia la capacidad de explicación de la variación presente en las cotizaciones al cierre, dado que es capaz de explicar un 99,9% de esta.

En línea con lo anterior, el error porcentual absoluto medio (MAPE) se reduce en un 95,34% en comparación con la misma métrica correspondiente al modelo de regresión lineal.

También ha utilizado la validación cruzada hacia adelante para evaluar el modelo de ensamble. Para ello se establecen en este caso 5 divisiones, lo que implica la existencia de cinco conjuntos de entrenamiento y cinco conjuntos de prueba, que sigue cada uno a un determinado conjunto de entrenamiento contiguo en el tiempo.

El algoritmo entrena con el primer conjunto de entrenamiento y efectúa posteriormente sus predicciones sobre el primer conjunto de prueba cuyos datos respetan la secuencia temporal de la serie, es decir, son continuos a los datos de entrenamiento del primer conjunto. Posteriormente el algoritmo continúa iterando entre las demás divisiones, lo que implica que seleccionará un nuevo conjunto de prueba y entrenará con los conjuntos anteriores a este. Este método permite evaluar el desempeño del modelo en diferentes partes de la serie temporal, sin centrarse meramente en el periodo de noviembre 2018 a mayo 2023.

Las predicciones realizadas por cada modelo son luego ponderadas por su peso en el modelo de ensamble, es decir, las predicciones correspondientes a *random forest* serán ponderadas por un 0,1 y las correspondientes a la regresión lineal por un 0,9, obteniendo las predicciones finales del modelo de ensamble, sobre las cuales se calculará la métrica promedio para evaluar su desempeño.

Tabla 10: Métrica promedio derivadas de la validación cruzada del modelo de ensamble

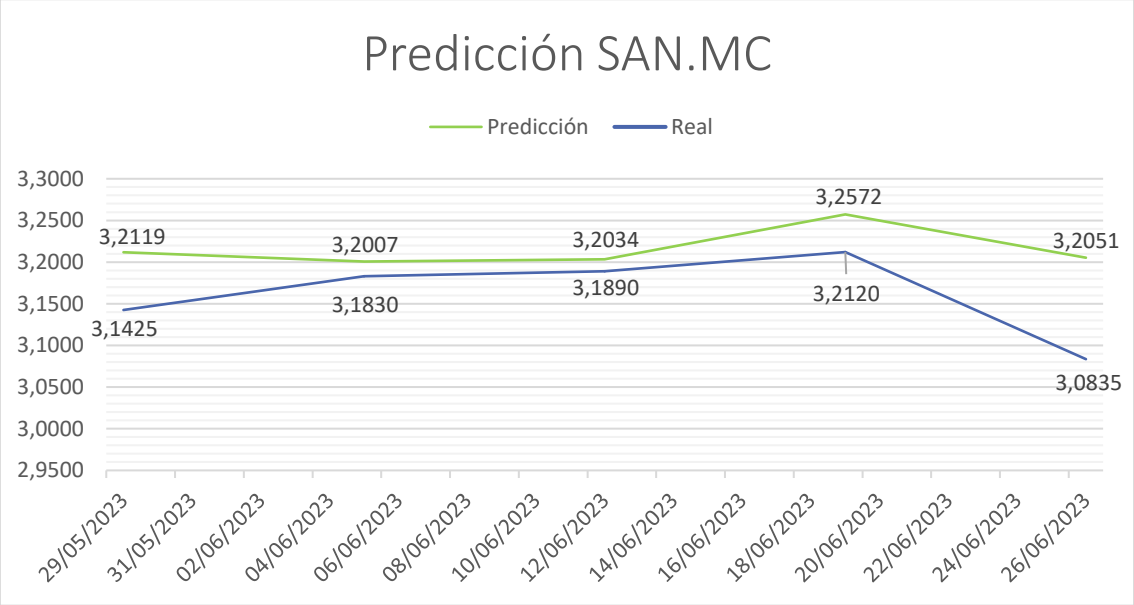
Estadísticas de la validación cruzada	
RMSE promedio	0,0639531

Fuente: elaboración propia a partir de las predicciones del modelo.

