ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

APLICACIÓN DE SUPPORT VECTOR MACHINE Y LONG SHORT TERM MEMORY PARA EL PRONÓSTICO DEL PRECIO DE LAS ACCIONES DEL BANCO MACRO DE ARGENTINA Y BANCOLOMBIA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL GRADO DE MAGÍSTER EN SISTEMAS DE INFORMACIÓN

JORGE XAVIER ROMERO CARRIÓN

jorge.romero03@epn.edu.ec

DIRECTORA: DIANA MARTÍNEZ MOSQUERA, PhD

diana.martinez@epn.edu.ec

CODIRECTOR: IVÁN CARRERA IZURIETA, PhD

ivan.carrera@epn.edu.ec

QUITO, DICIEMBRE 2023

APROBACIÓN DEL DIRECTOR

Como directora del trabajo de titulación: APLICACIÓN DE SUPPORT VECTOR MACHINE Y LONG SHORT TERM MEMORY PARA EL PRONÓSTICO DEL PRECIO DE LAS ACCIONES DEL BANCO MACRO DE ARGENTINA Y BANCOLOMBIA desarrollado por Jorge Xavier Romero Carrión, estudiante de la Maestría en Sistemas de Información Mención en Inteligencia de Negocios y Analítica de Datos Masivos, habiendo supervisado la realización de este trabajo y realizado las correcciones correspondientes, doy por aprobada la redacción final del documento escrito para que prosiga con los trámites correspondientes a la sustentación de la Defensa Oral.

Dra. Diana Martínez

Directora

APROBACIÓN DEL CODIRECTOR

Como codirector del trabajo de titulación: APLICACIÓN DE SUPPORT VECTOR MACHINE Y LONG SHORT TERM MEMORY PARA EL PRONÓSTICO DEL PRECIO DE LAS ACCIONES DEL BANCO MACRO DE ARGENTINA Y BANCOLOMBIA desarrollado por Jorge Xavier Romero Carrión, estudiante de la Maestría en Sistemas de Información Mención en Inteligencia de Negocios y Analítica de Datos Masivos, habiendo supervisado la realización de este trabajo y realizado las correcciones correspondientes, doy por aprobada la redacción final del documento escrito para que prosiga con los trámites correspondientes a la sustentación de la Defensa Oral.

Firmado electrónicamente por:
IVAN MARCELO
CARRERA IZURIETA

_____

Dr. Iván Carrera

Co-Director

DECLARACIÓN DE AUTORIA

Yo, Jorge Xavier Romero Carrión, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

La Escuela Politécnica Nacional puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la Normativa Institucional vigente.

Firmado electrónicamente por:
JORGE XAVIER ROMERO CARRION

_____

Jorge Xavier Romero Carrión

0104793211

DEDICATORIA

A Dios Padre,

A mis padres,

A mis hermanos,

A mis hermosas sobrinas.

AGRADECIMIENTOS

A Dios todo poderoso por darme la oportunidad de terminar con éxito este ciclo educativo. A las personas que estuvieron cerca de mi apoyándome en distintas formas. A la Escuela Politécnica Nacional por abrir me las puertas para poder empezar y culminar con éxito este título de cuarto nivel.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS.

RESUMEN

El objetivo principal del presente trabajo de desarrollo ha sido obtener la máxima precisión posible en los algoritmos de Inteligencia Artificial (AI, por sus siglas en inglés) de última generación, como el Aprendizaje Automático (ML) y los Modelos de Aprendizaje Profundo (DL), más específicamente el *Long Short-Term Memory* (LSTM) y *Support Vector Machine* (SVM), en el pronóstico de series temporales.

Durante la última década, la inmersión en modelos de AI ha abierto nuevos caminos en diferentes campos de estudio. Dentro de este ámbito, el pronóstico de series temporales financieras ha cobrado gran interés debido en gran parte a los nuevos enfoques que permiten alcanzar los modelos de ML y DL [1]. Estos nuevos enfoques comprenden el mejorar la capacidad predictiva en comparación con modelo estadísticos como ARIMA, SARIMA, GARCH, etc., la transferencia de conocimiento, y la aplicación de nuevas técnicas para el pronóstico de acciones de empresas como es el análisis de sentimientos de los inversionistas [1].

Una de las principales razones de la gran popularidad de los modelos de ML y DL en la última década se debe a que poseen una gran capacidad predictiva, superior a los alcanzados por lo modelos estadísticos nombrados en el párrafo anterior. Otra gran ventaja, es que poseen una gran cantidad de hiperparámetros[1], los mismos que son adaptables a las características específicas del problema que se desea investigar y resolver.

La novedad de la presente investigación radica en el hecho de aplicar el LSTM y SVM para pronosticar el precio de las acciones de dos instituciones financieras latinoamericanas que cotizan en la Bolsa de Nueva York, con el objetivo de que este tipo de investigación sirva como motivación para implementar dichos algoritmos en diferentes aspectos del mercado financiero y otras áreas.

Como forma de evaluar el rendimiento de los dos algoritmos de ML, se implementarán 3 métricas de error como el error cuadrático medio (RMSE), el error porcentual absoluto medio (MAPE) y el error absoluto medio (MAE).

Palabras Clave: Inteligencia Artificial (AI), Aprendizaje Automático (ML), Aprendizaje Profundo (DL), *Long Short-Term Memory* (LSTM), *Support Vector Machine* (SVM), precio de una acción, RMSE, MAPE, MAE.

[1] Ver Anexo B donde se explica lo que es un hiperparámetro

ABSTRACT

The main purpose of the current development project has been to acquire the ultimate accuracy of state-of-the-art Artificial Intelligence (AI) algorithms like Machine Learning (ML) and Deep Learning Models (DL), more specifically Long Short-Term Memory (LSTM) and Support Vector Machine (SVM), in the forecasting of time series.

During the last decade, immersion in AI models has opened new paths in different fields of study. Within this area, financial time series forecasting has gained great interest due in large part to new approaches that allow ML and DL models to achieve [1]. These new approaches range from improving the predictive capacity, in comparison with statistical models such as ARIMA, SARIMA, GARCH, etc., allowing the transfer of knowledge to new data, to the application of new techniques for forecasting stock prices such as investor's sentiment analysis.

One of the main reasons for the great popularity of ML and DL models in the last decade is because they hold an outstanding forecasting capability, superior to that achieved by the statistical models mentioned in the previous paragraph. Another great advantage is that they have a large number of hyperparameters, which can be tailored to meet the specific characteristics of the problem that needs to be investigated and tackled.

The novelty of the current investigation lies at the fact that LSTM and SVM will be applied to forecast the stock price of two Latin America Financial Institutions listed in the New York Stock Exchange, hopping this kind of investigation serves as an inspiring work to implement such algorithms in different issues of the financial landscape and others as well.

As a way to evaluate the performance of the two AI algorithms, 3 error metrics will be implemented such as Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) and Mean Absolute Error (MAE).

Keywords: Artificial Intelligence, Machine Learning, Deep Learning, Long Short-Term Memory, Support Vector Machine, Stock Price, RMSE, MAPE, MAE

1. Introducción.

1.1. Planteamiento del Problema.

A lo largo de las décadas, el pronóstico de series de tiempo siempre ha sido una de las áreas de mayor interés en las ciencias aplicadas [1]. Las variables que forman parte del mercado financiero[2] y su comportamiento a través del tiempo han jugado un papel muy importante en el creciente interés en el pronóstico de estas variables, pues el correcto y acertado pronóstico de una serie de tiempo marca la diferencia entre obtener una ganancia o pérdida económica [2].

A través del tiempo, los modelos clásicos o estadísticos como ARIMA han sido ampliamente utilizados durante varias décadas para el pronóstico de series de tiempo financieras[3], obteniendo óptimos resultados [3]. Sin embargo, estudios como Omer Nerat Sezer, M. Ugur Gudelek , Ahmet Murat Ozbayoglu [1] y Sima Siami-Namino, Neda Tavakoli, Akbar Siami Namin [4] han demostrado que modelos de *Machine Learning* (ML) y más concretamente modelos de *Deep Learning* (DL) presentan un mejor desempeño que el mencionado ARIMA según métricas de error como RMSE y MAE las cuales presentar un valor más pequeño. A más de este mejor desempeño en el pronóstico de series de tiempo, hay que tener en cuenta que los modelos de DL cuentan con una característica especial llamada *Transfer Learning* la misma que permite a investigadores de diversos campos hacer uso de modelos pre-entrenados, de esta forma se ahorran una considerable cantidad de tiempo en el desarrollo y entrenamiento de sus modelos [5].

Ante este escenario, el sector financiero se constituye en una muy interesante área para el pronóstico de series de tiempo haciendo uso de técnicas de ML y DL debido, entre otros aspectos, a las vastas implementaciones y el substancial impacto que genera la aplicación de estos modelos [1]. La inserción de modelos de ML y DL en el estudio y pronóstico de series temporales trae consigo un conjunto de nuevas oportunidades poco o no exploradas en el pasado, al menos en el contexto latinoamericano, como también un conjunto importante de retos. Por conjunto de oportunidades se entiende el hecho de poder obtener mejores resultados a la hora de pronosticar las series temporales y de aplicar nuevos enfoques como es el del análisis de sentimientos [6]. En lo referente a los retos, entre los más importantes se pueden destacar: el costo computacional de entrenar dichos algoritmos, el tiempo requerido, los distintos hiperparámetros y variables que entran en juego a la hora de seleccionar el mejor modelo con base en su poder predictivo [3].

A los retos anteriormente citados se debe adicionar aquel inherente al mercado de valores y al pronóstico del precio de las acciones de una determinada empresa. Este reto hace referencia al hecho de que el pronóstico del precio de una acción que cotiza en un mercado de valores requiere identificar los patrones más importantes en la trayectoria que presenta el precio de una acción, lo cual en muchos casos resulta complejo de conseguir debido a la falta de información y/o medios tecnológicos que permitan analizar los datos disponibles [7].

[2] Ver Capitulo 1.5 acerca de la respectiva referencia teórica.
[3] Ver Capitulo 1.5.3.4 acerca de la respectiva referencia teórica.

Ante este escenario, la importancia del actual proyecto de desarrollo radica en implementar un modelo de inteligencia artificial para el pronóstico de series de tiempo, precio de las acciones, de dos instituciones financieras latinoamericanas, estas son: Banco Macro de Argentina y BanColombia. Para la respectiva obtención de los datos que servirán de insumo para los algoritmos de ML y DL, se tomará como fuente de datos el portal web de Yahoo Finance[4] y como programa para realizar todo el proceso de visualización de datos, preprocesamiento, entrenamiento y evaluación se hará uso de Python y *Google Colab*, el cual es un ambiente de programación integrado que corre de forma nativa en Python en la nube de Google.

Los algoritmos de inteligencia artificial a implementar son *Support Vector Machine* (SVM) el cual pertenece a los modelos de ML, y el segundo algoritmo es *Long Short Term Memory* (LSTM) el cual pertenece a los modelos de DL y más concretamente a las Redes Neuronales Recurrentes[5].

La importancia en la aplicación de SVM en el pronóstico de series temporales radica en que su variante *Support Vector Regressor* ha sido precisamente construido para afrontar problemas de regresión, entre ellos, la predicción de diferente tipo de series temporales. A más de este hecho, se debe tomar en cuenta que SVM posee una gran cantidad de parámetros, también conocidos como hiperparámetros, que permiten al modelo adaptarse de mejor manera a la base de datos en cuestión y por ende mejorar el poder de predicción. Al respecto, Kyoung-jae Kim [2] manifiesta que encontrar el valor adecuado del parámetro C y el kernel resulta crucial a la hora de obtener óptimos resultados.

LSTM pertenece a la familia de las redes neuronales recurrentes, las cuales presentan un conjunto de propiedades idóneas para trabajar con series secuenciales, entre ellas las series de tiempo. Entre las principales razones por las que LSTM permite identificar patrones en datos de tipo secuencial está el hecho de que puede tomar simultáneamente varios valores de entrada o *input* y producir una secuencia de salida u *outputs*. Este tipo de metodología se conoce como *sequence-to-secuence* en la literatura referente a DL y es lo que permite a la red neuronal la predicción del precio de las acciones al tomar como *input* el precio pasado de una cierta acción en un determinado periodo de tiempo y obtener como resultado el pronóstico de esta para cierto día[6] [8].

Esta característica de LSTM hace posible que el *set* de entrenamiento converja más rápido y que se detecten dependencias de largo plazo en un cierto conjunto de datos [9]. Armin Lawi [10] y Loannis E. Livieris [11] recalcan la importancia del LSTM en el pronóstico de series temporales gracias a la identificación de dependencias de corto y largo plazo y a la flexibilidad que presenta el algoritmo al permitir explorar las distintas arquitecturas en sus capas internas o hidden layers[7].

1.2. Pregunta de Investigación:

¿Qué algoritmo entre el SVM y el LSTM presenta los mejores resultados en términos de métricas como: MAE, MAPE, RMSE, y qué configuración de cada uno permite obtener estos resultados?

[4] https://finance.yahoo.com/

[5] Ver la sección de Marco Teórico acerca de la respectiva definición.

[6] Mas acerca de cómo trabaja el LSTM es la sección de Marco Teórico.

[7] Los principales conceptos y características del SVM y el *LSTM* serán descritos en la sección de Marco Teórico

1.3. Objetivo General:

Aplicar SVM y LSTM para el pronóstico del precio de las acciones a corto plazo de dos bancos latinoamericanos.

1.4. Objetivos Específicos:

- Revisar la literatura concerniente al ML y DL aplicados a las series temporales financieras.
- Implementar el SVM y el LSTM a las series financieras temporales de las empresas seleccionadas en una franja de tiempo desde marzo de 2006 hasta marzo del 2023 obtenidas del portal web Yahoo Finance.
- Evaluar, a través de MAE, RMSE, este último como principal métrica de error, y MAPE, cuál es la metodología que presenta mejores resultados para pronosticar acciones en el corto plazo para las empresas seleccionadas.
- Analizar y evaluar los resultados obtenidos en los respectivos gráficos estadísticos para determinar la calidad de pronóstico de cada modelo, así como los parámetros de estos.

1.5. Marco Teórico

1.5.1. *Support Vector Machine* (SVM).

SVM es un poderoso y versátil modelo de ML, considerado uno de los modelos más populares en el campo del ML, capaz de resolver problemas de clasificación lineal y no lineal, regresión e inclusive puede ser implementado para la detección de valores anómalos [8].

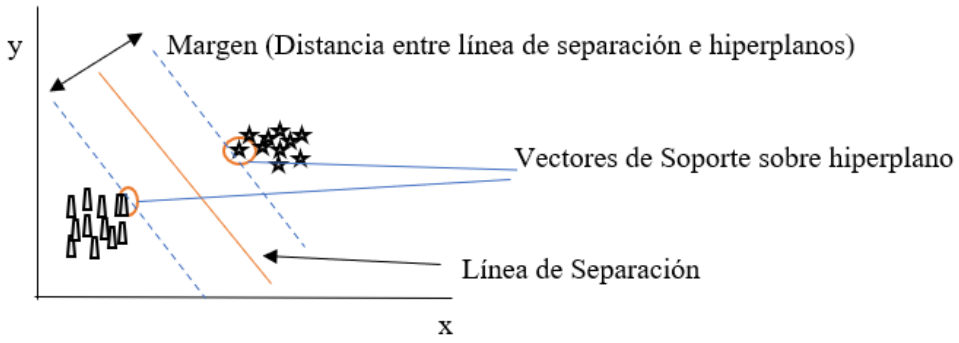


Figura 1 1. Clasificación Lineal del SVM y vectores de soporte.

Elaboración: Autor.

En la Figura 1.1 es posible observar que este algoritmo permite clasificar una cierta observación en una de dos clases para este ejemplo. El objetivo es maximizar el margen, el cual se lo puede

definir como la distancia que existe entre la línea de separación y las observaciones de entrenamiento más próximas a cada hiperplano, también llamadas vectores de soporte[8].

La variante de SVM a utilizar en el presente trabajo de desarrollo es el *Support Vector Regressor* (SVR), usado para tareas de regresión. El objetivo es el encontrar una función que mejor aproxime la relación entre las variables de entrada y la variable objetivo, mientras minimiza el error de predicción [5].

1.5.1.1. Hiperparámetros de SVM.

1.5.1.2. Kernel.

La idea detrás del hiperparámetro kernel en SVM radica en crear una combinación no lineal de las variables originales para proyectarlas en un espacio de altas dimensiones mediante una función de mapeo ϕ en donde los datos pasan a ser linealmente separables [12].

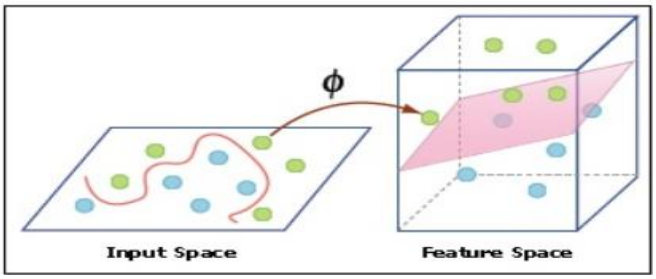


Figura 1.2. Representación gráfica de la aplicación de un kernel no lineal.

Tomado de: Aurelien Geron. Hands-on Machne Learning with Scikit-Learn, Keras & Tensor Flow. 2022 [8].

En Input Space de la Figura 1.2, se divisa que una línea recta horizontal o vertical no puede clasificar correctamente cada punto en una de las dos categorías, esto en un espacio de dos dimensiones. Al aplicar el kernel con su respectiva función de mapeo y proyectar cada punto a un espacio de 3 dimensiones para este ejemplo, cada observación es correctamente clasificada en una de las dos categorías, Feature Space de la Figura 1.2.

1.5.1.3. Epsilon (ε).

Este hiperparámetro controla el ancho del margen, es decir, trata de concentrar la mayor cantidad de observaciones dentro del mismo. Al tiempo que penaliza aquellas observaciones por fuera de dicho margen [8].

[8] La razón principal por la que se busca maximizar la distancia entre la línea de separación y los vectores de soporte obedece a que este tipo de diseño del algoritmo nos permite obtener un menor error de generalización y, por lo tanto, menor sobreajuste [8].

1.5.1.4. Hiperparámetro C.

Este hiperparámetro C también es conocido como regularizador, pues es el que controla el castigo por el sobreajuste y subajuste. Bajos niveles de C corresponden a una menor cantidad de violaciones en el margen, y a una mayor generalización[9] del modelo, y viceversa. Por lo tanto, es posible utilizar este hiperparámetro para controlar el ancho del margen [9]. En problemas de regresión, este hiperparámetro trabaja directamente con ε, pues al incrementarse C la tolerancia por observaciones afuera del margen ε también se incrementa.

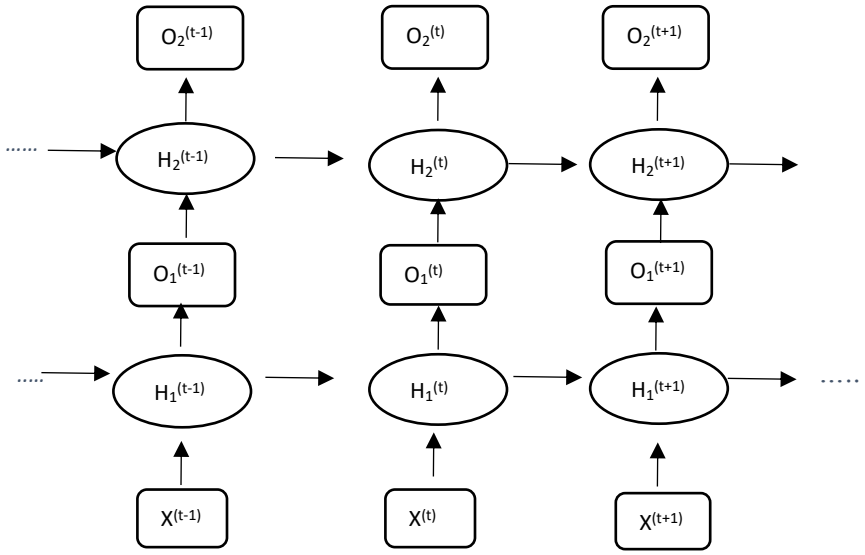
1.5.1.5. Gamma.

