Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Predicción de series temporales mediante técnicas de aprendizaje automático. Automatización del proceso

Autor: Guillermo Javier Nieto Benito

Tutores: Amparo Núñez Reyes

Fernando Pavón Pérez

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2023

Trabajo Fin de Grado
Grado en Ingeniería Electrónica, Robótica y Mecatrónica

Predicción de series temporales mediante técnicas de aprendizaje automático. Automatización del proceso

Autor:
Guillermo Javier Nieto Benito

Tutores:
Amparo Núñez Reyes
Profesora Titular

Fernando Pavón Pérez
CEO Gamco S.L

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2023

Trabajo Fin de Grado: Predicción de series temporales mediante técnicas de aprendizaje automático. Automatización del proceso

Autor: Guillermo Javier Nieto Benito

Tutores: Amparo Núñez Reyes
Fernando Pavón Pérez

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Este trabajo ha sido posible gracias al proyecto PREDDICCO (Predicciones y decisiones dinámicas con control de contingencias y optimización de la toma de decisiones en base a modelos predictivos y simulación) EXP 00136057 / TIC-20200448, financiado por el CDTI (Centro para el Desarrollo Tecnológico Industrial). Durante la fase del proyecto: Simulación de datos reales, generación de conjuntos de datos y simulación de la respuesta del sistema. Donde se ha disfrutado de una beca de investigación en AICIA (Asociación de Investigación y Cooperación Industrial de Andalucía), dentro del proyecto "Generation of datasets created by the Generative Adversial Network (GAN) to train predictive models" con referencia: PI-2201/24/2022.

En primer lugar, quiero agradecer a mis tutores, Amparo Núñez Reyes y Fernando Pavón Peréz, por depositar su confianza en mí y ofrecerme la posibilidad de ser parte de este proyecto, que ha hecho que me interese por el campo de la inteligencia artificial y el aprendizaje automático.

Gracias al equipo de GAMCO S.L, en especial a Álvaro Muñoz y, de nuevo, a Fernando Pavón, por su infinita paciencia y que, con sus consejos y conocimiento me han guiado a través del fascinante mundo de la inteligencia artificial y han conseguido despertar mi interés por este campo.

Por último, pero no menos importante, quiero agradecer a mi familia y amigos más cercanos por su apoyo y ánimos.

Guillermo Javier Nieto Benito

Sevilla, 2023

Resumen

La predicción de series temporales es el proceso de predecir valores o patrones futuros en una secuencia de datos ordenados en el tiempo. Esto involucra analizar el comportamiento pasado de una serie temporal y utilizar esta información para realizar predicciones sobre su comportamiento futuro. Gracias a esta predicción, las empresas pueden tomar decisiones informadas sobre la gestión del inventario, la asignación de recursos y la predicción de la demanda, entre otros. La predicción de series temporales es fundamental en campos como las finanzas, la meteorología, la demanda energética o la salud.

Existen numerosas técnicas para este fin y podemos encontrar desde métodos clásicos estadísticos hasta los métodos más recientes de aprendizaje automático: las redes neuronales de aprendizaje profundo. En este Trabajo Fin de Grado se propone emplear una de las últimas técnicas aplicadas con bastante éxito en la predicción de series temporales: las redes neuronales recurrentes con memoria a corto y largo plazo o LSTM (Long Short-Term Memory). También se utilizarán, a modo de comparación, dos modelos más simples: modelos lineales autorregresivos con entradas exógenas o ARX (AutoRegressive with eXogenous inputs) y modelos de red neuronal con función de base radial o RBF (Radial Basis Function). Se aplicarán estos tres modelos en la predicción de dos conjuntos de datos diferentes: el volumen de llamadas de un centro de atención al cliente (CAC) y el saldo diario acumulado de transacciones realizadas en un tipo de dispositivo de una gran entidad bancaria iberoamericana, de carácter confidencial, y ambos proporcionados por la empresa GAMCO S.L.

Una vez realizadas varias pruebas de predicción de forma manual y elegido un modelo de predicción, el último paso será automatizar el proceso (selección de entradas y ajuste de hiperparámetros) de predicción de series temporales, con el fin de simplificar y agilizar dicho proceso para futuras aplicaciones a diferentes series temporales.

Abstract

Time series forecasting is the process of predicting future values or patterns in a sequence of data points ordered over time. This involves analyzing the past behavior of a time series and using that information to make predictions about its future behavior. Thanks to this prediction, businesses can make informed decisions about inventory management, resource allocation and demand forecasting, among others. Time series forecasting is essential in fields such as finance, meteorology, energy demand or healthcare.

There are numerous techniques for this purpose and they range from classical statistical methods to the most recent methods in machine learning: deep learning neural networks. In this Final Degree Project it is proposed to use one of the latest techniques applied with great success in time series forecasting: LSTM or Long Short-Term Memory recurrent neural networks. Two simpler models will also be used for comparison: linear AutoRegressive models with eXogenous inputs or ARX and Radial Basis Function or RBF neural network models. These three models will be applied in the prediction of two different data sets: the volume of calls from a customer attention center (CAC) and the accumulated daily balance of transactions carried out on one type of device from a large Ibero-American banking entity, which is a confidential dataset, both provided by the company GAMCO S.L.

After performing several forecast tests manually and selecting a predictive model, the last step will be to automate the process (input selection and hyperparameter tuning) of time series forecasting, in order to simplify and speed up the process for future predictions of different time series.

Índice Abreviado

Índice

Índice de Figuras

Índice de Tablas

Notación

EMV	Estimación de máxima verosimilitud
LMS	Método de mínimos cuadrados
t	Tiempo
k	Observación
$S(t)$	Serie temporal con referencia de tiempo t
$\{s_k\}$	Conjunto de valores de la serie temporal $S(t)$
N	Número de observaciones
d	Desfase
$\boldsymbol{X}$	Matriz o conjunto de entradas de un modelo
X^i	Variable i-ésima de entrada de un modelo (vector columna)
X_k	Vector de entradas de un modelo en el instante k (vector fila)
x_k^i	Elemento i-ésimo de un conjunto de entradas, $\boldsymbol{X}$, en el instante k
m	Número de variables de entrada
$\boldsymbol{Y}$	Matriz o conjunto de salidas de un modelo
Y^j	Variable j-ésima de salida (etiqueta) de un modelo (vector columna)
Y_k	Vector de salidas de un modelo en el instante k (vector fila)
y_k^j	Elemento j-ésimo de un conjunto de salidas, $\boldsymbol{Y}$, en el instante k
n	Número de variables de salida
h	Horizonte de predicción
$\hat{y}_k$	Predicción realizada por un modelo en el instante k
CE	Conjunto de entrenamiento
CP	Conjunto de prueba
CN	Conjunto nuevo
AR	Modelo autorregresivo
ARX	Modelo autorregresivo con entradas exógenas
MA	Modelo de media móvil
$ARMA$	Modelo autorregresivo de media móvil
$ARIMA$	Modelo autorregresivo integrado de media móvil
$SARIMAX$	Modelo autorregresivo integrado de media móvil estacional con entradas exógenas

VAR	Modelo de autorregresión vectorial
MLP	Perceptrón multicapa
RBF	Función de base radial
NN	Red neuronal
RNN	Red neuronal recurrente
$LSTM$	Red neuronal con memoria de largo y corto plazo
μ	Media aritmética
σ^2	Varianza
$\rho_{X,Y} = corr(X,Y)$	Coeficiente de correlación entre las variables X e Y
$cov(X,Y)$	Covarianza entre las variables X e Y
$E[A]$	Esperanza matemática de A
a_i	Coeficiente lineal de la i-ésima entrada en un modelo ARX
$\phi_j(k)$	Salida de la j-ésima neurona en una red neuronal RBF en el instante k
$C_j(k)$	Vector de centros de la j-ésima neurona en una red neuronal RBF en el instante k
σ_j	Dispersión (anchura) de la gaussiana de la j-ésima neurona en una red neuronal RBF
$\omega_j(k)$	Peso de la conexión de la j-ésima neurona con la salida en una red neuronal RBF en el instante k
σ	Función de activación sigmoidal
$tanh$	Función de activación tangente hiperbólica
f_k	Valor de la puerta de olvido en una LSTM en el instante k
i_k	Valor de la puerta de entrada en una LSTM en el instante k
o_k	Valor de la puerta de salida en una LSTM en el instante k
C_k	Valor del estado de celda en una LSTM en el instante k
$\hat{C}_k$	Valor candidato del estado de celda en una LSTM en el instante k
h_k	Valor del estado oculto en una LSTM en el instante k
W	Matriz de pesos en una red neuronal
b	Sesgo de una conexión en una red neuronal
ECM	Error cuadrático medio
EAM	Error absoluto medio
$MAPE$	Error absoluto medio porcentual
$MAAPE$	Error arcotangente absoluto medio porcentual
$SMAPE$	Error absoluto medio porcentual simétrico
CAC	Centro de atención al cliente

1 Introducción

Una serie temporal es una secuencia de observaciones medidas en determinados momentos de tiempo y ordenadas cronológicamente. Los datos pueden estar espaciados a intervalos iguales (como el precio diario de las acciones) o desiguales (como las transacciones realizadas en un cajero automático). Cada punto de datos en una serie temporal está asociado con una marca de tiempo que indica cuándo se recopiló la observación. Matemáticamente, se puede expresar como:

$$S(t) = \{s_k\}_{k=1,2,3,...,N} = [s_1, s_2, s_3, ..., s_k, ..., s_N] \tag{1.1}$$

Donde k es el instante de tiempo, s_k es cada una de las observaciones de la serie temporal y N es el número de observaciones. Es importante mencionar que s_1 es la observación más antigua, mientras que s_N es la más reciente.

Las series temporales se pueden encontrar en diversos campos y aplicaciones, como:

- Finanzas: un ejemplo muy popular es el de los precios de acciones.
- Meteorología: temperatura, precipitaciones, humedad, velocidad del viento, radiación solar, etc.
- Salud: signos vitales (pulso, respiración), electrocardiogramas (ECG), niveles de glucosa, patrones de sueño (ligero, profundo, REM)
- Demanda: demanda eléctrica, demanda de productos, demanda de personal, tráfico, etc.

En la *Figura 1.1* se muestran algunos ejemplos reales de series temporales.

La recolección de datos de series temporales tiene varios propósitos:

- Análisis y comprensión: las series temporales se pueden estudiar y analizar en busca de patrones, tendencias o estacionalidad que presentan las variables a lo largo del tiempo. Esto puede ayudar a conocer mejor el comportamiento de la serie y las relaciones presentes en la misma, facilitando la toma de decisiones. Utilizando el centro de atención telefónica como ejemplo, si se sabe que los fines de semana y festivos se recibe un número menor de llamadas se puede reducir el personal utilizado en esos días.
- Monitorización y detección: la monitorización de series temporales en tiempo real permite detectar anomalías, cambios y eventos en la variable de interés. Esto es especialmente importante en áreas como el control de calidad o los sistemas de vigilancia.
- Predicción: mediante el análisis de los datos históricos es posible realizar predicciones sobre el comportamiento o valores futuros de una serie temporal. Por tanto, el análisis de series

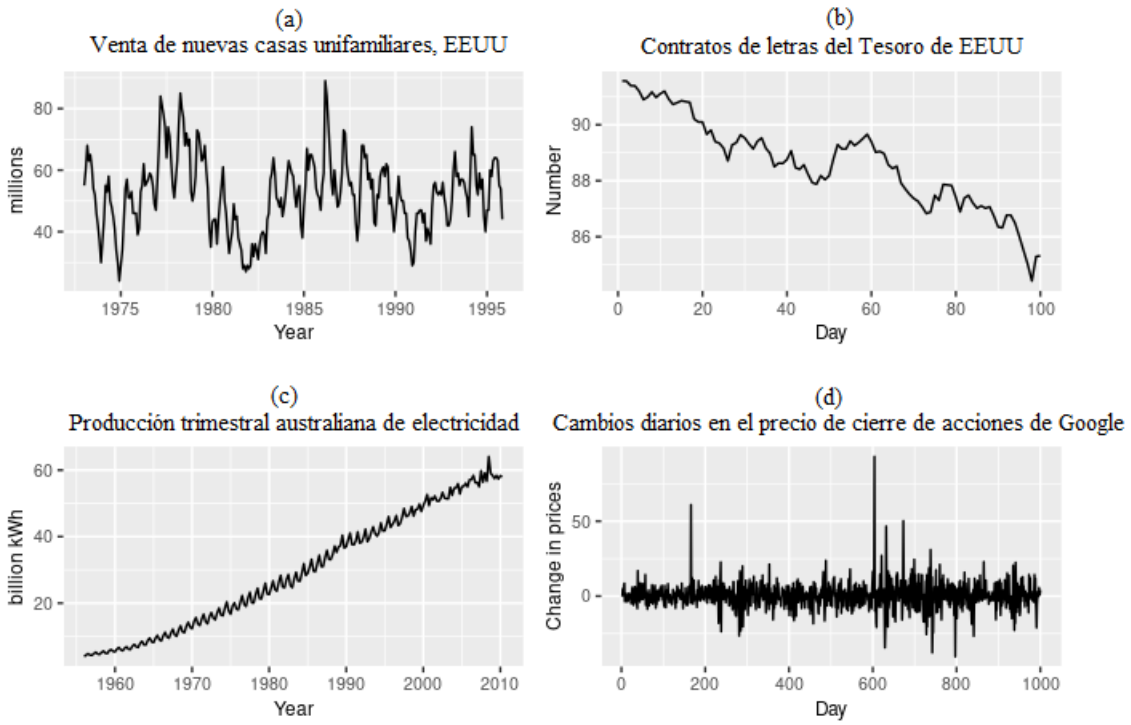


Figura 1.1 Ejemplos de series temporales reales [1].

temporales suele ser el paso previo a la predicción.

La predicción de series temporales resulta crucial para la toma de decisiones y la planificación estratégica en los diferentes campos y aplicaciones ya mencionados. Por ejemplo, la predicción de la demanda eléctrica ayuda a optimizar la generación de energía y asegurar la distribución eficiente de la misma; la predicción de series temporales meteorológicas es fundamental para la seguridad pública (huracanes, tornados...) y de gran importancia en otros sectores como la agricultura o el transporte; la predicción de los signos vitales de una persona permite la detección temprana de condiciones médicas, así como la preparación y la intervención inmediata en casos críticos.

Matemáticamente, la predicción de valores futuros de una serie temporal en un instante determinado de tiempo, k, se puede expresar como:

$$\hat{s}_{k+h} = f(s_{k-1}, s_{k-2}, ..., x_k^1, x_k^2, ...) \tag{1.2}$$

Donde:

- h: Es el horizonte de predicción, que indica el instante de tiempo futuro que se quiere predecir.
- $\hat{s}_{k+h}$: Representa el valor futuro y desconocido a predecir de la serie temporal.
- $s_{k-i}, i = 1, 2, ...$: Representa los valores pasados de la serie temporal. Se denominan entradas autorregresivas.
- $x_k^i, i = 1, 2, 3, ...$: Representa los valores de series temporales externas a la serie temporal original. Se denominan entradas exógenas.
- $f()$: Representa la función o modelo que relaciona los valores de entrada con el valor futuro a predecir.

El objetivo en la tarea de predicción es encontrar el conjunto de entradas ($\{s_{k-i}, x^i\}$), el tipo de modelo (f) y los parámetros del mismo (hiperparámetros) que mejor representan y aproximan la serie temporal s, y con ello sus valores futuros, s_{k+h}.

Por tanto, se puede dividir el proceso de predicción de series temporales en tres grandes tareas:

- Elección del modelo.
- Selección de entradas.
- Ajuste/Optimización de hiperparámetros.

En general, estas tareas pueden llegar a ser tediosas y repetitivas, por lo que resulta de gran interés automatizar estos procesos. La automatización de estas tareas presenta una serie de ventajas, como la simplificación de dichos procesos, una mayor eficiencia de trabajo y una mayor consistencia en los resultados al reducir errores humanos.

Es por estos motivos que surge el aprendizaje automático automatizado (AutoML, Automated Machine Learning), que hace referencia a la resolución de una tarea de aprendizaje automático de forma automatizada, tal que no se requiera ningún (o muy poco) esfuerzo manual. La importancia de AutoML reside en la simplificación, facilitación y agilización del desarrollo de aplicaciones de aprendizaje automático, reduciendo el tiempo y los recursos necesarios para la creación de modelos robustos y precisos.

1.1 Objetivos

Este trabajo se encuentra enfocado en la predicción de series temporales utilizando modelos lineales ARX y no lineales basados en redes neuronales RBF y LSTM y la automatización del proceso de creación de modelos para facilitar y agilizar la predicción de nuevas series temporales.

Para el estudio se han empleado dos conjuntos de datos que permiten, por un lado, predecir las llamadas entrantes en un centro de atención al cliente y por otro lado predecir el saldo diario acumulado de transacciones realizadas en diferentes buzoneras (dispositivos físicos de ingreso de efectivo) de un gran banco iberoamericano.

El objetivo final consiste en la automatización del proceso de creación de modelos predictivos de series temporales.

La metodología empleada en la presente investigación consta de los siguientes pasos:

- Preparación de grandes conjuntos de datos de series temporales (preprocesamiento y análisis), ambos facilitados por la empresa GAMCO S.L:
  - Conjunto de datos del servicio de atención al cliente 1004 de Telefónica entre los años 2001-2003.
  - Conjunto de datos más recientes (2019-2022) y confidenciales pertenecientes a una gran entidad bancaria iberoamericana
- Creación manual de diferentes modelos predictivos para algunas series temporales:
  - Modelos lineales autorregresivos con entradas exógenas (ARX, AutoRegressive model with eXogenous inputs).
  - Redes neuronales con función de base radial (RBF, Radial Basis Function).

  - Redes neuronales recurrentes con memoria de corto y largo plazo (LSTM, Long Short-Term Memory).
- Evaluación y comparación de los modelos predictivos.
- Automatización del proceso de creación de modelos predictivos y aplicación a diferentes series temporales.