Dado que rango de fluctuación de las cotizaciones al cierre desde enero de 2000 hasta mayo de 2023 ha ido desde €1,48 hasta €12,87. Un RMSE promedio de 0,063 indica que, en promedio, las predicciones del modelo han diferido aproximadamente €0,063 de los valores de cotización semanal al cierre reales de SAN.MC, mejorando considerablemente respecto del modelo de regresión lineal que contaba con un RMSE promedio de €0,4, es decir un 84% menor.

Considerando lo anterior, se ha ejecutado una predicción del precio de SAN.MC a 5 semanas partiendo del 29 de mayo de 2023 y utilizando para ello el modelo de ensamble entrenado que comentamos en este apartado.

Figura 18: Aplicación del modelo de ensamble a la predicción del precio al cierre de SAN.MC por 5 semanas continuas



Fuente: elaboración propia a partir de las predicciones efectuadas con el modelo de ensamble y las cotizaciones semanales al cierre obtenidas de Yahoo! finanzas.

En la gráfica puede apreciarse como el modelo capta tanto las tendencias alcistas como bajistas del periodo. Siendo las predicciones de las primeras tres semanas más certeras y perdiendo precisión cuando se aleja en el tiempo de los datos aplicados para el entrenamiento.

Tabla 11: Métricas derivadas de las predicciones del precio al cierre de SAN.MC durante 5 semanas, partiendo del 29 de mayo de 2023.

Estadísticas del modelo	
MSE	0,0020451
MAE	0,0536573
MAPE	0,0166888

Fuente: elaboración propia a partir de las predicciones del modelo de ensamble.

Las métricas obtenidas a partir de los resultados de la predicción por 5 semanas no varían significativamente respecto de aquellas que derivaron de la evaluación del modelo. Tanto el error medio cuadrático (MSE) como el error medio absoluto (MAE)

derivados de esta evaluación indican un buen desempeño, puesto que las predicciones distan en promedio, en 0,0536 en de los valores reales al cierre, en términos de valor absoluto, ascendiendo la mayor diferencia presente a €0,12 en la quinta semana. Por lo que se confirma que la puesta en práctica de este daría lugar a resultados óptimos de predicción siempre que el modelo continúe actualizándose con los datos reales disponibles en la fuente seleccionada semana a semana, consiguiendo reducir de tal forma el error incremental en que se incurre luego de la tercera semana.

6. Conclusión

El objetivo fundamental de este trabajo ha sido analizar cuatro modelos de predicción de activos financieros, para los que se selecciona el modelo de random forest, regresión lineal, SVR y redes neuronales recurrentes (LSTM). Para ello, se tomó como caso de estudio el precio semanal al cierre de la acción SAN.MC, obteniendo las cotizaciones históricas desde enero del 2000 hasta mayo de 2023 de la fuente Yahoo! finanzas.

Se ha explicado el proceso relativo a como cada modelo se nutre de datos para reconocer los patrones subyacentes en los precios históricos al cierre para luego generalizarlos y aplicar tales relaciones a la hora de predecir el precio futuro de la acción.

Una vez comprendido el funcionamiento de cada modelo, se procedió a analizar el desempeño que presentaban en el conjunto de prueba, que fue establecido para todos ellos como el 20% del dataset. A partir de tal evaluación, se determinó que el modelo de regresión lineal contaba con gran capacidad de capturar tendencias y eventos inhabituales del mercado, al adaptarse fácilmente a la caída del precio de SAN.MC sufrida durante la pandemia del COVID-19. Sin embargo, era el modelo de random forest el que tenía mayor capacidad de precisión en sus predicciones, aunque presentaba la dificultad de que tales predicciones no se efectuaban de manera continua semana a semana, sino que se trataba de fechas aleatorias que representaban el 20% del conjunto de prueba dentro del total del dataset.