Tal como sucede con el hiperparametro C, Gamma está inversamente relacionado con su distancia. De tal forma que mientras más grande sea el valor de este hiperparámetro, la línea de separación es más ajustada y propensa a un sobreajuste, y viceversa [5].

Por lo tanto, a un mayor nivel de gamma, el rango de influencia de cada observación es más pequeño, y la línea de separación termina siendo más irregular. En contraste, a valores más pequeños de gamma, las observaciones tienen un mayor grado de influencia y la línea de separación termina siendo más flexible y suave [8].

1.5.2. *Long Short-Term Memory* (LSTM).

Una *Recurring Neural Network* (RNN) es una arquitectura de DL especialmente adaptada para procesar secuencias de datos. Sin embargo, el mayor inconveniente con este tipo de red neuronal es que sufre de un problema de corta memoria denominado desvanecimiento del gradiente[10]. Esto último significa que la información de uno de los primeros elementos insertados dentro de la red no tendrá efecto alguno en fases posteriores en la secuencia de la red [9].



[9] La generalización en ML y DL hace referencia a la capacidad de un modelo para adaptarse adecuadamente a datos nuevos que no se encuentran en su conjunto de entrenamiento original. Sebastian Raschka, Machine Learning with PyTorch and Scikit-Learn, Packt, 2022 [9].

[10] El gradiente es simplemente la función que le dice a la red como cambiar los pesos de esta. Sebastian Raschka, Machine Learning with PyTorch and Scikit-Learn, Packt, 2022 [9].

Figura 1.3. Ilustración de una RNN multicapa

Elaboración: Autor.

En la Figura 1.3 se visualiza que cada unidad escondida, representada por la letra h, recibe dos diferentes conjuntos de entrada: la función de la capa de entrada y la función de activación de la misma capa oculta del tiempo previo, esto es t-1. Al respecto, Sebastian Raschka, Yuxi Liu, y Vahid Mirjalili [9] explican brevemente cómo funciona este tipo de red neuronal:

- Capa = 1. En un primer punto, t = 0, las unidades ocultas, representadas como $\mathbf{h_1^{(t)}}$ están inicializadas con valor de cero o valores pequeños. Luego en un siguiente punto en el tiempo, t >0, las unidades ocultas reciben sus valores de entrada de la observación, $\mathbf{x^{(t)}}$, y de los valores ocultos en la misma capa, pero en un punto temporal previo, $\mathbf{h_1^{(t-1)}}$.
- Capa =2. La segunda capa oculta, $\mathbf{h_2^{(t)}}$, recibe sus valores de entrada de las salidas de la capa anterior en el presente punto temporal, $\mathbf{o_1^{(t)}}$, y de sus propios valores ocultos del punto temporal previo. $\mathbf{h_2^{(t-1)}}$**.**

Debido a que en cada capa recurrente debe recibir una secuencia como entrada, todas las capas recurrentes con excepción de la última debe retornar una secuencia como valor de salida, aclarando que el comportamiento de la última capa recurrente depende del tipo de problema [9].

En tanto que el LSTM añade una celda a la arquitectura del RNN y de esta forma logra contrarrestar el problema del desvanecimiento del gradiente en donde información pasada deja de tener impacto en el aprendizaje de la red neuronal.

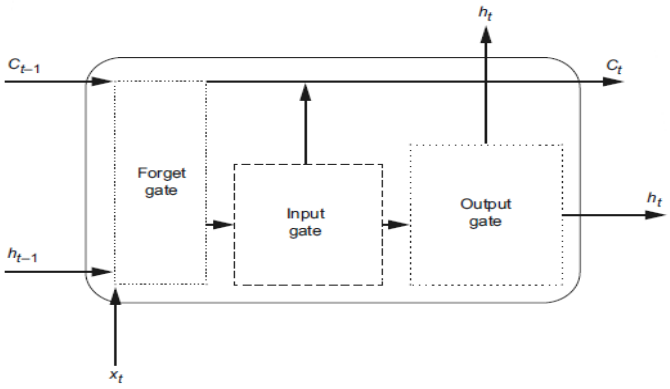


Figura 1.4. Arquitectura interna de una LSTM.

Tomado de: Marco Peixeiro. Time Series Forecasting in Python, Manning, 2022 [13].

En la Figura 1.4 se puede visualizar la existencia de un nuevo estado oculto denominado C es lo que hace posible a la red neuronal el poder almacenar información pasada por un largo periodo de tiempo, resolviendo de esta forma el problema del desvanecimiento del gradiente. En LSTM se tienen dos elementos que pasan por cada una de las etapas de la red LSTM, asegurando de esta forma que la información del pasado es usada como entrada para cada etapa de la secuencia en proceso [8].

Otro punto para considerar en la Figura 1.4 es la existencia de 3 puertas (*gates*): *Forget gate, Input gate y Output gate*.

- *Forget gate*. Es la Puerta que permite a la celda de memoria reiniciar su estado sin crecer indefinidamente. Es la encargada de decidir qué información es permitida para avanzar a través de la red y que información no tomar en cuenta [9].
- *Input gate*. Es la Puerta encargada de actualizar el estado de la celda, controlando de esta forma que partes de la celda deben ser agregadas al estado de largo plazo [9].
- *Output gate*. Esta Puerta decide como actualizar los valores de las unidades escondidas, de esta forma controla que partes del estado de largo plazo deberían ser leídos y producidos en la última fase [9].

1.5.2.1. ***Hiperparámetros de LSTM***.

1.5.2.2. Algoritmos Optimizadores.

Estos algoritmos son aquellos que ajustan los parámetros del algoritmo de red neuronal durante la etapa de *training* con el objetivo de minimizar la función de pérdida de una manera iterativa [14]. Entre los algoritmos más comúnmente usados se destacan:

- *Stochastic Gradient Descent with momentum*. El descenso del gradiente se basa en una función convexa y que transforma iterativamente a sus parámetros con el objetivo de minimizar una cierta función a su mínimo local [10].
  El descenso del gradiente estocástico es una extensión del algoritmo anteriormente explicado y mitiga la principal desventaja de este, la cual es que requiere una considerable cantidad de memoria para cargar toda la base de datos [14]. Este algoritmo oscila en diferentes direcciones y actualiza los pesos en cada iteración.
  La principal desventaja en este algoritmo radica en que es ruidoso, pues toma varias oscilaciones [15] como se muestra en la Figura 1.5 (a), traduciéndose en una mayor cantidad en el tiempo de cómputo. Para solucionar este inconveniente se hace uso del descenso del gradiente estocástico con *momentum* el cual añade solo una fracción de la actualización previa a la última actualización [9], de acuerdo con la Figura 1.5 (b).

(a) (b)

Figura 1.5. Descenso del gradiente estocastico sin momentum (a) y con momentum (b).

Tomado de: Sebastian Raschka. Machine Learning with PyTorch and Scikit-Learn, Packt, 2022 [9].

- *AdaGrad* (*Adaptive Gradient Descent*). Para todos los casos anteriores, la tasa de aprendizaje permaneció constante. Para AdaGrad, el algoritmo, dicha tasa varía para cada uno de los pesos de la red [9]. Esto gracias a que realiza pequeños niveles de actualizaciones a los parámetros asociados con variables con alta frecuencia de ocurrencia, y viceversa. Su ventaja es que no necesita una actualización manual de su tasa de aprendizaje, pues, esta cambia automáticamente con cada iteración [8]. La desventaja es que su tasa de aprendizaje se degrada a tal punto que el algoritmo termina parando por completo mucho antes de alcanzar el mínimo global [9].
- *RMSprop* (*Root Mean Squared Propagation*). Ayuda a mitigar en gran parte la desventaja del AdaGrad. Esto lo logra al acumular solamente los gradientes de las iteraciones más recientes [8].
- *Adam* (*Adaptive Momentum estimation*). Es considerado como una combinación del *RMSprop* y *SGD with momentum* [9]. Adam dinámicamente calcula las tasas de aprendizaje individual basándose en el gradiente pasado y el segundo momento [8]. Al incorporar tanto el primer y segundo momento de los gradientes, Adam logra obtener una tasa de aprendizaje adaptativa, la cual eficientemente se mueve a través de la optimización de pesos de la respectiva red neuronal, logrando de esta forma una convergencia más rápida y un mayor poder de estimación de toda la red [9].

1.5.2.3. Función de Pérdida.

Una función de pérdida es una función matemática que mide la diferencia entre el resultado previsto del modelo y el resultado real, es decir, provee una medida cuantitativa del rendimiento del modelo y sus hiperparámetros. La importancia de este tipo de funciones radica en que permiten al modelo con el que se está trabajando lograr una optimización eficiente [14].

Ejemplos de funciones de pérdida para regresión se tienen: *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE). El MSE es la función de pérdida más usada según la literatura revisada, por lo tanto y al tratarse de un problema de regresión el que se va a desarrollar en la presente investigación, se hará uso de esta función [14].

1.5.2.4. Regularización.

La regularización es el proceso de agregar restricciones al modelo para evitar el sobreajuste y mejorar el alcance de generalización de este. Implica agregar un término de penalización a la función de pérdida durante el entrenamiento [15]. De tal forma que a mayor el peso individual del vector, mayor el nivel de penalización. Las técnicas de regularización más comunes son las siguientes [9]:

- Introducir una penalidad por la complejidad del modelo vía regularización L1 y L2.
- Dropout.

Se procederá a brevemente describir cada una.

Regularización L1 o Regresión Lasso. Este tipo de reduce los Bs, en la función objetivo, de las variables independientes menos importantes a un valor de cero, eliminándolas del modelo [5].

Regularización L2 o Regresión Ridge. Este tipo de regularización reduce los coeficientes de las variables independientes menos importantes, pero no las elimina [9].

$$\frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} + \hat{y}^{(i)}\right)^2 + \lambda||w||_2^2 \qquad (1)$$

$$\frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} + \hat{y}^{(i)}\right)^2 + \lambda||w|| \qquad (2)$$

La ecuación 1 y la ecuación 2 permiten visualizar la regularización L2 y L1 respectivamente. En ambos casos, al incrementar el termino de regularización (λ) se procede a aproximar los pesos a cero, completamente cero en el caso de la regularización L1. Haciendo hincapié que en regularización L1, la penalidad es la suma de los valores absolutos de los coeficientes de los pesos de las matrices de vectores, por lo que los pesos pueden ser reducidos a un valor de cero absoluto.

Dropout. El *Dropout* es una técnica de regularización que se ha constituido como la más popular a la hora de combatir el sobreajuste en el uso de redes neuronales, pues permite incrementar el rendimiento del modelo entre un 1 y 2% [8].

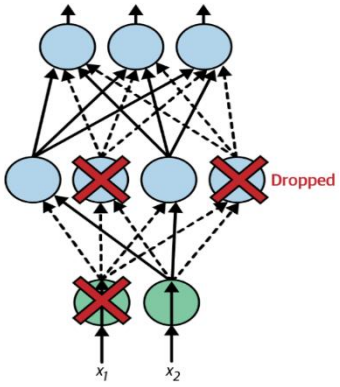


Figura 1.6. Regularización de una red neuronal haciendo uso del Dropout.

Tomado de: Aurelien Geron. Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow, OREILLY, 2022 [8].

En cada fase de entrenamiento, cada neurona tiene una probabilidad *p* de ser temporalmente desconectada, ver Figura 1.6, lo cual significa que será completamente ignorada durante el proceso de entrenamiento, pero puede estar activa en la siguiente fase. El hiperparámetro *p* es conocido como *"dropout rate"* y su valor tiende a oscilar entre 10% y 50%, con un valor entre 20% y 30% es las redes neuronales recurrentes [8].

1.5.2.5. Función de Activación.

Las funciones de activación son funciones matemáticas aplicadas a la salida de cada neurona en una red neuronal. Ayudan a introducir no linealidad en el modelo y permiten que la red modele relaciones complejas en los datos [15].

Las principales funciones de activación para problemas de regresión son:

Tangencial (tanh). La Función de Activación Tangencial es muy similar a la función sigmoidea, la única diferencia es que es simétrica alrededor del origen. El rango de valores en este caso es de -1 a 1; por lo tanto, las entradas a las siguientes capas no siempre serán del mismo signo [14].

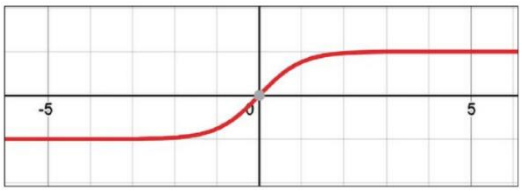


Figura 1.7. Función de Activación Tangencial con resultado en el rango (-1,1).

Tomado de: Amita Kapoor, Antonio Gulli, Sujit Pal. Deep Learning with TensorFlow & Keras, PACKT, 2022 [14].

La función tangencial se define como:

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \qquad (3)$$

La ecuación (3) presenta el terminó *e* el cual representa una constante matemática y es la base del logaritmo natural. Los valores de este término en el numerador y denominador permiten a esta función tomar cualquier valor real como entrada y obtiene como resultado valores en el rango de -1 y 1. De tal forma que al ser más grande el valor de entrada en términos absolutos, valor positivo, más cercano el resultado a 1. En tanto que al ser más pequeño el valor de entrada en términos absolutos, valor negativo, más cercano el resultado a -1 [14].

En la Figura 1.7 se observa que el rango de resultados oscila entre -1 y 1 en esta función de activación. Además, la misma es centrada a los valores de cero y los gradientes no están restringidos a moverse en una dirección en particular [8].

ReLU (REctified Linaer Unit). Esta función tiene la ventaja de no activar todas las neuronas al mismo tiempo y ayudar a resolver el problema del desvanecimiento del gradiente, discutido anteriormente, que afecta principalmente a las redes neuronales recurrentes. Esto debido a que el máximo valor del gradiente de esta función es uno [14].

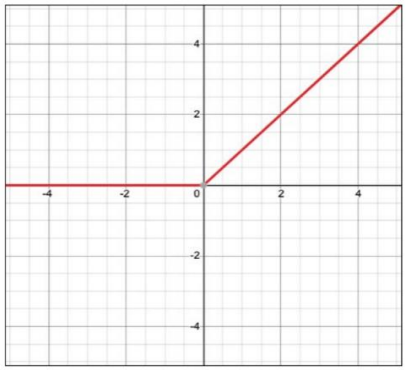


Figura 1.8. Función de Activación ReLUultado en el rango (-1,1).

Tomado de: Amita Kapoor, Antonio Gulli, Sujit Pal. Deep Learning with TensorFlow & Keras, PACKT, 2022 [14].

La función ReLU está definida como:

$$f(x) = \max(0, x) \qquad (4)$$

Se observa en la Figura 1.8 que la función es cero para valores negativos y crece linealmente para los positivos. Para valores negativos las neuronas no se activan y viceversa. Hecho que también se confirma mediante la ecuación (4), ya que el valor de la función oscila entre 0 y x. Esta característica convierte a esta función de activación computacionalmente más eficiente que las dos funciones anteriormente analizadas [14].

1.5.2.6. Tasa de aprendizaje.

La tasa de aprendizaje es posiblemente el hiperparámetro más importante. En general, la tasa de aprendizaje óptima es aproximadamente la mitad de la tasa de aprendizaje máxima [8].

Una forma de encontrar una buena tasa de aprendizaje es entrenar el modelo durante unos cientos de iteraciones, comenzando con una tasa de aprendizaje muy baja (por ejemplo, 10 a 5) y aumentando gradualmente hasta un valor muy grande (por ejemplo, 10). Esto se hace al multiplicar la tasa de aprendizaje por un factor constante en cada iteración [8].

1.5.2.7. BATCH.

El tamaño del *batch* puede tener un impacto significativo en el rendimiento de un modelo y tiempo de entrenamiento. El principal beneficio de utilizar un *batch* de tamaño grande es que los aceleradores de hardware como las GPU pueden procesarlos de manera eficiente [8].

Por ello, muchos investigadores y practicante en el área de ML recomiendan utilizar el tamaño de *batch* más grande que pueda caber en la RAM de la GPU. Sin embargo, hay un problema: en la práctica, los *batch* grandes a menudo conducen a inestabilidades en la fase de entrenamiento, especialmente al comienzo de esta, lo cual puede resultar en que el modelo no pueda realizar una correcta generalización o al menos no en la misma magnitud que un modelo con un tamaño de *batch* pequeño [8].

1.5.2.8. EPOCH.

Es un hiperparámetro más del descenso del gradiente que controla el número de pasadas completas a través del conjunto de datos de entrenamiento [9]. Una época significa que cada muestra del conjunto de datos de entrenamiento ha tenido la oportunidad de actualizar los parámetros internos del modelo. Una época se compone de uno o más batches [15].

Una vez se han revisado los principales conceptos relacionados a LSTM y SVM y sus hiperparámetros, se procederá a introducir los conceptos básicos del mercado financiero para una mejor comprensión del presente trabajo.

1.5.3. Mercado Financiero.

Los mercados financieros son vitales para el buen funcionamiento de las economías capitalistas [16]. Estos mercados pueden incluir activos o valores que cotizan en bolsas reguladas o se negocian en el mercado extrabursátil[11]. Cuando fallan, pueden producirse perturbaciones económicas, incluidas recesiones y un aumento del desempleo [16].

1.5.3.1. Acciones.

Representan la propiedad de una empresa o corporación y un derecho proporcional sobre sus activos y ganancias [16]. Tener acciones significa que un accionista es propietario de una porción de la empresa igual al número de acciones que posee como proporción del total de acciones en circulación de la empresa. Por ejemplo, una persona o entidad que posea 100.000 acciones de una empresa con un millón de acciones en circulación tendría una participación del 10% en ella [16].

1.5.3.2. Bolsa de Valores.

Constituye el lugar físico donde se negocian diferentes instrumentos financieros, incluidas acciones, materias primas y bonos. Las negociaciones reúnen a corporaciones y gobiernos con inversores y ayudan a proporcionar liquidez en el mercado, lo que significa que hay suficientes compradores y vendedores para que las operaciones puedan procesarse de manera eficiente y sin demoras [17].

1.5.3.3. Precio de Cierre Ajustado.

Es el precio final al que el activo subyacente se negocia durante el horario habitual del mercado en un día determinado [18]. En tanto que el precio de cierre ajustado, *Adjusted Close Price* en inglés, modifica el precio de cierre de una acción para reflejar el valor de esa acción después de contabilizar cualquier acción corporativa como divisiones de acciones, dividendos y ofertas de derechos.

El precio de cierre ajustado se considera la valoración más precisa de una acción u otro valor. A menudo se lo utiliza al examinar los rendimientos históricos [16]. Por ejemplo: En el caso que el precio de cierre para cierta empresa es USD 20 el viernes. Después del cierre de los mercados en aquel viernes, la empresa anuncia un dividendo de USD 1,5 por acción. Por lo que el precio ajustado para el viernes sería USD 18,5, el cual se deriva de: USD 20 – USD 1,5.

1.5.3.4. Análisis y pronósticos de Series de Tiempo Financieras.

El análisis de series de tiempo es una forma específica de analizar una secuencia de puntos de datos recopilados durante un intervalo de tiempo. En el análisis de series de tiempo, los analistas registran puntos de datos a intervalos consistentes durante un período de tiempo determinado [5].

[11] Ver Anexo B para la respectiva revisión conceptual.

Lo que distingue a los datos de series de tiempo de otros datos es que el análisis puede mostrar cómo las variables cambian con el tiempo. En otras palabras, el tiempo es una variable crucial porque muestra cómo se ajustan los datos a lo largo de un horizonte temporal. A más de proporcionar una fuente adicional de información y un orden establecido de dependencias entre los datos [5].

1.5.4. Modelo CRISP-DM.

El marco de investigación CRISP-DM (*Cross-Industry Standard Process for Data Mining*) publicado por primera vez en 1999, por la empresa IBM, como metodología de investigación para la estandarización de procesos de minería de datos en diferentes industrias, ha llegado a constituirse como la metodología más común para analítica, ciencia y minería de datos [19].

Este modelo tiene un ciclo de vida que consiste en 6 fases, como se muestra en la Figura 1.9, con flechas indicando las más importantes dependencias de frecuencia entre las distintas fases.

La secuencia entre estas fases no tiene un sentido estricto, de hecho, muchos proyectos aplicados a la vida real se mueven a través de este esquema en cualquier dirección según sea la necesidad del proyecto[12].