1.2 Entorno de desarrollo

El trabajo se ha desarrollado en la plataforma MATLAB, debido a que ofrece una amplia gama de funciones y herramientas especializadas en el análisis y procesamiento de datos, lo que facilita la manipulación y visualización de series temporales.

Además, MATLAB proporciona el Deep Learning Toolbox, que es un conjunto de funciones y algoritmos avanzados para la construcción y el entrenamiento de redes neuronales de aprendizaje profundo, y que se utiliza en este trabajo para la creación de los modelos LSTM.

También es importante mencionar el Parallel Computing Toolbox, que ofrece funciones para el procesamiento en paralelo del código y que se implementa en este trabajo junto a la automatización del proceso de creación de modelos para reducir el tiempo empleado.

1.3 Estructura del documento

El documento se divide en 5 capítulos que se resumen a continuación:

- Capítulo 1: Introducción.
  En el primer capítulo se introduce el concepto de serie temporal, los diferentes campos y aplicaciones donde podemos encontrarlas y la importancia que presentan. Además, se explican los objetivos que se pretenden alcanzar en este Trabajo Fin de Grado.
- Capítulo 2: Estado del arte.
  En este capítulo se comienza revisando la literatura en el ámbito de la predicción de series temporales, presentando la evolución de las soluciones adoptadas por diferentes autores para este problema a lo largo del tiempo hasta la actualidad. Seguidamente, se presenta una breve revisión sobre el aprendizaje automático automatizado. Por último, se menciona la propuesta para este trabajo.
- Capítulo 3: Metodología.
  Una vez introducidos los conceptos iniciales y revisada la literatura, en este capítulo se explica de forma detallada la metodología seguida en este Trabajo Fin de Grado para abordar el problema de estudio. Primero, se definen una serie de conceptos y notación utilizados a lo largo del trabajo, además de explicar los pasos generales seguidos para la creación de modelos predictivos mediante aprendizaje supervisado. A continuación, se detallan las técnicas concretas aplicadas en los diferentes pasos, así como los distintos modelos empleados. Para finalizar, se presentan los algoritmos de automatización propuestos.
- Capítulo 4: Aplicación. Conjuntos de datos reales
  En este capítulo se introducen las métricas empleadas para la evaluación del rendimiento de los modelos de predicción de series temporales. En primer lugar, se realizan las pruebas con la serie temporal de volumen de llamadas (CAC) para comparar el rendimiento de los diferentes modelos ante un problema sencillo debido a su periodicidad. En segundo lugar, se presentan las pruebas realizadas para el conjunto de datos de saldo diario acumulado, que se

trata de un problema más complejo, y así comprobar si se obtienen las mismas conclusiones. Para finalizar, se parte del tipo de modelo que haya presentado mejores resultados y se aplican los algoritmos de automatización definidos a una serie temporal ya predicha, así como una nueva serie temporal perteneciente al segundo conjunto de datos.

- Capítulo 5: Conclusiones
  Por último, se resumen las conclusiones obtenidas.

2 Estado del arte

2.1 Modelos de predicción de series temporales

El análisis y la predicción de series temporales comenzó en las décadas de 1920 y 1930, con el trabajo de G. U. Yule [8] y J. Walker [9]: la primera aplicación real de modelos autorregresivos (AR) a conjuntos de datos.

Durante este tiempo, se introdujo el modelo de media móvil (MA) para eliminar las fluctuaciones periódicas en las series temporales, como las fluctuaciones debidas a la estacionalidad [10]. Herman Wold introdujo los modelos autorregresivos de media móvil (ARMA) para series estacionarias [11], pero fue incapaz de derivar una función de probabilidad que permitiera la estimación de máxima verosimilitud (EMV) de los parámetros del modelo [10]. Algunos ejemplos de aplicación de estos modelos a la predicción de series temporales se pueden encontrar en [12][13]

No fue hasta 1970 que esto último se logró, cuando se publicó el libro "Time Series Analysis: Forecasting and Control" de George E. P. Box y Gwilym M. Jenkins [14]. Antes de su desarrollo, los procedimientos de predicción de series temporales se basaban en modelos *ad hoc* y técnicas heurísticas. En particular, existía un conjunto de técnicas bajo el nombre de "suavizado exponencial", caracterizadas por el hecho de que un único modelo era empleado para la predicción de cualquier serie temporal. Estas técnicas podían ser programadas para generar predicciones para cualquier serie temporal de forma completamente automática [15]. Entre estas técnicas destacaba el método de Holt-Winters (1960) [16][17] como extensión del suavizado exponencial [18], que incorporaba componentes de tendencia y estacionalidad de la serie temporal.

El método de Box-Jenkins propuesto en el libro mencionado no se trataba de un método automático, sino que proporcionaba al usuario una serie de modelos (AR, MA, ARMA) y una estrategia mediante la cual, para una serie temporal concreta, se elegía un modelo de los mencionados según las propiedades de la serie temporal bajo estudio [15]. Este método consistía en tres etapas principales: identificación, estimación y diagnóstico [10][19], que se resumen a continuación.

En la etapa de identificación, se busca determinar el tipo de modelo y el orden del mismo que mejor se ajusten a los datos. Esto se consigue mediante gráficas de la serie temporal, análisis de la función de autocorrelación y otra información relevante [20]. En la etapa de estimación, se utilizan métodos estadísticos como la EMV o el método de los mínimos cuadrados (LMS) para estimar los parámetros del modelo identificado. Esto implica ajustar el modelo a los datos de la serie temporal y encontrar los valores óptimos de los parámetros. Finalmente, en la etapa de diagnóstico, se evalúa el rendimiento del modelo ajustado. Esto implica realizar pruebas de diagnóstico para verificar si

las predicciones realizadas por el modelo son adecuadas y ésto se consigue analizando la función de autocorrelación de los errores obtenidos en las predicciones [20].

El método de Box-Jenkins fue un hito importante en la predicción de series temporales al proporcionar al usuario un enfoque sistemático para la creación de modelos predictivos [19], encontrando aplicaciones de este método en artículos como [21][22]. Desde entonces, se desarrollaron y refinaron muchas extensiones y variantes del enfoque original, como los modelos ARIMA (AutoRegressive Integrated Moving Average) [23], SARIMA (Seasonal AutoRegressive Integrated Moving Average) [24] o SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous inputs) [25].[26]

A principios de la década de 1980, el econometrista Christopher Sims introdujo los modelos de autorregresión vectorial (VAR) [27] para modelar las interdependencias y relaciones dinámicas entre múltiples variables de una serie temporal (multivariante). Los modelos VAR son una generalización de los modelos AR donde, en lugar de modelar una sola variable en función de sus propios valores pasados, estos capturan las relaciones entre múltiples variables al incluir valores pasados de todas las variables en el sistema. Un ejemplo de aplicación es [28]

En esta misma década surgieron los modelos autorregresivos condicionales heterocedásticos o ARCH (1982) [29] y su variante generalizada, GARCH (1986) [30], aplicados principalmente a series temporales financieras. Una de las características que suelen presentar estas series es la heterocedasticidad condicional, que significa que la varianza de la serie temporal no es constante a lo largo de la misma, sino que varía con el tiempo. Este hecho no se tiene en cuenta en los modelos clásicos AR, ya que una de las premisas de estos es tener una varianza constante, tanto condicional como incondicional, a lo largo de la serie temporal [31]. Es por esto que se desarrollan los modelos ARCH, que solucionaban este problema.[32][33] son algunos ejemplos de predicción de series temporales mediante estos modelos.

Una limitación de todos estos modelos mencionados es su incapacidad para extraer o capturar las características que presentan las entradas exógenas. Sin embargo, la restricción principal de estos modelos basados en los modelos ARMA es una base lineal preestablecida, que plantea dificultades para que estos modelos aprendan y capturen dinámicas no lineales de una serie temporal determinada [26].

Con el crecimiento exponencial en la disponibilidad de datos y la capacidad computacional, comenzaron a ganar popularidad enfoques basados en los datos, en contraposición a los enfoques basados en el conocimiento que se utilizaban desde el principio. Mientras que los enfoques basados en el conocimiento dependen de expertos y experiencia en el campo de estudio para establecer reglas y relaciones presentes en un sistema, los enfoques basados en los datos emplean técnicas de Aprendizaje Automático para analizar grandes cantidades de datos y extraer conocimiento, patrones y relaciones ocultas presentes en ellos. Entre estos enfoques nos encontramos con las redes neuronales, árboles de decisión [34][35] y máquinas de soporte vectorial (SVM) [36][37].

A finales de 1980 y principios de 1990 comenzaron a utilizarse redes neuronales artificiales (ANN) en la predicción de series temporales, empezando con el perceptrón multicapa (MLP) [38][39]. Las redes neuronales se caracterizan por utilizar funciones de activación no lineales en cada nodo o neurona, lo que les permite capturar y modelar relaciones no lineales de un conjunto de datos. En este sentido, destacan las funciones de activación de base radial o RBF [38][40][41], capaces de capturar relaciones no lineales complejas e irregulares. Además, las redes neuronales tienen la propiedad de aproximación universal, que significa que son capaces de aproximar cualquier función

continua con mayor o menor precisión, lo que las convierte en un modelo flexible para un gran rango de aplicaciones.

Al mismo tiempo, comenzaron a utilizarse las redes neuronales recurrentes (RNN) como un enfoque prometedor para capturar dependencias temporales y modelar relaciones dinámicas complejas en conjuntos de datos secuenciales [42]. Este tipo de redes neuronales se caracterizaban por presentar conexiones recurrentes que permitían a la información propagarse no sólo hacia delante sino también hacia atrás, formando un bucle de realimentación que permitía a la RNN retener y procesar información secuencial o temporal. Estas conexiones convertían a las RNNs en una arquitectura profunda, formada por muchas capa, y, por tanto, son una de las primeras formas de aprendizaje profundo o Deep Learning (DL), aunque este término no apareció hasta más adelante, en 2006 [43]. Sin embargo, estas primeras versiones de RNN presentaban dificultades durante el entrenamiento de las mismas debido a valores desproporcionados de gradientes al tratar con secuencias de datos largas. Es lo que se conoce como el problema del gradiente explosivo/desvanecedor, lo cual limitó la eficacia de estos modelos y, por tanto, su utilización. No obstante, encontramos algunos artículos de aplicación de estos modelos a la predicción de series temporales como [44][45]

En 1997, Hochreiter y Schmidhuber [46] introdujeron la arquitectura LSTM (Long Short-Term Memory), una variante de RNN diseñada específicamente para solucionar los problemas de las RNNs originales, principalmente el problema del gradiente explosivo/desvanecedor. Esta arquitectura se caracteriza por presentar celdas de memoria y mecanismos de compuerta especializados. Las celdas de memoria permiten a la LSTM guardar y utilizar información durante periodos de tiempo prolongados, permitiendo al modelo capturar relaciones temporales de largo plazo presentes en las series temporales. Los mecanismos de compuerta facilitan la retención y utilización eficiente de la información al controlar el flujo de esta dentro de la red. Por estos motivos, las redes LSTM destacan en el modelado de relaciones dinámicas complejas y patrones irregulares y en la elaboración de predicciones precisas. Las redes LSTM ganaron una popularidad significativa y fueron aplicadas en la predicción de series temporales debido a su arquitecura única y sus características. Se encuentran numerosas aplicaciones de estos modelos en la literatura [47][48][49][50]

De la arquitectura LSTM surgen otras íntimamente relacionadas. La LSTM bidireccional [51] procesa la información de entrada un sentido (hacia delante) y en el otro (hacia atrás) para capturar el contexto pasado y futuro de forma simultánea [52][53]. La arquitectura de LSTMs apiladas [54], que utiliza varias LSTMs en serie [52][55]. La arquitectura GRU (Gated Recurrent Unit) [56] surge con el objetivo de simplificar la arquitectura LSTM y reducir su complejidad computacional [57][58]. Otros ejemplos de aplicación de estas variantes de LSTM a la predicción de series temporales son [59][60][61][62].

2.2 Aprendizaje automático automatizado (AutoML)

El aprendizaje automático automatizado ha evolucionado a lo largo de la última década motivado principalmente por la creciente necesidad de facilitar el aprendizaje automático y hacerlo más accesible para un público más amplio.

Las raíces de AutoML se remontan a los primeros esfuerzos en la optimización de hiperparámetros, con trabajos como [63] (2011) o [64] (2012).

En 2013 se introdujo Auto-WEKA [65], que marcó un esfuerzo temprano en la selección automatizada de modelos, además del ajuste de hiperparámetros. Este sistema surge de WEKA, un software de código abierto escrito en Java que proporciona herramientas para el preprocesamiento

y visualización de datos, y la implementación de algoritmos de aprendizaje automático.

En 2015 se introdujo Auto-Sklearn [66] como una extensión de la idea de Auto-WEKA, incluyendo la automatización del preprocesamiento de los datos y la extracción de características. Se basa en la librería de aprendizaje automático sickit-learn en el lenguaje de programación Python. Más tarde se desarrolla una versión mejorada (Auto-Sklearn 2.0) que aumenta la precisión y reduce el tiempo empleado con respecto a la primera versión (Auto-Sklearn 1.0) [67].

Neural Architecture Search (NAS) ganó importancia con artículos como [68] y [69]. Estos introdujeron la idea de utilizar algoritmos para descubrir automáticamente arquitecturas de redes neuronales.

Gigantes tecnológicos como Google y Microsoft introdujeron plataformas comerciales de AutoML (Google Cloud AutoML y Azure AutoML, ambas en 2018), lo que hizo que AutoML fuera accesible a una audiencia más amplia.

En este Trabajo Fin de Grado se propone automatizar los procesos de selección de entradas y optimización de hiperparámetros, partiendo de un modelo elegido previamente tras realizar unas primeras pruebas de forma manual. Para ello, se utilizarán dos conjuntos de datos, facilitados por la empresa GAMCO S.L: el volumen de llamadas entrantes en un centro de atención al cliente y el saldo diario acumulado de transacciones realizadas en las buzoneras (dispositivos de ingreso de efectivo) de una gran entidad bancaria iberoamericana.

3 Metodología

En este capítulo se comienza definiendo y aclarando una serie de conceptos y notación relacionados con las series temporales y la resolución de problemas de aprendizaje automático. Seguidamente, se define una serie de pasos generales para la resolución de este tipo de problemas, cuyo objetivo final es la creación de modelos predictivos. Por último, se detallan las técnicas concretas empleadas en el desarrollo de este Trabajo Fin de Grado.

3.1 Conceptos y notación

En este primer apartado se definen algunos conceptos y se presenta la notación utilizada a lo largo del trabajo.

3.1.1 Series temporales: variables autorregresivas y variables objetivo o de predicción

Una de las características de las series temporales y, quizás la más importante (ya que las diferencia de cualquier otro conjunto de datos), es la existencia de lo que se conoce como dependencia temporal. Esto quiere decir que el valor de una serie temporal en un instante determinado, t, está relacionado, en mayor o menor medida, con valores de la misma serie en instantes pasados: $t-1$, $t-2$, $t-3$,... De este concepto surge otro de gran importancia en la predicción de series temporales: las variables autorregresivas.

Una variable autorregresiva de una serie temporal es la propia serie temporal retardada en el tiempo. A ese retardo se le llama desfase y se denotará como d. No obstante, el desfase también se puede utilizar para adelantar la serie temporal, definiendo así la variable objetivo o de predicción. Para hablar de instantes desfasados se utiliza la notación $t+d$.

El desfase indica qué valor pasado o futuro hay que tomar de la serie temporal original para cada instante de tiempo. Es importante aclarar que el valor de una serie temporal en $t+d$, con $d<0$ puede seguir siendo un valor futuro si los datos llegan con retraso. Por ejemplo, si se tiene una serie temporal de intervalo horario y los datos nos llegan con dos horas de retraso, los valores de dicha serie en $t-1$ y $t-2$ serían desconocidos y, por tanto, no se podrían definir variables autorregresivas con esos desfases, sino que se considerarían variables objetivo o de predicción.

A continuación, se introduce la notación empleada en este trabajo. Se puede expresar una serie temporal cualquiera como:

$$S(t) = \{s_k\}_{k=1,2,3,...,N} = [s_1, s_2, s_3, ..., s_k, ..., s_N] \tag{3.1}$$

Para las definiciones siguientes se considerará, por simplicidad, que desfases negativos ($d < 0$) indican pasado y desfases positivos ($d \geq 0$) indican futuro. Una variable autorregresiva de una serie temporal, $S(t)$, se expresará como:

$$X(t) = S(t+d) = \{s_{k+d}\}_{k=1,2,...,N} = [s_{1+d}, s_{2+d}, ..., s_{k+d}, ..., s_{N+d}], \quad d < 0 \tag{3.2}$$

De igual manera, se puede definir una variable objetivo o de predicción como:

$$Y(t) = S(t+d) = \{s_{k+d}\}_{k=1,2,...,N} = [s_{1+d}, s_{2+d}, ..., s_{k+d}, ..., s_{N+d}], \quad d \geq 0 \tag{3.3}$$

Como se puede intuir, al aplicar un desfase cualquiera a una serie temporal se pierden datos en la variable desfasada, ya que no se conocen los datos de la serie temporal en los instantes de tiempo donde $(k+d) \notin [1,2,...,N]$. El número de datos perdidos será igual al valor absoluto del desfase, $|d|$.

3.1.2 Creación de un modelo predictivo mediante aprendizaje supervisado

En la *Figura 3.1* se muestra la jerarquía de los distintos algoritmos de aprendizaje automático según el tipo de entrenamiento: supervisado, no supervisado o de refuerzo.

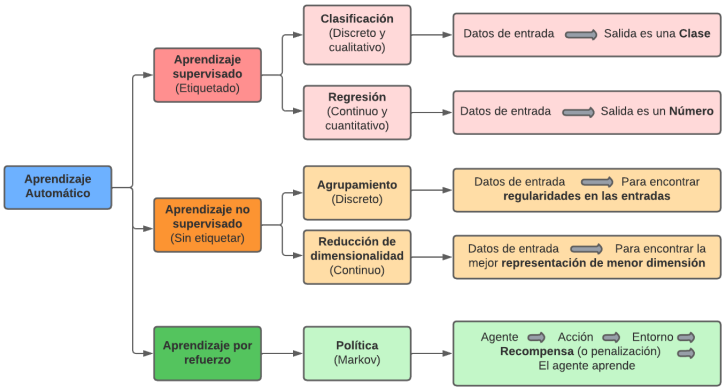


Figura 3.1 Diagrama de los algoritmos de aprendizaje automático [2].

