Revista ECA Sinergia
ISSN-e: 2528-7869
https://www.revistas.utm.edu.ec/index.php/ECASinergia
Vol. 13 Núm. 2 (107-117) Mayo - Agosto 2022
revistaecasinergia@gmail.com
Universidad Técnica de Manabí
DOI: https://doi.org/10.33936/eca_sinergia.v13i2.4149

Procedimiento para la previsión de tipos de cambio. Aplicación de redes neuronales artificiales EUR/USD

Procedure for the forecast of exchange rates. Applying artificial neural networks: EUR/USD

Fidel de la Oliva de Con 000-0002-1284-9218 fdelaoliva@gmail.com
Reinaldo Molina Fernández 0000-0003-1042-5654 reinaldo.molina@fcf.uh.cu
David Díaz Rodríguez 0000-0002-0927-9795 daviddrsch@gmail.com

Universidad de La Habana, Cuba.

Recepción: 2021-11-11 / **Aceptación:** 2022-05-12

Código Clasificación JEL: C45, O24

Citación/como citar este artículo: De la Oliva, F., Molina, R. y Díaz, D.(2022). Procedimiento para la previsión de tipos de cambio. Aplicación de redes neuronales artificiales EUR/USD. ECA Sinergia, 13(2), 107-117. https://doi.org/10.33936/eca_sinergia.v13i2.4149

Revista ECA Sinergia
ISSN-e: 2528-7869
https://www.revistas.utm.edu.ec/index.php/ECASinergia
Vol. 13 Núm. 2 (107-117) Mayo - Agosto 2022
revistaecasinergia@gmail.com
Universidad Técnica de Manabí
DOI: https://doi.org/10.33936/eca_sinergia.v13i2.4149

RESUMEN

La previsión del tipo de cambio constituye un elemento fundamental en la estrategia de cobertura frente al riesgo cambiario, lo que impulsa la búsqueda de nuevas técnicas que sean más eficaces. Con el objetivo de facilitar el uso de algoritmos de inteligencia artificial para este propósito, se creó el procedimiento expuesto en este trabajo. Este procedimiento está basado en el uso de métodos cuantitativos para la obtención de la previsión, siendo el más importante la utilización de las redes neuronales artificiales, concretamente las denominadas long short-term memory. Se lleva a cabo la construcción de distintos modelos de los cuales se elige el óptimo usando como criterio de selección la raíz cuadrada del error cuadrático medio. Tras la obtención de modelos que se desempeñan con una efectividad comprendida en el rango de 97.28% - 99.55% se demostró la eficacia del procedimiento.

Palabras clave: inteligencia artificial, pronósticos, redes neuronales recurrentes, series de tiempo.

ABSTRACT

The forecast of the exchange rate is an essential part of foreign exchange risk management strategy, which drives the search for new techniques that are more effective. The procedure presented in this work was created in order to facilitate the use of artificial intelligence algorithms for exchange rate forecasting. The procedure is based on the use of quantitative methods to obtain the forecast, the most important being the use of artificial neural networks, specifically those called long short-term memory. The construction of different models is carried out, from which the optimal one is chosen using the square root of the mean square error as the selection criterion. The efficacy of the procedure was demonstrated after obtaining models that perform with an effectiveness comprised in the range of 97.28% - 99.55%.

Keywords: artificial intelligence, prognostic, recurrent neural networks, time series.

INTRODUCCIÓN

La previsión del tipo de cambio es una de las temáticas más tratadas en las investigaciones financieras, por su importancia en la actividad económica internacional. Debido a la complejidad del problema de predecir el comportamiento de los tipos de cambio y con el objetivo de obtener los mejores resultados los investigadores han estudiado el uso de diversas técnicas.

En Cuba se han realizado una serie de investigaciones con el objetivo de predecir el tipo de cambio para una mejor cobertura del riesgo cambiario. Estas investigaciones han abordado distintos métodos para la predicción como los análisis de mercados, fundamental y técnico, como se exponen en De la Oliva y García (2020).

Con el avance de la inteligencia artificial (AI, por sus siglas en inglés, artificial intelligence) y el empleo de las redes neuronales artificiales (ANN, por sus siglas en inglés, artificial neural network) se abre una oportunidad para implementar nuevas técnicas para el análisis de las series de tiempo.