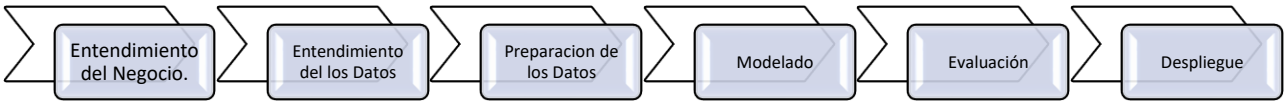


Figura 1.9. Representación gráfica del proceso CRISP-DM.

Tomado de: https://www.ibm.com/docs/en/spss-modeler/saas?topic=guide-introduction-crisp-dm.

Existen 6 fases del modelo CRISP-DM las cuales serán brevemente revisadas en este punto.

1. Entendimiento del negocio.
   Se debe tener en cuenta que todo buen proyecto empieza con un profundo entendimiento de las necesidades del negocio, y por supuesto los proyectos de minería de datos no son la excepción. Esta fase se enfoca en los siguientes puntos:
   - Entendimiento de objetivos.
   - Análisis de la situación.
   - Determinación de los objetivos de la minería de datos.

2. Entendimiento de los datos.

[12] Para la descripción de cada fase del modelo CRISP-DM se ha tomado la información de la página web de IBM. https://www.ibm.com/docs/en/spss-modeler/saas?topic=guide-introduction-crisp-dm.

Esta fase permite identificar, colectar y analizar la base de datos que permitirá conseguir los objetivos del proyecto. La misma posee los siguientes subprocesos:

- Obtención de la base de datos.
- Descripción de la base de datos.
- Exploración y verificación de la calidad.

3. Preparación de los datos.
   En esta fase se prepara el conjunto de datos que servirá de insumo para los algoritmos a utilizar. Básicamente consta de 4 subprocesos:
   - Selección de los datos.
   - Limpieza de los datos.
   - Integración de la base de datos.
   - Formatear/Transformar la base de datos en las unidades de medida que se desea.

4. Modelado.
   En esta fase se construirán y evaluarán varios modelos basados en diferentes técnicas de modelado. Esta fase tiene a su vez 4 subprocesos:
   - Selección de técnicas de modelado.
   - Generación de técnicas de testeo.
   - Construcción del modelo.
   - Evaluación del modelo.

5. Evaluación.
   Determinar qué modelo mejor representa los objetivos del negocio y la determinación de los pasos a seguir es el objetivo en esta fase. 3 subprocesos para seguir:
   - Evaluación de resultados.
   - Revisión del proceso.
   - Determinar los siguientes pasos a seguir.

6. Despliegue.
   Después de aplicar correctamente todas las fases descritas anteriormente, llega el momento de desplegarlo, lo cual básicamente significa darle una utilidad práctica.
   - Planificación del despliegue del modelo.
   - Planificación del monitoreo y mantenimiento del modelo.
   - Producir el informe final.
   - Revisión del proyecto.

1.5.5. Métricas de Error.

1.5.5.1. *Mean Absolute Percentage Error (MAPE)-Error Porcentual Absoluto Medio*.

EL MAPE mide el rendimiento de un modelo de predicción y se puede calcular como el error porcentual absoluto promedio para cada período de tiempo menos los valores reales dividido por los valores reales [5].

$$\text{MAPE} = \frac{1}{n} \sum \frac{|yi - \hat{y}i|}{yi} \qquad (5)$$

Donde:

n = es el número total de observaciones.

yi = Es el valor real para un cierto punto en el tiempo.

$\hat{y}i$ = Es el valor pronosticado para un cierto punto en el tiempo.

$\sum$ = Notación de suma de los valores absolutos para cada punto pronosticado en el tiempo.

El MAPE constituye la sumatoria de las desviaciones entre el valor real y valor pronosticado de cierta variable dividida para el valor real y multiplicado por 1/número total de observaciones, ver ecuación (5).

Esta métrica de error es la medida más común utilizada para pronosticar el error, debido a que es más fácil de interpretar, ya que está expresado como un porcentaje. Lo que lo convierte en independiente a la escala de los valores con los que trabaja [5].

1.5.5.2. *Root Mean Squared Error (RMSE)-Raíz del Error Cuadratico Medio*.

El Error Cuadrático Medio (*Mean Squared Error*, MSE) es calculado como el promedio de las diferencias al cuadrado entre los valores predichos por el modelo y los valores reales para cada observación en el conjunto datos [20].

En tanto que la raíz del error cuadrático constituye la raíz cuadrática del MSE, lo cual significa que las unidades del RMSE son las mismas que las unidades originales de la variable objetivo que está siendo pronosticada [5].

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum |\hat{y}i - yi|^2} \qquad (6)$$

La ecuación (6) representa la fórmula para obtener el RMSE la cual es la raíz cuadrada del MSE.

Mientras el valor resultante de esta métrica sea más pequeño, el modelo implementado tendrá un mayor 'poder' predictivo.

1.5.5.3. *Mean Absolute Error (MAE)-Error Absoluto Medio.*

El Error Absoluto Medio también mide el error en el pronóstico en las mismas unidades de la variable pronosticada, pero la diferencia con la anterior métrica radica en el hecho de que los cambios en el MAE son lineales y por lo tanto intuitivos *[5]*.

$$\text{MAE} = \frac{1}{n} \sum |\hat{y}i - yi| \qquad (7)$$

La ecuación del MAE, (7), es muy similar a la del RMSE solo que esta es simplemente la sumatoria del promedio de las diferencias entre valor real y pronosticado de una determinada variable. Esta métrica no proporciona mayor o menor peso a los diferentes tipos de errores, al contrario del MSE y RMSE que tienden a inflar los errores con un valor absoluto más grande. Por el contrario, sus valores tienden a incrementarse linealmente con incrementos en el error [21].

De igual forma que con las anteriores métricas revisadas, mientras menor sea el valor del MAE, mayor el poder de predicción del modelo.

1.6. Revisión de Literatura.

Investigaciones relacionadas con la estimación de series de tiempo financieras y más concretamente con el pronóstico del precio de las acciones de una empresa o un cierto grupo de empresas haciendo uso de modelos de ML y DL en diferentes partes del mundo durante las últimas décadas son numerosas y diversas. En esta sección se procederá a revisar los trabajos más relevantes relacionados con la temática de la presente investigación.

En el pronóstico de series de tiempo financieras haciendo uso de técnicas de inteligencia artificial predomina fuertemente el uso de DL y en menor medida algunos algoritmos de ML. Dentro de este campo, Armin Lawi, Hendra Mesra y Supri Amir [10] implementan el LSTM y GRU en su estudio para el pronóstico del precio de las acciones de empresas como: Amazon (AMZN), Ball Corp (BLL), Google (GOOGL) y *Qualcomm Incorporated* (QCOM). Haciendo uso de 3 métricas de error, *Mean Absolute Percentage Error* (MAPE), *Root Mean Squared Percentage Error* (RMSPE) y *Rooted Mean Dimensional Percentage Error* (RMDPE), y tomando como única variable de entrada al precio de cierre ajustado, se encontró que los dos modelos presentan un muy buen poder predictivo pues todas las 3 métricas presentaron un nivel de rendimiento por arriba del 94% para todas las 4 empresas. A más de esto, se observa que LSTM obtuvo resultados más precisos para AMZN y QCOM.

A.J.P. Samarawickrama y T.G.I. Fernando [22] también aplican GRU y LSTM en su investigación sobre el pronóstico del precio diario de las acciones de la bolsa de valores de Sri Lanka. 3 empresas

son seleccionadas tanto del sector financiero como comercial, en el periodo correspondiente entre enero 2002 y junio 2013. Utilizando como única variable de entrada el precio de cierre ajustado se estudió el impacto que tiene el número de capas ocultas en el poder predictivo de cada modelo, encontrándose que a un menor número de capas ocultas en los dos algoritmos utilizados, mejor el rendimiento del modelo y que LSTM obtiene menores valores de error que el GRU.

No solamente comparaciones de rendimiento entre LSTM y GRU se ha detectado en las investigaciones de referencia analizadas, también diferentes arquitecturas en el LSTM. Tal es el caso del estudio llevado a cabo por Anita Yadav, C K Jha y Aditi Sharan [23] en donde se implementa dos versiones del LSTM para estimar el precio de las acciones en el mercado de acciones de la India. Una de las versiones considera el estado, *stateful*, y la otra sin estado, *stateless*. Se toman en cuenta para la base de datos 4 distintas empresas que cotizan en la bolsa de valores de dicho país, entre 2008 y 2019. Entre las principales conclusiones se destaca el hecho de que el rendimiento del LSTM es muy dependiente a los valores de sus hiperparámetros y que la diferencia en poder predictivo de las dos versiones de LSTM no es estadísticamente significativas. Por último, se menciona el hecho de que los mejores resultados en términos de predicción fueron alcanzados con una sola capa oculta.

Otra muy interesante investigación es la de Loannis Livieris, Emmanuel Pintelas y Panagiotis Pintelas [11] en donde a través de un modelo hibrido LSTM y *Convolutional Neural Network* (CNN) se pronostica el precio del oro con datos diarios entre enero 2014 y abril 2018 obtenidos del portal web finance.yahoo.com. Se comparó el rendimiento de este modelo combinado con LSTM con una y dos capas ocultas y con SVM. Si bien SVM y LSTM obtuvieron grandes resultados en términos de las métricas de error RMSE y MAE, el modelo combinado CNN-LSTM obtuvo ligeramente mejores resultados, pero a la vez se enmarca que este modelo es altamente sensible a los valores de sus hiperparámetros y es sumamente complejo de implementar.

Un modelo hibrido ARIMA-*Artificial Neural Network* (ANN) ha sido aplicado en la investigación desarrollada por Julián Daniel Jaramillo [3] para pronosticar el precio de las acciones de 4 empresas colombianas que cotizan en la bolsa de valores de Nueva York (*New York Stock Exchange, NYSE*). Trabajando con el logaritmo del precio de cierre ajustado, con un rezago para la variable objetivo y con un numero de neuronas que varía entre 1 y 10, se encontró que esta modalidad hibrida supera en rendimiento predictivo al ARIMA para todos los casos estudiados a excepción de uno en el que el modelo econométrico clásico ARIMA presento mejores resultados. Según los autores esto se debe a que, para esta empresa en particular, Grupo Aval, ARIMA siempre obtuvo muy buenas estimaciones por lo que para las restantes 3 empresas el modelo hibrido tenía amplio margen para superar a ARIMA.

A más de las arquitecturas revisadas, la investigación realizada por Li-Pang Chen [24] implementa LSTM, CNN y SVM para el pronóstico del precio de las acciones de 4 empresas de distintos sectores que cotizan en la bolsa de valores NYSE. Con una base de datos que comprende el periodo enero 2002 a marzo 2020 obtenidas del portal web Yahoo Finance, se utilizaron métricas como MAPE y RMSE como principales evaluadores del modelo. Así, se determinó que para el periodo 2015-2019, todos los modelos obtuvieron buenos resultados, pero desde enero 2020 en adelante la predicción considerablemente disminuye debido en gran parte a la pandemia del COVID-19. En

términos de rendimiento, el modelo hibrido CNN-LSTM obtuvo un mejor poder predictivo que LSTM, y de los 3 modelos SVR obtuvo los mejores resultados. El autor argumenta que este gran resultado de SVR se debe a la gran cantidad de hiperparámetros que maneja este algoritmo y su flexibilidad.

Analizar el efecto que tienen distintos valores del kernel de SVM en el rendimiento del pronóstico de series de tiempo financieras es de gran importancia. Es así como la investigación realizada por Altan Karasu, [7] se enfoca en determinar el valor del kernel que permite obtener la mejor pronóstico del tipo de cambio de USD/TRY y EUR/TRY con el fin de optimizar el portafolio de inversiones de uno o varios agentes de inversión. El periodo analizado es entre junio 2010 y mayo 2019 con datos diarios. Se encontró que el valor del kernel para SVM que obtiene el menor valor del MAE es de 4 tanto para el pronóstico de tipo de cambio dólar estadounidense y lira turca y para el euro y la lira turca.

El estudio realizado por Kyoung-jae Kim [2] hace uso del SVM para pronosticar la dirección del precio diario del índice de la bolsa de valores de Corea del Sur. Se trabaja con datos desde 1989 a 1998, y se toma el precio de cierre ajustado como variable explicativa. Se determinó que hiperparámetros de SVM como C y el Kernel son muy importantes a la hora de obtener óptimos pronósticos.

Una vez revisados los principales trabajos relacionados con el tema del presente proyecto de desarrollo, es necesario hacer hincapié en la importancia de los hiperparámetros de cada algoritmo a utilizar. Para el caso del SVM, el tipo de kernel y el valor de *epsilon* permiten lograr un mayor o menor poder predictivo; en tanto que el valor de *C* como parámetro de regularización permite limitar la existencia del *overfitting*[13]. En cuanto a LSTM, se debe tener presente que la función de activación y el algoritmo optimizador permiten mejorar el poder de predicción de esta red neuronal. En tanto que el número de capas ocultas, numero de neuronas por capa oculta y el regularizador *dropout* permiten controlar el *overfitting*. Destacando el hecho de que un trabajo de estas características en donde se realice un pronóstico del precio de las acciones en empresas del sector financiero o de cualquier otro sector de la economía no ha sido implementado en Latinoamérica. Por lo tanto, los resultados aquí obtenidos deberían servir de guía para futuras investigaciones que se deseen realizar en la materia.

[13] Revisar el glosario de términos en la sección de Anexos.

2. Metodología

El pronóstico de series de tiempo es un proceso meticuloso, que requiere llevar a cabo un conjunto de pasos rigurosos que permita a los algoritmos implementados obtener óptimos resultados en términos de su capacidad predictiva. Dentro de los pasos más importantes a desarrollar e implementar en los respectivos conjuntos de datos, se destacan la selección y análisis estadístico de la variable que servirá como *input* para los dos modelos planteados, escalamiento de los valores, esto es: normalización o estandarización. Todo este minucioso procedimiento requiere establecer una metodología científica acorde a la problemática y que garantice el cumplimiento de los objetivos de la investigación.

En lo referente al pronóstico de series de tiempo financieras, es importante mencionar que los modelos de series de tiempo pueden ayudar a descifrar patrones a partir de datos históricos y utilizarlos para formar proyecciones más precisas. Los inversionistas y profesionales financieros pueden perfeccionar sus pronósticos incorporando métodos de pronóstico de series de tiempo. Esto puede conducir a mejores predicciones de tendencias de ventas, ingresos y gastos [2].

En esta sección, se procederá a desarrollar cada uno de los puntos del Modelo CRISP-DM, con excepción del punto 6 el cual está fuera del objetivo del presente trabajo, explicados dentro del marco teórico.

2.1 Entendimiento del negocio.

En esta fase se procede a identificar el contexto en el que se analiza el presente proyecto de desarrollo, determinado sus objetivos y alcance.

Gran parte de este proceso pasa por el hecho de contextualizar el público objetivo del presente trabajo quienes lo conforman inversionistas que desean tener un mayor grado de certeza acerca de la posible trayectoria que tomará el precio de una determinada acción. Y por otro lado están los investigadores quienes desean conocer sobre nuevas técnicas en la predicción de series temporales.

De cualquier forma, el presente proyecto de desarrollo busca acentuar las bases para futuras investigaciones acerca del pronóstico de series de tiempo aplicando modelos de ML y DL en cualquier tipo de contexto.

2.1.1. Determinación de los objetivos del Negocio.

En el presente proyecto de desarrollo se pondrá en efecto la ejecución de dos algoritmos, uno de ML y otro de DL como son: SVM y LSTM respectivamente. Esto con el objetivo de:

- Determinar el valor de los hiperparámetros óptimos que permiten alcanzar la mejor capacidad de pronóstico.

- Utilizar los distintos hiperparámetros de los dos algoritmos para controlar el *overfitting* en el proceso de entrenamiento respectivo.
- Identificar la configuración más optima, de cada algoritmo, en términos de capacidad predictiva y generalización.
- Realizar el pronóstico de las acciones del BMA y BCol minimizando el valor de las metricas de error: RMSE, MAPE, MAE.
- Identificar cuál de los dos algoritmos es el más idóneo para el pronóstico de series de tiempo en función de capacidad predictiva, tiempo de entrenamiento y cantidad de hiperparámetros.

2.1.2. Análisis de la Situación.

Para el desarrollo del presente trabajo, se ha hecho uso del interpretador de código Python *Google Colab* el cual representa una plataforma *open source* para correr Python de forma nativa. No ha sido necesario el uso de otros lenguajes de programación o software.

Python contiene un gran cantidad de librerías muy útiles a la hora de explorar y visualizar diferentes observaciones. Para el caso particular de la presente investigación, se usó *Pandas* para la exploración de datos y análisis estadístico descriptivo. Mientras que, para la visualización de las distintas gráficas, se usó la librería *Matplotlib.*

En lo que respecta a los modelos SVR y LSTM es importante mencionar que se ha hecho uso de las librerías de Python *Scikit Learn* y *Keras,* las cuales cuentan con funciones especializadas para el entrenamiento de distintas bases de datos. A más de esto, estas librerías traen implementadas consigo las 3 metricas de error que se utilizó en el presente trabajo, estas son: RMSE, MAPE y MAE.

2.2 Entendimiento de los datos.

Esta fase comprende la obtención, descripción, exploración y verificación de los datos obtenidos del portal web *Yahoo Finance.* Se desarrolla cada punto con el objetivo de explicar de una forma clara y resumida cada procedimiento que ha llevado a que se obtengan las dos bases de datos en los formatos necesarios para servir de entrada a los algoritmos LSTM y SVM y posteriormente aplicar las respectivas metricas de error como medida de evaluación del procedimiento.

2.2.1 Obtención de la Base de Datos.

Para poder entrenar a los modelos LSTM y SVM se hizo uso de dos documentos en formato csv descargados directamente del portal web Yahoo Finance. Estos documentos poseen información

del precio de cierre ajustado de las acciones de BMA y BCol desde el 01 de abril de 2006 hasta el 31 de marzo de 2023. Ambas bases de datos se encuentran dólares reales[14] estadounidenses.

2.2.2 Descripción de la Base de Datos.

Los datos de las dos empresas financieras corresponden al periodo comprendido entre abril 2006 y marzo 2023. La base de datos se encuentra en formato csv (*comma-separated values*) y permiten observar el precio de las acciones de cada banco en un día determinado en función del precio de apertura, cierre, máximo valor y mínimo valor, además del volumen generado en cada día. Se cuenta en total con 4278 observaciones para cada uno de los bancos seleccionados, acotando el hecho de que los sábados y domingo no presentan información debido a que los mercados financieros no laboran los fines de semana.

2.2.3 Exploración y verificación de la calidad de los Datos.

Una vez se han descrito las características más básicas de la base de datos a trabajar, se procederá a explorar los mismos.

```
Banco = 'BMA'

BMA = yf.download(Banco, start='2006-04-01', end='2023-03-31')
BMA
```

```
[**********100%**********]  1 of 1 completed
```

Date	Open	High	Low	Close	Adj Close	Volume
2006-04-03	23.000000	23.100000	22.670000	22.700001	14.835738	160200
2006-04-04	22.900000	23.600000	22.900000	23.600000	15.423942	454600
2006-04-05	23.600000	23.750000	23.400000	23.400000	15.293227	84400
2006-04-06	23.200001	23.299999	22.690001	22.690001	14.829206	60000

Figura 2.1. Variables y Valores del precio de las acciones del Banco Macro de Argentina.

Los datos son descargados directamente de la página web de *yahoo finance* a través de la API yfinance de Python, con fecha inicial 01 de abril de 2006 hasta el 31 de marzo de 2023. Una observación a tomar en cuenta es que los precios de las acciones de ambas instituciones financieras están calculados en dólares estadounidenses, ver Figura 2.1. Este precio representa el precio real o valor de mercado lo cual refleja la cantidad de dinero que los inversionistas pagarán por una acción basados en información pasada, presente y predicciones [25]. En este punto se debe aclarar el hecho de que la propia casa de valores, NYSE, se encarga de transformar los precios de las acciones en moneda local, pesos argentinos en el caso del Banco Macro y pesos colombianos para el BanColombia, a dólares estadounidenses.