En este sentido, la predicción de series temporales se considera un problema de aprendizaje supervisado y, concretamente, un problema de regresión, puesto que el objetivo es predecir un valor numérico (no categórico).

En los problemas de aprendizaje supervisado se trabaja con conjuntos de datos "etiquetados" para el entrenamiento. Esto quiere decir que los modelos se entrenan con parejas de entradas-salida, de forma que el modelo conoce el valor de la salida para cada conjunto de valores de entrada. El objetivo es que el modelo aprenda y capture la relación existente entre dichas variables de entrada y la variable objetivo.

En la predicción de series temporales se utilizan, principalmente, las variables autorregresivas de la propia serie temporal como entradas, pero también es común utilizar lo que se conoce como variables exógenas, que son series temporales externas a la serie temporal a predecir y pueden afectar su comportamiento. Por ejemplo, datos de tiempo, festivos u otros datos que puedan tener

cierta relación con la serie temporal a predecir.

El conjunto de datos empleado en un problema de aprendizaje supervisado suele expresarse como matriz, $\boldsymbol{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$, donde sus dos elementos son las matrices:

$$\boldsymbol{X} = [X^1, X^2, ...X^i, ..., X^m] \tag{3.4}$$
$$\boldsymbol{Y} = [Y^1, Y^2, ...Y^j, ..., Y^n] \tag{3.5}$$

Donde:

- $\boldsymbol{X}$: Representa la matriz o conjunto de entradas.
- $X^i = \{x_k^i\}_{k=1,2,...,N}$: Es la i-ésima variable de entrada (vector columna).
- x_k^i: Es el valor k-ésimo de la variable de entrada X^i.
- m: Es el número de variables de entrada.
- $\boldsymbol{Y}$: Representa la matriz o conjunto de salidas.
- $Y^j = \{y_k^j\}_{k=1,2,...,N}$: Es la j-ésima variable de salida (vector columna).
- y_k^j: Es el valor k-ésimo de la variable de salida Y^j.
- n: Es el número de variables de salida.

En este trabajo se considera una sola salida, por lo que $n = 1$ y, por tanto, $\boldsymbol{Y} = Y^j = Y$.

Un concepto importante que se utilizará más adelante es el de vector de entrada, que es el conjunto de entradas para un instante determinado de tiempo, k, y se corresponde con cada fila del conjunto $\boldsymbol{X}$. Se denotará un vector de entrada como $X_k = [x_k^1, x_k^2, ..., x_k^m]$.

A continuación, se explican los pasos seguidos para la creación de un modelo predictivo cualquiera, que será necesario para entender algunas técnicas empleadas más adelante. En la *Figura 3.2* se ilustra este proceso.

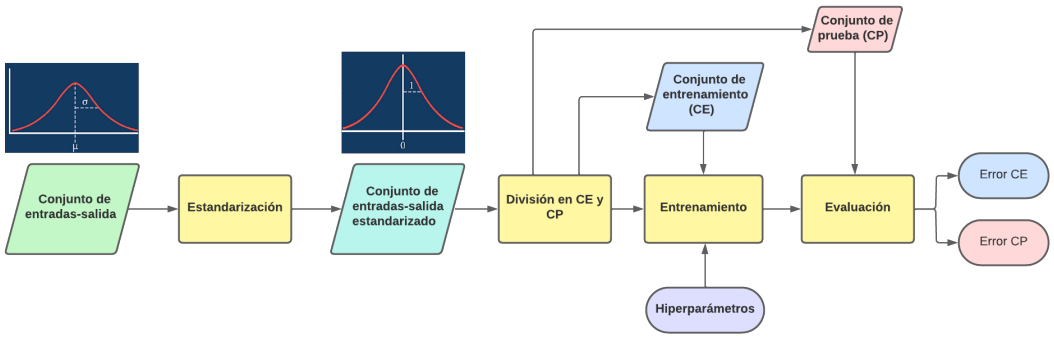


Figura 3.2 Ilustración del proceso de creación de un modelo predictivo.

1. Construcción de un conjunto de entradas-salida (a menudo se denominará como Cx o CX), $\boldsymbol{D}$. En el siguiente apartado se explican las técnicas de selección de entradas empleadas.

2. Estandarización del conjunto, de forma que cada variable tenga media nula ($\mu = 0$) y desviación típica unidad ($\sigma = 1$). Para ello se aplica la siguiente transformación a cada variable del conjunto, incluida la salida:

$$Z^i = \frac{X^i - \overline{X}^i}{\sigma(X^i)} \tag{3.6}$$

   La estandarización evita problemas derivados de la presencia de diferentes escalas entre las variables del conjunto, como la predominancia de unas variables frente a otras o la sensibilidad de algunos algoritmos a la magnitud relativa de las variables.

3. División en conjunto de entrenamiento (CE) y conjunto de prueba (CP). El primero será usado para entrenar el modelo y el segundo se utilizará para su evaluación. En este trabajo se utiliza la división 80-20 (80 % de los datos para el entrenamiento o CE y 20 % de los datos para la evaluación o CP). Los conjuntos de datos no se barajan, puesto que es importante preservar la dependencia temporal presente en los datos de series temporales.

4. Entrenamiento del modelo. En este paso se estiman los valores de los parámetros que definen el modelo (por ejemplo, los coeficientes lineales en los modelos ARX).

5. Evaluación del modelo. Una vez entrenado el modelo, se realizan predicciones para ambos conjuntos de entrenamiento y prueba utilizando diferentes métricas de evaluación, como pueden ser el error cuadrático medio (ECM) o el error absoluto medio porcentual (MAPE). Esto resulta en un error (diferencia entre el valor esperado y la predicción) en CE y un error en CP que definen el rendimiento del modelo. El error en CP es el más indicativo, puesto que se trata de un caso que no ha utilizado el modelo para el entrenamiento y, por tanto, se asemeja más a la realidad. No obstante, es importante tener en cuenta ambos errores y la diferencia entre ellos (una diferencia elevada puede ser indicio de sobreentrenamiento).

Por último, se puede dividir un modelo predictivo en tres componentes principales:

- **Tipo de modelo**.
- **Conjunto de entradas** utilizadas para el entrenamiento.
- **Hiperparámetros** que definen el modelo.

En el siguiente apartado se detallan los algoritmos y métodos empleados para la obtención y optimización de cada componente.

3.2 Metodología aplicada

En esta sección se presenta una visión general de la metodología aplicada y los distintos pasos seguidos. Además, se detallan los algoritmos y técnicas empleados en cada paso. El procedimiento se puede dividir en cuatro grandes pasos: preparación de los datos, elección del modelo, selección de entradas y ajuste de hiperparámetros, que aparecen representados en la *Figura 3.3*.

1. Preparación de los datos

El primer paso para la predicción de una serie temporal es la obtención y preparación de los datos del problema. En este trabajo los conjuntos de datos han sido facilitado por la empresa GAMCO S.L., por lo que se omite el paso de obtención de datos. El siguiente paso es la preparación de los datos, que se puede dividir en tres tareas ordenadas:

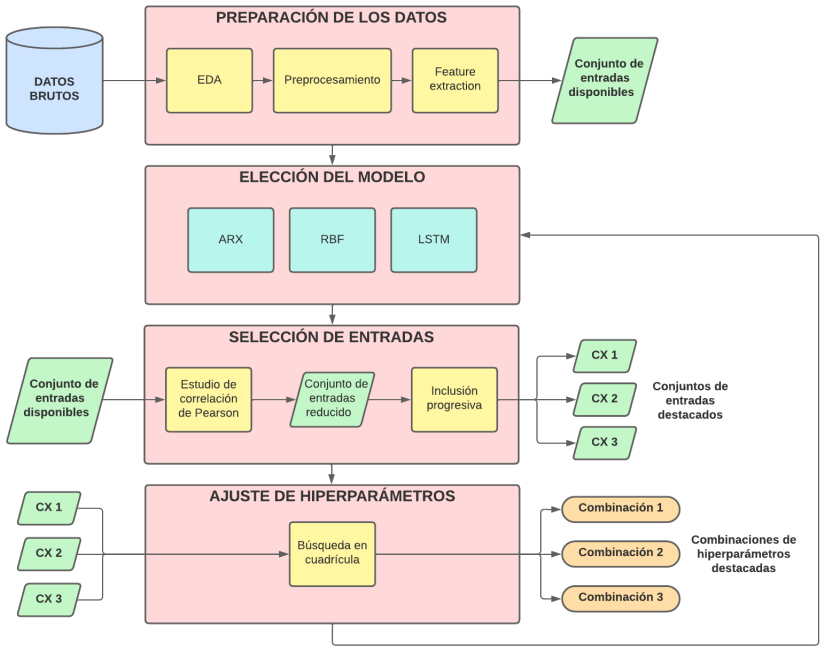


Figura 3.3 Ilustración de la metodología aplicada.

1. Análisis exploratorio de los datos (EDA, Exploratory Data Analysis), que engloba diferentes subtareas:
   - Identificación de las diferentes variables o series temporales que componen el conjunto de datos, así como información básica de cada una: paso temporal (horario, diario, irregular, etc.), número de registros, rango temporal.
   - Visualización de las series temporales. Esto tiene varios objetivos: obtención de una idea general sobre la serie temporal e identificación de patrones, tendencia, estacionalidad y anomalías (valores atípicos, valores faltantes, valores negativos cuando no procede, errores). Esta tarea es la base para el posterior preprocesamiento.
   - Análisis estadístico de las series temporales: media, varianza y rango de valores para entender mejor la distribución de los datos. Este análisis se puede realizar sobre distintos periodos de tiempo. Por ejemplo, si se tiene una serie temporal diaria, se puede realizar un análisis estadístico para cada día de la semana, y con ello determinar la influencia del día de la semana en el comportamiento de los datos.
2. Preprocesamiento de los datos, que consiste en aplicar una solución para los casos mencionados en el punto anterior: filtrado de valores atípicos, rellenado de datos faltantes, corrección o eliminación de valores erróneos, etc.. La solución aplicada dependerá del objetivo que se persigue y la naturaleza de los datos. Este paso es fundamental para evitar errores y resultados subóptimos e indeseados en el modelado de las series temporales.
3. "Feature extraction/engineering": definición y extracción de posibles características o variables relevantes para utilizar como entradas en los modelos predictivos y que puedan mejorar el rendimiento de los mismos. Algunas variables de ejemplo son: media, mínimo y máximo móvil según diferentes ventanas de tiempo, variables de tiempo (día, hora, mes) y variables de tipo seno y coseno aplicados a variables de tiempo. El objetivo es proporcionar al modelo variables que ayuden a capturar características de la serie temporal, como tendencia, estacionalidad, periodicidad o cambios en el comportamiento ante distintos eventos (fines de semana, festivos, principio de mes, etc.).

2. Elección del modelo

Por motivos que se explicarán en el siguiente apartado, el segundo paso seguido es la elección del tipo de modelo empleado: suavizado, lineal, basado en red neuronal, etc. Como ya se ha comentado, en este trabajo se utilizan tres modelos diferentes: **ARX**, **RBF** y **LSTM**, que se detallan a continuación.

Modelos autorregresivos con entradas exógenas (ARX)

El modelo ARX es una extensión del modelo autorregresivo (AR) que incorpora variables exógenas (X) para modelar la relación entre la variable objetivo y otras variables explicativas.

Un modelo autorregresivo (AR) representa una serie temporal como combinación lineal de sus valores pasados, mientras que un modelo de entradas exógenas (X) representa una serie temporal como combinación lineal de una serie de variables externas.

La salida de un modelo ARX de serie temporal en un instante concreto de tiempo, k, con una única variable de salida y múltiples entradas se puede representar mediante la siguiente ecuación, que distingue los términos de variables autorregresivas y exógenas por separado:

$$\hat{y}_k = \hat{s}_{k+h} = \sum_{i=1}^{m} a_i s_{k+d(i)} + \sum_{j=1}^{n} b_j x_k^j, \quad d(i) < 0 \tag{3.7}$$

Donde:

- k: Hace referencia a la k-ésima observación del conjunto de datos.
- h: Representa el horizonte de predicción.
- $\hat{y}_k = \hat{s}_{k+h}$: Representa la predicción del modelo en el instante k.
- i: Hace referencia a la i-ésima entrada autorregresiva.
- m: Es el número de entradas autorregresivas.
- a_i: Es el coeficiente de la i-ésima entrada autorregresiva.
- $d(i)$ Representa el desfase asociado a la i-ésima entrada autorregresiva
- $s_{k+d(i)}$: Representa la k-ésima observación de la i-ésima entrada autorregresiva.
- j: Hace referencia a la j-ésima entrada exógena.
- b_j: Es el coeficiente de la j-ésima entrada exógena.
- n: Es el número de entradas exógenas.
- x_k^j: Representa la k-ésima observación de la j-ésima entrada exógena.

Sin embargo, es más simple expresar la predicción haciendo referencia al conjunto de entradas, $\boldsymbol{X}$ y utilizando la notación ya empleada:

$$\hat{y}_k = \sum_{i=1}^{m} a_i x_k^i \tag{3.8}$$

Donde x_k^i es el elemento i-ésimo del vector de entradas X_k en el instante k.

Los coeficientes a_i son los parámetros del modelo ARX que se estimarán durante el entrenamiento.

En este sentido, se emplea el algoritmo de Mínimos Cuadrados Recursivos (MCR) ponderados, mediante el cual se calculan recursivamente los coeficientes a_i para minimizar la función de coste de mínimos cuadrados. Para este algoritmo y por tanto, para la creación de modelos ARX, se emplean dos hiperparámetros:

- γ: Factor de olvido. Determina la importancia de observaciones pasadas en el algoritmo MCR. Controla la velocidad a la cual se olvidan las observaciones pasadas en relación con observaciones más recientes. El factor de olvido toma valores en el rango $0 < \gamma \leq 1$. Un valor de 1 significa que todas las observaciones son igual de importantes, mientras que un valor menor a 1 da menor importancia a observaciones pasadas. En la práctica suele tomar valores entre 0.98 y 1 [70].
- δ: Es un factor que multiplica a la matriz identidad, I, para inicializar la matriz de covarianza de las observaciones ponderadas, $P(0)$.

Redes neuronales con función de base radial (RBF)

La idea fundamental de las redes neuronales RBF aplicadas a tareas de regresión es la aproximación o interpolación de funciones mediante sumas de funciones de base radial. Una función de base radial es una función no lineal cuyo valor depende de la distancia entre la entrada y un punto fijo. Algunos ejemplos de funciones de base radial son: multicuadrática, cuadrática inversa y gaussiana, entre otras. En la *Figura 3.4* se ilustra la idea de aproximación de funciones mediante sumas de funciones de base radial, concretamente gaussianas.

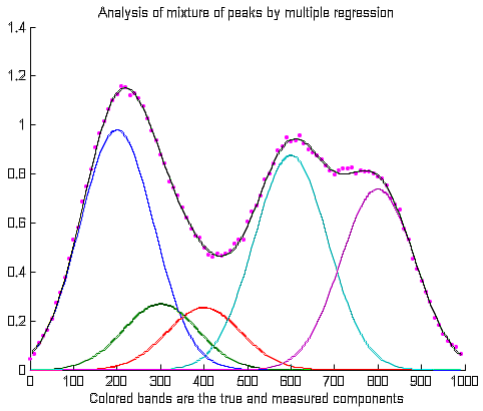


Figura 3.4 Aproximación de una función mediante suma de gaussianas [3].

Por tanto, las redes neuronales RBF son modelos no lineales capaces de capturar patrones o relaciones no lineales entre las variables independientes y la variable objetivo del problema.

Una red neuronal RBF está generalmente formada por tres capas: una capa de entrada, una capa oculta y la capa de salida. En la *Figura 3.5* se representa la estructura típica de una red RBF de múltiples entradas y una salida (MISO), como es el caso que se estudia en este trabajo.

La capa de entrada recibe el vector de entradas, por lo que tendrá tantas neuronas como entradas tenga la red. Esta capa se encuentra directamente conectada con la capa oculta, sin aplicar ningún peso en las conexiones.

La capa oculta está formada por un número variable de neuronas y la elección de dicho número es

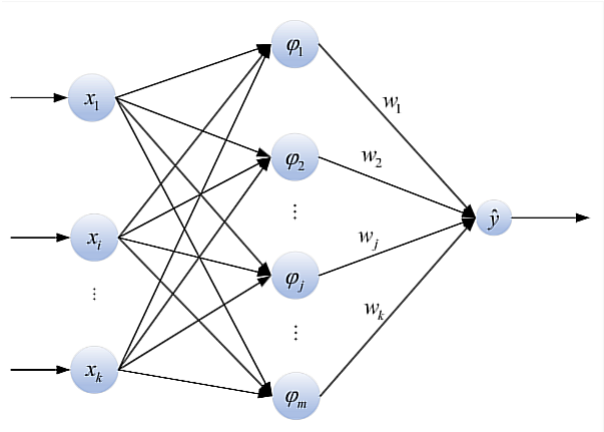


Figura 3.5 Arquitectura de una red neuronal de función de base radial [4].

una tarea complicada. Este es uno de los hiperparámetros de las redes RBF y, al igual que cualquier otro hiperparámetro, deberá ser optimizado experimentalmente.