Hasta el momento en el equipo de trabajo no se ha investigado mucho acerca de la posibilidad de emplear las técnicas y métodos de la AI como las ANN para la creación de modelos que permitan predecir series de tiempo econométricas, por lo que esta investigación se propuso como objetivo elaborar un procedimiento para su utilización.

El porqué del uso de las ANN como técnica para la previsión de series de tiempo, como expone Brownlee (2019), se debe a cualidades de las mismas como la robustez ante los residuos y la capacidad de modelar relaciones no lineales. Según lo referido por Chollet y Allaire (2018) las ANN poseen la propiedad de construirse a partir de múltiples entradas y salidas, lo que Brownlee (2019) define como la capacidad de trabajar con distintas variables que influencian en la serie; además de poder obtenerse pronósticos multivariados y predecirse varias observaciones futuras.

Las ANN pueden clasificarse como unidireccionales (FNN, por sus siglas en inglés, feedforward neuronal networks) y bidireccionales (RNN, por sus siglas en inglés, recurrent neuronal networks). En el texto Brownlee (2019b) el autor señala las siguientes limitaciones de las FNN: número de variables de entrada fijo al igual que los métodos tradicionales de pronóstico de series de tiempo; número de variables de salida también fijo, lo que significa que para cada patrón de entrada, se debe producir una salida y la limitación fundamental de tener que especificar la dependencia temporal por adelantado en el diseño del modelo, esta dependencia es casi siempre desconocida y debe descubrirse y extraerse del análisis detallado en una forma fija.

Las RNN no cuentan con las mencionadas limitaciones de las FNN lo que ha hecho que este tipo de red demuestre una mayor eficacia para la solución de diversos problemas como la previsión de series de tiempo. Entre las RNN destacan por su desempeño en problemas de regresión las long short-term memory (LSTM), como se explica en Brownlee (2019) el desempeño sobresaliente de estas ANN se debe a su capacidad de determinar que observaciones de entrada son relevantes para una determinada salida.

Debido a lo expuesto con anterioridad el procedimiento que se propone se basa en la creación de distintos modelos de redes LSTM para predecir una serie de tiempo y determinar cuál es el más eficaz.

METODOLOGÍA

El procedimiento desarrollado está basado en el uso de métodos cuantitativos como: la recopilación de los datos cuantitativos correspondientes a la serie de tiempo a predecir; el uso de modelos de redes LSTM; y la evaluación de los resultados obtenidos por los modelos generados usando como criterio de selección la raíz cuadrada del error cuadrático medio (RMSE, por sus siglas en inglés, root mean square error).

Recopilación de datos

Los datos cuantitativos utilizados en el ejemplo que se muestra en la discusión fueron recopilados de la página investing.com. Se seleccionó el par EUR/USD, debido a la importancia de esta tasa de cambio tanto para sus respectivas economías, como para aquellas que las usan como intermediario de otras monedas.

Redes LSTM

Todas las RNN tienen la forma de una cadena de módulos repetitivos de una ANN. Pero como se expone en Olah (2015) y Chollet

Revista ECA Sinergia
ISSN-e: 2528-7869
https://www.revistas.utm.edu.ec/index.php/ECASinergia
Vol. 13 Núm. 2 (107-117) Mayo - Agosto 2022
revistaecasinergia@gmail.com
Universidad Técnica de Manabí
DOI: https://doi.org/10.33936/eca_sinergia.v13i2.4149

y Allaire (2018) en la RNN estándar este módulo repetitivo tendrá una estructura muy simple, como una sola capa y por lo general con la función de activación tanh (tangente hiperbólica), mientras que en las LSTM el módulo de repetición tiene una estructura diferente, en lugar de tener una sola capa, hay cuatro que interactúan de una manera muy especial.