[14] Ver Anexo B para revisar la respectiva referencial conceptual.

```
Banco = 'CIB'

BCol = yf.download(Banco, start='2006-04-01', end='2023-03-31')
BCol
```

```
[**********100%**********]  1 of 1 completed
```

Date	Open	High	Low	Close	Adj Close	Volume
2006-04-03	35.000000	35.990002	34.950001	35.799999	19.123852	458700
2006-04-04	35.950001	36.180000	35.810001	35.919998	19.187954	216100
2006-04-05	35.939999	35.939999	34.750000	34.830002	18.605694	307000
2006-04-06	34.840000	34.970001	34.299999	34.840000	18.611034	231800

Figura 2.2. Variables y Valores del precio de las acciones del BanColombia.

Las Figuras 2.1 y 2.2 permiten observar los 4 primeros valores del precio de las acciones para cada banco y las distintas variables con las que se dispone en las bases de datos obtenidas.

En la Tabla 2.1 es posible visualizar el tipo de variable y los valores no nulos para cada una de las columnas con las que se trabaja. Se cuentan 4278 valores no nulos, lo cual corresponde al mismo número de valores totales de la muestra. Todas las columnas corresponden a valores flotantes, ya que se cuenta con valores decimales, no así para la columna volumen en donde se tienen valores enteros.

Banco Macro de Argentina (BMA)		
Columna	Valores No Nulos	Tipo de Dato
Open	4278	float64
High	4278	float64
Low	4278	float64
Close	4278	float64
Adj Close	4278	float64
Volume	4278	int64

Tabla 2.1. Valores No nulos y tipo de dato BMA.

Exactamente los mismos valores pueden ser observados para el BanColombia, de acuerdo con la Tabla 2.2, lo cual no sorprende considerando que los datos son extraídos de la misma fuente y bajo el mismo formato. Nuevamente, el total de observaciones coincide con el total de valores no nulos por lo que se evidencia para las dos bases de datos que no existen valores nulos o valores que no sean considerados por Python como flotantes.

Banco Colombia (BCol)		
Columna	Valores No Nulos	Tipo de Dato
Open	4278	float64
High	4278	float64
Low	4278	float64
Close	4278	float64
Adj Close	4278	float64
Volume	4278	int64

Tabla 2.2. Valores No nulos y tipo de dato BCol.

Las Tablas 2.3 y 2.4 presentan interesantes cifras estadísticas, en donde si se compara el precio medio de las acciones de cada banco en el periodo de tiempo estudiado, se puede observar que no existe una diferencia substancial. Por ejemplo, el precio de cierre ajustado de cada acción del Banco Macro es de USD 27,75, mientras que el del BanColombia es de USD 29,68.

	Open (Precio de la acción al empezar el día)	High (Máximo precio de la acción en el día)	Low (Mínimo precio de la acción en el día)	Close (Precio de la acción al terminar el día)	Adj Close (Precio de la acción ajustado al terminar el día)	Volume (Volumen negociado)
Número Observaciones	4278	4278	4278	4278	4278	4.28E+03
Media	35.84	36.566	35.086	35.823	27.756	2.27E+05
Desviación Estandar	25.13	25.51	24.73	25.12	20.33	2.77E+05
Valor Minimo	6.55	7.11	4.92	6.83	4.59	3.20E+03
Primer Cuartil	17.11	17.5	16.62	17.03	13.3	9.08E+04
Mediana (Segundo Cuartil)	26.98	27.54	26.47	26.99	19.61	1.57E+05
Tercer Cuartil	45.16	46.33	44	45.2	35.35	2.81E+05
Valor Maximo	135.13	136.1	133.5	135.46	108.3	7.17E+06

Tabla 2.3. Principales valores estadísticos del Banco Macro, Argentina.

La primera columna la cual representa el precio de la respectiva acción al empezar un determinado día presenta una media de USD 35.84. Este valor se mantiene relativamente constante desde la segunda columna hasta la cuarta, las cuales representan el máximo precio, mínimo precio y precio al terminar el día de la acción respectivamente, tal y como se observa en la tabla 2.3.

La mediana o segundo cuartil el cual constituye una medida estadística más precisa que la media al no dar un peso significativo a los valores extremos tiene un valor de USD 26.99 para la columna precio al terminar el día, cuarta columna desde la izquierda. Este valor permite determinar que la mitad del periodo total de la serie temporal, entre 01 de abril de 2006 y 31 de marzo de 2023, tiene un valor para la variable precio de cierre superior a USD 26.99. Mientras que la otra mitad del periodo total de la serie temporal presenta un valor menor a USD 26.99 para la misma variable.

El mínimo valor del precio de la acción del día, tercera columna, presenta un valor de USD 44 para el tercer cuartil. Este valor significa que el 75% del periodo total de la serie temporal el valor mínimo de la acción en un determinado día fue de USD 44. Mientras que el restante 25% del tiempo, el valor mínimo de la acción en un día ha sido superior a esta cantidad, llegando a un máximo de USD 133.5.

Por último, la columna de la derecha representa el total de acciones negociados, compradas y vendidas, del BMA. El valor promedio del número de acciones negociados por día es de 227.000, con un valor mínimo de 3.200 y máximo de 7.170.000.

	Open (Precio de la acción al empezar el día)	High (Máximo precio de la acción en el día)	Low (Mínimo precio de la acción en el día)	Close (Precio de la acción al terminar el día)	Adj Close (Precio de la acción ajustado al terminar el día)	Volume (Volumen negociado)
Número Observaciones	4278	4278	4278	4278	4278	4.28E+03
Media	42.43	42.95	41.88	42.42	29.68	3.83E+05
Desviación Estandar	13.04	13.07	12.99	13.04	9.02	2.69E+05
Valor Minimo	15.7	16.59	15	15.33	8.92	3.61E+03
Primer Cuartil	31.7	32.2	31.2	31.65	22.61	2.19E+05
Mediana (Segundo Cuartil)	40.11	40.6	39.57	40.12	29.52	3.19E+05
Tercer Cuartil	53.49	53.95	52.77	53.4	37.89	4.59E+05
Valor Maximo	70.48	70.62	70.06	70.5	46.48	5.93E+06

Tabla 2.4. Principales valores estadísticos del BanColombia.

La Tabla 2.4 permite visualizar los principales estadísticos relacionados a las 6 variables del precio de las acciones del BCol. El precio de la acción al empezar el día presenta un valor mínimo de USD 15.7 y un valor máximo de USD 70.48 lo cual representa un incremento de 4.5 veces.

La segunda columna representa el máximo precio de la acción en el día, y el estadístico primer cuartil tiene un valor de USD 32.2. Esto significa que en un 25% de tiempo del total de la serie temporal el valor máximo de la acción del BCol durante el día presente este valor, mientras que el 75% restante de la serie temporal tiene un valor superior a USD 32.2.

El precio de la acción del BCol al terminar el día, cuarta columna, tiene un valor mínimo de USD 15.33 y un valor máximo de USD 70.5. El promedio de acciones de este banco que se negocian por día es de 383.000, con un valor mínimo de 36.100 y máximo de 5.930.000.

Por la tanto una notable diferencia entre los dos bancos analizados es el número mínimo de acciones que se negocian al día. Pues para BMA esta cifra es de 3.200, mientras que para BCol esta cantidad es mucho más grande, 36.100. Este característica, más el hecho de que se negocian en promedio una mayor cantidad de acciones del BCol que del BMA lleva a determinar que las acciones del BCol son mucho más demandades por los diferentes inversionistas durante el periodo de análisis.

Otro punto a tomar en cuenta y que representa una diferencia significativa entre las dos instituciones es el hecho de que el valor mínimo del precio ajustado de cierre de las acciones del Banco Macro fue de USD 4,51 y el valor máximo USD 106,34, es decir, su precio se incrementó en 24 veces aproximadamente entre abril de 2006 y abril de 2023. En tanto que para el BanColombia esta misma cifra es de 5,2, pasando de USD 8,63 a USD 44.94, durante el mismo periodo. Este indicador sirve para determinar que el precio de las acciones del BMA son mucho más volátiles que las del BCol. En la sección de resultados se analizará si este hecho determina en algún grado la capacidad predictiva de los dos algoritmos.

La Figura 2.3 permite comparar las fluctuaciones del precio de cierre ajustado de los dos bancos entre sí.

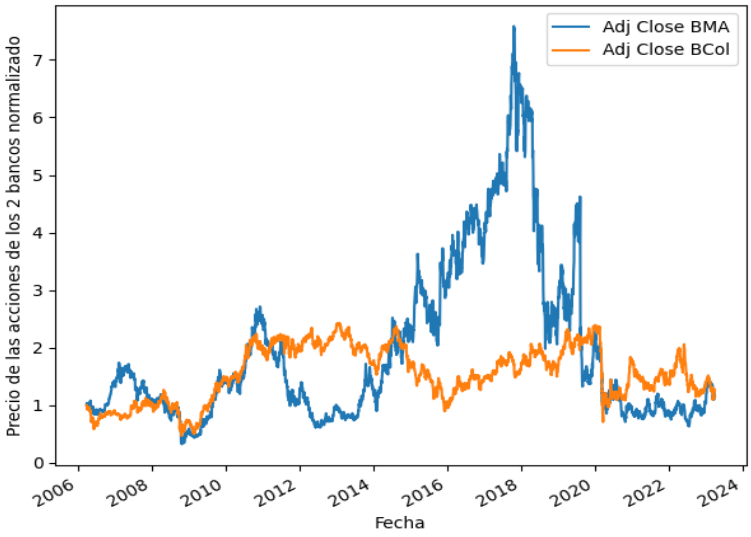


Figura 2.3. Precio normalizado de las acciones de las dos empresas.

Para poder construir la gráfica de la Figura 2.3 se procedió a normalizar[15] cada base de datos. Este procedimiento permite observar que el precio de cierre ajustado del BCol ha permanecido relativamente constante sin mayores oscilaciones. En tanto, que para el BMA presenta una fuerte fluctuación sobre todo en el periodo entre 2015 y 2020. El eje de las ordenadas permite distinguir cuantas veces se incrementó el precio de la acción de los dos bancos en una fecha determinada en comparación con el precio para el primer día del que se tiene información. Por ejemplo, para el BMA el precio de la acción en el año 2018 fua aproximadamente 7 veces más el precio de la acción en abril de 2006.

Fecha	BMA. Precio de cierre ajustado normalizado	BCol. Precio de cierre ajustado normalizado
4/3/2006	1.00	1.00
4/4/2006	1.039	1.003
4/5/2006	1.03	0.973
4/6/2006	0.999	0.973
4/7/2006	0.993	0.933

Tabla 2.5. Precio normalizado para BMA y BCol.

Mediante la Tabla 2.5 es posible visualizar la evolución del precio normalizado para ambos bancos durante los primeros 5 días. Al estar los precios divididos para el primer día, 03 de abril de 2006, se observa como los siguientes días evolucionan en función de este día. Por lo que un valor menor a 1 se interpreta como un decremento del precio tomando como base el primer día, mientras que un valor mayor a 1 se considera un incremento.

[15] Se procedió a dividir el precio de cada acción para el respectivo precio del primer día, logrando de esta forma que los precios de ambas empresas tengan el mismo punto de partida.

En las Figuras 2.4 y 2.5 se puede visualizar el análisis de estacionariedad[16] aplicado a las series temporales del precio de cierre ajustado del BCol y BMA respectivamente.

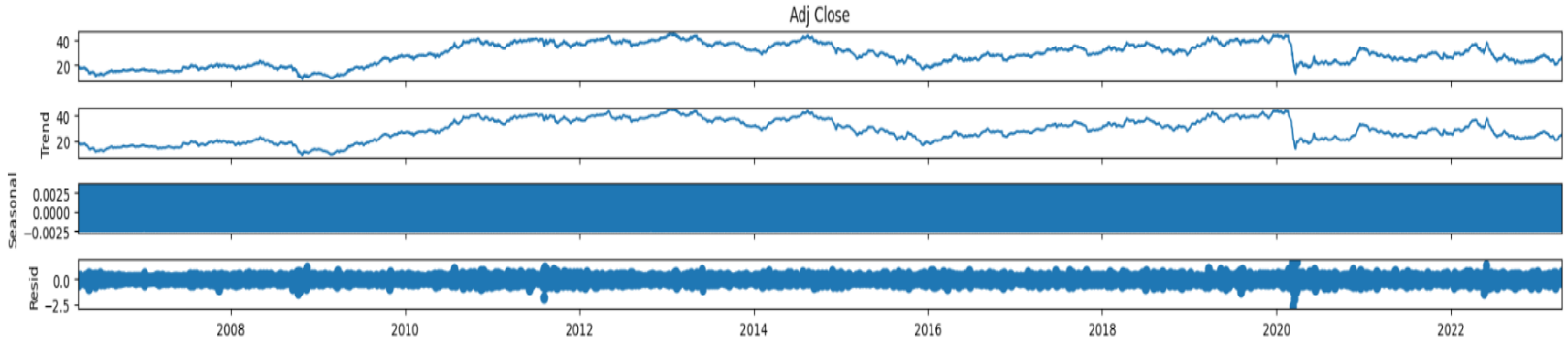


Figura 2.4. Descomposición de Estacionariedad en componente tendencial, temporal y residual. BCol.

Al observar las Figuras 2.4 y 2.5 se identifica que tanto la serie temporal del BCol y BMA respectivamente presentan un leve componente tendencial, segunda ilustración, pues no existen mayores oscilaciones para el periodo de tiempo analizado.

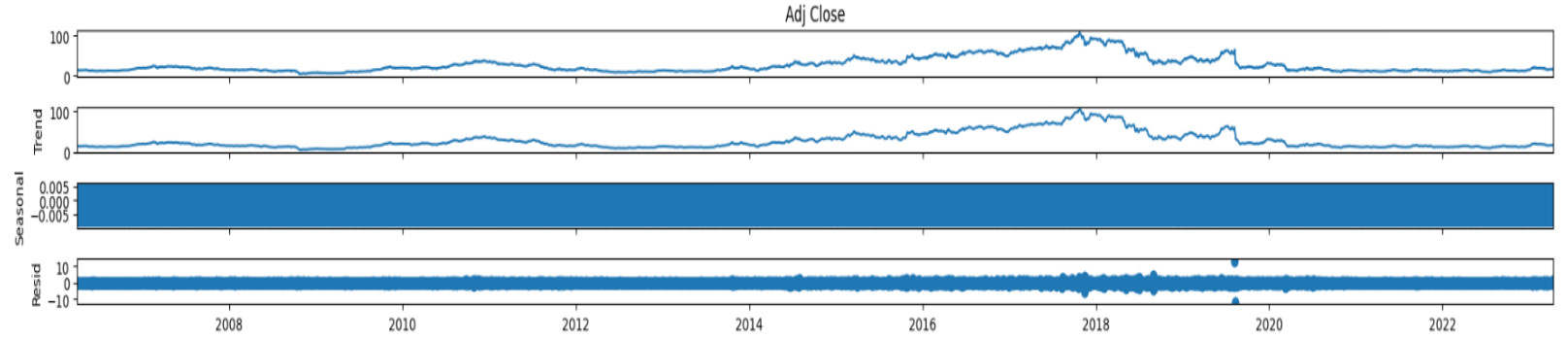


Figura 2.5. Descomposición de Estacionariedad en componente tendencial, temporal y residual. BMA.

[16] Ver Anexo B para la respectiva referencia conceptual.

El componente estacional, el cual representa la tercera ilustración en las Figuras 2.4 y 2.5, es inexistente pues se divisa una línea gruesa horizontal. En caso de existir estacionalidad, se observarían líneas en zigzag a lo largo del periodo de tiempo analizado.

En cuanto al componente residual de ambas series temporales, es posible observar que el residuo se mantiene en un rango de valores cercano a cero durante todo periodo de tiempo analizado. A este fenómeno se lo conoce como ruido blanco[17].

Este análisis permite determinar que las dos series de tiempo objeto de estudio en el presente trabajo, no presentan una tendencia marcada que sea creciente o decreciente. Esta característica permite a los dos algoritmos, LSTM y SVM, incrementar la capacidad predictiva ya que no están forzados a aprender el componente tendencial y estacional de las dos series temporales [8].

2.3 Preparación de los datos.

Esta fase permite tomar los datos en su forma natural y transformarlos en el formato que se necesita para poder trabajar con los dos algoritmos a implementar. Debido a la naturaleza propia del objetivo de estudio del presente proyecto de desarrollo y al hecho de que las observaciones obtenidas no presentan ningún valor atípico ni datos nulos, esta fase no requirió mayor costo computacional.

2.3.1 Selección de los Datos.

En esta fase se procede a determinar la variable que servirá de insumo para los algoritmos a implementar. Para el efecto se ha elegido a la variable *Adjusted Close Price*, precio de cierre ajustado, como variable predictora. James C. Van Horne y John M. Wachowicz, Jr. [16] argumentan que el precio de cierre ajustado se considera la valoración más precisa de una acción. Esto último, debido a que esta variable modifica el precio de cierre de una acción para reflejar el valor de esa acción después de contabilizar el accionar corporativo como divisiones de acciones, dividendos y ofertas de derechos[18].

En cuanto al número de observaciones, se toman en cuenta todas las 4278 observaciones que, como se aclaró en las anteriores secciones, corresponde al número de días para los cuales se tiene información sobre el precio de las acciones de ambos bancos entre abril 2006 y marzo 2023 de lunes a viernes.

[17] Ver Anexo B.

[18] Ver sección 1.5.3.3.

2.3.2 Limpieza de los Datos.

Tal y como se puede observar en las tablas 2.1 y 2.2, no existen valores nulos o valores que sean no numéricos de tipo float64 para la variable de entrada Precio de Cierre Ajustado en el caso de los dos bancos estudiados.

Por lo tanto, la variable en cuestión cumple con los parámetros necesarios para servir de entrada para los dos algoritmos a implementar, no requiriendo ningún tratamiento en esta fase.

2.3.3. Formatear/Transformar la base de Datos en las unidades de medida y estructura que se desea.

Cuando se implementa un proyecto de ML o DL es muy importante llevar a cabo un procedimiento denominado escalamiento de variables el cual consiste en llevar a cabo la estandarización de las variables de entrada. Estandarización conlleva centrar las variables de entrada con un valor de media cero y una desviación estándar de uno, de tal forma que estas variables tienen los mismos parámetros que una distribución normal estándar, lo cual hace que el aprendizaje de los pesos sea más sencillo [9].

Las ventajas en usar esta técnica son [9]:

- Algoritmos como SVM, Redes Neuronales, Regresión Logística, entre otros, cuentan con valores en los pesos de cero o cercanos a cero, por lo que contar con una media en las variables de entrada en este rango, resulta beneficioso para los mencionados algoritmos.
- La estandarización mantiene la información original de las variables de entrada como en el caso de los valores anormales, *outliers*, y al mismo tiempo permite a los algoritmos ser menos sensibles a estos valores.

$$x^{(i)}std = \frac{x^{(i)} - \mu x}{\sigma x} \qquad (8)$$

En donde:

$X^{(i)}$ = Es una observación particular.

μx = Es la media o valor promedio de la variable en cuestión.

σx = Es la desviación estándar de la variable en cuestión.

La ecuación (8) permite transformar los valores de una cierta variable en unidades estándar al tomar cada valor y restarle la media de toda la variable y dividirla para la desviación estándar de esta última. Hay que tomar en cuenta que cada unidad estandarizada mide la distancia entre una observación con su valor promedio en términos de la desviación estándar de esta observación [26].

Como se explicó en las anteriores secciones, la variable que servirá para pronosticar el precio de las acciones es Precio de cierre ajustado. Por lo tanto, se procede a re-escalar esta variable.

Fecha	Precio a Dia 0	Precio a Dia 1	Precio a Dia 2	Precio a Dia 3	Precio a Dia 4	Precio Objetivo
2006-03-31	14.162241	14.020185	14.576052	14.452523	14.014007	13.927539
2006-04-03	14.020185	14.576052	14.452523	14.014007	13.927539	13.649609
2006-04-04	14.576052	14.452523	14.014007	13.927539	13.649609	13.501374
2006-04-05	14.452523	14.014007	13.927539	13.649609	13.501374	13.587844
2006-04-06	14.014007	13.927539	13.649609	13.501374	13.587844	13.865779
2006-04-07	13.927539	13.649609	13.501374	13.587844	13.865779	13.989303

Figura 2.6. Precio Ajustado Valor Real en USD, BMA.