Cada neurona de la capa oculta contiene un vector "prototipo", que se denominará vector de centros, y en cada una de ellas se aplica una función de base radial como función de activación. La función de activación más utilizada en las redes RBF es la función gaussiana [71] debido a sus propiedades, que permiten y facilitan las tareas interpolación y generalización. Mediante esta función, la activación de cada neurona decrece con la distancia del vector de entradas al vector de centros. Matemáticamente, se puede expresar la salida de cada neurona de la capa oculta como:

$$\phi_j(k) = e^{-\frac{(\|X_k - C_j(k)\|)^2}{2\sigma_j^2}} \tag{3.9}$$

Donde:

- j: Hace referencia a la j-ésima neurona.
- k: Hace referencia al instante de tiempo k del conjunto de datos.
- $\phi_j(k)$: Representa la salida de la j-ésima neurona en el instante k.
- $X_k = [x_k^1, x_k^2, ..., x_k^m]$: Es el vector de entradas de la k-ésima observación, común a todas las neuronas.
- m: Es la dimensión del vector de entradas, es decir, el número de variables de entrada.
- $C_j(k) = [c_{1,j}(k), c_{2,j}(k), ..., c_{n,j}(k)]$: Es el vector de centros de la j-ésima neurona en el instante k.
- $\|X_k - C_j(k)\|$: Es la distancia euclídea entre X_k y $C_j(k)$
- σ_j: Es la dispersión (anchura) de la gaussiana de la j-ésima neurona.

En la *Figura 3.6* se representa la función gaussiana para distintos valores de σ. Este es otro de los hiperparámetros que se emplearán en las redes RBF y es una forma de controlar cómo de rápido decae la activación de cada neurona en función de la distancia de la entrada al centro de la neurona.

Por último, la capa de salida está formada por una única neurona y calcula la salida de la red

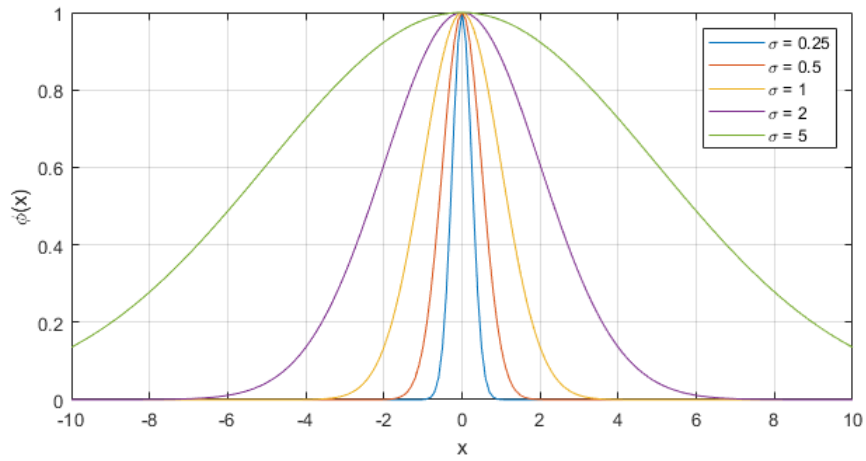


Figura 3.6 Función gaussiana con $\mu = 0$ y diferentes valores de σ.

como combinación lineal de sus entradas, que se corresponden con las salidas de cada neurona de la capa oculta. Matemáticamente, la salida de la red neuronal se puede expresar como:

$$\hat{y}_k = \sum_{j=1}^{p} \omega_j(k)\phi_j(k) \tag{3.10}$$

Donde:

- k: Hace referencia al instante de tiempo k del conjunto de datos.
- $\hat{y}(k)$: Representa la predicción realizada por el modelo RBF en el instante k.
- j: Hace referencia a la j-ésima neurona.
- p: Es el número de neuronas.
- $\omega_j(k)$: Representa el peso entre la j-ésima neurona y la salida en el instante k.
- $\phi_j(k)$: Representa la salida de la j-ésima neurona en el instante k.

En el caso de los modelos RBF se emplearán los siguientes hiperparámetros:

- nn: Número de neuronas en la capa oculta.
- σ: Amplitud de la gaussiana de cada neurona. Es la misma para todas.
- α: Tasa de aprendizaje para el entrenamiento.
- $nPasadas$: Número de pasadas por todas las muestras del conjunto de entrenamiento.
- γ: Factor de olvido del algoritmo de entrenamiento MCR ponderados.

Redes neuronales LSTM (Long Short-Term Memory)

Las redes neuronales LSTM son un tipo de red neuronal recurrente (RNN) que han ganado popularidad en el ámbito del aprendizaje profundo y, concretamente, en tareas que tratan con datos secuenciales, como procesamiento del lenguaje, reconocimiento de voz o predicción de series temporales. Uno de los motivos detrás de este incremento de popularidad es la capacidad de las redes LSTM para capturar relaciones a largo plazo. Además, las redes LSTM consiguen solucionar uno de los principales problemas presente en las RNNs: el problema del gradiente explosivo/desvanecedor, que provoca inestabilidad (gradiente explosivo) o no convergencia a soluciones óptimas o cercanas

(gradiente desvanecedor) durante la etapa de entrenamiento.

La idea básica detrás de las redes LSTM es el empleo de dos "memorias" o estados: uno para largo plazo (estado de celda, C) y otro para corto plazo (estado oculto, h). Cada celda LSTM se compone de tres "puertas" que controlan el flujo de información a través de la unidad: puerta de olvido (f_k), puerta de entrada (i_k) y puerta de salida (o_k), que se pueden ver en la *Figura 3.7*. En resumen, se podría decir que las redes LSTM aprenden cuándo deben recordar y cuándo deben olvidar.

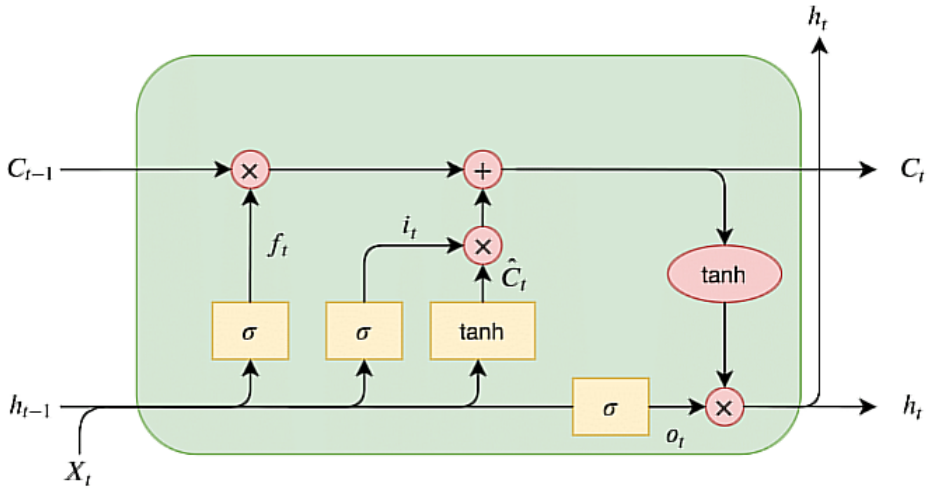


Figura 3.7 Arquitectura de una unidad LSTM [5].

La función de cada puerta se puede explicar de forma intuitiva:

- La puerta de olvido ("forget gate") actualiza la memoria de largo plazo (estado de celda), controlando qué porcentaje de información debe olvidar.
- La puerta de entrada ("input gate") determina qué porcentaje de información de la entrada debe añadir a la memoria de largo plazo.
- La puerta de salida ("output gate") actualiza la memoria de corto plazo (estado oculto), controlando qué porcentaje de información debe recordar.

El funcionamiento explicado se obtiene gracias a dos funciones de activación:

- Sigmoide (σ): Esta función convierte los valores de entrada al rango [0, 1]. Como se puede ver en la *Figura 3.7*, la sigmoide se utiliza siempre como uno de los factores en los diferentes productos de una celda LSTM. Esto se puede entender como aplicar un determinado porcentaje (0-100 %) al otro factor del producto.
- Tangente hiperbólica (*tanh*): Esta función convierte los valores de entrada al rango [-1, 1]. Con esta función se obtienen memorias potenciales de corto y largo plazo, que se combinan con la sigmoide para determinar qué porcentaje de esas memorias potenciales se debe guardar en cada estado.

Se definen las siguientes ecuaciones matemáticas que describen el funcionamiento de una celda LSTM:

$$f_k = \sigma(W_f \cdot [h_{k-1}, X_k] + b_f) \tag{3.11}$$
$$i_k = \sigma(W_i \cdot [h_{k-1}, X_k] + b_i) \tag{3.12}$$
$$o_k = \sigma(W_o \cdot [h_{k-1}, X_k] + b_o) \tag{3.13}$$
$$\hat{C}_k = \sigma(W_C \cdot [h_{k-1}, X_k] + b_C) \tag{3.14}$$
$$C_k = i_k \cdot \hat{C}_k + f_k \cdot C_{k-1} \tag{3.15}$$
$$h_k = o_k \cdot \tanh(C_k) \tag{3.16}$$

Donde:

- f_k, i_k, o_k: Son las salidas de las puertas de olvido, entrada y salida, respectivamente, en el instante k.
- W_f, W_i, W_o, W_C: Son los pesos de la celda LSTM y serán estimados durante el entrenamiento.
- b_f, b_i, b_o, b_C: Son los sesgos de la celda LSTM y serán estimados durante el entrenamiento.
- h_{k-1}: Valor del estado oculto en el instante $k-1$.
- X_k: Vector de entradas en el instante k
- $\hat{C}_k$: Valor candidato a estado de celda en el instante k.
- C_k: Valor del estado de celda en el instante k.
- h_k: Valor del estado oculto en el instante k. Es el valor de salida de la celda LSTM.

Para el entrenamiento de las LSTMs se utilizará el algoritmo de optimización Adam [72] (acrónimo de Adaptive Moment Estimation), que es una extensión del método SGDM (descenso por gradiente estocástico). Además, para evitar el sobreentrenamiento se utilizan dos métodos de regularización: la parada temprana y el abandono ("dropout"). Se emplean los siguientes hiperparámetros para los modelos LSTM:

- *nn*: Número de neuronas (celdas LSTM).
- α: Tasa de aprendizaje.
- $p_{Dropout}$: Probabilidad de abandono. Toma valores en el rango [0, 1].
- *miniBatchSize*: Tamaño del mini-lote. Determina el número de muestras procesadas en paralelo durante cada pasada del entrenamiento. Los parámetros son actualizados según el valor medio de los gradientes de cada mini-lote.
- *maxPasadas*: Determina cuántas pasadas como máximo se realizan durante el entrenamiento.
- *pacienciaCP*: Determina cuántas pasadas deben transcurrir sin que haya mejora en el CP para detener el entrenamiento.

3. Selección de entradas

El objetivo en este paso es la obtención de un conjunto de entradas que proporcionen buenos resultados en la evaluación de los modelos.

Los métodos de selección de entradas se pueden clasificar en tres categorías principales:

- Filtrado: son generalmente un paso de preprocesamiento para reducir el número de entradas posibles. Utilizan una métrica de puntuación para medir el grado de relación entre una entrada y la salida a predecir y seleccionar aquellas de mayor puntuación o bien que superen un cierto umbral. La principal ventaja de estos métodos es su rapidez y bajo coste computacional.
- Envoltura: miden la relevancia de un subconjunto de entradas al entrenar y evaluar un modelo con dicho conjunto. Por tanto, estos métodos seleccionan las mejores entradas para el modelo de predicción empleado, no de forma general, lo cual es una ventaja. Otra ventaja es que tienen en cuenta las relaciones entre las entradas, incluyendo interacciones y redundancias. La principal desventaja de estos métodos es su elevado coste computacional, ya que es necesario realizar el entrenamiento y evaluación de los modelos.
- Embebido: estos métodos implementan la selección de entradas en el propio modelo de predicción. Durante el entrenamiento, el modelo ajusta sus parámetros y determina los pesos/importancia apropiados para cada entrada y así producir el mejor resultado de predicción.

En este trabajo se utiliza un método híbrido que combina un método de filtrado para reducir el número de entradas a probar y un método de envoltura para seleccionar las mejores entradas para el modelo de predicción empleado. Como método de filtrado se emplea el **coeficiente de correlación de Pearson**, que mide el grado de relación lineal entre dos variables. El método de envoltura elegido es el de **selección hacia adelante** o **inclusión progresiva**, que se trata de un método iterativo donde en cada iteración se añade a un conjunto base aquella entrada que mejores resultados proporciona en la evaluación de modelos. Por este motivo es necesario haber elegido previamente el modelo que se va a utilizar.

Definición de grupos de entradas

En este trabajo el proceso de selección de entradas se realiza por grupos de entradas. Es decir, de forma previa a la selección de entradas se definen unos grupos de entradas relacionadas entre sí y se aplica el método comentado sobre cada grupo. De esta forma, se evalúa la influencia de diferentes entradas relacionadas en la capacidad de predicción del modelo utilizado. Algunos ejemplos de estos grupos de entradas son:

- Valores pasados de la serie temporal original, sin modificar, según diferentes desfases.
- Media/mínimo/máximo móvil de la serie temporal según distintos desfases.
- Variables de tipo fecha (día de la semana, mes, día del mes) con desfase igual a la salida a predecir.
- Seno y coseno de variables de tipo fecha (día de la semana, mes, día del mes) con desfase igual a la salida a predecir.

A continuación, se explica en mayor detalle el proceso de selección de entradas llevado a cabo para cada grupo de entradas.

Estudio de correlaciones de Pearson

Se comienza el proceso de selección de entradas con un método de filtrado para reducir en gran medida el número de variables de entrada a probar. El coeficiente de correlación de Pearson para dos variables X e Y con medias μ_X y μ_Y y varianzas σ_X^2 y σ_Y^2, se define como:

$$\rho_{X,Y} = corr(X,Y) = \frac{cov(X,Y)}{\sigma_X^2 \sigma_Y^2} = \frac{E[(X-\mu_x)(X-\mu_Y)]}{\sigma_X^2 \sigma_Y^2}, \quad si \;\; \sigma_X^2 \sigma_Y^2 > 0 \tag{3.17}$$

Donde *cov* es la covarianza entre las variables X e Y, y E es el operador de esperanza.

La correlación de Pearson indica el grado de relación lineal existente entre dos variables y puede tomar valores en el rango [-1,1]. A continuación, se explican los casos extremos para entender mejor este concepto:

- $\rho_{X,Y} = 1$: Indica que las variables X e Y tienen una correlación perfecta positiva (cuando una aumenta, la otra también lo hace en la misma medida).
- $\rho_{X,Y} = 0$: Indica que las variables X e Y son independientes entre sí, no existe relación lineal entre ellas.
- $\rho_{X,Y} = -1$: Indica que las variables X e Y tienen una correlación perfecta negativa (cuando una aumenta, la otra disminuye en la misma medida).

En la *Figura 3.8* se muestran gráficos de dispersión para distintos casos de X e Y y el valor que toma el coeficiente de correlación de Pearson en dichos casos.

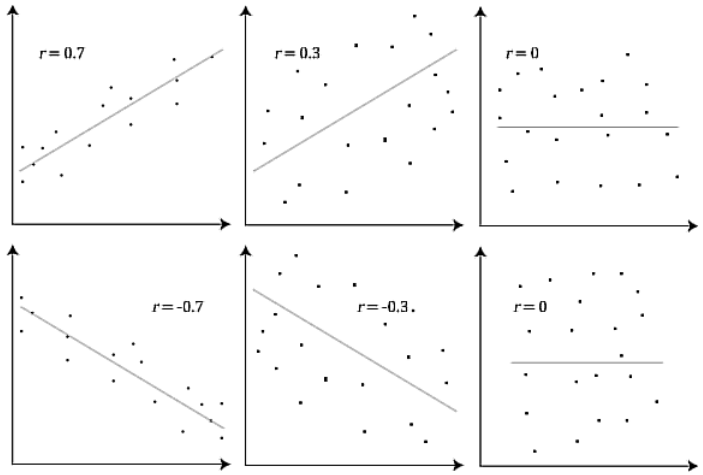


Figura 3.8 Gráficas de dispersión para distintos casos de X e Y y coeficiente de correlación de Pearson en dichos casos [6].

Por tanto, al obtener el coeficiente de correlación de Pearson para cada entrada del conjunto de entradas disponibles se tiene una primera idea de qué entradas pueden ser de utilidad al modelo para predecir la variable objetivo.

Es importante mencionar que es igual de útil una correlación elevada positiva como una negativa. Es decir, lo que realmente interesa al final es el valor absoluto de la correlación, independientemente del sentido que tenga.

Selección hacia adelante / Inclusión progresiva ("Forward selection")

Partiendo del conjunto de entradas reducido tras el estudio de correlación de Pearson, la siguiente técnica empleada es el método de envoltura conocido como selección hacia delante o inclusión progresiva. Este método consiste en ir añadiendo de una en una aquellas entradas que presentan los mejores resultados (menores errores) en la evaluación de modelos. Para ello es necesario elegir un modelo e hiperparámetros iniciales y evaluar cada subconjunto de entradas, eligiendo para la siguiente iteración aquel conjunto que proporciona el mayor rendimiento. Este proceso se repite

hasta dejar de obtener mejora en el rendimiento de los modelos.

En la *Figura 3.9* se ilustra el proceso explicado anteriormente con un ejemplo muy simple.

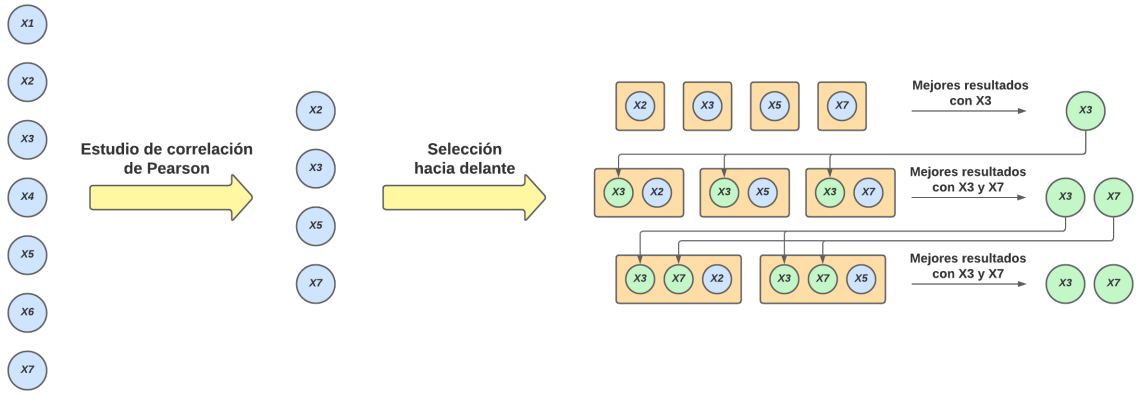


Figura 3.9 Ilustración del procedimiento de selección de entradas empleado.