Como exponen Olah (2015) y Chollet y Allaire (2018), la clave para las LSTM es el estado de la celda, que corre directamente por toda la cadena de las capas que componen una LSTM, con solo algunas interacciones lineales menores, haciendo que la información fluya de manera fácil y sin cambios. Los textos mencionados explican además que las LSTM tienen la capacidad de eliminar o agregar información al estado de la celda, cuidadosamente regulado por estructuras definidas como "puertas", estas comprenden una opción para dejar pasar o no información a la red. Las "puertas" se componen de una capa de red neuronal sigmoidea, cuya función de activación es la sigmoide, y una operación de multiplicación puntual. Esta capa sigmoide, como explican Olah (2015) y Chollet y Allaire (2018), genera números entre cero y uno, que representan el porciento de información aceptada, correspondiente a una entrada.

Una LSTM tiene tres de estas puertas, para proteger y controlar el estado de la célula, como se le denomina al conjunto de sus componentes, y el funcionamiento de la LSTM se puede describir según Chollet y Allaire (2018) y Olah (2015) en cuatro pasos, que según este último son:

1. Decidir qué información se olvida del estado de la celda a través de la puerta, forget gate layer. Esta puerta ve a ht-1 y xt, y genera un número entre 0 (deshacerse) y 1 (mantener) para cada número en el estado de la celda Ct - 1, ecuación 1.

$$f_t = \sigma\left(W_f \,.\left[h_{t-1,x_t}\right] + b_f\right) \qquad [1]$$

2. Decidir ı input gate layer decide qué valores actualizar y luego, una capa tanh crea un vector de nuevos valores candidatos que podrían agregarse al estado, ecuaciones 2 y 3.

$$i = \sigma\left(W_i \,.\left[h_{t-1,x_t}\right] + b_i\right) \qquad [2]$$

$$\widetilde{C_t} = \tanh\left(W_c \,.\left[h_{t-1,x_t}\right] + b_c\right) \qquad [3]$$

3. Se actu or por, olvidando lo necesario, luego se agrega. Estos son los nuevos valores candidatos, escalados según cuánto se necesita actualizar cada valor de estado, ecuación 4.

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \qquad [4]$$

4. Se gene ide qué partes del estado de la celda es la salida; luego el estado de la celda pasa a través de una función tanh (escalando los valores entre −1 y 1) y se multiplican por la salida de la puerta, output gate, ecuaciones 5 y 6.

$$o_t = \sigma\left(W_o \left[h_{t-1,x_t}\right] + b_o\right. \qquad [5]$$

RMSE

$$h_t = o_t * \tanh(C_t) \qquad [6]$$

Se evaluaron lots modelos obtenidos mediante el cálculo del RMSE debido a que como se expone en de la Oliva et al. (2018) es uno de los criterios de selección más comúnmente utilizados en la evaluación de problemas de regresión. El RMSE se calcula mediante la fórmula que se expone en Hirekerur (2020), ecuación 7.

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad [7]$$

Donde:

y = valor real
yˆ = predicción
n = cantidad de observaciones o predicciones

RESULTADOS

Se propone un procedimiento que parte desde la preparación del entorno de trabajo y culmina en la evaluación de los modelos obtenidos. Entre las distintas etapas del procedimiento se destaca la tercera, plantear y responder preguntas de investigación, que determina en mayor medida los resultados de aplicar el procedimiento. A continuación, se ilustran todas las etapas del procedimiento:

A. Preparar las condiciones para la correcta y fácil aplicación del procedimiento
B. Importar los datos
C. Plantear y responder preguntas de investigación
D. Cargar y pre-procesar los datos
E. Crear las entradas y las salidas
F. Construir y entrenar los modelos
G. Evaluar y seleccionar modelo

DISCUSIÓN

A. Preparar las condiciones para la correcta y fácil aplicación del procedimiento
Etapa preparatoria que responde a las necesidades básicas de conocer acerca del software y adecuar el entorno de trabajo a las necesidades de los usuarios. Durante la aplicación de la propuesta esta etapa fue llevada a cabo en dos pasos:

I. Configurar las distintas ventanas de RStudio. Para facilitar el trabajo con el software, se siguieron las indicaciones expuestas en Philips (2018) y se creó una ventana rscript, facilitándose la escritura y la ejecución de los códigos necesarios en el resto del procedimiento.