Considerando la relevancia atribuible a que tales predicciones se realicen de manera semanal continua, se optó por desarrollar un modelo que combine las fortalezas de la regresión lineal con la precisión de random forest. Para ello se atribuyeron pesos ponderados a cada modelo, que ascendían a 0,9 y 0,1 , respectivamente, a efectos de obtener un modelo de ensamble que genere predicciones continuas semanales de los precios al cierre de SAN.MC, pero con una mayor precisión que la brindada por el modelo de regresión lineal.

Tras evaluar el modelo de ensamble, se llegó a la conclusión de que las métricas de desempeño habían evolucionado favorablemente, llegando a reducir el MAPE en un 95,34% y logrando un R2 del 99,9%, lo que indica que el modelo cuenta con una gran capacidad para explicar las variaciones en las cotizaciones trabajadas.

Asimismo, se probó la efectividad de este a la hora de predecir el precio al cierre de la acción seleccionada durante 5 semanas, notando una reducción de la efectividad a partir de la tercera semana, lo que ratifica la importancia de que el modelo trabaje semana a semana con las ultimas cotizaciones al cierre disponibles en la fuente seleccionada.

En definitiva, el presente modelo podría brindar ser de guía al momento de operar a corto plazo puesto que ayuda a identificar la tendencia del mercado. Sin embargo, es importante considerar que, por limitaciones temporales y materiales, el mismo no considera aspectos adicionales que influirían en los precios de las acciones, como ser el volumen de operaciones o bien, el sentimiento de mercado. Resultaría de gran interés explorar en un futuro la influencia de dichos aspectos con el objeto de aportar resultados más comprehensivos respecto de la situación del mercado.

Bibliografía

Breiman, L. (2001). Random Forest. *Kluwer Academic Publishers*.

Debasish, K. (2022). An Overview on Long Short Term Memory (LSTM). *Data Science Blogathon*.

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2da Edición.* O'Reilly Media, Inc.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* MIT Press.

Hochreiter, S., & Schmidhuber, J. (1997). Short long-term memory. *Neural Computation*.

Isasi Viñuela, P., & Galván León, I. (2004). *Redes neuronales artificiales, un enfoque práctico.* Pearson.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). *An introduction to statistical learning with applications in R.* Springer.

Jang, S. K. (2020). Machine learning identifies inhibitors of the SARS-CoV-2 main protease. Nature, 586(7827), . *Nature*, 113-119.

Kim, S., & Kim, H. (2016). A new metric of absolute percentage error for intermittent demand forecasts. *International journal of forecasting*.

Meshram, I., & Kulal, P. (2022). A comparative study of SVM, LSTM and LR algorithms for stock market prediction using OHLC data. *Research gate*.

Miranda Henrique, B., Amorim Sobreiro, V., & Kiruma, H. (2018). Stock price prediction using support vector regression on daily and. *Science Direct* .

Mood, A., Graybill, F., & Boes, D. (1974). *Introduction to the theory of statistics.* McGraw-Hill International book company.

Rencher, A., & Schaalje, G. (2007). *Linear models in statistics.* Wiley-interscience.

Rob J., H., & Athanasopoulos, G. (2021). *Forecasting: principles and practice.* Texts.

Saini, A. (2021). An Introduction to Random Forest Algorithm for beginners. *Data Science Blogathon*.

Samuel, A. (1959). *Some studies in machine learning using the game of checkers.* . IBM Journal of Research and Development. 210-229.

Sijian, T. (2023). Predicting BMW Stock Price Based on Linear Regression,. *College of Literature, Science, and Arts, University of Michigan, Ann Arbor, Michigan, the United*.

Staudemeyer, R., & Rothstein Morris, E. (2019). Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks. *Reaserch Gate*, 6.

Yilin, M., Ruizhu, H., & Xiaoling, F. (2019). Stock prediction based on random forest and LSTM neural network. *Research Gate*.