4. Ajuste de hiperparámetros

Tras el proceso de selección de entradas se tendrá uno o varios conjuntos de entrada destacados, es decir, que han proporcionado los mejores resultados en la evaluación. El segundo componente fundamental en un modelo de predicción son los hiperparámetros del mismo, que son parámetros que afectan al entrenamiento y que el usuario debe optimizar. Para este propósito existen diversos métodos: búsqueda manual, búsqueda en cuadrícula, búsqueda aleatoria, optimización bayesiana, etc.

En este trabajo se emplea el método de **búsqueda en cuadrícula**, que se explica a continuación.

Búsqueda en cuadrícula ("Grid Search")

El método de búsqueda en cuadrícula consiste en definir una cuadrícula de valores posibles para cada hiperparámetro del modelo. Para cada combinación única de hiperparámetros se crea un modelo y se evalúa. El objetivo es encontrar la combinación óptima de hiperparámetros que proporcione los mejores resultados.

La ventaja de este método es la capacidad de obtener una combinación de hiperparámetros óptima o cercana. Sin embargo, tiene un gran inconveniente, que es el aumento desproporcionado del coste computacional a medida que se introducen más hiperparámetros y valores posibles en la cuadrícula. Por este motivo, la búsqueda en cuadrícula empleada en este trabajo no ha sido muy extensa, sino que se ha aplicado a los hiperparámetros más influyentes y se ha intentado minimizar el número de valores introducidos en la cuadrícula.

Tras repetir los pasos 3 y 4 para los tres modelos utilizados, se tendrá uno o varios mejores modelos de cada tipo, cada uno con su evaluación ya realizada. Por tanto, el último paso será comparar el

rendimiento de todos estos modelos y determinar cuál es más adecuado para el problema tratado. En el primer apartado del capítulo 4 se explican las métricas utilizadas para este fin.

3.3 Algoritmos de automatización

En este apartado se explican los algoritmos de automatización empleados.

Un punto a tener en cuenta es que se ha implementado un mecanismo de paralelización del método, agilizando aún más el proceso de optimización de modelos.

3.3.1 Selección de entradas

Como se ha explicado anteriormente, el paso inicial en esta tarea consiste en definir previamente grupos de entradas de entre todas las disponibles con el objetivo de realizar pruebas individuales con un tipo concreto de entrada y así determinar su impacto o influencia en el modelo.

Para cada uno de estos grupos se aplicará la metodología de selección de entradas, es decir, filtrado por correlación de Pearson y selección hacia adelante, pero de forma automatizada.

A continuación, se explica el algoritmo de automatización empleado:

Estudio de correlaciones de Pearson

La automatización de este método es inmediata. Se ha optado por seleccionar las N (parámetro definido por el usuario) entradas de mayor correlación en vez de especificar un umbral para el filtrado, ya que de esta forma el número de entradas a probar es constante y conocido.

Cabe mencionar que si un grupo de entradas presenta un número reducido de entradas (< 10-20), este método se omite.

Selección hacia adelante

Para la automatización de este método es necesario definir un criterio (métrica) de evaluación que permita al sistema seleccionar las mejores entradas según dicho criterio en cada iteración del proceso. En este caso se ha optado por utilizar el Error Cuadrático Medio (ECM) en el conjunto de prueba (CP).

En resumen, el método consiste en recorrer cada grupo de entradas filtrado, evaluar la influencia de cada una de las entradas e ir añadiendo a un conjunto base aquellas que consiguen mejorar las predicciones realizadas por el modelo.

Se ha incluido un mecanismo de paciencia y parada temprana, de forma que el sistema continúa iterando hasta transcurrir P (parámetro definido por el usuario) iteraciones sin que exista mejora en el criterio elegido (ECM en CP), momento en el cual se pasa directamente al siguiente grupo de entradas. De esta forma se alcanza un equilibrio entre la capacidad de encontrar el conjunto de entradas óptimo y el tiempo empleado en el método.

Otro punto importante es que entre grupos de entradas se actualiza el conjunto base, tomando aquel conjunto de entradas de mayor rendimiento de entre todos los conjuntos evaluados hasta el momento, de forma que si en un grupo de entradas no ha habido mejora con respecto a conjuntos anteriores, el conjunto base no presentará ninguna entrada perteneciente a dicho conjunto, evitando

así la adición de entradas irrelevantes o redundantes. Se muestra el algoritmo resultante de selección automática de entradas como *Algorithm 1*.

Algorithm 1: Pseudocódigo del algoritmo de selección automatizada de entradas

Entrada: Grupos de entradas, salida a predecir, número de entradas a filtrar, paciencia en la selección hacia adelante, combinaciones de hiperparámetros

```
Inicializar conjunto de entradas base vacío;
for grupoEntradas = 1, 2, ..., numGruposEntradas do
    /* Estudio de correlaciones de Pearson */;
    for entrada = 1, 2, ..., numEntradas do
        Calcular la correlación de la entrada actual con la salida a predecir
    Filtrar las N entradas de mayor correlación (entradasPrueba);
    /* Selección hacia adelante */;
    while paciencia > 0 and numEntradasPrueba > 0 do
        for entradasPrueba = 1, 2, ..., numEntradasPrueba do
            Construir conjunto de entradas de prueba añadiendo la entrada de prueba actual
              al conjunto base;
            for combinacion = 1, 2, ..., numCombinaciones do
                Entrenar modelo LSTM con el conjunto de entradas de prueba y la
                  combinación de hiperparámetros actuales;
                Evaluar el modelo creado utilizando como métrica el ECM en CP;
        Identificar el conjunto de entradas de prueba que ha resultado en menor ECM en CP;
        Reemplazar el conjunto base con el conjunto de entradas de prueba seleccionado;
        Eliminar del conjunto de entradas de prueba la nueva entrada añadida al conjunto
          base;
        if no existe mejora en ECM en CP con respecto a las iteraciones anteriores del
          grupo de entradas actual then
            Se decrementa el contador de paciencia o intentos restantes;
        else
            Se actualiza el mínimo ECM en CP obtenido para este grupo de entradas;
            Se reinicia el contador de paciencia a su valor inicial;
    Identificar el conjunto de entradas de prueba que ha resultado en menor ECM en CP de
      entre todas las pruebas realizadas desde el inicio del algoritmo;
    Reemplazar el conjunto de entradas base por dicho conjunto de entradas;
```

Salida: Conjunto de entradas de mayor rendimiento según ECM en CP

3.3.2 Ajuste de los hiperparámetros

Puesto que el método de búsqueda en cuadrícula ya es un método automatizado, en este trabajo se propone automatizar la realización de múltiples búsquedas en cuadrícula (que llamaremos barridos) variando diferentes hiperparámetros cada vez (por pares o de forma individual, para evitar una carga computacional excesiva) y seleccionando las mejores combinaciones de hiperparámetros entre iteraciones usando de nuevo como criterio el ECM en CP.

Se muestra el algoritmo resultante de ajuste automatizado de hiperparámetros como *Algorithm 2*

Algorithm 2: Pseudocódigo del algoritmo de ajuste automatizado de hiperparámetros

Entrada: Conjunto de entradas, salida a predecir, conjunto de valores de cada hiperparámetro, barridos a realizar (hiperparámetro individual o parejas de hiperparámetros), valores por defecto de hiperparámetros

```
for barrido = 1, 2, ..., numBarridos do
    Inicializar hiperparámetros a valores por defecto;
    Identificar hiperparámetro/s a barrer;
    Definir todas las combinaciones posibles de hiperparámetros manteniendo constantes los
     hiperparámetros que no se barren;
    for combinacion = 1, 2, ..., numCombinaciones do
        Entrenar modelo con el conjunto de entradas elegido y la combinación de
         hiperparámetros actual;
        Evaluar el modelo creado utilizando como métrica el ECM en CP;
    Identificar la combinación de hiperparámetros que ha resultado en menor ECM en CP;
    Actualizar los valores de hiperparámetros por defecto con los valores de dicha
     combinación;
```

Salida: Combinación de hiperparámetros de mayor rendimiento según ECM en CP para el conjunto de entradas utilizado

Es importante mencionar que, aunque en este trabajo se ha utilizado como criterio el ECM en CP en ambos algoritmos, se podría cambiar dicho criterio en cualquier momento de forma fácil e inmediata.

4 Aplicación. Conjuntos de datos reales

Una vez definida la metodología seguida, se procede a aplicar las distintas técnicas mencionadas a conjuntos de datos reales. Para comenzar, se explicarán las distintas métricas empleadas para la evaluación de los modelos predictivos. Seguidamente, se utilizará un conjunto de datos de un centro de atención al cliente como primera prueba de los algoritmos explicados. En segundo lugar, se abordará el problema principal de predicción del saldo diario acumulado de transacciones de una gran banco. El objetivo será comparar los resultados obtenidos para los distintos modelos predictivos y determinar cuál presenta mayor rendimiento. Una vez elegido un modelo de predicción, se procederá a aplicar la automatización a la misma serie temporal predicha por último, así como una nueva serie temporal para demostrar la capacidad de generalización del método automatizado.

4.1 Métricas de evaluación

Existen numerosas métricas de evaluación aplicables a los problemas de regresión y cada una representa el rendimiento de los modelos de forma distinta. Por ello, se han empleado cuatro métricas distintas, que se detallan a continuación.

Antes de explicar cada una de las métricas empleadas, se va a definir una nomenclatura común a todas ellas:

- y_k: Representa la salida real en el instante k.
- $\hat{y}_k$: Representa la predicción realizada por el modelo en el instante k.
- z_k: Representa el valor estandarizado de la salida real en el instante k.
- $\hat{z}_k$: Representa el valor estandarizado de la predicción realizada por el modelo en el instante k.
- N: Es el número de observaciones.

A lo largo de este apartado se mencionará con frecuencia el término "error", haciendo referencia al error existente entre las predicciones realizadas por el modelo y las salidas reales, o lo que es lo mismo, la diferencia entre ambos. Con esto, se define a continuación cada una de las métricas:

- **Error cuadrático medio (ECM)**

  Es el valor medio de los errores al cuadrado. Para que el resultado sea más interpretable se toma la raíz cuadrada de este resultado. La fórmula matemática es la siguiente:

$$ECM = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(z_k - \hat{z}_k)^2} \tag{4.1}$$

Elevar los términos de error al cuadrado tiene varios propósitos: asegura que el valor del ECM sea siempre positivo y, además, enfatiza y penaliza los errores más elevados.

- **Error absoluto medio (EAM)**

  Es el valor medio de los errores absolutos. Matemáticamente se expresa como:

$$EAM = \frac{1}{N}\sum_{k=1}^{N}|z_k - \hat{z}_k| \tag{4.2}$$

  La única diferencia que presenta con respecto al ECM es que el EAM mide todos los errores por igual, dando la misma importancia a errores bajos como elevados.

- **Error medio porcentual absoluto arcotangente (MAAPE)**

  MAAPE son las siglas de "Mean Arctangent Absolute Percentage Error".

  El MAAPE surge del MAPE ("Mean Absolute Percentage Error"), una de las métricas más populares para tareas de predicción. El MAAPE se obtiene aplicando la función arcotangente a los distintos términos del MAPE, por lo que conserva las ventajas de este último: independencia a la escala (por lo que se puede usar para comparar resultados de diferentes series), se puede interpretar de forma intuitiva como un error porcentual absoluto y es fácil de calcular [73]. Además, presenta otra serie de ventajas con respecto al MAPE [73]:

  – El rango limitado de la función arcotangente del MAAPE ($[0, \frac{\pi}{2}]$) soluciona el problema presente en el MAPE de tender a infinito a medida que el valor esperado tiende a cero.

  – Penaliza los errores negativos y positivos de una forma más equilibrada que el MAPE.

  – Es más robusto que el MAPE ante valores atípicos ("outliers").

  Matemáticamente, el MAAPE (en tanto por unidad) se describe como:

$$MAAPE = \frac{1}{N}\sum_{k=1}^{N} arctan\left(\left|\frac{y_k - \hat{y}_k}{y_k}\right|\right) \tag{4.3}$$

- **Error medio porcentual absoluto simétrico (SMAPE)**

  El SMAPE aparece con el objetivo de solucionar la asimetría existente en el MAPE. Existen diferentes versiones del SMAPE, y la que se usará en este trabajo se expresa matemáticamente (en tanto por unidad) como:

$$SMAPE = \frac{1}{N}\sum_{k=1}^{N}\frac{|y_k - \hat{y}_k|}{|y_k| + |\hat{y}_k|} \tag{4.4}$$

  Con esta definición, el SMAPE puede tomar valores en el rango $[0,1]$ o $[0,100]$ % en términos porcentuales.

Es importante mencionar que las dos primeras métricas (EAM, ECM) se han calculado con valores normalizados siguiendo una distribución normal de media nula ($\mu = 0$) y varianza unidad ($\sigma^2 = 1$). Con esto se pretende obtener valores unitarios o porcentuales de EAM y ECM, lo cual permite comparar resultados de series temporales con diferente escala.

4.2 Predicción del volumen de llamadas en un centro de atención al cliente (CAC)

Como primera prueba se utiliza el conjunto de datos de un centro de atención al cliente (CAC) para predecir el volumen de llamadas recibidas.

Este conjunto presenta registros desde julio de 2001 hasta marzo de 2003, con un paso temporal horario y un total de $N = 5760$ observaciones.

A continuación, se aplica la metodología explicada previamente.

1. Preparación de los datos

Análisis exploratorio de los datos (EDA)

El primer paso en cualquier tarea de predicción de series temporales es representar y observar la evolución de la serie temporal en el tiempo. En la *Figura 4.1* se representa el volumen de llamadas de un CAC a lo largo de 6 semanas.

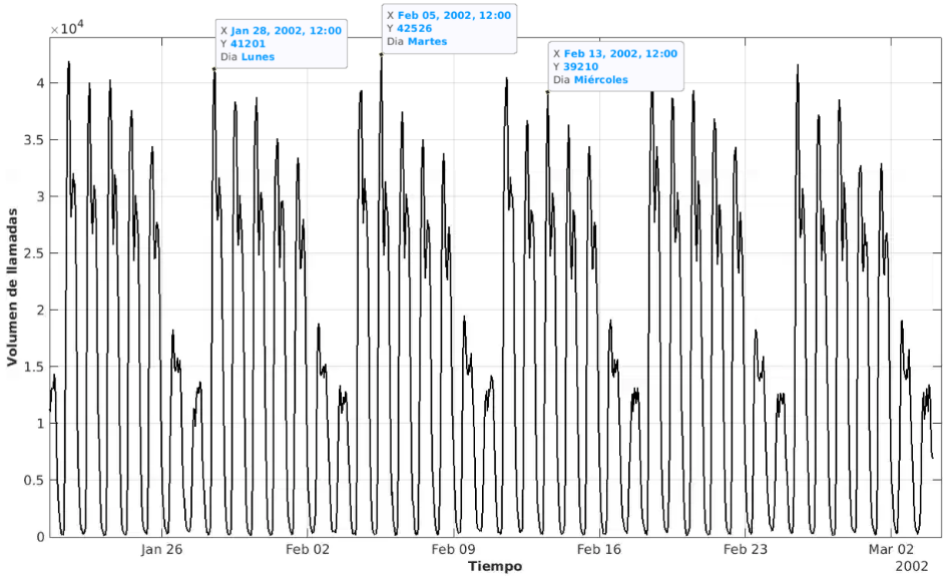


Figura 4.1 Volumen de llamadas de un CAC en un periodo de 6 semanas.

De esta figura se puede extraer información relevante para el modelado:

- Cada día se repite el mismo patrón: a partir de las 7-8h se observa un incremento progresivo en el volumen de llamadas recibidas, hasta alcanzar su máximo diario a las 12h, generalmente. De 12h a 15h hay una caída y, de nuevo, un aumento progresivo hasta las 17-18h de lunes a viernes y hasta las 19-20h los sábados y domingos. A partir de esas horas, se da un decremento progresivo hasta el día siguiente. Por tanto, esta serie presenta cierta periodicidad diaria.
- También se aprecia una periodicidad semanal, teniendo los días laborables (L-V) un comportamiento similar, los sábados un volumen mucho menor de llamadas y los domingos similar a los sábados, pero con un volumen incluso menor. Gracias a esta información se define el tipo de día, agrupando aquellos días de la semana con un comportamiento similar. Así, se tienen tres tipos de días: de lunes a viernes, sábado y domingo. A partir de esta variable se

definirán otras autorregresivas que tomarán como referencia el tipo de día de la salida y se comprobará más adelante que mejoran el rendimiento de los modelos.

- En general, no aparecen valores atípicos, irregularidades o cambios bruscos en los datos ni se observan datos faltantes.

Todas estas características hacen que la serie temporal sea fácil de predecir, lo cual se comprobará más adelante en los resultados.

El horizonte de predicción será de una semana a la misma hora que la actual y como la serie es horaria, $h = 24 \cdot 7 = 168$. Es decir, para cada instante de tiempo se va a predecir el volumen de llamadas recibidas dentro de 168 horas.

Preprocesamiento

Como se ha mencionado en el último punto del paso anterior, no se observan anomalías en los datos, por lo que en este caso no se realiza ningún preprocesamiento sobre el conjunto.

Feature extraction/engineering

Este problema ya ha sido estudiado previamente [74][75] y, por tanto, la definición de entradas potenciales ya está resuelta. En la *Tabla 4.1* se muestran las diferentes variables de entrada de las que se parte, indicando las abreviaturas correspondientes y el rango de desfases utilizado:

Tabla 4.1 Conjunto de entradas potenciales definidas.