II. Instalar los paquetes necesarios. Con el objetivo de lograr un trabajo más fluido en las restantes etapas del procedimiento, en este paso se instalaron todos los paquetes o herramientas necesarias y sus dependencias siguiendo las instrucciones que se muestran en Duncan (2020).
Los paquetes instalados son keras y tensorflow, para el trabajo con ANN, y los llamados tseries, forecast y scales para el trabajo con series de tiempo.

B. Importar los datos
Se comenzó importando la base de datos obtenida de investing.com correspondiente a los datos históricos del par EUR/USD. Una vez importada se apreció que estaba conformada por los valores de apertura, cierre, máximo, mínimo. Se contaba hasta la fecha de

Revista ECA Sinergia
ISSN-e: 2528-7869
https://www.revistas.utm.edu.ec/index.php/ECASinergia
Vol. 13 Núm. 2 (107-117) Mayo - Agosto 2022
revistaecasinergia@gmail.com
Universidad Técnica de Manabí
DOI: https://doi.org/10.33936/eca_sinergia.v13i2.4149

en la que se aplicó el procedimiento con 5654 observaciones, correspondientes a una frecuencia diaria desde el 4 de enero de 1999 hasta el 4 de septiembre del 2020. Se trabajó con los valores al cierre, ya que el objetivo perseguido en la aplicación de la propuesta fue el de predecir las futuras observaciones de esta variable basándose solo en el comportamiento histórico de la misma.

C. Plantear y responder preguntas de investigación

Se plantean y responden preguntas de investigación acorde a las características de cada problema. Esta es la etapa de mayor importancia ya que se determinan los aspectos a evaluar en la búsqueda del modelo óptimo.

Las preguntas a las que se dieron respuesta al plantear la investigación fueron:

¿Será necesario transformar la serie? Es una pregunta relevante ya que autores como Keydana (2017) y Brownlee (2017) recomiendan transformar las series de tiempo para la construcción de modelos de redes LSTM. Por lo que se evaluaron las siguientes tres opciones:

- No transformar la serie: serie original.

- Primera transformación: crear una serie de tiempo estacionaria mediante la diferenciación en el orden necesario de la serie original y escalar la resultante en un rango entre 0 y 1.

- Segunda transformación: crear una serie de tiempo estacionaria aplicando primero a la serie original una transformación logarítmica y diferenciar está en el orden necesario para que cumpla con el supuesto de estacionariedad y luego escalar la serie diferenciada en un rango entre -1 y 1.

¿Cuál es la estructura optima de los vectores de entrenamiento? Teniéndose en cuenta que el problema de predicción de series de tiempo es usualmente abordado en machine learning mediante el aprendizaje supervisado, como se expone en Brownlee (2016), se determinó que los vectores de entrenamientos estuviesen conformados por vectores de entrada y vectores de salida, para lo que se realizaron las siguientes dos preguntas:

¿Cuál es el horizonte temporal? El horizonte temporal se determina en dependencia de la cantidad de observaciones a futuro que se desea predecir, este será también el número de observaciones de los vectores de salida. Para ilustrar el procedimiento se seleccionó un horizonte temporal de un mes, unas 23 observaciones a predecir.

¿Cuántas observaciones deben tener los vectores de entrada? Se recomienda para esto realizar un proceso iterativo incrementando de uno en uno el número de observaciones en los vectores de entrada, hasta llegar al punto en el que la efectividad de los modelos se vea afectada por el incremento de observaciones. Lo cual implica una alta carga de pruebas por lo que para dinamizar este proceso se consideraron solo tres posibilidades:

- El número de observaciones de los vectores de entrada será igual a los de salida.
- El número de observaciones de los vectores de entrada será el doble de los de salida.
- El número de observaciones de los vectores de entrada será el triple de los de salida.

Determinadas las posibles estructuras para los vectores de entrenamiento se pasó a dar respuesta a la siguiente pregunta. ¿Cuál es la estructura óptima del modelo? Se recomienda probar múltiples números de capas y de neuronas para determinar en qué forma se ven afectados los resultados según la estructura. Para agilizar este proceso se decidió utilizar un modelo simple con una capa LSTM y una capa de salida, como el expuesto en el ejemplo expuesto de Keydana (2017), se determinó además que el número de neuronas de la capa LSTM sería igual al número de observaciones en el vector de entradas y el número de neuronas de la capa de salida sebe corresponder con la cantidad de observaciones de los vectores de salida.