La Figura 2.6 permite observar un subconjunto de la variable precio de cierre ajustado en valor real, es decir en dólares estadounidenses. La columna 'Precio a Día 0' representa el precio de cierre ajustado al día en cuestión. Por ejemplo, para el caso de la primera observación, 31 de marzo 2006, Precio a Día 0 es el precio de la acción al día 31 de marzo. La columna 'Precio a Día 1' representa el precio de cierre ajustado al siguiente día en cuestión. Para el caso de la primera observación, este precio corresponde al precio de la acción al día 3 de abril de 2006, el cual constituye el segundo día para el cual se tiene información acerca del precio de la acción del BMA. Este proceso se repite hasta llegar al 5to día[19], el cual representa el precio objetivo o precio pronosticado para el día en cuestión. Para el caso del 31 de marzo de 2006 el precio pronosticado para este día es el precio de la acción en 5 días, esto es el 7 de abril de 2006.

Fecha	Precio a Dia 0	Precio a Dia 1	Precio a Dia 2	Precio a Dia 3	Precio a Dia 4	Precio Objetivo
2006-03-31	-0.678313	-0.685472	-0.658890	-0.665212	-0.686727	-0.691239
2006-04-03	-0.685200	-0.658559	-0.664863	-0.686390	-0.690898	-0.704631
2006-04-04	-0.658251	-0.664540	-0.686067	-0.690566	-0.704306	-0.711774
2006-04-05	-0.664240	-0.685772	-0.690248	-0.703989	-0.711457	-0.707607
2006-04-06	-0.685500	-0.689958	-0.703687	-0.711149	-0.707286	-0.694215
2006-04-07	-0.689692	-0.703414	-0.710854	-0.706972	-0.693878	-0.688263

Figura 2.7. Precio Ajustado Valor estandarizado, BMA.

[19] Se establece el 5to día para hacer alusión al hecho de que el precio pronosticado de cada acción para el día lunes, por ejemplo, está en función del precio del siguiente lunes y así sucesivamente. Se sigue la metodología empleada en el libro titulado: Time Series Analysis with Python Cookbook de la editorial Packt y desarrollada por Atwan, Tarek A. [5]

En tanto que, en la Figura 2.7 se distingue el precio estandarizado, es decir luego de haber aplicado el procedimiento de re-escalamiento de la variable[20] para el BCol.

Fecha	Precio a Dia 0	Precio a Dia 1	Precio a Dia 2	Precio a Dia 3	Precio a Dia 4	Precio Objetivo
2006-03-31	18.025801	18.490652	18.552624	17.989645	17.994816	17.261385
2006-04-03	18.490652	18.552624	17.989645	17.994816	17.261385	17.044460
2006-04-04	18.552624	17.989645	17.994816	17.261385	17.044460	17.132261
2006-04-05	17.989645	17.994816	17.261385	17.044460	17.132261	17.586782
2006-04-06	17.994816	17.261385	17.044460	17.132261	17.586782	17.638433
2006-04-07	17.261385	17.044460	17.132261	17.586782	17.638433	17.411169

Figura 2.8. Precio Ajustado Valor Real en USD, BCol.

El proceso de estandarización desarrollado para el BMA también se lo aplica a BCol. La Figura 2.8 permite visualizar los valores previos a estandarizarlos aplicando el mismo procedimiento empleado para BMA.

Fecha	Precio a Dia 0	Precio a Dia 1	Precio a Dia 2	Precio a Dia 3	Precio a Dia 4	Precio Objetivo
2006-03-31	-1.074527	-1.025601	-1.019602	-1.080292	-1.080419	-1.159394
2006-04-03	-1.024947	-1.018990	-1.079668	-1.079740	-1.158695	-1.182550
2006-04-04	-1.018337	-1.079046	-1.079116	-1.158004	-1.181847	-1.173178
2006-04-05	-1.078383	-1.078495	-1.157367	-1.181152	-1.172476	-1.124660
2006-04-06	-1.077832	-1.156734	-1.180512	-1.171782	-1.123967	-1.119147
2006-04-07	-1.156057	-1.179875	-1.171144	-1.123281	-1.118454	-1.143406

Figura 2.9. Precio Ajustado Valor estandarizado, BCol.

En la Figura 2.9 se aprecia el precio de cierre ajustado estandarizado para el BCol haciendo uso de la clase denominada *Standardize* en Python[21] y la función *fit_transform*.

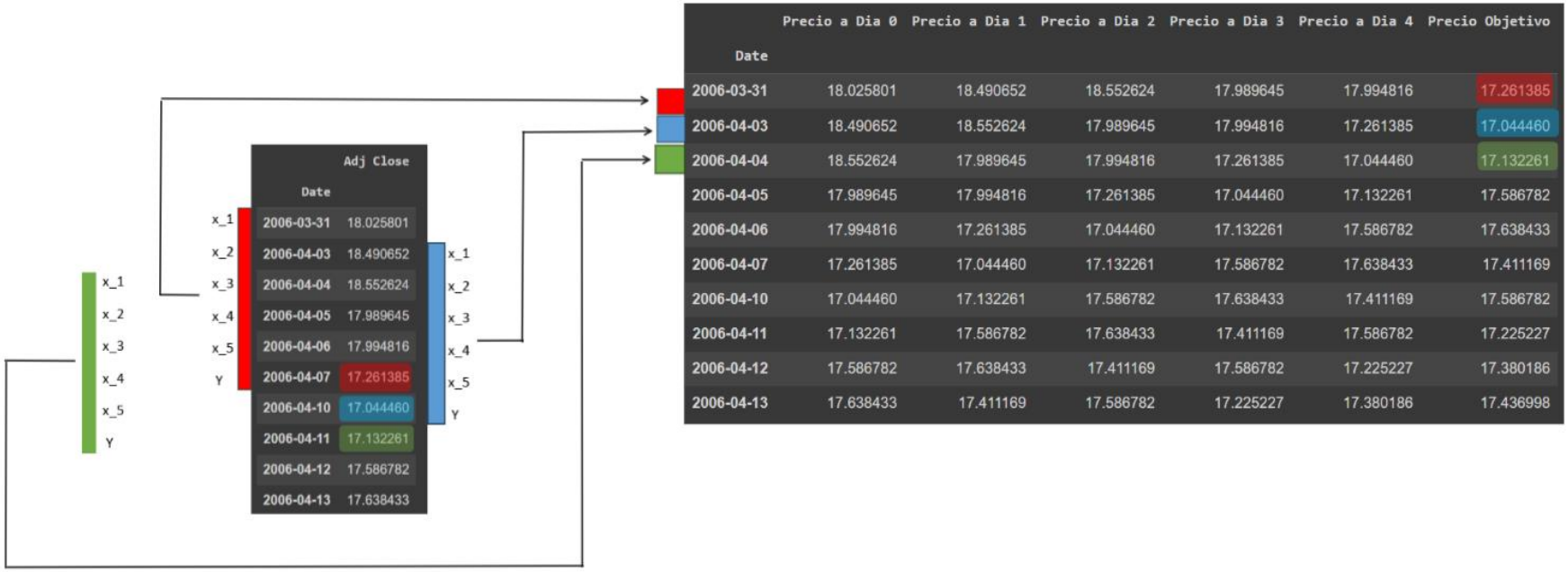
Una vez descrito el proceso de estandarización aplicado a la variable objetivo, se procederá a explicar la arquitectura implementada en la base de datos para la obtención de un modelo con una variable dependiente en función de sus rezagos, también conocidos como variables independientes.

La Figura 2.10 permite ilustrar cómo con una venta temporal de 5 periodos, (a), se obtienen 5 columnas (b), las mismas que trabajan como variables independientes, y una columna, Y, que sirve como variable objetivo. El número total de columnas depende del valor del tamaño de la ventana temporal, número de columnas = tamaño de la ventana temporal + 1.

[20] Se hizo uso de una clase llamada Standardize la cual se encarga de estandarizar la variable en cuestión y a la vez de reconvertirle en su valor original para efectos de pronóstico. El respectivo código se encuentra en el repositorio de github perteneciente al autor y listado en la sección de Anexo A.

[21] Ver Anexo A para revisar el correspondiente código en Jupyter Notebook.

Para el primer día en cuestión, 31 de marzo 2006, se tiene que el precio de la acción a pronosticar para este día es su precio en 5 días[22], es decir el precio al 7 de abril del mismo año. Para el segundo día, esto es 3 de abril de 2006, el precio pronosticado es el precio al 5to día, esto es 10 de abril, y así sucesivamente.



Date	Precio a Dia 0	Precio a Dia 1	Precio a Dia 2	Precio a Dia 3	Precio a Dia 4	Precio Objetivo
2006-03-31	18.025801	18.490652	18.552624	17.989645	17.994816	17.261385
2006-04-03	18.490652	18.552624	17.989645	17.994816	17.261385	17.044460
2006-04-04	18.552624	17.989645	17.994816	17.261385	17.044460	17.132261
2006-04-05	17.989645	17.994816	17.261385	17.044460	17.132261	17.586782
2006-04-06	17.994816	17.261385	17.044460	17.132261	17.586782	17.638433
2006-04-07	17.261385	17.044460	17.132261	17.586782	17.638433	17.411169
2006-04-10	17.044460	17.132261	17.586782	17.638433	17.411169	17.586782
2006-04-11	17.132261	17.586782	17.638433	17.411169	17.586782	17.225227
2006-04-12	17.586782	17.638433	17.411169	17.586782	17.225227	17.380186
2006-04-13	17.638433	17.411169	17.586782	17.225227	17.380186	17.436998

Figura 2.10. Serie Temporal original para el BCol (a). Serie Temporal transformada en función de 5 periodos de tiempo (b).

Elaboración: Autor.

Al pronosticar el precio de la acción en un determinado día como su precio en el enésimo día, $5.^0$ día en la Figura 2.10 (a), esencialmente se está transformando una serie de tiempo uni-variada en una regresión múltiple[23]. Por lo tanto, con una ventana temporal de 5, se obtiene un problema de regresión múltiple de 5 variables independientes, y una variable dependiente para un total de 6 columnas, tal

[22] Con 5 días se obtiene una ventana temporal de 5. Con 10 días la ventana temporal es de 10, etc.

[23] Esto se logra al aplicar una función denominada *one_step_forecast* en Python. Revisar la respectiva referencia de código en el repositorio de github listado en el Anexo A.

como se muestra en Figura 2.10 (b)[24]. En resumen, la metodología implementada permite que cada variable objetivo, precio pronosticado, constituye la misma variable rezagada 5 periodos, como se muestra en la Figura 2.10 (b) columna Precio Objetivo.

2.4 Modelado.

En esta fase se describirán los modelos a implementar, las técnicas de testeo, la construcción del modelo y la evaluación.

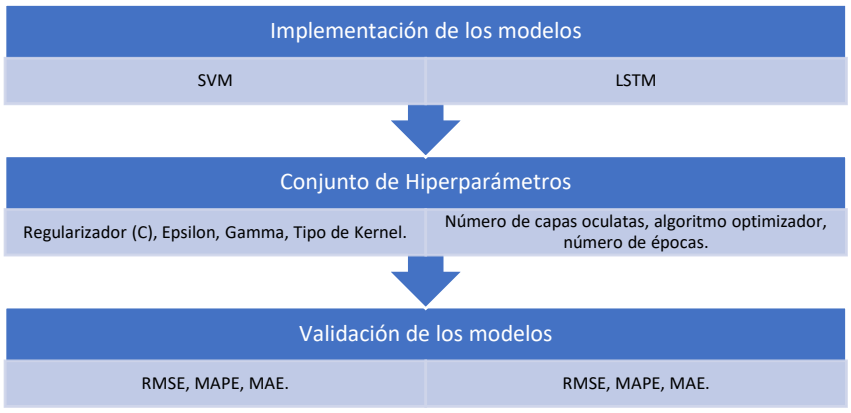


Figura 2.11. Fase de Modelación de los dos algoritmos a implementar y aplicación de métricas de evaluación de rendimiento.

La Figura 2.11 describe los procesos a implementar durante la fase de modelado. Para ambos algoritmos se procede a probar un conjunto de hiperparámetros con distintos valores para cada uno. Posteriormente, se mide el error de cada arquitectura de modelo mediante las métricas de error ya descritas: RMSE, MAPE y MAE.

2.4.1 Selección de técnicas de modelado.

En el desarrollo del presente proyecto de investigación se hace uso de los siguientes algoritmos:

- *Support Vector Machine* (SVM)
- *Long Short-Term Memory* (LSTM)

SVM.

[24] Este tipo de metodología ha sido utilizada en el libro titulado: Time Series Analysis with Python Cookbook de la editorial Packt y desarrollada por Atwan, Tarek A. [5]

Este modelo se implementó de la siguiente manera[25]:

1. Se transforman las dos series temporales uni-variable en una función de 5 rezagos.
   #Código implementado en *Google Colab.*
   La función *one_step_forecast* toma dos argumentos, el primero es la base de datos que se ha cargado y el segundo es el tamaño de la ventana temporal que como se explicó anteriormente es de 5 que corresponde a 5 días[26].

   ```
   BCol_os = one_step_forecast(BCol, 5)
   BMA_os = one_step_forecast(BMA, 5).
   ```

2. Obtención del conjunto de entrenamiento y *test*.
   Se aplica la función s*plit_data* el cual tomo como argumento el objeto obtenido en el paso 1, esto es BCol_os y BMA_os.

   ```
   train_BCol, test_BCol = split_data(BCol_os)
   train_BMA, test_ BMA = split_data(BMA _os)
   ```

3. Estandarización de las dos bases de datos y división en set de entrenamiento y test.
   Se hace uso de la clase *Standardize*[27], la cual, a más de estandarizar a las variables objetivo, tiene un método que permite invertir el proceso, llamado *inverse_y*, de tal forma que la variable pronosticada vuelva a estar en sus unidades originales, para este trabajo esto es dólares estadounidenses.
   Esta clase toma como argumento el objeto obtenido en el paso 1.

   ```
   BCol_estand = Standardize(BCol_os)
   BMA _estand = Standardize(BMA_os)
   ```

4. Obtención del conjunto de entrenamiento y *test* estandarizado.
   Se hace uso del método *fit_transform* de la clase *Standardize* el cual permite dividir los datos estandarizados en *set* de entrenamiento y *test*.
   Este método toma como argumento a los objetos obtenidos en el paso 2, esto es *train* y *test* de cada una de las dos bases de datos.

   ```
   train_BCol,                    test_BCol                    =
   BCol_estand.fit_transform(train_BCol, test_BCol)
   ```

[25] Ver Anexo A link hacia el repositorio github para observar el código implementado en todo el proceso, incluyendo la carga de las bases de datos, proceso exploratorio y transformación de variables, y la construcción de las distintas funciones y clases que hacen posible la implementación de los dos algoritmos de ML y DL.
[26] Ver Figura 2.9 para observar cómo se pasa de una serie uni-variable a una función con una variable dependiente y 5 variables independientes que se constituyen en sus rezagos.
[27] Ver Anexo A link hacia repositorio github para más detalles acerca de la construcción de esta clase de objeto de Python y sus diferentes métodos.

```
train_BMA, test_ BMA = BMA_estand.fit_transform(train_ BMA,
test_ BMA)
```

5. Obtención del *set* de entrenamiento y *test* para las variables independientes y dependiente respectivamente de cada base de datos: BCol y BMA.

```
#BCol
x_train_BCol,           y_train_BCol                        =
train_BCol.drop(columns=['Precio  Objetivo']),  train_BCol['
Precio Objetivo ']
x_test_BCol, y_test_BCol  = test_BCol.drop(columns=[' Precio
Objetivo ']), test_BCol[' Precio Objetivo ']

#BMA
x_train_BMA,  y_train_BMA  =  train_BMA.drop(columns=['Precio
Objetivo']), train_BMA[' Precio Objetivo ']
x_test_BMA,  y_test_BMA   =  test_BMA.drop(columns=['  Precio
Objetivo ']), test_BMA[' Precio Objetivo ']
```

6. Una vez se han realizado todos los pasos previos y se ha procedido a determinar el valor óptimo de los hiperparámetros[28], se entrena el algoritmo con estos hiperparámetros.

```
#BCol
svr_BCol_optimizado      =      SVR(kernel='rbf',      C=0.5,
epsilon=0.0001, gamma=0.001).fit(x_train_BCol, y_train_BCol)

#BMA
svr_BMA_optimizado      =      SVR(kernel='rbf',      C=0.5,
epsilon=0.0001, gamma=0.001).fit(x_train_BMA, y_train_BMA).
```

7. Habiendo entrenado los respectivos modelos, se procede a la predicción de cada uno de ellos.
   Como se explicó anteriormente, la clase *Standardize* posee un método llamado *inverse_y* el mismo que permite convertir los valores estandarizados a valores en unidades originales, dólares estadounidenses.

```
#Prediccion BCol
prediccion_BCol = svr_BCol_optimizado.predict(x_test_BCol)
prediccion_BCol = BCol_estand.inverse_y(prediccion_BCol)
```

[28] Ver Anexo A link hacia repositorio github para más detalles acerca del proceso de optimización de hiperparámetros.

#Prediccion BMA
```
prediccion_BMA = svr_BMA_optimizado.predict(x_test_BMA)
prediccion_BMA = BMA_estand.inverse_y(prediccion_BMA)
```

8. Se procede a transformar el objeto predicción_BCol y predicción_BMA en un *dataframe* de Python para poder comparar los valores reales y pronosticados por SVM.

#BCol
```
train = BCol_os[:len(train_BCol)]
test_BCol = BCol_os[len(train_BCol):]
test_BCol['Precio de Cierre Pronosticado'] = prediccion_BCol
```

#BMA
```
train = BMA_os[:len(train_BMA)]
test_BMA = BMA_os[len(train_BMA):]
test_BMA['Precio de Cierre Pronosticado'] = prediccion_BMA
```

9. Finalmente se comparan los valores reales y pronosticados para el periodo 15 de marzo de 2023 y 31 de marzo 2023.

#BCol
```
test_BCol.loc['2023-03-15':'2023-03-31', ['Precio de Cierre
Real', 'Precio de Cierre Pronosticado']].round(2)
```

#BMA
```
test_BMA.loc['2023-03-15':'2023-03-31', ['Precio de Cierre
Real', 'Precio de Cierre Pronosticado']].round(2)
```

LSTM.

Los pasos implementados en SVM también se ha implementado para LSTM a excepción del 4, 5, 6, y 8 en donde se ha cambiado levemente la clase *Standardize* con el objetivo de contar con un *set* de validación que permita comparar el valor de perdida en este set con el *set* de entrenamiento y de esta forma evitar la presencia de *overfitting* en el modelo. A más de este cambio, se debe tomar en cuenta que el entrenamiento de LSTM difiere del SVM, como se verá a continuación:

4. Estandarización de las dos bases de datos y división en set de entrenamiento, test y validación.

#BCol

```
train_BCol, test_BCol, val_BCol = BCol_estand.fit_transform()
```

#BMA

```
train_BMA, test_BMA, val_BMA = BMA_estand.fit_transform()
```

5. Obtención del set de entrenamiento, validación y test para las variables independientes y dependiente respectivamente haciendo uso de la función *feature_target_ts*[29] la cual toma como argumentos a los objetos obtenidos en el paso 4, estos son: *train, test, val* para BCol y BMA respectivamente.

#BCol

```
(y_train_BCol, y_val_BCol, y_test_BCol, x_train_BCol, x_val_BCol, x_test_BCol)    =    features_target_ts(train_BCol,    val_BCol, test_BCol)
```

#BMA

```
(y_train_BMA,  y_val_BMA,  y_test_BMA,  x_train_BMA,  x_val_BMA, x_test_BMA) = features_target_ts(train_BMA, val_BMA, test_BMA)
```

6. Entrenamiento de LSTM.

#BCol

```
BCol_Optimizado.compile(optimizer='adam', loss='mean_squared_error')
```

```
BCol_Optimizado.fit(x_train_BCol, y_train_BCol, batch_size = 32, epochs = 83)
```

#BMA

[29] Ver Anexo A link hacia repositorio github para la respectiva referencia.

```
BMA_Optimizado.compile(optimizer='adam',
loss='mean_squared_error')

BMA _Optimizado.fit(x_train_ BMA, y_train_ BMA, batch_size = 32,
epochs = 120)
```

8. Construcción del *dataframe* teniendo en cuenta el *set* de validación para poder comparar los valores reales y pronosticados por LSTM.