Descripción de la entrada potencial	Abreviatura	Tipo entrada	Rango desfases
Volumen de llamadas	V	Autorregresiva	[-1440 -1]
Volumen medio de las 7 horas centradas en el desfase	V_{mean7}	Autorregresiva	[-1440 -3]
Tipo de día	TD	Exógena	[168]
Volumen del día previo del mismo tipo que la salida desde el instante actual	V_{damt}	Autorregresiva	[168]
Volumen del segundo día previo del mismo tipo que la salida desde el instante actual	V_{2damt}	Autorregresiva	[168]
Volumen del día previo del mismo tipo que la salida a la hora previa desde el instante actual	V_{damt_ha}	Autorregresiva	[168]
Hora	H	Exógena	[168]
Día del mes	dM	Exógena	[168]
Mes	M	Exógena	[168]
Día de la semana por frecuencia de aparición	dS	Exógena	[168]
Día de la semana ordenado (1-7)	dS_{ord}	Exógena	[168]

Hay que tener en cuenta varios puntos sobre la *Tabla 4.1*:

- El desfase máximo empleado para las variables autorregresivas es $d = -1440 = -24 \cdot 60$, es decir, se utilizan valores de hasta 2 meses en el pasado.
- Las únicas variables exógenas que se utilizan son de tipo calendario y concretamente, del instante a predecir ($d = 168$).
- El desfase para las variables autorregresivas que utilizan el tipo de día ($damt$) indica la referencia de tipo de día, que en este caso será la salida ($d = 168$).

Siempre se empieza por añadir variables autorregresivas a los modelos, ya que son más explicativas para la variable objetivo. En este caso, se tienen: V, V_{mean7}, V_{damt}, V_{2damt} y V_{damt_ha}. Una vez se deja de observar mejora en el rendimiento con entradas autorregresivas se pasa a incluir entradas exógenas.

Como es la primera prueba de creación de modelos, se explicará el proceso en detalle, indicando todos los pasos seguidos en la selección de entradas y el ajuste de hiperparámetros, así como los resultados obtenidos. Para el resto de pruebas se presentará únicamente el resumen de resultados, ya que el proceso de creación de modelos seguido es siempre el mismo.

2. Elección del modelo

Comenzamos por el modelo más sencillo, el modelo lineal ARX.

3. Selección de entradas

Definición de grupos de entradas

Se definen los siguientes grupos de entradas:

Tabla 4.2 Grupos de entradas definidos.

Grupo de entradas	Entradas incluidas
1	V
2	V_{mean7}
3	V_{damt}, V_{2damt}, V_{damt_ha}
4	TD, H, dM, dS, dS_{ord}

A continuación, se muestra únicamente el proceso de selección de entradas para el primer grupo de entradas definido a modo de ejemplo, ya que sería idéntico para el resto de grupos.

Estudio de correlaciones de Pearson

Al tener tantas entradas posibles (1440) no es factible probarlas todas y por eso el primer paso será realizar un estudio de correlaciones de Pearson con el objetivo de reducir en gran medida el número de entradas que se utilizarán en los modelos, tomando únicamente aquellas que presenten mayor correlación con la salida. En la *Figura 4.2* se muestra la correlación de las diferentes entradas autorregresivas $V(t+d)$ con la salida a predecir, $V(t+168)$.

De esta gráfica se obtienen dos conclusiones principales:

- Las correlaciones más elevadas se dan para los desfases múltiplos de -168 ($d = -k \cdot 168$), es decir, el volumen de llamadas hace 1, 2, ..., k semanas a la misma hora que la actual, lo cual tiene sentido debido a la periodicidad semanal que presenta la serie temporal.
- Los valores más elevados de correlación son muy cercanos a 1, lo que implica que las variables correspondientes tienen una relación lineal muy elevada con la salida y por tanto, es probable que den buenos resultados.

En la *Tabla 4.3* se muestran las 20 entradas de mayor correlación, que se utilizan en el método de inclusión progresiva.

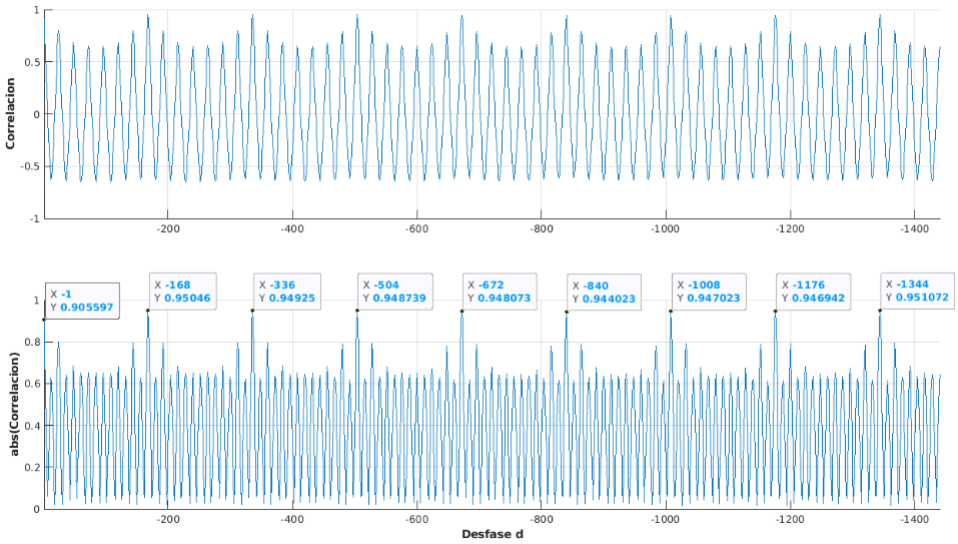


Figura 4.2 Correlaciones de Pearson (arriba) de las entradas autorregresivas $V(t+d)$ con la salida $V(t+168)$ y valor absoluto de dichas correlaciones (abajo).

Tabla 4.3 Entradas autorregresivas $V(t+d)$ de mayor correlación con la salida $V(t+168)$.

d	$\|\boldsymbol{\rho}\|$
-1344	0.9511
-168	0.9505
-336	0.9492
-504	0.9487
-672	0.9481
-1008	0.9470
-1176	0.9469
-840	0.9440
-1	0.9056
-1343	0.8951
-167	0.8917
-1175	0.8916
-503	0.8911
-671	0.8911
-335	0.8908
-1007	0.8908
-169	0.8898
-337	0.8885
-1345	0.8876
-839	0.8874

Selección hacia delante / Inclusión progresiva

Para comenzar, se debe elegir unos valores de hiperparámetros iniciales correspondientes al modelo elegido en el punto 2. Se utilizan los siguientes valores:

- $\gamma = 1$
- $\delta = 10$

Con esto, se entrena un modelo ARX con cada una de las entradas elegidas anteriormente y se representa el rendimiento de los modelos según las distintas métricas explicadas.

Antes de seguir, es necesario mencionar que se denominará a cada conjunto único de entradas como Cx o CX y cada uno tendrá un número identificador asociado para diferenciarlo de otros. Por ejemplo, se puede tener un CX 1 con la entrada $V(t-1344)$ y un CX 2 con la entrada $V(t-168)$. A continuación, se muestra el proceso de selección de entradas autorregresivas de la variable V.

- 1 entrada

En la *Figura 4.3* se muestra el rendimiento (en CE, en CP y la media entre ambos conjuntos) de modelos ARX con 1 entrada autorregresiva según diferentes métricas.

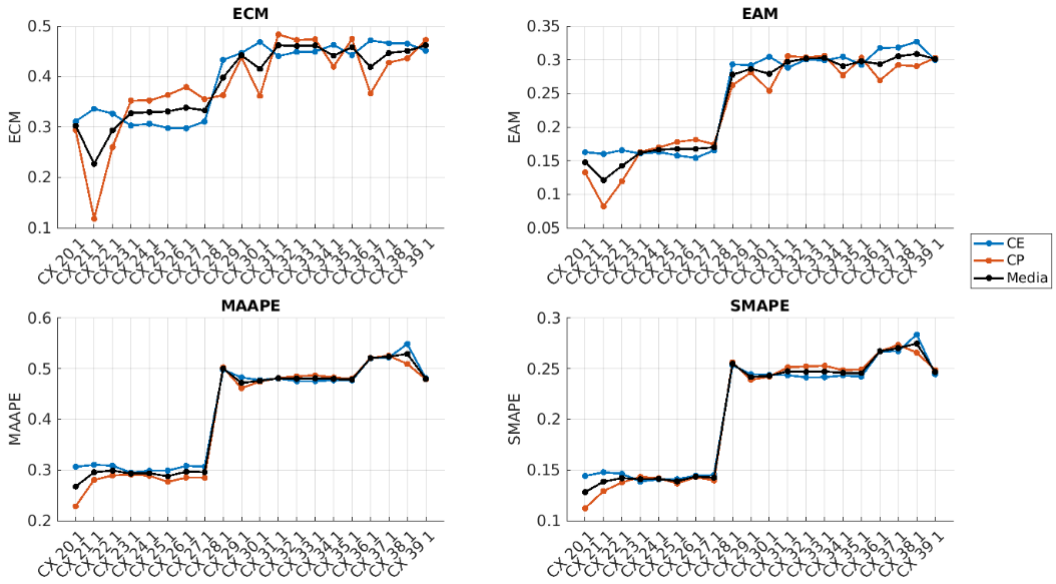


Figura 4.3 Rendimiento de modelos ARX con 1 entrada autorregresiva según diferentes métricas.

En cada subfigura se representan los errores correspondientes (CE, CP y media) obtenidos en las predicciones del modelo ARX de cada conjunto de entradas o CX. El orden de los CX es el mismo que el de la *Tabla 4.3*, por lo que el CX 20 contiene la entrada $V(t-1344)$, el CX 21 contiene la entrada $V(t-168)$, etc. Se puede ver que en el caso de los ARX las entradas de mayor correlación parecen tener mejor rendimiento en general.

Según el método de selección hacia adelante, ahora se debe elegir un CX del cual partir para la siguiente iteración. Observando la *Figura 4.3*, se puede ver que los errores son menores para los tres primeros CXs. El segundo CX presenta una diferencia muy grande en ECM entre CE y CP, lo cual no suele ser buena señal, por lo que se ignora ese conjunto. Se elige el CX 20, que utiliza la entrada $V(t-1344)$, como base para la siguiente iteración.

- 2 entradas

Partiendo de la entrada $V(t-1344)$, se genera un conjunto de entradas por cada una de las entradas restantes de mayor correlación. Por tanto, en este caso se tienen 19 CXs, para cada cual se entrena un modelo ARX y, de nuevo, se representa el rendimiento de cada uno. Además, se representa

también el modelo correspondiente al CX elegido en la iteración anterior para tener una referencia de comparación, que en este caso es el CX 20.

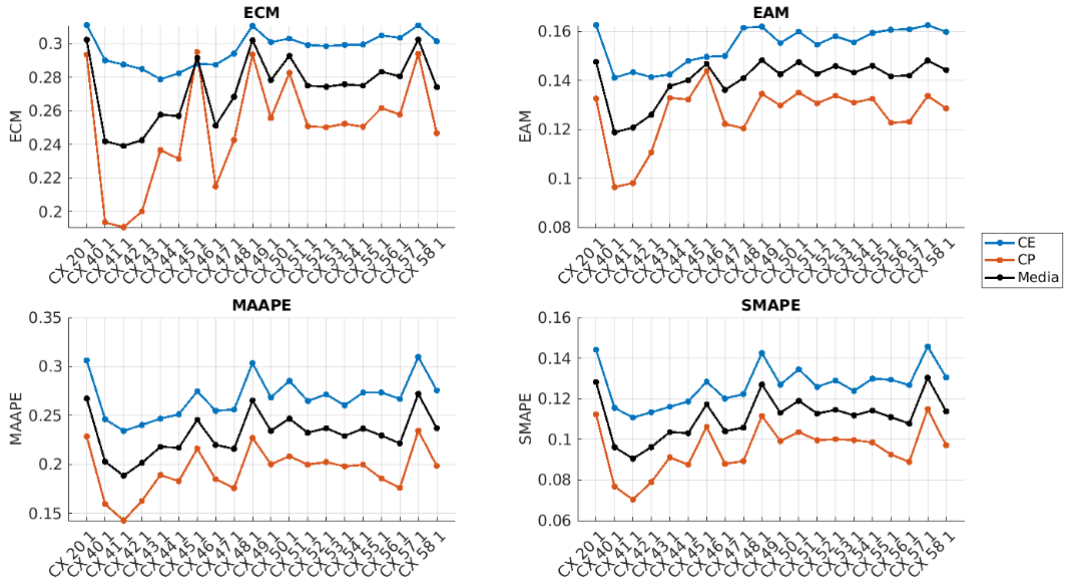


Figura 4.4 Rendimiento de modelos ARX con 2 entradas autorregresivas según diferentes métricas.

En la *Figura 4.4* se puede ver que algunos CXs presentan mejor rendimiento que el último CX elegido. En este caso, se elige el CX 41, ya que parece tener los menores errores en general. Este conjunto presenta las entradas $V(t-1344)$ y $V(t-168)$.

- 3 entradas

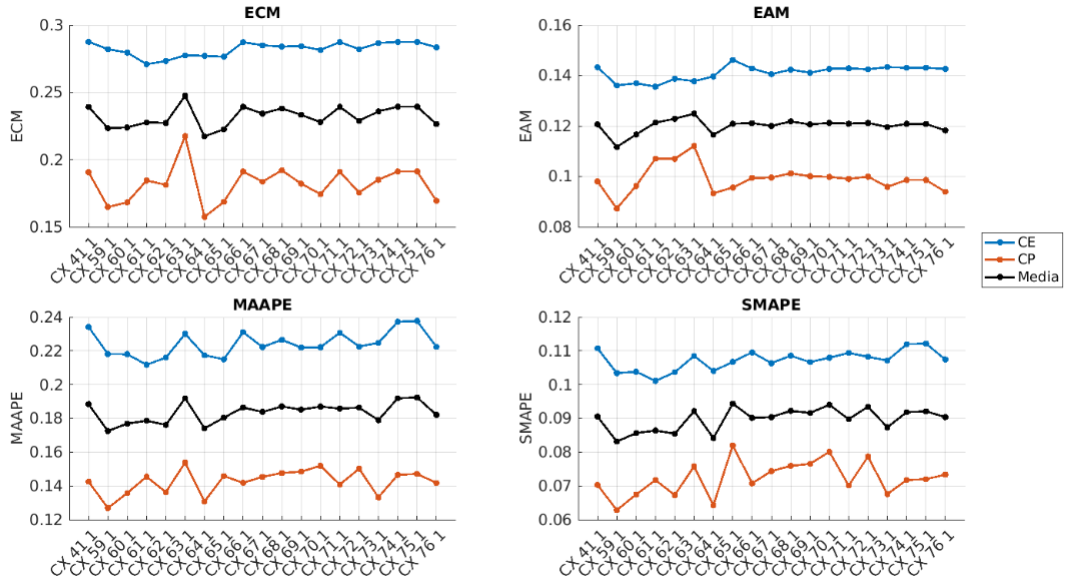


Figura 4.5 Rendimiento de modelos ARX con 3 entradas autorregresivas según diferentes métricas.

En esta iteración se elige el CX 59, ya que parece presentar los menores errores, aunque también se podría haber elegido el CX 64 por tener un rendimiento similar. Este conjunto incluye la entrada $V(t-336)$.

- 4 entradas

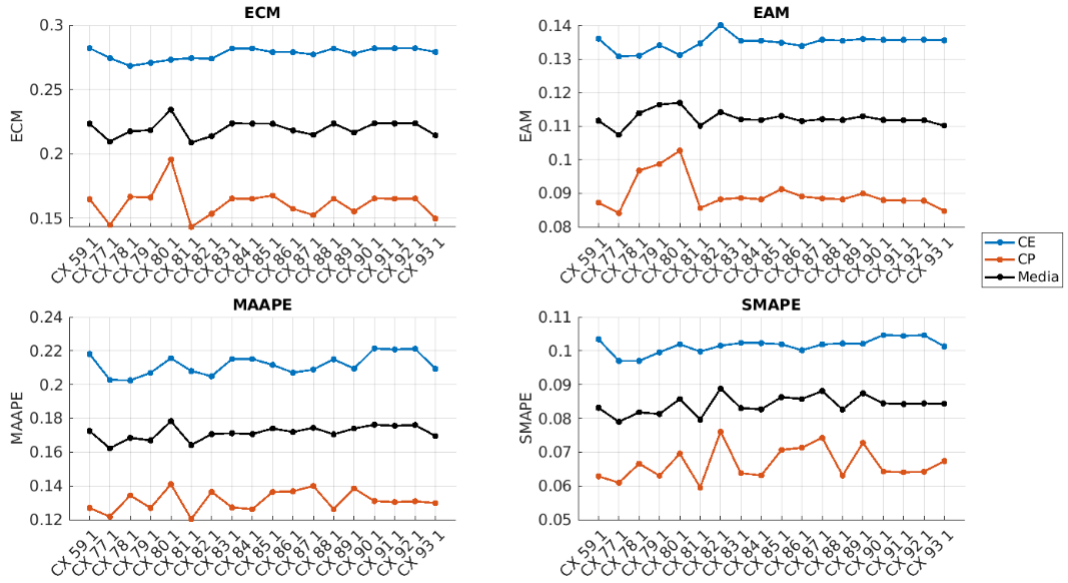


Figura 4.6 Rendimiento de modelos ARX con 4 entradas autorregresivas según diferentes métricas.

En esta iteración se podría elegir el CX 77 o el CX 81, ya que presentan resultados similares y mejores que el CX 59. Se elige el CX 77, que añade la entrada $V(t-504)$.