Obtenidas las estructuras a investigar y habiendo planteado usar tres series, se tienen nueve posibles modelos, lo que lleva a la siguiente pegunta ¿Cuántos modelos de iguales características se deben construir? Una pregunta que los autores consideran fundamental en el procedimiento, debido a que el aprendizaje en una ANN es un proceso estocástico como exponen Torres (2018), Chollet y Allaire (2018), por lo es necesario construir distintos modelos con iguales características para poder determinar una efectividad esperada si la investigación realizada es replicada. Para responder a esta pregunta y agilizar el proceso investigativo se planteó construir 10 modelos de iguales características.

Por lo tanto, al dar respuesta a las preguntas planteadas en la presente etapa se obtuvo el siguiente diseño de la investigación:

- Utilizar muestras de tres series con el objetivo de determinar cómo afecta la efectividad de una red neuronal la serie utilizada.
- Utilizar tres estructuras de vectores de entrenamiento y de ANN para determinar cómo estos aspectos afectan la efectividad de las mismas.
- Construir y entrenar diez modelos de iguales características, para evidenciar la aleatoriedad en el proceso de aprendizaje en una red neuronal y poder determinar una efectividad esperada más ajustada si la investigación es replicada.

D. Cargar y preprocesar los datos
En esta etapa se cargan y pre-procesan los datos si durante la etapa anterior se determinó realizar alguna transformación a la serie original.

Debido a que en la ilustración del procedimiento en la tercera etapa se planteó que se trabajaría además de con la serie original con series de tiempos transformadas, se ejecutaron dichas transformaciones. Como ambas transformaciones de la serie parten de que las serie que se obtendrá sea estacionaria se verificó si la serie original y las obtenidas cumplían con este supuesto mediante el procedimiento expuesto en de la Oliva et al. (2018) consiste en el análisis del correlograma y la realización de las pruebas de Dickey-Fuller y Phillips Perron.

El análisis del correlograma fue realizado a través del uso de la función ggtsdisplay() del paquete forecast y las pruebas de Dickey-Fuller y Philips Perron para el supuesto de estacionariedad fueron realizadas a través de las funciones adf.test() y pp.test() del paquete tseries.

E. Crear las entradas y las salidas
Esta etapa tiene como objetivo crear los distintos vectores de entrenamiento planteados durante la etapa C. para cada una de las series a utilizar.
Para construir los vectores de entrenamiento se determinó que se utilizarían las primeras 5487 observaciones de la serie, desde el 4 de enero de 1999 hasta 13 de enero del 2020. Para construir los vectores de entrenamiento se utilizó la estrategia denomina rolling windows utilizada en Keydana (2017) y Brownlee (2017), que consiste en dividir la serie en muestras de n número de observaciones y que cada muestra comenzará y terminará en la observación posterior que la muestra anterior.

Resultados de los vectores creados que utilizaron como base la serie original

- Vectores de entrada se utilizó el periodo entre el 4 de enero de 1999 hasta el 12 de diciembre del 2020, obteniéndose: vectores con 23 observaciones, 5442 muestras; con 46 observaciones, 5419 muestras; con 69 observaciones 5396 muestras.
- Vectores de salida con 23 observaciones para vectores de entrada con 23 observaciones se utilizó el periodo del 4 de febrero de 1999 hasta 13 de enero del 2020, obteniéndose 5442 muestras.
- Vectores de salida con 23 observaciones para vectores de entrada con 46 observaciones se utilizó el periodo del 9 de marzo de 1999 hasta 13 de enero del 2020, obteniéndose 5419 muestras.
- Vectores de salida con 23 observaciones para vectores de entrada con 69 observaciones se utilizó el periodo del 9 de abril de 1999 hasta 13 de enero del 2020, obteniéndose 5396 muestras.

Por otro lado, los vectores creados teniendo como base alguna de las series resultantes de las transformaciones tienen una muestra menos, esto se debe a que en ambas transformaciones se aplicó una diferencia de orden uno, lo que implica que estas series no tengan la primera observación.