#BCol

```
train = BCol_os[:len(train_BCol)]

val = BCol_os[len(train_BCol):-len(test_BCol)]

test_BCol = BCol_os[-len(test_BCol):]

test_BCol['Precio de Cierre Pronosticado'] = prediccion_BCol
```

#BMA

```
train = BMA_os[:len(train_BMA)]

val = BMA_os[len(train_BMA):-len(test_BMA)]

test_BMA = BMA_os[-len(test_BMA):]

test_BMA['Precio de Cierre Pronosticado'] = prediccion_BMA
```

2.4.2 Generación de técnicas de testeo.

Para la fase de evaluación de los dos algoritmos utilizados, se hace uso de las 3 métricas de error descritas en la sección de marco teórico, RMSE, MAPE y MAE.

#Codigo implementado en *Google Colab.*

Previo a la construcción de las métricas de error se procede a importar las librerías necesarias.

```
from sktime.performance_metrics.forecasting
import(MeanAbsolutePercentageError, MeanSquaredError,
MeanAbsoluteError)
```

En un Segundo paso, se procede a instanciar cada función importada.

```
mape = MeanAbsolutePercentageError()

mse = MeanSquaredError()
```

```
mae = MeanAbsoluteError()
```

2.4.2.1. Mean Absolute Percentage Error (MAPE).

#Se construye cada métrica de error en función de los valores reales, *y_test*, y los valores pronosticados, prediccion_BCol y prediccion_BMA.

#BCol

```
mape_BCol = mape(y_test_BCol, prediccion_BCol)
```

#BMA

```
mape_BMA = mape(y_test_BMA, prediccion_BCol)
```

2.4.2.2. Root Mean Squared Error (RMSE).

#Al no existir la función RMSE, se procede en un primer paso a calcular el *MSE* para posteriormente obtener la raíz cuadrada del valor, lo cual como se vio en el apartado de marco teórico representa el RMSE.

#BCol

```
rmse_BCol = np.sqrt(mse(y_test_BCol, prediccion_BCol))
```

#BMA

```
rmse_BMA = np.sqrt(mse(y_test_BMA, prediccion_BMA))
```

2.4.2.3. Mean Absolute Error (MAE).

#BCol

```
mae_BCol = mae(y_test_BCol, prediccion_BCol)
```

#BMA

```
mae_BMA = mae(y_test_BMA, prediccion_BCol)
```

2.4.3 Construcción del modelo.

En esta fase, una vez que las dos bases de datos que presentan el precio cierre ajustado de las acciones del BMA y BCol están listas para ser entrenadas en los algoritmos LSTM y SVM respectivamente, se procede a la construcción de las diferentes arquitecturas a usar en los dos algoritmos mencionados.

Para ambos bancos se trabajó con un set de entrenamiento equivalente al 70% del total de la base de datos, y el 30% restante fue utilizado para el set de testeo, ver Tabla 2.6.

Set de entrenamiento y testeo para ambas empresas.	
Set de entrenamiento	2995 (70%)
Set de testeo	1284 (30%)

Tabla 2.6. Set de Entrenamiento y Testeo para los dos bancos analizados.

2.4.3.1. SVM.

En cuanto a la implementación del SVM se debe mencionar que se trabajó con distintos valores en sus diferentes hiperparámetros.

Support Vector Regression			
Valores para los distintos hiperparámetros			
C	*Epsilon*	*Gamma*	*Kernel*
0.5	0.0001	0.001	Lineal
1	0.01	0.1	Radial Basis Function (RBF)
10	0.1	10	Polinomial (Poly)
500	1		

Tabla 2.7. Hiperparámetros utilizados para el SVR.

Como se observa en la Tabla 2.7, se toman en cuenta 4 distintos valores para el hiperparámetro C, mientras que para los distintos hiperparámetros, *Epsilon*, *Gamma* y kernel, se probó con 3 diferentes valores. Durante la fase de entrenamiento se prueba cada uno de estos valores, para cada hiperparámetro y se registra el valor del error de predicción.

En lo referente al SVM, se trabajó como sigue:

- Se utilizó la función *GridSearchCV* de *Scikit-Learn* para encontrar los valores óptimos de los hiperparámetros: C, *gamma*, *épsilon*; y así también para el tipo de kernel más adecuado.
- Para la estandarización y división de la base de datos en set de entrenamiento y testeo, se construyó una clase denominada *Standardize*[30] la cual aplica todos estos procesos en un solo paso para ambas bases de datos.
- Se construyó una función llamada *one_step_forecast* la cual toma como argumento la venta temporal que para el entrenamiento de ambas bases de datos tiene el valor de 5. De esta forma, se logra representar la relación que tiene el precio de un cierto día de la semana con el precio del mismo día de la siguiente semana, por ejemplo, de lunes a lunes, martes a martes, etc.
- Una vez que se encontraron los valores óptimos para cada uno de los hiperparámetros se procedió a entrenar al SVM con estos valores tanto para el BCol como para el BMA para posteriormente realizar el respectivo pronóstico.
- Finalmente, se listaron en forma descendente todos las diferentes arquitecturas trabajadas con el SVR en función de las 3 métricas de error RMSE, MAPE y MAE.

2.4.3.2. LSTM

Al utilizar el LSTM también se trabajó con distintas arquitecturas como: Número de capas, número neuronas en las distintas capas, algoritmo optimizador, función de activación y el tamaño del *batch*.

Long Short-Term Memory				
Valores para los distintos hiperparámetros				
Número de capas Ocultas	*Algoritmo Optimizador*	*Tamaño Batch*	*Número de Neuronas por capa oculta*	*Función de Activación*
1	Adam	16	20	
2	SGD	32	32	Tangencial
3	RMSProp	64	70	ReLU
5	AdaGrad			

Tabla 2 8. Hiperparámetros utilizados para el LSTM.

La Tabla 2.8 detalla los hiperparámetros y los distintos valores de estos con los que se trabaja en la fase de entrenamiento. Es así como, para el caso de número de capas ocultas, se tiene un valor entre 1 y 5. En lo que respecta a los algoritmos optimizadores, se entrenó a la red con: *Adam, SGD with momentum, RMSProp, AdaGrad*. Al igual que en el caso con SVM, se entrenó al LSTM con el valor de cada uno de estos hiperparámetros, registrándose el error en el pronóstico para cada caso.

[30] Ver Anexo A para apreciar el respectivo código.

En lo referente al modelo de redes neuronales recurrentes (LSTM), se trabajó como sigue:

- Para el cálculo del tamaño del batch se empezó con un numero de 32 unidades, para posteriormente probar el valor óptimo de este hiperparametro mediante la función *GridSearch* de *Sklearn*, entrenando las distintas arquitecturas del LSTM con valores del batch que se detallan en la tabla 2.8.
- Para obtener el número de épocas con las que entrenar a los dos bases de datos, se hizo uso del método *EarlyStopping*[31] el cual evalúa el valor de pérdida del modelo y por ende su rendimiento en cada iteración y en caso de que este valor de perdida no disminuya, el entrenamiento se detiene indicando el número de épocas necesarias para obtener cierto rendimiento. Esta característica permite a la red neuronal no sobre entrenarse, y por lo tanto, se limita la existencia de *overfitting*. El valor máximo para el numero de épocas a entrenar haciendo uso de esta función es de 500[32].
- En lo que respecta a los algoritmos optimizadores, cuatro fueron probados: *Adam, RMSProp, SGD with momentum, Adadelta*. En cada una de estas configuraciones, se utilizó el valor por *default* de los demás hiperparámetros. Es decir, cuando entrenó a la red con el optimizador *Adam*, no se modificó el valor del tamaño del *Batch*, numero de neuronas ocultas, etc. Pues se utilizó el valor por *default.* Esta metodología se repitió para todos los optimizadores.
- Se trabajó con un distinto número de capas ocultas, empezando desde 1 para posteriormente ir incrementando esta cifra hasta 5, observando el rendimiento del modelo en cada caso.

Habiéndose descrito en este capítulo todos los pasos del modelo CRISP-DM en donde se ha detallado todo el análisis estadístico descriptivo implementado, exploración y verificación de la calidad de los datos, transformación de los datos, fase de modelamiento, implementación de metricas de error como mecanismo de evaluación de la calidad de pronósticos de los distintos algoritmos, en el siguiente capítulo se describen los resultados obtenidos una vez se ha cumplido con todos estos procedimientos previos.

[31] Ver Anexo A donde se encuentra link hacia el github del autor con los códigos respectivos.

[32] Ver Anexo B en donde se analiza el valor perdido en el set entrenamiento y validación a través de sus respectivas curvas.

3. Resultados

En esta sección se resumen los principales resultados encontrados luego de haber puesto en práctica la metodología y el procedimiento descritos en la sección anterior.

Luego de haber transitado por cada una de las distintas fases previas las cuales permitieron obtener una base de datos con el formato requerido para los algoritmos estudiados, se procede a describir los principales resultados obtenidos.

3.1 LSTM.

El LSTM presenta varios hiperparámetros que pueden ser modificados, posibilitando el poder construir un mejor modelo final en términos de capacidad predictiva.

Para el efecto, en esta sección se detallan todos los hiperparámetros que fueron probados, así como los distintos valores de las métricas de error que se obtuvieron con las distintas configuraciones. Esto último, con el objetivo final de determinar la mejor arquitectura de LSTM en el pronóstico de las acciones del BCol y BMA.

	RMSE	MAPE	MAE
BCol. LSTM Dos capas ocultas	0.066810	84.406349	0.050627
BCol. LSTM Una capa oculta	0.067276	84.707359	0.050597
BCol. LSTM Tres capas ocultas	0.097932	181.803085	0.081001
BCol. LSTM Cinco capas ocultas	0.147078	183.845520	0.112117

Tabla 3 1. LSTM con diferentes número de capas ocultas y las respectivas métricas de error usadas para el BCol.

El primer parámetro que se probó en el LSTM ha sido el número de capas ocultas existentes en la red. La Tabla 3.1 permite visualizar cada una de las cuatro configuraciones usadas para el efecto y estas van desde una capa oculta hasta cinco capas ocultas, omitiendo el numero de 4 para no caer en redundancias. Para cada una de ellas se capturó el valor de las 3 métricas de error utilizadas en el presente trabajo de desarrollo, y se obtuvo como resultado que la configuración por el número de capas ocultas que mayor poder de predicción logra para el BCol es la de dos capas ocultas seguida muy de cerca de una capa. Asimismo, es posible observar que la configuración con 5 capas ocultas presentó el peor resultado pues las tres métricas de error presentar los mayores valores en comparación con las configuraciones con menor número de capas ocultas, ver tabla 3.1.

Siguiendo este mismo procedimiento para el BMA, se obtuvieron los resultados que se muestran en la Tabla 3.2.

	RMSE	MAPE	MAE
BMA. LSTM Una capa oculta	0.027008	2.771262	0.020851
BMA. LSTM Tres capas ocultas	0.050934	5.419922	0.042688
BMA. LSTM Dos capas ocultas	0.066348	7.811426	0.062028
BMA. LSTM Cinco capas ocultas	0.185169	22.411730	0.179174

Tabla 3 2. LSTM con diferentes número de capas ocultas y las respectivas métricas de error usadas para el BMA.

Se observa que la configuración por número de capas ocultas con mejor capacidad de pronóstico es la de una capa, seguida por la de tres, dos y cinco. Si bien el rendimiento del LSTM en las dos bases de datos trabajadas no permite identificar exactamente el hecho de que si una sola capa oculta es capaz de identificar de una manera casi precisa los patrones que toma cierta variable a través del tiempo, lo que si se presenta como una patrón más 'sencillo' de identificar es el hecho de que contar con 5 capas ocultas en los modelos de LSTM para los dos bancos aquí estudiados obtiene un relativo bajo poder de pronóstico.

Como lo explican Sebastian Raschka, Yuxi Liu y Vahid Mirjalili [9], contar con un mayor número de capas ocultas lleva a que el valor de los gradientes sea extremadamente pequeño al tiempo que se propagan a través de la red durante la fase de entrenamiento. A más de esto se debe considerar que ante una mayor cantidad de capas ocultas existe un mayor número de parámetros que aprender para la red neuronal lo cual lleva a que esta carezca de poder de generalización y por ende se produzca el *overfitting* [9].

En la Tabla 3.3 se presentan los distintos resultados obtenidos con 4 distintos algoritmos optimizadores: *Adam, SGD with momentum, Adadelta* y *RMSprop* para el BCol.

	RMSE	MAPE	MAE
BCol. Optimizador Adam	0.069329	92.751305	0.052639
BCol. Optimizador RMSprop	0.076333	135.089859	0.058344
BCol. Optimizador Adagrad	0.155179	209.438782	0.121238
BCol. Optimizador SGD	0.179886	340.239288	0.155228

Tabla 3 3. LSTM con diferentes tipos de algoritmo optimizador y las respectivas métricas de error usadas para el BCol.

El algoritmo optimizador que permitió obtener el menor valor en términos de las 3 métricas de error utilizadas es *Adam* seguido de cerca por *RMSprop, Adagrad* en tercero y por último *SGD.*

	RMSE	MAPE	MAE
BMA. Optimizador Adam	0.040267	4.369713	0.033564
BMA. Optimizador Adagrad	0.042225	4.466066	0.033552
BMA. Optimizador SGD	0.044946	4.702649	0.037774
BMA. Optimizador RMSprop	0.064034	7.235543	0.058211

Tabla 3.4. LSTM con diferentes tipos de algoritmo optimizador y las respectivas métricas de error usadas para el BMA.

Para el caso del BMA, el optimizador Adam también obtuvo los mejores resultados, como se muestra en la Tabla 3.4, al presentar el menor valor en las 3 métricas de error. Aunque es importante mencionar que los 4 optimizadores utilizados han permitido obtener resultados relativamente óptimos.

Sebastian Raschka, Yuxi Liu y Vahid Mirjalili [9] argumentan que los optimizadores Adam y RMSprop son versiones mejoradas del *Gradient Descent* y todas sus variantes debido a que ambos mitigan el problema de la variación de los gradientes, el cual se basa en que una proporción de gradientes tiene un valor pequeño y otros un valor grande. *Adam* y *RMSprop* adaptan la tasa de aprendizaje y reducen significativamente el problema del decrecimiento de la tasa de aprendizaje presente en otros tipos de algoritmos optimizadores.

Debido a las características del algoritmo optimizador *Adam* descritas en la sección de teoría y a los resultados obtenidos luego de entrenar las distintas configuraciones del LSTM en base a los diferentes optimizadores, ver Tablas 3.3 y 3.4, el pronóstico del precio de las acciones para los dos bancos se hará utilizando el mencionado algoritmo optimizador.

	RMSE	MAPE	MAE
BCol. Tamaño Batch 32	0.072059	123.718918	0.055249
BCol. Tamaño Batch 64	0.072809	92.683426	0.055507
BCol. Tamaño Batch 16	0.074175	126.845627	0.057565

(a)

	RMSE	MAPE	MAE
BMA. Tamaño Batch 32	0.029975	3.084904	0.023285
BMA. Tamaño Batch 16	0.039240	4.201008	0.032778
BMA. Tamaño Batch 64	0.063411	6.755686	0.055093

(b)

Tabla 3.5. LSTM con diferente tamaño de batch y las respectivas métricas de error usadas para el BCol (a) y BMA (b).

Distintos valores para el tamaño del *Batch*, entre 16, 32 y 64, son observables en la Tabla 3.5. Obteniendo el mejor resultado para la configuración de 32 unidades tanto para el caso del BCol (a) y BMA (b).

De esta forma se comprueba que un tamaño de *Batch* de 32 unidades es el óptimo para realizar el pronóstico respectivo.

	RMSE	MAPE	MAE
BCol Numero de Neuronas 32	0.072059	123.718918	0.055249
BCol. Numero de Neuronas 70	0.077201	135.747101	0.060334
BCol. Numero de Neuronas 20	0.079433	144.418182	0.062268

	RMSE	MAPE	MAE
BMA. Numero de Neuronas 32	0.029975	3.084904	0.023285
BMA. Numero de Neuronas 70	0.042895	4.638610	0.035615
BMA. Numero de Neuronas 20	0.058953	6.483890	0.052607

(a) (b)

Tabla 3.6. LSTM con diferente número de neuronas en las capas ocultas y las respectivas métricas de error usadas para el BCol (a) y BMA (b).

Al incrementarse el número de neuronas en las capas internas de la red se incrementa también el número de parámetros a optimizar [14]. Esta última aseveración se puede constatar parcialmente en la Tabla 3.6 en donde se divisa que el número de neuronas optimo en cada capa oculta de la red es de 32. Aunque también es importante enfatizar que en un principio al incrementarse el número de neuronas también se mejora el rendimiento del modelo. Pero al ser este número 70, el poder predictivo del LSTM decrece considerablemente en comparación con la configuración de 32 neuronas tanto para el caso del BCol y BMA.

Finalmente, se ha entrenado a la red con dos distintas funciones de activación, estas son la ReLU y la Tangencial hiperbólica, ambas descritas en el apartado teórico.

	RMSE	MAPE	MAE
BCol Funcion Activacion Tanh	0.078946	141.200699	0.061695
BCol. Funcion Activacion ReLU	0.099690	170.068710	0.079503

(a)

	RMSE	MAPE	MAE
BMA. Funcion Activacion Tanh	0.025511	2.665313	0.020018
BMA. Funcion Activacion ReLU	0.048125	5.187182	0.041345

(b)

Tabla 3.7. LSTM con función de activación Tanh y ReLU y las respectivas métricas de error usadas para el BCol (a) y BMA (b).

En Tabla 3.7 (a) y (b) es posible visualizar que la función de activación Tangencial Hiperbólica obtiene el mejor rendimiento en términos predictivos tanto para el BCol como para BMA. Las características de esta función, como se detalló en el apartado teórico, permiten que los gradientes no estén restringidos a moverse en cierta dirección lo cual da paso a que se obtenga el menor valor en todas las métricas de error usadas para el efecto.

Una vez se ha procedido a identificar los mejores hiperparámetros en términos del valor de las métricas de error más pequeñas para las distintas configuraciones anteriormente analizadas, en la Tabla 3.8 se detalla el valor de estos.

LSTM				
Hiperparámetros Optimizados				
Número de capas ocultas	*Algoritmo Optimizador*	*Tamaño del Batch*	*Numero de Neuronas por capa oculta*	*Función de Activación*
2	Adam	32	32	Tangencial Hiperbólica

(a)

LSTM				
Hiperparámetros Optimizados				
Número de capas ocultas	*Algoritmo Optimizador*	*Tamaño del Batch*	*Numero de Neuronas por capa oculta*	*Función de Activación*
1	Adam	32	32	Tangencial Hiperbólica

(b)

Tabla 3.8. Valor de los hiperparámetros óptimos para BCol (a) y BMA (b).

Analizando la Tabla 3.8, se concluye que el valor y tipo de hiperparámetro óptimo tanto para el BCol, en (a), como para el BMA en (b), son los mismos a excepción del número de capas ocultas. Estos resultados permiten confirmar la robustes de los resultados aquí obtenidos y a la vez hacen

posible realizar el proceso de pronóstico con menor grado de incertidumbre sobre los resultados a obtener en este. Sobre todo, en el caso que se desee utilizar los modelos aquí entrenados para pronosticar en otro tipo de datos.

En la Tabla 3.9 se presentan los valores reales y pronosticados para el caso del BCol haciendo uso de los hiperparámetros presentados en la tabla 3.8 (a). El periodo del pronóstico corresponde del 15 de marzo al 31 de marzo de 2023.

Fecha	Precio de Cierre Real	Precio de Cierre Pronosticado
2023-03-15	21.06	21.100000
2023-03-16	20.54	20.900000
2023-03-17	20.85	20.400000
2023-03-20	21.54	20.700001
2023-03-21	22.02	21.400000
2023-03-22	22.56	21.900000
2023-03-23	23.11	22.400000
2023-03-24	23.54	23.000000
2023-03-27	24.16	23.400000
2023-03-28	24.22	24.100000
2023-03-29	24.25	24.200001
2023-03-30	24.32	24.200001
2023-03-31	24.54	24.299999

Tabla 3.9. Precio de cierre ajustado real y pronosticado en USD de las acciones del BCol. LSTM.

Se puede observar en la Tabla 3.9 que el modelo obtiene muy buenos resultados, pues los valores pronosticados están muy cercanos a los reales. Tomando como ejemplo la primera y la última observación, esto es 15 de marzo y 31 de marzo, la diferencia entre valor real y valor pronosticado es de USD 0,04 y USD 0,25 respectivamente.