- 5 entradas

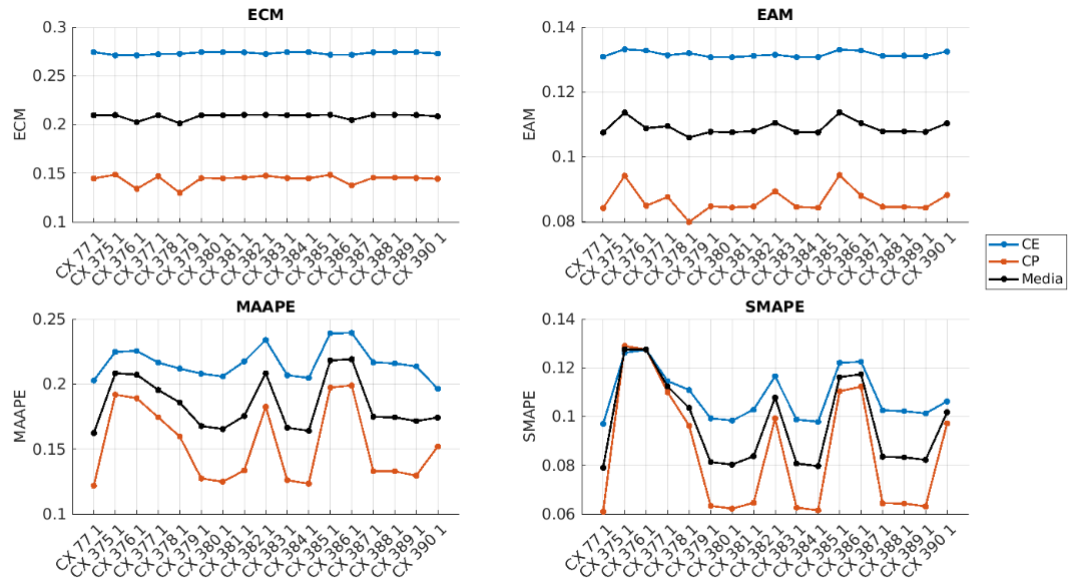


Figura 4.7 Rendimiento de modelos ARX con 5 entradas autorregresivas según diferentes métricas.

En la *Figura 4.7* se observa que, aunque parece haber mejora en ECM y EAM, en los errores porcentuales se consiguen resultados similares o peores con los nuevos CXs de esta iteración. Por este motivo, se deja de iterar y se toma el conjunto de entradas $\{V(t-1344), V(t-168), V(t-336), V(t-504)\}$ como mejor conjunto.

Se repite este proceso para el resto de grupos de entradas y se elabora la *Tabla 4.4* a modo de resumen del proceso de selección de entradas, indicando los distintos errores obtenidos en el conjunto de prueba (CP). La primera columna hace referencia al número de entradas, *m*, que también se corresponde con la iteración del proceso.

Tabla 4.4 Resumen del proceso de selección de entradas y errores obtenidos usando modelos ARX.

m	Conjunto de entradas	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
1	$V(t-1344)$	0.2934	0.1325	0.2284	0.1123
2	$V(t-1344), V(t-168)$	0.1907	0.0980	0.1425	0.0703
3	$V(t-1344), V(t-168), V(t-336)$	0.1648	0.0872	0.1269	0.0628
4	$V(t-1344), V(t-168), V(t-336), V(t-504)$	0.1447	0.0841	0.1217	0.0609
5	$V(t-1344), V(t-168), V(t-336), V(t-504), V_{damt}(t+168)$	**0.0801**	**0.0513**	**0.1060**	**0.0563**

4. Ajuste de hiperparámetros

Una vez se deja de obtener mejora en el rendimiento de los modelos al introducir entradas, el siguiente paso es realizar una búsqueda en cuadrícula para ajustar y optimizar los hiperparámetros. El conjunto de valores de la cuadrícula definida será el siguiente:

- $\gamma = [0.98, 0.985, 0.99, 0.995, 1]$
- $\delta = [0.1, 1, 3, 6, 10, 15, 20]$

En la *Figura 4.9* aparece representado gráficamente el conjunto de combinaciones disponibles para ilustrar el concepto de búsqueda en cuadrícula.

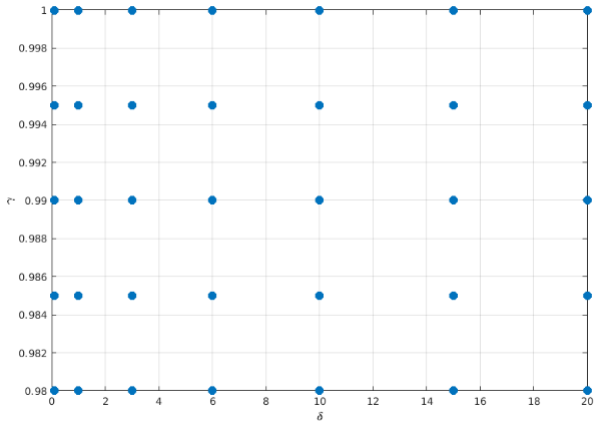


Figura 4.8 Cuadrícula de valores definidos para γ y δ.

Una vez definida la cuadrícula de valores, se crea un modelo ARX por cada combinación única de γ y δ con un conjunto de entradas destacado tras el proceso de selección de entradas. A continuación, se muestran los resultados obtenidos tomando como ejemplo el ECM:

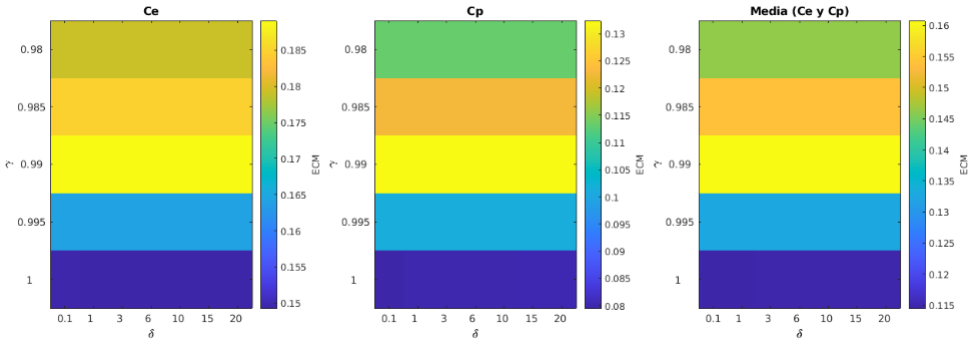


Figura 4.9 Rendimiento según ECM de modelos ARX en función de γ y δ.

Observando la *Figura 4.9* se obtienen las siguientes conclusiones:

- El hiperparámetro δ parece tener muy poca influencia en el rendimiento de los modelos ARX.
- El valor óptimo de γ es 1.

Con esto, se da por finalizado el proceso de ajuste de hiperparámetros y con ello, la creación de modelos ARX. A continuación, se muestra el mejor modelo ARX obtenido:

Tabla 4.5 Modelo ARX con mejor rendimiento.

Conjunto de entradas	γ	δ	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
$V(t-1344), V(t-168), V(t-336), V(t-504), V_{damt}(t+168)$	1	0.1	0.0795	0.0507	0.1047	0.0552

Como último paso se puede representar la salida predicha por el modelo junto a la salida real para observar si realmente las predicciones se corresponden con los errores obtenidos. En la *Figura 4.10* aparece representado el saldo real y el saldo predicho por el mejor modelo ARX tanto en CE como en CP para un intervalo de tiempo de 2 semanas. Aunque la evaluación del modelo en el CP refleja mejor el rendimiento real del mismo, también es útil la evaluación en el CE.

También se puede representar la salida predicha frente a la salida real. En el caso ideal se observaría una línea recta, tal que para cada valor de salida real, se tiene el mismo valor de salida predicha. En la *Figura 4.11* se representa este tipo de gráfica tanto para el CE como para el CP.

Se comprueba que las predicciones son muy cercanas a los valores reales de la serie, como se esperaba desde un principio debido a las características que presenta la serie temporal y como bien han reflejado las distintas métricas empleadas.

Una vez se ha aclarado el proceso de creación de modelos con un ejemplo práctico, para el resto de pruebas se presentará únicamente un resumen de los resultados, así como explicaciones de algunas características de los modelos empleados.

Redes neuronales RBF

Se comienza eligiendo los valores de hiperparámetros iniciales:

- $nn = 30$
- $\sigma = 2$

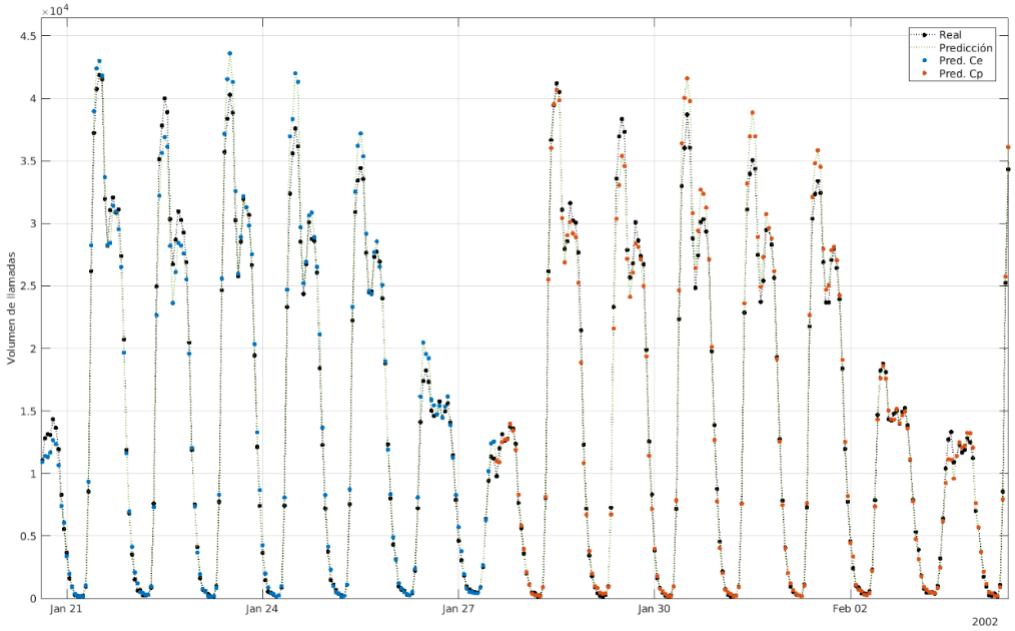


Figura 4.10 Salida predicha y salida real en CE y CP para el mejor modelo ARX.

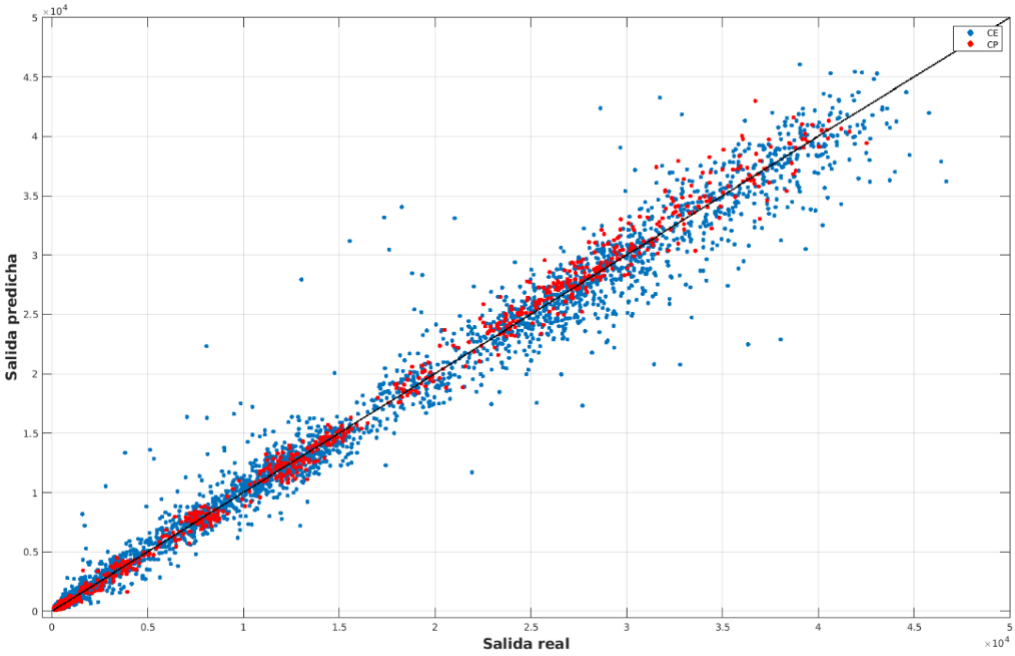


Figura 4.11 Salida predicha frente a salida real en CE y CP para el mejor modelo ARX.

- $\alpha = 0.001$
- $nPasadas = 50$
- $\gamma = 1$

Antes de presentar los resultados obtenidos es importante mencionar que en el caso de RBFs la elección del "mejor" conjunto de entradas no suele ser tan inmediato, ya que suele haber conjuntos con resultados similares o resultados muy distintos según la métrica observada. Por esta razón, no se elige únicamente un conjunto de entradas en cada iteración, sino que se eligen varios que presenten

buenos resultados según diferentes criterios. En la *Tabla 4.7* se muestra el resumen del proceso de selección de entradas.

Tabla 4.6 Resumen del proceso de selección de entradas y errores obtenidos usando modelos RBF.

m	Conjunto de entradas	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
1	$V(t-1176)$	0.9702	0.8534	0.7699	0.9127
1	$V(t-1344)$	0.9786	0.8644	0.7708	0.9148
2	$V(t-1176), V(t-168)$	0.1772	0.0935	0.2071	0.1287
2	$V(t-1344), V(t-336)$	0.1329	0.0894	0.2616	0.1718
3	$V(t-1176), V(t-168), V(t-1175)$	0.1656	0.0914	0.1904	0.1026
3	$V(t-1344), V(t-336), V(t-1175)$	0.1573	0.0861	0.1512	0.0821
3	$V(t-1344), V(t-336), V(t-167)$	0.1122	0.0690	0.1151	0.0602
4	$V(t-1344), V(t-336), V(t-1175), V(t-168)$	0.1070	0.0663	0.1622	0.0942
4	$V(t-1344), V(t-336), V(t-1175), V_{mean7}(t-169)$	0.1154	0.0726	0.1525	0.0808
4	$V(t-1344), V(t-336), V(t-167), V_{damt}(t+168)$	0.1031	0.0657	0.1570	0.0873
5	$V(t-1344), V(t-336), V(t-1175), V(t-168), V_{damt}(t+168)$	0.1047	0.0616	0.1549	0.0940
5	$V(t-1344), V(t-336), V(t-1175), V_{mean7}(t-169), V_{damt}(t+168)$	0.1024	0.0619	0.1512	0.0814

Se puede representar gráficamente la evaluación de estos modelos RBF según las distintas métricas empleadas. En la *Figura 4.12* se representa en cada subfigura el error correspondiente en CP frente al error correspondiente en CE. De esta forma, cada punto se corresponde con un modelo y cuanto más cerca esté un modelo de la esquina inferior izquierda mayor será el rendimiento general del mismo.

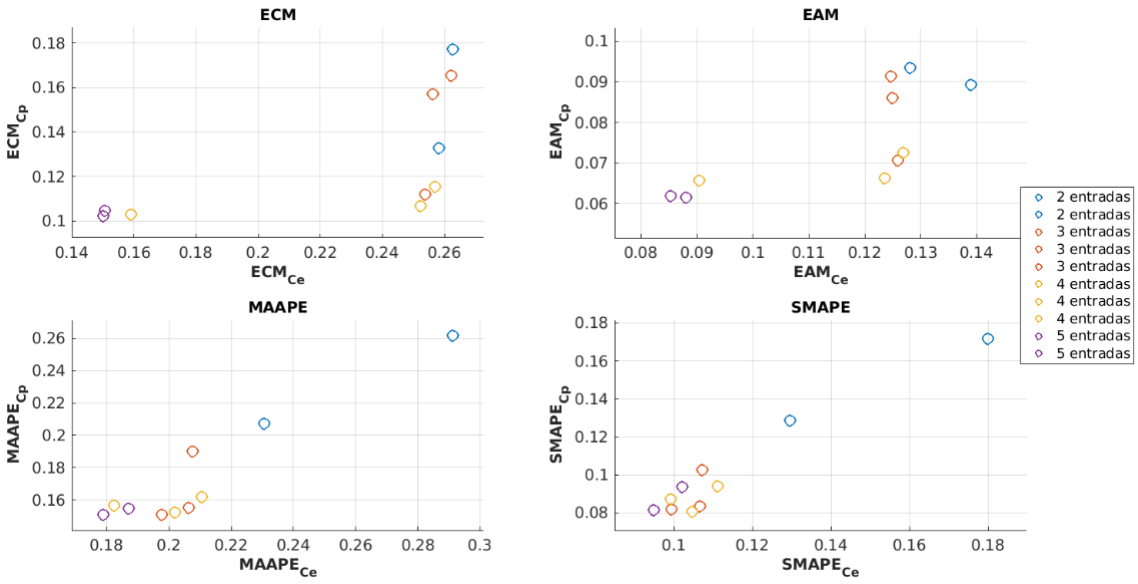


Figura 4.12 Comparativa de modelos RBF con distintos conjuntos de entrada.

Seguidamente, se realiza una búsqueda en cuadrícula partiendo de un conjunto de entradas que haya dado buenos resultados. Por ejemplo, se utiliza el conjunto $\{V(t-1344), V(t-336), V(t-1175), V_{mean7}(t-169)\}$. Los hiperparámetros más influyentes en los modelos RBF son el número de neuronas, nn, y la amplitud de la gaussiana, σ. Se define la siguiente cuadrícula de valores:

- $nn = [10, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43, 46, 49]$
- $\sigma = [1.5, 2, 2.5, 3, 3.5, 4, 4.5]$

De nuevo, se crea un modelo RBF por cada combinación única de parámetros y se obtiene la siguiente mejor combinación: $nn = 46$, $\sigma = 4.5$. Con esta nueva mejor combinación de hiperparámetros se vuelve a crear un modelo RBF para los conjuntos de entrada destacados y se obtienen los siguientes resultados:

Tabla 4.7 Modelos RBF con conjuntos de entradas destacados tras el ajuste de hiperparámetros.

m	Conjunto de entradas	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
3	$V(t-1344), V(t-336), V(t-167)$	0.1131	0.0690	0.1151	0.0602
4	$V(t-1344), V(t-336), V(t-1175), V(t-168)$	0.1035	0.0657	0.1104	0.0574
4	$V(t-1344), V(t-336), V(t-1175), V_{mean7}(t-169)$	0.1142	0.0702	0.1156	0.0596
4	$V(t-1344), V(t-336), V(t-167), V_{damt}(t+168)$	0.1031	0.0809	0.1029	0.0530
5	$V(t-1344), V(t-336), V(t-1175), V(t-168), V_{damt}(t+168)$	0.0804	0.0529	0.1080	0.0549
5	$V(t-1344), V(t-336), V(t-1175), V_{mean7}(t-169), V_{damt}(t+168)$	0.0799	0.0519	0.1050	0.0536

De forma similar a la *Figura 4.12* se pueden representar los modelos que ya se tenían frente a estos nuevos con los hiperparámetros ya ajustados. En la *Figura 4.13* se comprueba que los mejores modelos RBF se obtienen tras realizar la búsqueda en cuadrícula.