F. Construir y entrenar los modelos
Esta etapa se dividió en dos pasos, construir los modelos y entrenar el modelo, pasos en los que se construyen y entrenan todos los modelos planteados en la etapa C, usando para ello las distintas muestras creadas en la etapa anterior.

Como se mencionó con anterioridad los modelos construidos contaron con una estructura simple, una capa LSTM y una capa de salida. Para la capa LSTM se usó la configuración por defecto, lo que quiere decir que la función de activación utilizada es la denominada tangente hiperbólica como se explica en Olah (2015). Para compilar los modelos se utilizaron las más sugeridas configuraciones: optimizador adam y el error cuadrático medio como se ve Keydana (2017) y Brownlee (2016).

Revista ECA Sinergia
ISSN-e: 2528-7869
https://www.revistas.utm.edu.ec/index.php/ECASinergia
Vol. 13 Núm. 2 (107-117) Mayo - Agosto 2022
revistaecasinergia@gmail.com
Universidad Técnica de Manabí
DOI: https://doi.org/10.33936/eca_sinergia.v13i2.4149

El entrenamiento de los modelos fue realizado con un batch size de uno, lo que se explica en Brownlee (2018) como la cantidad de muestras a analizar antes de actualizar los parámetros del modelo y con valor de 5 en el parámetro epoch que es el número de veces que las muestras de entrenamiento serán pasadas al modelo, Brownlee (2018).

Como resultado de esta etapa se obtuvo el desempeño esperado de los distintos modelos construidos, siendo:

- Para los modelos entrenados con los vectores construidos a partir de los valores de la serie tipo, rango en error cuadrático medio: 0.0081-0.0002.
- Para los modelos entrenados con los vectores construidos a partir de los valores de la serie con la primera transformación, rango en error cuadrático medio: 0.0141-0.0130.
- Para los modelos entrenados con los vectores construidos a partir de los valores de la serie con la primera transformación, rango en error cuadrático medio: 0.0355-0.0351.

G. Evaluar y seleccionar modelo

Esta etapa tiene como primer objetivo evaluar el desempeño de los modelos construidos. Para evaluar la efectividad del modelo se utilizó la estrategia denomina walk forward validation. Como expone Brownlee (2017), esta técnica parte de utilizar el método de división en muestras explicado con anterioridad, rolling windows y consta de los siguientes pasos.

1. Se pasa al modelo una muestra de entrada para obtener salidas.
2. Se comparan las salidas obtenidas con la muestra de salida real.
3. Se actualiza el modelo entrenándolo con la muestra de entrada y la muestra de salida real.
4. Se repiten los tres pasos anteriores con cada conjunto de muestras entrada-salidas.
5. Se calcula el error del modelo promediando los resultados arrojados en las distintas comparaciones.

Para cumplir con el objetivo esta etapa fue dividida en tres pasos: crear vectores de entradas y salidas; predecir y transformar las predicciones; evaluar modelo y seleccionar óptimo.
En el primero de los pasos de esta etapa se crearon los distintos vectores de entradas y salidas utilizando la misma estrategia que en la etapa E.

- Para los vectores de entrada con 23 observaciones se utilizó el periodo comprendido entre el 13 de noviembre del 2019 al 4 de agosto del 2020.
- Para los vectores de entrada con 46 observaciones se utilizó el periodo comprendido entre el 11 de octubre del 2019 al 4 de agosto del 2020.
- Para los vectores de entrada con 69 observaciones se utilizó el periodo comprendido entre el 10 de septiembre del 2019 al 4 de agosto del 2020.

- Para el vector de salida que fue el mismo en todos los casos se utilizó el periodo comprendido entre el 16 de diciembre del 2019 al 4 de septiembre del 2020.

Cada uno de los vectores obtenidos está comprendido por un total de 167 muestras.

En el segundo paso de esta etapa fueron pasados al modelo los vectores de entrada correspondientes. A las predicciones obtenidas por los modelos que fueron entrenados por vectores conformados a partir de las series transformadas se le realizaron las siguientes transformaciones antes de realizar las comparaciones con los valores reales.

- Primera transformación: Se escalaron a los valores mínimos y máximos de la serie diferenciada correspondiente y se invirtió la diferenciación.
- Segunda transformación: Se escalaron a los valores mínimos y máximos de la serie diferenciada correspondiente, se invirtió la diferenciación y se aplicó el antilogaritmo.