Fecha	Precio de Cierre Real	Precio de Cierre Pronosticado
2023-03-15	15.97	16.070000
2023-03-16	15.54	16.020000
2023-03-17	15.53	15.620000
2023-03-20	16.46	15.470000
2023-03-21	16.60	16.190001
2023-03-22	16.95	16.570000
2023-03-23	16.95	16.900000
2023-03-24	16.25	16.950001
2023-03-27	16.81	16.360001
2023-03-28	16.34	16.620001
2023-03-29	16.45	16.370001
2023-03-30	16.23	16.370001
2023-03-31	16.43	16.209999

Tabla 3.10. Precio de cierre ajustado real y pronosticado en USD de las acciones del BMA. LSTM.

Al igual que en el caso del BCol, Cuando se observa en la Tabla 3.10, LSTM logra pronosticar el precio de las acciones del BMA con bastante precisión, ya que valores reales y estimados son muy similares. Por ejemplo, para el 21 de marzo el precio real fue de USD 16,6 y el precio pronosticado fue de USD 16,19, siendo la diferencia de USD 0,39.

Para obtener una mejor perspectiva del poder predictivo del LSTM, se presenta la Figura 3.1 que permite visualizar tanto el valor real como el pronosticado para BMA.

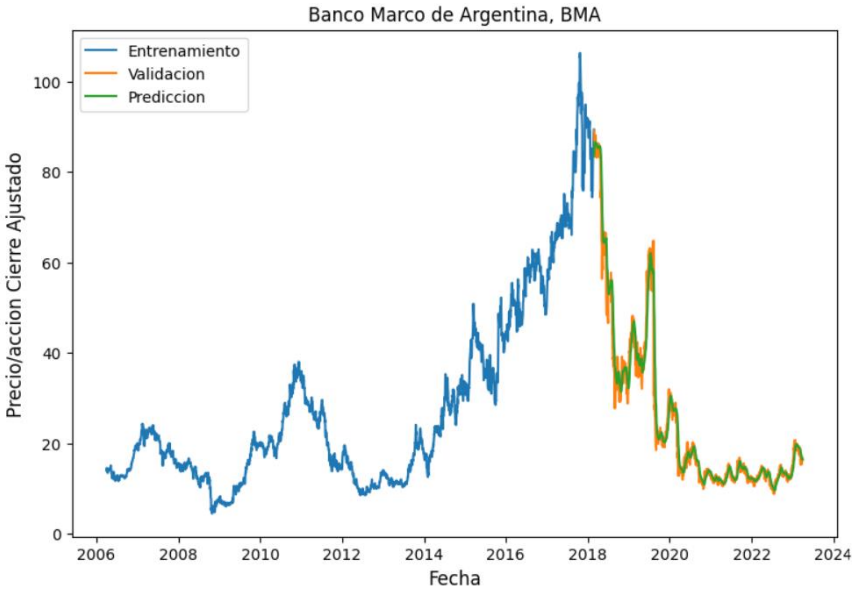


Figura 3.1. Modelo LSTM para el BMA. Valores reales y pronosticados.

Es posible observar que el LSTM logra reproducir bastante bien la trayectoria del precio de las acciones del BMA por cuánto la línea de color verde que representa los valores pronosticados se encuentra sobre la línea de color amarillo que representa los valores reales.

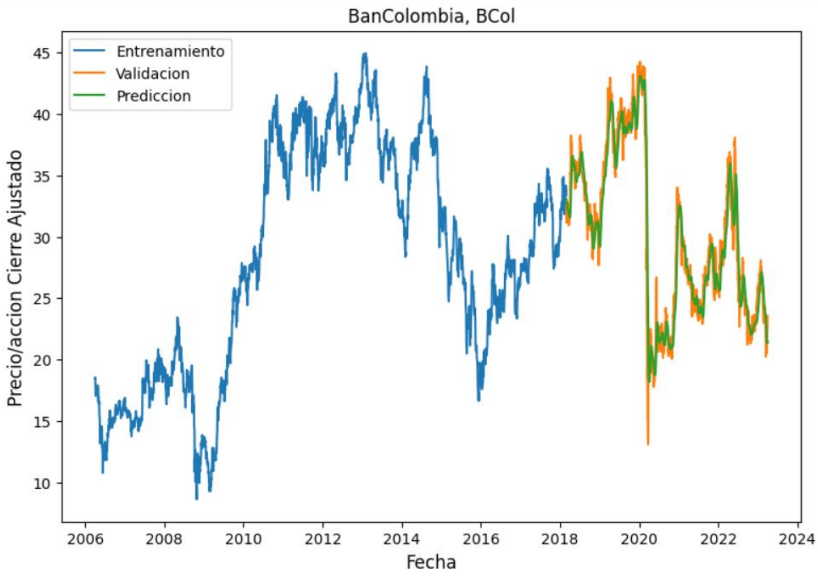


Figura 3.2. Modelo LSTM para el BCol. Valores reales y pronosticados.

La Figura 3.2 visualiza los valores reales y pronosticados por el LSTM para el BCol. Se observa que la red neuronal hace un muy buen trabajo al estimar la trayectoria de los valores reales, de nuevo la curva de valores pronosticados esta por sobre la curva de valores reales. Considerando además que, durante el 2020, año de la pandemia del COVID-19, el LSTM no pudo pronosticar una considerable caída en el precio de la acción del banco en cuestión, lo cual podría indicar una relativa desventaja de este modelo.

Nombre del Modelo	RMSE	MAPE	MAE
LSTM optimizado	0.072	123.72	0.055

(a)

Nombre del Modelo	RMSE	MAPE	MAE
LSTM optimizado	0.04	4.37	0.033

(b)

Tabla 3 11. Valor de las metricas de error usando LSTM con hiperparámetros optimizados para BCol (a) y BMA (b).

La Tabla 3.11 muestra el valor de las 3 métricas de error, habiendo entrenado a los datos de los dos bancos con el LSTM con hiperparámetros optimizados. Al observar los valores de estas métricas para BMA, (b), se concluye que el hecho de que el precio de las acciones de este banco haya mostrado una mayor volatilidad en el periodo de tiempo analizado no ha llevado a que el LSTM pueda pronosticar eficientemente el precio de cierre ajustado para el banco en cuestión.

Por último, se presentan las Figuras 3.3 y 3.4 en donde se visualiza las curvas de entrenamiento y validación en base al número de épocas.

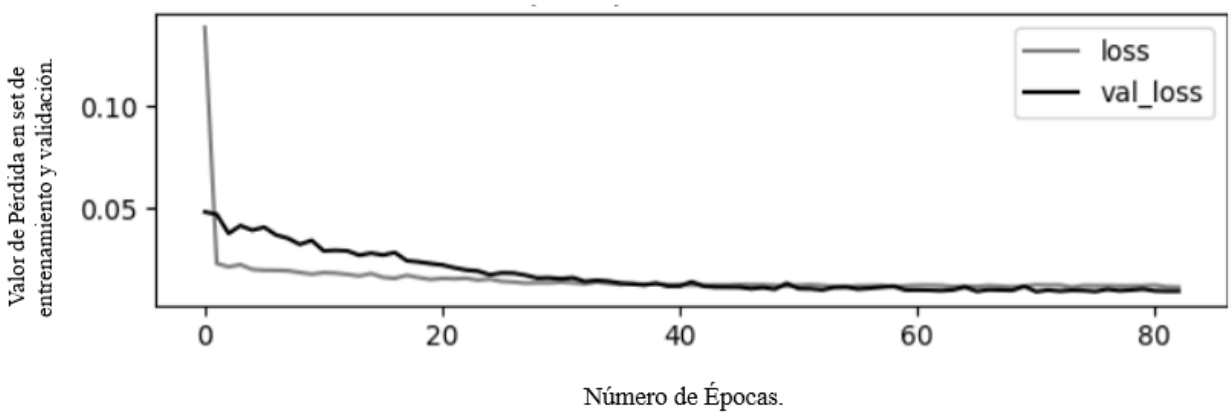


Figura 3.3. Valor de Pérdida en set de entrenamiento y validación por época de entrenamiento para BCol.

Según lo observado en la Figura 3.3, la red neuronal LSTM entrenó hasta aproximadamente la época 82 en la base de datos correspondiente al BCol y al no existir una mejora en el rendimiento del modelo, es decir la función de perdida no disminuye, el entrenamiento se detiene.

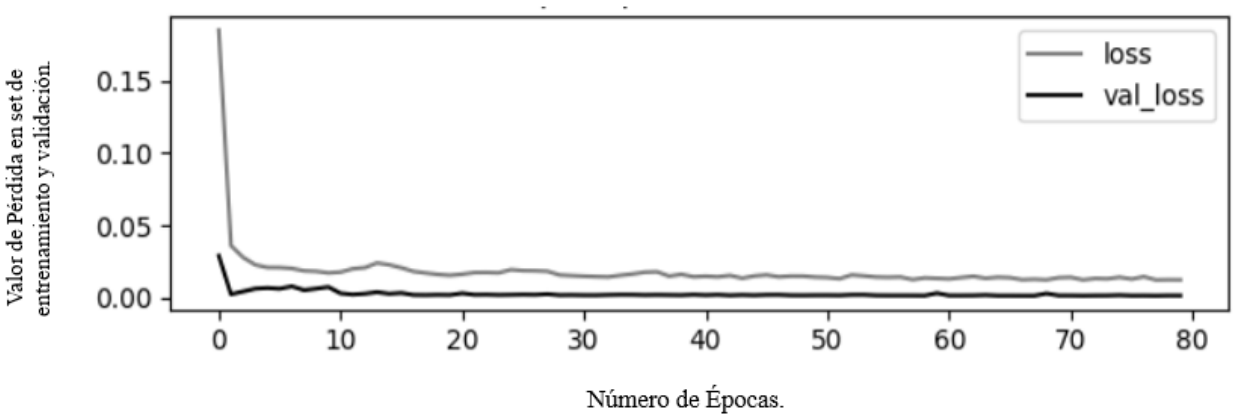


Figura 3.4. Valor de Pérdida en set de entrenamiento y validación por época de entrenamiento para BMA.

La Figura 3.4 permite visualizar que la red neuronal LSTM entrenó hasta aproximadamente la época 79 en la base de datos correspondiente al BMA y al no existir una mejora en el rendimiento del modelo, el entrenamiento se ha detenido. Esta característica de las redes neuronales no permite la existencia de overfitting [14].

El objetivo es minimizar la función de pérdida, y tanto en las Figura 3.3 y 3.4 se observa que la pérdida en el set de entrenamiento y la pérdida en el set de validación van disminuyendo lentamente y no separándose la una de la otra, lo cual en caso de separarse sería una clara señal de existencia de overfitting [5].

3.2. SVM.

En lo referente al SVM, como se detalló en el apartado número 2 del presenta trabajo de desarrollo, se probaron diferentes valores con 3 hiperparámetros: C, *Epsilon* y *Gamma*, los cuales dieron distintos resultados para cada uno de los kernels: Lineal, '*RBF*', '*Poly*'. A continuación, se presenta el valor para las 3 métricas de error.

Nombre del Modelo	RMSE	MAPE	MAE
Kernel Radial con hiperparam. optimizados 2	0.072938	0.291227	0.051258
Kernel Lineal con hiperparam. optimizados	0.072947	0.290177	0.051206
Kernel Radial con hiperparam. optimizados	0.072990	0.291898	0.051280
Kernel Poly Grado 1 con hiperparam. optimizados	0.072991	0.290659	0.051264
Kernel Lineal con hiperparam. valor default	0.073528	0.291811	0.051548
Kernel Radial con hiperparam. valor default	0.075435	0.298938	0.052840
Kernel Radial con hiperparam. optimizados 3	0.108790	0.447401	0.075924
Kernel Poly Grado 1 con hiperparam. optimizados 2	0.128128	0.473566	0.093226
Kernel Poly Grado 3 con hiperparam. valor default	0.398147	0.939451	0.350330
Kernel Poly Grado 3 con hiperparam. optimizados	0.536988	0.873309	0.475984
Kernel Poly Grado 2 con hiperparam. optimizados	1.079762	1.169649	0.777210

Tabla 3.12. Diferentes configuraciones del SVM y las 3 métricas de error usadas: RMSE, MAPE y MASE para el BCol.

En la Tabla 3.12 es posible visualizar varios puntos que es importante mencionar:

- Todos los modelos con hiperparámetros optimizados obtienen mejores resultados, a excepción del kernel polinomial grado 2 y 3, que los valores por *default*.
- Los modelos que obtuvieron el mejor poder de predicción han sido el kernel optimizado Lineal, Polinomial grado 1 y Radial. La diferencia en poder predictivo entre ellos es casi inexistente, pues las 3 métricas de error muestran resultados prácticamente idénticos.
- El kernel Polinomial tiene un grado de 3 por *default*, por lo que al entrenar la base de datos del BCol con este modelo se obtienen muy malos resultados tanto para valores optimizados de los hiperparametros como para los *default*.
- Si bien los kernels Radial optimizados 2[33] y 1, Lineal y Polinomial Grado 1 obtuvieron el mejor poder de predicción, el valor del hiperparámetro *C* fue grande, 500, lo cual influye inversamente en el poder de generalización del modelo. Hay que recordar que como se vio en la sección teórica, si el valor de este hiperparámetro es relativamente grande[34] el modelo no va a poder generalizar lo cual significa que no podrá pronosticar correctamente datos nuevos para los cuales no fue entrenado, traduciéndose en un modelo con *overfitting*.

[33] Ver tabla 3.13 (a).
[34] El valor default de este hiperparámetro es 1.

- Teniendo en cuenta este último punto, se ha trabajado con un valor para *C* de 0.5, manteniendo constante el valor de los demás hiperparámetros, para controlar el *overfitting*. Los dos modelos con esta característica son kernel Radial con hiperparámetros optimizados 3[35], y kernel Polinomial Grado 1 con hiperparámetros optimizados 2. Al observar la tabla 3.12 se puede concluir que, de las dos configuraciones anteriormente descritas, es la del kernel Radial la que logra conseguir el menor valor de error. Por lo tanto, se hará referencia a este último.

SVM			
Hiperparametros Optimizados			
Hiperparametro C	*Hiperparametro Epsilon*	*Hiperparametro Gamma*	*Kernel*
500	0.0001	0.001	Radial

SVM			
Hiperparametros Optimizados reduciendo *overfitting.*			
Hiperparametro C	*Hiperparametro Epsilon*	*Hiperparametro Gamma*	*Kernel*
0.5	0.0001	0.001	Radial

Tabla 3.13. Valor de hiperparámetros optimizados del SVM para el BCol. (a) C valor de 500. (b) C valor de 0.5.

La Tabla 3.13 permite conocer las cifras de los distintos hiperparámetros optimizados cuyo kernel es el Radial para el BCol. Como se explicó en el párrafo anterior, se trabajará con el valor más pequeño del hiperparámetro *C*, para evitar la presencia de *overfitting*. Por lo tanto, los valores observables en la tabla 3.13 (b) serán utilizados para realizar el respectivo pronóstico del precio de las acciones del BCol.

Siguiendo el mismo procedimiento anteriormente descrito, se obtuvieron los siguientes resultados para el BMA que pueden ser observados en la Tabla 3.14.

[35] Ver Tabla 3.13 (b),

Nombre del Modelo	RMSE	MAPE	MAE
Kernel Lineal con hiperparam. optimizados	0.072751	0.155693	0.036271
Kernel Poly Grado 1 con hiperparam. optimizados	0.072779	0.155647	0.036269
Kernel Radial con hiperparam. optimizados	0.072875	0.157443	0.036267
Kernel Radial con hiperparam. optimizados 2	0.072910	0.157524	0.036259
Kernel Lineal con hiperparam. valor default	0.073156	0.149602	0.037888
Kernel Radial con hiperparam. valor default	0.083209	0.181855	0.049053
Kernel Radial con hiperparam. optimizados 3	0.095665	0.235729	0.051874
Kernel Poly Grado 1 con hiperparam. optimizados 2	0.121392	0.277788	0.078027
Kernel Poly Grado 3 con hiperparam. valor default	0.563120	1.502095	0.421756
Kernel Poly Grado 2 con hiperparam. optimizados	0.591500	1.805803	0.511925
Kernel Poly Grado 3 con hiperparam. optimizados	0.601267	1.506624	0.453439

Tabla 3.14. Diferentes configuraciones del SVM y las 3 métricas de error usadas: RMSE, MAPE y MASE para el BMA.

En la Tabla 3.14 es posible distinguir las siguientes características:

- Al igual que con el BCol, las configuraciones del SVM para el BMA que presentan mayor poder de predicción son los que tienen el kernel optimizado Lineal, Polinomial grado 1 y Radial con hiperparámetros optimizados obtienen mejores resultados que los valores por *default*.
- Los kernels que obtuvieron peores resultados han sido el polinomial grado 2 y grado 3, tanto con hiperparámetros optimizados y no.
- De igual forma que con los resultados obtenidos en BCol, la configuración del SVM para el BMA que permite contralar los negativos efectos del *overfitting* es el kernel Radial con valores optimizados 3.

SVM			
Hiperparametros Optimizados			
Hiperparametro C	*Hiperparametro Epsilon*	*Hiperparametro Gamma*	*Kernel*
500	0.01	0.001	Lineal

(a)

SVM			
Hiperparametros Optimizados reduciendo *overfitting.*			
Hiperparametro C	*Hiperparametro Epsilon*	*Hiperparametro Gamma*	*Kernel*
0.5	0.0001	0.001	Radial

(b)

Tabla 3 15. Valor de hiperparámetros optimizados del SVM para el BMA. (a) C valor de 500. (b) C valor de 0.5

La Tabla 3.15 permite conocer el valor de los distintos hiperparámetros optimizados con kernel Lineal, (a), y con kernel Radial, (b), para el BMA. Siguiendo el mismo principio anteriormente explicado sobre el *overfitting*, a pesar de que la configuración mostrada en (a) permite obtener el menor valor posible en las 3 métricas de error, se hará uso la configuración mostraba en (b). De esta forma, son estos los valores de los hiperparámetros y kernel utilizados para realizar el pronóstico del precio de las acciones del BMA.

En la Tabla 3.16 se observan los valores reales y pronosticados usando los valores de hiperparámetros y kernel observados en las Tablas 3.13 (b) y 3.15 (b) para el BCol y BMA respectivamente. El periodo corresponde entre el 15 de marzo y 31 de marzo del 2023.

Fecha	Precio de Cierre Real	Precio de Cierre Pronosticado
2023-03-15	21.06	20.82
2023-03-16	20.54	20.96
2023-03-17	20.85	20.81
2023-03-20	21.54	20.87
2023-03-21	22.02	21.09
2023-03-22	22.56	21.34
2023-03-23	23.11	21.70
2023-03-24	23.54	22.22
2023-03-27	24.16	22.75
2023-03-28	24.22	23.28
2023-03-29	24.25	23.68
2023-03-30	24.32	23.96
2023-03-31	24.54	24.16

Tabla 3 16. Precio de cierre ajustado real y pronosticado en USD de las acciones del BCol. SVM.

La Tabla 3.16 permite visualizar los valores reales, en la segunda columna Precio de Cierre Real, y los valores pronosticados, en la tercera columna Precio de Cierre Pronosticado, del BCol habiéndose entrenado con SVM y más específicamente SVR. A simple vista se divisa que el modelo hace un buen trabajo en pronosticar el precio de las acciones, pues los valores pronosticados y reales son semejantes. Si se toman en cuenta como ejemplo la primera y la última observación en la Tabla 3.16, 15 de marzo y 31 de marzo respectivamente, se ve que el error de pronóstico es relativamente pequeño pues para el primer caso es de USD 0,24 (21,06-20,82) y para el segundo es de USD 0,38 (24,54-24,1).

Una representación gráfica del comportamiento de los valores reales y pronosticados es posible observarla en la Figura 3.5. En ella, se puede observar el valor real, color amarillo, y el valor

pronosticado, color verde, del precio de las acciones del BCol. La trayectoria de ambas curvas es muy similar, demostrándose de esta manera la buena capacidad predictiva del SVM y sus hiperparámetros optimizados.

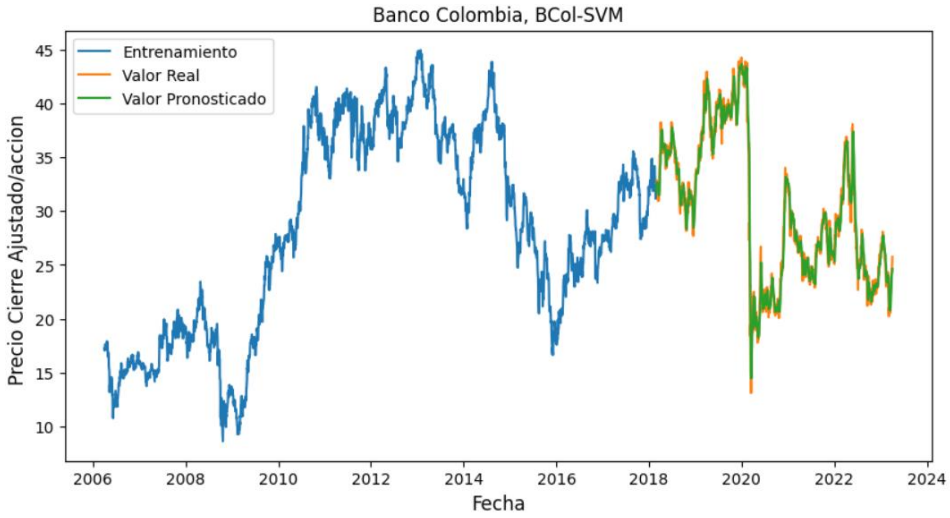


Figura 3.5. Modelo SVM para el BCol. Valores reales y pronosticados.

Siguiendo la misma metodología aplicada para el caso del BCol, se obtienen los siguientes resultados para el BMA.

Fecha	Precio de Cierre Real	Precio de Cierre Pronosticado
2023-03-15	15.97	15.80
2023-03-16	15.54	15.91
2023-03-17	15.53	15.72
2023-03-20	16.46	15.67
2023-03-21	16.60	15.93
2023-03-22	16.95	16.04
2023-03-23	16.95	16.27
2023-03-24	16.25	16.52
2023-03-27	16.81	16.56
2023-03-28	16.34	16.63
2023-03-29	16.45	16.55
2023-03-30	16.23	16.47
2023-03-31	16.43	16.33

Tabla 3 17. Precio de cierre ajustado real y pronosticado en USD de las acciones del BMA. SVM.

En la Tabla 3.17 es posible comparar el valor real y pronosticado del precio de las acciones del BMA para el periodo entre el 15 de marzo y 31 de marzo de 2023. Se puede visualizar que tanto los valores reales y pronosticados tienen un valor muy cercano. Por ejemplo, para el 21 de marzo la diferencia entre precio real y pronosticado ha sido de USD 0,67.

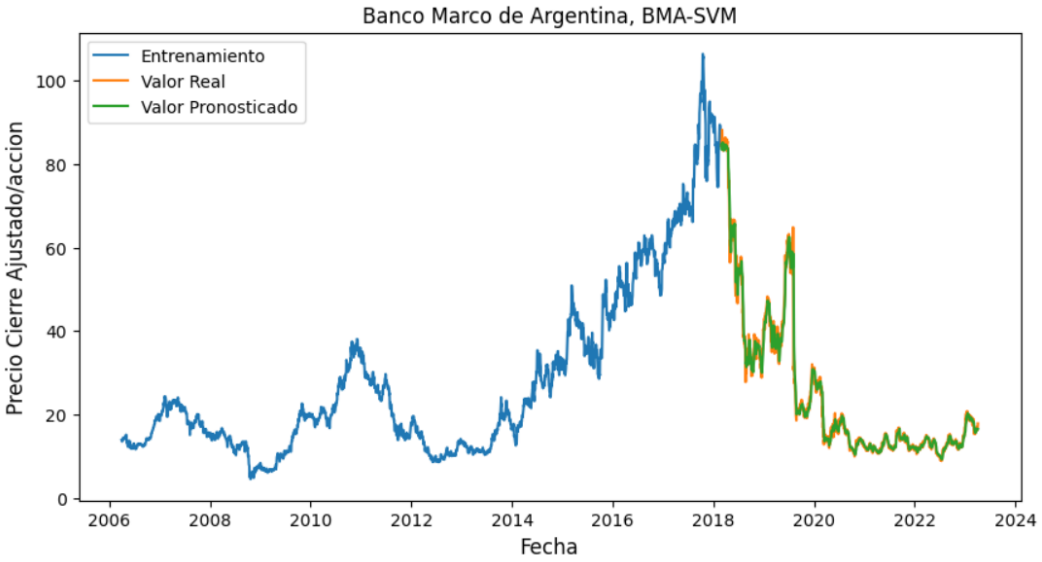


Figura 3.6. Modelo SVM para el BMA. Valores reales y pronosticados.

La Figura 3.6 permite confirmar visualmente el buen desempeño del SVM en el pronóstico de las acciones para el BMA. Se visualiza que la curva que representa el valor pronosticado esta sobre la cuerva que representa el valor real para todo el periodo temporal de la serie. Al igual que con LSTM, SVM también pudo pronosticar correctamente el precio de las acciones del BMA a pesar de que el precio de cierre ajustado de este banco ha presentado una trayectoria más volátil que la del BCol.

De esta forma se evidencia que SVM y su variante SVR, diseñada principalmente para tareas de regresión y pronóstico, pudo pronosticar correctamente el precio de las acciones de los dos bancos latinoamericanos estudiados en el presente trabajo.

4. Conclusiones y Recomendaciones.

A lo largo del presente proyecto de desarrollo se modelaron dos algoritmos, LSTM y SVM, con la finalidad de pronosticar con la mayor precisión los precios de las acciones de dos bancos latinoamericanos, BMA, BCol.

Se revisó la literatura correspondiente al ML y DL y se pudo determinar que los valores de los hiperparámetros de SVM y LSTM son muy importantes en la capacidad predictiva del modelo, por lo que se justifica de esta forma el haber buscado en este proyecto de desarrollo los valores óptimos. Se determinó que el LSTM ha dado muy buenos resultados en cuanto a su capacidad de pronosticar series temporales y que cuando se combina con otro tipo de algoritmos como ARIMA y CNN el poder de predicción es aún mayor. Si bien en la literatura que sirvió como referencia en el presente trabajo no se hizo un uso exhaustivo de SVM, la revisión de esta sirvió para reafirmar la importancia de los hiperparámetros de este algoritmo no solo como herramienta para mejorar la capacidad predictiva, sino además para controlar el *overfitting.*

Una vez se culminó con la fase de preparación de los datos, se pasó a la modelación de cada uno de los algoritmos, en donde se encontraron algunos puntos importantes que cabe mencionarlos. A través de las distintas gráficas y tablas en donde se visualizó el valor de las 3 métricas de error usadas en el presente trabajo y las curvas de observaciones reales y pronosticadas se determinó que:

En cuanto al SVM, el kernel radial y lineal, y los hiperparámetros *C, Epsilon* y *Gamma* con un valor de 0.5, 0.0001 y 0.001 respectivamente permitieron conseguir la mejor capacidad de predicción en términos de las 3 métricas de error utilizadas, RMSE, MAPE, MAE, tanto para el BCol como para el BMA. Esta configuración del SVM a más de lograr la mejor capacidad predictiva, permite controlar el *overfitting* y de esta manera los resultados aquí obtenidos pueden ser generalizados en un conjunto de datos para los que el algoritmo SVM no ha sido entrenado.

En el caso del LSTM, los hiperparámetros probados han sido el número de capas ocultas, algoritmo optimizador, función de activación, tamaño del *Batch* y el número de neuronas en cada capa oculta de la red neuronal. Luego de testear el rendimiento del LSTM mediante las mismas métricas de error utilizadas para SVM se determinó que los hiperparámetros optimizados para el BCol y el BMA son: *Adam* como algoritmo optimizador, Tangencial Hiperbólica como Función de Activación, Tamaño del *Batch* y número de neuronas por capa ocultas con valor de 32 unidades. La única diferencia entre las dos bases de datos trabajadas ha sido el número de capas ocultas. En donde para el caso del BMA este valor corresponde a 1, y para el BCol es de 2.

Los valores de estos hiperparámetros, para el caso LSTM, a más de permitir obtener una gran capacidad predictiva, han permitido regularizar a la red y evitar el *overfitting.* Estos hiperparámetros han sido el número de neuronas por capa oculta y el número total de capas ocultas. A más de este procedimiento, se utilizó el regularizador *dropout* el cual ha permitido apagar temporalmente una cierta cantidad de neuronas durante una fase del proceso de entrenamiento para luego prender estas neuronas y apagar otro subconjunto de ellas en una siguiente fase de entrenamiento y así sucesivamente hasta que se complete todo el proceso de entrenamiento.

Si bien SVM y LSTM presentaron un gran poder de predicción, pues las métricas de error utilizadas en la investigación obtienen valores pequeños, el LSTM logró obtener los menores valores. Se debe recordar que se eligió la configuración del modelo que mejor capacidad predictiva obtuvo y pudo controlar el *overfitting.* A más de este hecho, LSTM cuenta con una mayor cantidad y flexibilidad de distintos hiperparámetros para mejorar la capacidad predictiva y controlar el *overfitting,* como número de capas ocultas, número de neuronas por capa oculta, funciones de activación, *dropout,* etc. Por lo tanto, se recomienda el uso de LSTM para el pronóstico de series temporales ya que el mismo está construido precisamente para trabajar con datos de tipo secuencial.

Los gráficos y las tablas estadísticas mostraron que la serie temporal del BMA es mucho más volátil que la del BCol. Sin embargo, al momento de realizar las predicciones se observó, a través de las métricas de error, que los dos algoritmos pudieron pronosticar satisfactoriamente el precio de las acciones del BMA. Por lo tanto, para el actual proyecto de desarrollo se concluye que SVM y LSTM se adaptan muy bien a escenarios de volatilidad y logran aprender eficientemente estos patrones.

Un hecho a tomar en consideración en cuanto al LSTM es que cuando se presentan colapsos o incrementos pronunciados en una serie temporal, esta red neuronal no logra identificar y pronosticar con la suficiente eficiencia esta oscilación. Pues en el caso del BCol ante una caída pronunciada del precio de su acción en el año 2020 durante la pandemia del COVID-19, el LSTM no logra capturar este movimiento y por consecuencia su capacidad predictiva se ve seriamente afectada.

Ante esta situación, se recomienda que estudios posteriores que deseen profundizar acerca de la temática aquí estudiada podrían probar con otro tipo de configuración de los hiperparámetros aquí analizados. O, por el contrario, se podría tomar otro tipo de enfoque como es el de combinar el LSTM y CNN (*Convolutional Neural Networks),* el cual permite filtrar el ruido propio de las series de tiempo, por lo que se les conoce como resistentes al ruido.

Otra ventaja de los modelos híbridos CNN-LSTM es que posibilita la predicción del precio de acciones en función del análisis de sentimientos de los inversores. Estos tipos de modelos híbridos permiten capturar la perspectiva de los inversores en un cierto conjunto de acciones, logrando este tipo de metodologías un mejor rendimiento en comparación con modelos que cuentan únicamente con LSTM.

5. Referencias Bibliográficas.

[1] M. U. G. A. M. O. Omer Nerat Sezer, "Financial Time Serie Forecasting with Deep Learning: A Systematic Literature Review: 2005-2019," University of Economics and Technology., Ankara, 2019.

[2] K.-j. Kim, "Financial Time Series Forecasting using Support Vector Machines," Dongguk University, South Korea, 2003.

[3] J. D. J. Perafan, "Capacidad predictiva del modelo ARIMA-ANN en las acciones financieras colombianas.," Universidad de los Andes, Bogotá , 2020.

[4] N. T. A. S. N. Sima Siami-Namino, "A Comparison of ARIMA and LSTM in Forecasting Time Series," IEEE, 2018.

[5] T. A. Atwan, Time Seires Analysis with Python Cookbook., Packt, 2022.

[6] Z. W. y. H. W. Nan Jing, "A hybrid model integrating deep learning with investor sentiment analysis for stock price prediction," ELSEVIER, Shangai, 2021.

[7] A. A. a. S. Karasu, "The Effect of Kernel Values in Support Vector Machines to forecasting performance of financial time series and cognitive decison making," The Journal of Cognitive Systems., 2019.

[8] A. Geron, Hand-on Machine Learning with Scikit-Learn, Keras & Tensor Flow, Sebastopol: OREILLY, 2022.

[9] Y. L. V. M. Sebastian Raschka, Machine Learning with PyTorch and Scikit-Learn., Birmingham: Packt, 2022.

[10] H. M. a. S. A. Armin Lawi, "Implementation of Long Short-Term Memory and Gated Recurrent Units on grouped time-series data to predict stock prices accurately," Springer Open, 2022.

[11] E. P. P. P. Loannis E. Livieris, "A CNN-LSTM model for gold price time-series forecasting," Springer, London, 2019.

[12] Y. Vasiliev, "Python for Data Science," no starch press, 2021.

[13] M. Peixeiro, Time Seires Forecasting in Python, Manning Publications Co, 2022.

[14] A. G. S. P. Amita Kappor, Deep Learning with TensorFlow and Keras, Birmingham: Packt, 2022.

[15] F. Chollet, DEEP LEARNING with Python, New York: MANNING, 2022.

[16] J. M. W. J. James C. Van Horne, Fundamentals of Financial Management, Edinburgh: Pearson Education, 2009.

[17] D. R. Harper, "Investopedia," Getting to Know the Stock Exchanges, 10 Abril 2022. [Online]. Available: https://www.investopedia.com/articles/basics/04/092404.asp#:~:text=A%20stock%20exchange%20is%20a,communicate%20buy%20and%20sell%20orders.. [Accessed 8 Octubre 2023].

[18] C. Majaski, "Investopedia," 11 Abril 2023. [Online]. Available: https://www.investopedia.com/articles/investing/052313/financial-markets-capital-vs-money-markets.asp. [Accessed 8 Octubre 2023].

[19] L. C. C. F. Fernando Martínez, "CRISP-DM Twenty Years Later: From Data Mining Processed to Data Science Trajectories.," IEEE, 2019.

[20] M. I. A. B.-T. a. A. N. Yonina C. Eldar, "Robust Mean-Squared Error Estimation in the Presence of Model Uncertainties.," IEEE, 2005.

[21] M. Joseph, Modern Time Series Forecasting with Python, Birmingham: packt, 2022.

[22] T. F. A.J.P. Samarawickrama, "A Recurrent Neural Network Approach in Predicting Daily Stock Prices. An Application to the Sri Lankan Stock Market," University of Sri Jayawardenepura, Sri Lankan, 2017.

[23] C. K. J. A. S. Anita Yadav, "Optimizing LSTM for time series prediction in Indian stock market," ELSEVIER, New Delhi, 2020.

[24] L.-P. Chen, "Using Machine Learning Algorithms on Prediction of Stock Price," University of Western Ontario, Canada, 2020.

[25] M. J. BOYLE, "Calculate the Difference between Nominal Value and Real Value of Stock.," Investopedia, 28 Septiembre 2023. [Online]. Available: https://www.investopedia.com/ask/answers/050515/how-can-you-calculate-difference-between-nominal-value-and-real-value-stock-shares.asp. [Accessed 12 Diciembre 2023].

[26] M. V. Wegberg., "Standardization Process of Systems Technologies: Creating a Balance between Competition and Cooperation.," University of Maastricht, The Netherlands., 2004.

[27] amazon, "aws.amazon.com," Amazon, 2023. [Online]. Available: https://aws.amazon.com/what-is/overfitting/. [Accessed 15 Diciembre 2023].

[28] A. B. Pérez, "enciclopediafinanciera.com," Enciclopedia Financiera, [Online]. Available: http://www.enciclopediafinanciera.com/definicion-estacionalidad.html. [Accessed 15 Diciembre 2023].

[29] J. Villavicencio, "Introducción a Series de Tiempo," estadisticas.gobierno.

[30] linkedin, "linkedin.com," Linkedin, [Online]. Available: https://www.linkedin.com/advice/1/how-do-you-interpret-learning-curves-validation#:~:text=A%20learning%20curve%20plots%20the,it%20generalizes%20to%20new%20data.. [Accessed 15 Diciembre 2023].

[31] P. Radhakrishnan, "Towardsdatascience.com," Towardsdatascience, 09 Agosto 2017. [Online]. Available: https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a. [Accessed 15 Diciembre 2023].

[32] M. J. Kramer, "investopedia.com," Investopedia., 23 Marzo 2022. [Online]. Available: https://www.investopedia.com/terms/o/over-the-countermarket.asp#:~:text=An%20over%2Dthe%2Dcounter%20(,a%20central%20exchange%20or%20broker.. [Accessed 15 Diciembre 2023].

[33] Santander, "Santander.com," Santander, 03 Julio 2023. [Online]. Available: https://www.santander.com/en/stories/nominal-real-value#:~:text=In%20economics%2C%20the%20nominal%20value,things%20that%20have%20financial%20value.. [Accessed 15 Diciembre 2023].

[34] J. J. Ross Westerfield, Corporate Finance, New York: Mc Graw Hill Education, 2016.

[35] R. Seethalakshmi, "Analysis of stock market predictor variables using Linear Regression," SASTRA DEEMED UNIVERSITY, Tamilnadu, India, 2018.

[36] V. A. Vadim Kramar, "Time-Series Forecasting of Seasonal Data Using Machine Learning Methods," MDPI, Basel, Switzerland , 2023.

[37] E. E.-D. H. A. E.-S. N. A. E.-B. Marwa Sharaf, "StockPred: a framework for stock Price prediction," Springer, Menoufia, Egipto., 2021.

6. Anexos

6.1. Anexo A. Codigo Python

Código Python empleado en todo el proceso de la investigación.

https://github.com/XavierRomero90/Modelos-de-DL-ML

6.2. Anexo B. Glosario de Términos.

Overfitting. El sobreajuste es un comportamiento indeseable del aprendizaje automático que se produce cuando el modelo de aprendizaje automático proporciona predicciones precisas para los datos de entrenamiento, pero no para los datos nuevos [27].

Análisis de Estacionariedad. Comprende una serie de procesos para determinar si la serie temporal en cuestión posee las características de estacional. Esto es que, el valor promedio y la varianza de la variable en cuestión sean constantes a través del tiempo.

Tendencia. La tendencia representa el cambio a largo plazo en el nivel de una serie temporal. Este cambio puede ser hacia arriba, aumento de nivel, o hacia abajo, disminución de nivel.

Serie Estacional. La estacionalidad es una característica de una serie temporal en la que los datos experimentan cambios regulares y predecibles que se repiten cada año calendario [28].

Ruido Blanco. Una serie de tiempo es ruido blanco si las variables son independientes y están distribuidas idénticamente con una media cero. Esto significa que todas las variables tienen la misma varianza y cada valor tiene una correlación cero con todos los demás valores de la serie [29].

Curva de aprendizaje. Una curva de aprendizaje traza las puntuaciones de entrenamiento y validación de su modelo en función de la cantidad de ejemplos de entrenamiento. Este tipo de curvas ayudan a evaluar qué tan bien aprende el modelo de los datos y cómo se generaliza a datos nuevos. Una puntuación alta significa que un determinado modelo se ajusta bien a los datos, mientras que una puntuación baja significa que el modelo comete muchos errores. [30].

Curva de validación. Una curva de validación traza las puntuaciones de entrenamiento y validación del modelo en función de un único hiperparámetro. Si una curva de validación muestra un pico en un determinado valor del hiperparámetro, significa que este valor ofrece el mejor equilibrio entre sesgo y varianza [30].

Hiperparámetro. Un hiperparámetro es un parámetro que controla el comportamiento del modelo, como la cantidad de capas ocultas en una red neuronal o la intensidad de la regularización en una regresión lineal. Ajustar un determinado hiperparámetro ayuda a encontrar el valor óptimo que maximice el rendimiento de un cierto modelo [31].

Mercado Extrabursátil. El mercado extrabursátil se refiere a la negociación de valores que se realiza fuera de las principales bolsas de valores. Hay más de 12.000 valores negociados en el mercado Over-The-Counter (OTC), incluidas acciones, fondos cotizados en bolsa, bonos, materias primas y derivados. A diferencia de las bolsas tradicionales como la Bolsa de Valores de Nueva York (NYSE) o el Nasdaq, no existe una ubicación física asociada con el mercado OTC. Más bien, todas las transacciones se producen de forma electrónica y directa entre dos partes en un mercado descentralizado [32].

Precio Real: El valor real de un ítem, también llamado precio relativo, constituye su valor nominal ajustado por la inflación [33].