Luego de obtener cada predicción en valores de la serie tipo, se utilizó el vector de entrada utilizado para la predicción en la actualización del modelo, hasta utilizar el conjunto de muestras de entradas y haber obtenido cada predicción en valores equivalentes a la serie original.

En el tercer paso la evaluación de los modelos fue realizada a través del cálculo de la raíz cuadrada del error cuadrático medio (RMSE por sus siglas en inglés, root mean squared error), criterio óptimo de selección, como se exponen De la Oliva et al. (2018) y De la Oliva y Molina (2019). De los errores obtenidos en cada una de las 167 predicciones realizadas utilizando los distintos modelos se obtuvieron los resultados expuestos en la Tabla No.1:

Tabla 1. Resultados obtenidos en los modelos.

Modelos entrenados con vectores creados a partir de la serie original		
Modelos entrenados con vectores con 23 entradas		
RMSE medio: 0.013561192	RMSE mínimo: 0.00367950	RMSE máximo:0.03220582
Modelos entrenados con vectores con 46 entradas		
RMSE medio: 0.013675444	RMSE mínimo: 0.00351732	RMSE máximo: 0.03351387
Modelos entrenados con vectores con 69 entradas		
RMSE medio: 0.013731994	RMSE mínimo: 0.00356540	RMSE máximo: 0.0338307
Modelos entrenados con vectores creados a partir de la serie con primera transformación		
Modelos entrenados con vectores con 23 entradas		
RMSE medio: 0.046626800	RMSE mínimo: 0.01701220	RMSE máximo: 0.07465840
Modelos entrenados con vectores con 46 entradas		
RMSE medio: 0.043105726	RMSE mínimo: 0.01437569	RMSE máximo: 0.07119603
Modelos entrenados con vectores con 69 entradas		
RMSE medio: 0.039225007	RMSE mínimo: 0.01026937	RMSE máximo: 0.06730419
Modelos entrenados con vectores creados a partir de la serie con segunda transformación		
Modelos entrenados con vectores con 23 entradas		
RMSE medio: 0.028140840	RMSE mínimo: 0.00598571	RMSE máximo: 0.05740401
Modelos entrenados con vectores con 46 entradas		
RMSE medio: 0.025698432	RMSE mínimo: 0.00456011	RMSE máximo: 0.05420584
Modelos entrenados con vectores con 69 entradas		
RMSE medio: 0.026414246	RMSE mínimo: 0.00565094	RMSE máximo: 0.05556464

Fuente: Elaboración propia

De los resultados expuestos en la tabla anterior se pudo deducir que específicamente para el caso ilustrado:

- La efectividad de los modelos se ve afectada según la serie utilizada para el entrenamiento en un rango entre el 1,20% y el 3,30%.
- La efectividad de los modelos se ve afectada según su estructura: en un rango entre el 0,005% y el 0,017% para los entrenados con la serie original; en un rango entre el 0,352% y el 0,74% para los entrenados con la serie con la primera transformación; y en un rango entre el 0,071% y el 0,2444% para los entrenados con la serie con la segunda transformación.

Además, se determinó como modelo óptimo el segundo modelo construido y entrenado con los vectores de 23 entradas y 23 salidas a partir de la serie original, el cual obtuvo un RMSE de 0.01352.

CONCLUSIONES

La obtención del modelo óptimo para la previsión del valor de cierre del par EUR/USD se realiza mediante el procedimiento propuesto que se basa en la realización de una investigación en la que se construyen múltiples modelos de ANN teniendo en cuenta diversos parámetros y se selecciona el más eficaz usando como criterio de selección el RMSE.

El modelo óptimo obtenido puede ser explotado de forma continua, hasta que su eficacia decaiga, mediante el entrenamiento de el mismo con las nuevas observaciones de la serie. La eficacia del modelo puede decaer

Revista ECA Sinergia
ISSN-e: 2528-7869
https://www.revistas.utm.edu.ec/index.php/ECASinergia
Vol. 13 Núm. 2 (107-117) Mayo - Agosto 2022
revistaecasinergia@gmail.com
Universidad Técnica de Manabí
DOI: https://doi.org/10.33936/eca_sinergia.v13i2.4149

si las observaciones futuras de la serie sobrepasan ampliamente el rango de valores histórico o si ocurren cambios bruscos continuos en los comportamientos futuros de la variable. Una vez la eficacia del modelo haya decaído más allá de los límites establecidos se repite el procedimiento para hallar un nuevo modelo óptimo.

REFERENCIAS BIBLIOGRÁFICAS

Allaire, J. J., & Chollet, F. (2020). keras: R Interface to 'Keras'. R package version 2.3.0.0. https://CRAN.R-project.org/package=keras

Allaire, J. J., & Tang,Y. (2020). tensorflow: R Interface to 'TensorFlow'. R package version 2.2.0. Recuperado de: https://CRAN.R-project.org/package=tensorflow

Brownlee, J. (2016). «Time series forecasting as Supervised Learning». Machine Learning Mastery. Recuperado de de: https://machinelearningmastery.com/time-series-forecasting-supervised-learning/

Brownlee, J. (2017). «How to use timesteps in LSTM networks for time series forecasting». Machine Learning Mastery. Reccuperado de: https://machinelearningmastery.com/use-timesteps-lstm-networks-time-series-forecasting/

Brownlee, J. (2018). «Difference Between a Batch and an Epoch in a Neural Network». Machine Learning Mastery. Recuperado de: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

Brownlee, J. (2019). «The promise of recurrent neural networks for time series forecasting». Machine Learning Mastery. Recuperado de: https://machinelearningmastery.com/promise-recurrent-neural-networks-time-series-forecasting/

Chollet, F., & Allaire, J. J. (2018). Deep Learning with R. Manning Publication.

De la Oliva, F.,& García, A.(2020). «Procedimiento para un pronostico de la tasa de cambio euro-dólar». Revista CoFin Habana, 1. Recuperado de: http://www.cofinhab.uh.cu/index.php/RCCF/articledownload/387/377

Duncan, E.W. (2020). «A Guide to (Re-)Installing R and Related Software». Rpubs. Recuperado de: https://rpubs.com/Earlien/guide-to-installing-R

Hirekerur, R. (2020). «A Comprehensive Guide To Loss Functions – Part1:Regression». Medium. Obtenido de: https://medium.com/analytics-vidhya/a-comprehensive-guide-to-loss-functions-part-1-regression-ff8b847675d6

Hyndman, R., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O'Hara-Wild, M., Petropoulos, F., Razbash, S., Wang, E., & Yasmeen, F. (2020). _forecast: Forecasting functions for time series and linear models_. R package version 8.12. Recuperado: http://pkg.robjhyndman.com/forecast.

Hyndman, RJ., & Khandakar ,Y. (2008). “Automatic time series forecasting: the forecast package for R.” _ Journal of Statistical Software_, *26*(3), 1-22. Recuperado de: http://www.jstatsoft.org/article/view/v027i03.

Keydana, S. (23 de mayo de 2017). RPubs. «Time series forecasting - with deep learning». Rpubs. Recuperado

de: https://rpubs.com/zkajdan/279967

Olah, C. (27 de agosto de 2015). «Understanding LSTM networks». Colah's blog. Recuperado de: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Phillips, N.D (2018). YaRrr! The Pirate's Guide to R. Bookdown. Se encuentra en: Recuperado de https://bookdown.org/ndphillips/YaRrr/

R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Recuperado de https://www.R-project.org/

RStudio Team (2019). RStudio: Integrated Development for R. RStudio, Inc., Boston, MA. Recuperado de http://www.rstudio.com/.

Torres, J. (2018). «Learning process of a neural network». Medium. Se encuentra en: https://towardsdatascience.com/how-to-artificial-neural-networks-learn-773e46399fc7

Trapletti, T., & Hornik,K. (2019). tseries: Time Series Analysis and Computational Finance. R package version 0.10-47. Recuperado de https://CRAN.R-project.org/package=tseries

Wickham, H., & Seidel,D. (2020). scales: Scale Functions for Visualization. R package version 1.1.1. Recuperado de https://CRAN.R-project.org/package=scales