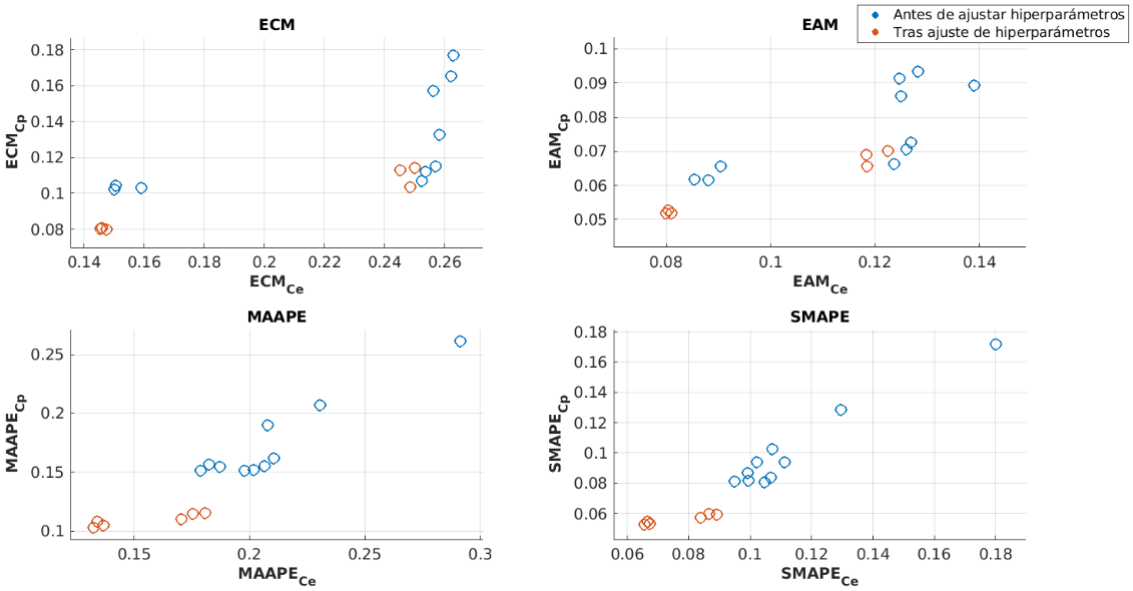


Figura 4.13 Comparativa de modelos RBF antes y después de ajustar hiperparámetros.

Redes neuronales recurrentes LSTM

Se utilizan los siguientes valores iniciales de hiperparámetros:

- $nn = 10$
- $\alpha = 0.05$
- $p_{Dropout} = 0.2$
- $miniBatchSize = 168$
- $maxPasadas = 200$
- $pacienciaCP = 20$

Antes de continuar es importante mencionar que en el caso de los modelos LSTM la inicialización de parámetros (no hiperparámetros) influye de forma significativa en el resultado obtenido. Por este motivo, se ha decidido crear cinco modelos para cada conjunto de entradas e hiperparámetros únicos. Idealmente, se crearía un número mayor de modelos para obtener resultados lo más fieles a realidad como sea posible, pero esto resulta muy costoso computacionalemente, por lo que se limita el número de modelos creados a cinco.

A la hora de representar los resultados se hace uso de los diagramas de caja (y bigotes) o "boxplot", que representan un resumen de la distribución de un conjunto de datos (en el caso de los resultados esos datos son los errores obtenidos por los diferentes modelos). En la figura *Figura 4.14* se ilustra la representación de diagrama de caja.

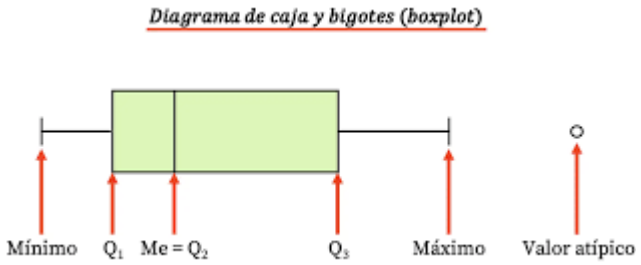


Figura 4.14 Explicación del diagrama de caja y bigotes [7].

Los valores Q1, Q2 y Q3 representan los cuartiles, que son medidas estadísticas que dividen el conjunto de datos en cuatro partes. Así, Q1 representa el valor debajo del cual se encuentra el 25 % de los valores, Q2 (mediana) el 50 % y Q3 el 75 %. Aunque este tipo de gráfica sería más útil en el caso de tener más muestras (solo se crean cinco modelos y por tanto, se tienen cinco valores de errores para cada métrica), sigue siendo de utilidad al proporcionar información relevante sobre los resultados obtenidos.

En la *Figura 4.15* se ilustra la aplicación de este tipo de gráfica a una iteración del proceso de selección de entradas.

A continuación, se muestra un resumen del proceso de selección de entradas. Los errores indicados se corresponden con los mínimos obtenidos de entre los cinco modelos de cada conjunto de entradas e hiperparámetros únicos.

Tabla 4.8 Resumen del proceso de selección de entradas para modelos LSTM.

m	Conjunto de entradas	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
1	$V(t-1344)$	0.2589	0.1217	0.2341	0.1165
1	$V(t-168)$	0.1147	0.0787	0.1827	0.0856
2	$V(t-1344), V(t-168)$	0.1074	0.0692	0.1976	0.0927
3	$V(t-1344), V(t-168), V(t-1)$	0.0944	0.0626	0.1753	0.0806
4	$V(t-1344), V(t-168), V(t-1), V(t-336)$	0.0916	0.0605	0.1486	0.0700
5	$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167)$	0.0858	0.0590	0.1397	0.0665
6	$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167), V_damt(t+168)$	0.0748	0.0512	0.1321	0.0622
7	$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167), V_{damt}(t+168), V_{2damt}(t+168)$	0.0732	0.0510	0.1096	0.0546
8	$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167), V_{damt}(t+168), V_{2damt}(t+168), dS_{ord}(t+168)$	0.0691	0.0479	0.1325	0.0620
9	$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167), V_{damt}(t+168), V_{2damt}(t+168), dS_{ord}(t+168), dS(t+168)$	0.0667	0.0458	0.1240	0.0619

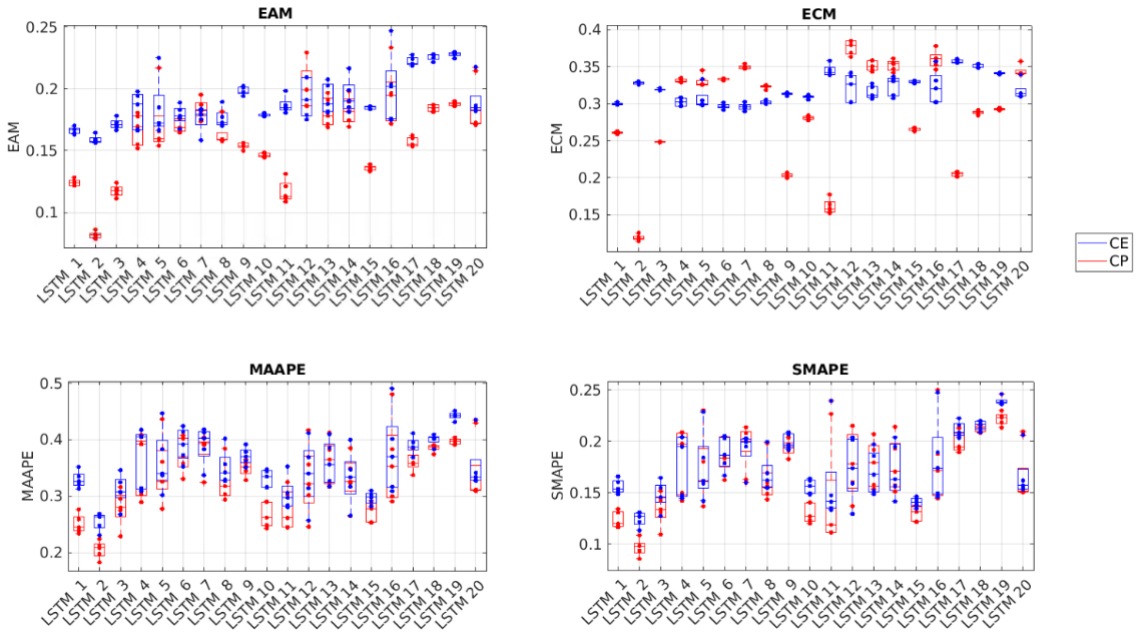


Figura 4.15 Representación de resultados de modelos LSTM mediante diagramas de caja.

En la *Tabla 4.8* se puede ver que en algunas iteraciones algunos errores aumentan con respecto a la iteración anterior. Esto es porque las conclusiones obtenidas varían con la métrica que se observa y se tome de referencia. En este caso, las decisiones tomadas se han basado principalmente en los errores EAM y ECM, pero a la hora de elegir modelos destacados se han tenido en cuenta todas las métricas.

Se realiza un ajuste de hiperparámetros con el último conjunto de entradas elegido y se crean modelos LSTM con los nuevos hiperparámetros y conjuntos de entrada destacados. En la *Tabla 4.9* se muestran algunos modelos LSTM destacados según distintas métricas.

Tabla 4.9 Modelos LSTM destacados tras el ajuste de hiperparámetros.

Conjunto de entradas	nn	α	$p_{Dropout}$	$miniBatchSize$	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167), V_{daml}(t+168), V_{2daml}(t+168), dS_{ord}(t+168), dS(t+168)$	30	0.05	0.2	168	0.0660	0.0450	0.1013	0.0598
$V(t-1344), V(t-168), V(t-1), V(t-336), V(t-167), V_{daml}(t+168), V_{2daml}(t+168), dS_{ord}(t+168), dS(t+168)$	20	0.05	0	168	0.0798	0.0519	0.1031	0.0506

Comparación de modelos

En este apartado se comparan los mejores modelos de cada tipo obtenidos para determinar cuál presenta mayor rendimiento en este problema. En la *Figura 4.16* se representan los modelos destacados de cada tipo mediante la gráfica de error en CP frente a error en CE.

Primero, se muestran en la *Tabla 4.10* los mejores modelos de cada tipo según ECM en CP.

Tabla 4.10 Mejores modelos de cada tipo según ECM en CP.

Modelo	ECM_{CP}	EAM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
ARX	0.0795	0.0507	0.1047	0.0552
RBF	0.0799	0.0519	0.1050	0.0536
LSTM	0.0660	0.0450	0.1013	0.0598

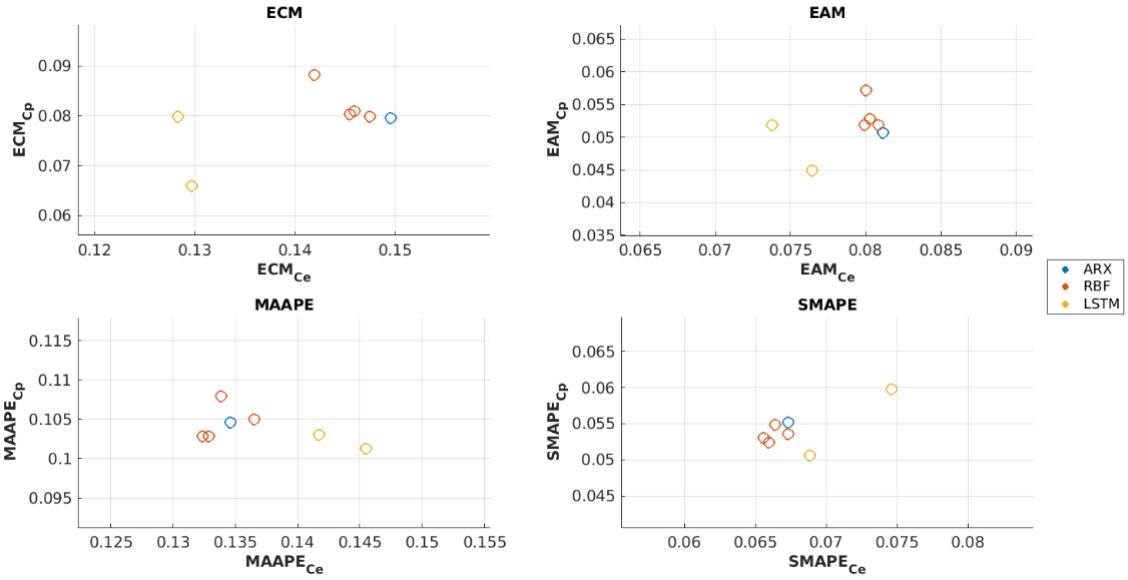
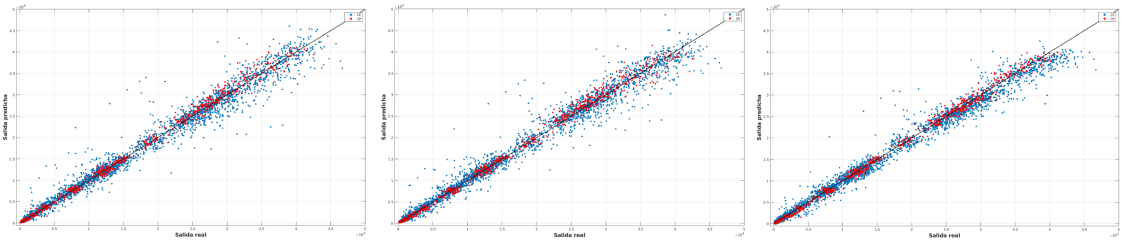


Figura 4.16 Comparación del rendimiento de los diferentes tipos de modelos para la predicción del volumen de llamadas en un CAC.

A continuación, se representa en la *Figura 4.17* la salida predicha frente a la salida real para cada uno de los modelos de la *Tabla 4.10* a modo de comparación.



(a) Modelo ARX. **(b)** Modelo RBF. **(c)** Modelo LSTM.

Figura 4.17 Salida predicha frente a salida real en CE y CP para los diferentes modelos.

Las conclusiones de resultados varían según la métrica y conjunto observados. En la *Tabla 4.11* se muestra un resumen de dichas conclusiones.

Tabla 4.11 Mejor modelo según la métrica y conjunto observados.

Conjunto / Métrica	CE	CP
ECM	LSTM	LSTM
EAM	LSTM	LSTM
MAAPE	RBF	LSTM
SMAPE	RBF	LSTM

Por tanto, el modelo que mejores resultados presenta en general es el LSTM, siendo el de ma-

yor rendimiento en el conjunto de prueba (CP) y superado únicamente por el RBF en los errores porcentuales (MAAPE, SMAPE) para el conjunto de entrenamiento (CE).

4.3 Predicción del saldo acumulado diario de transacciones bancarias

En este caso se tiene un conjunto de datos financieros pertenecientes a un gran banco iberoamericano. En este conjunto se identifican diferentes unidades monetarias, sucursales y sistemas de efectivo, que definen numerosas series temporales.

Antes de continuar es necesario mencionar que por motivos de confidencialidad no se mencionará ningún nombre relativo al banco, sucursales, etc. De igual manera, tampoco se mostrarán las series temporales originales, sino que se normalizarán los datos siguiendo una distribución normal ($\mu = 0$, $\sigma = 1$) y se ignorará la unidad monetaria empleada.

Para este trabajo se utilizará una serie temporal de una unidad monetaria y sucursal concretas y en cuanto al sistema de efectivo se trabajará con lo que se conocen como buzoneras. Estos dispositivos solo permiten el ingreso de efectivo por parte de los clientes. Se denomina transacción a cada ingreso individual realizado por un cliente. Como se puede deducir, se trata de una serie temporal con un intervalo de recolección de datos irregular y desconocido, puesto que no se puede saber cuándo un cliente va a realizar una transacción. Por este motivo, se preprocesan los datos para obtener la serie temporal de saldo acumulado diario de transacciones, de forma que el periodo de la serie temporal es de un día y es constante a lo largo de la serie. En este caso, el horizonte de predicción será de una semana, es decir, $h = 7$ días.

Otro punto importante a tener en cuenta en este problema es que los datos son recibidos con dos días de retraso, por lo que los desfases que se utilicen para las variables de entradas deberán cumplir que $d \leq -2$.

El conjunto de transacciones originales para la sucursal y divisa concretas presenta un total de $N = 72185$ registros que van desde el 5 de julio de 2019 hasta el 3 de noviembre de 2022. Tras el preprocesamiento y transformación de los datos y obtener la serie temporal de saldo acumulado diario de transacciones el número de observaciones se reduce a $N = 1189$.

Tras un primer análisis de la serie temporal se detecta que el número de días entre las fechas inicial y final de la serie temporal es de 1218. Sin embargo, en la serie temporal se tienen datos de 1189 días. Por tanto, hay 29 días (2.38 % del total de días) que no presentan datos de transacciones, lo cual podría deberse a un fallo en el dispositivo o en la captura de las transacciones, por ejemplo. Este será uno de los problemas a resolver.

Primero, se comienza representando la serie temporal (*Figura 4.18*).

La primera conclusión que se obtiene a simple vista es que parece ser una serie temporal irregular y caótica. En la *Figura 4.19* se muestran dos rangos de tiempo diferentes más reducidos.

En la *Figura 4.19 (a)* se demuestra la falta de datos de transacciones en determinados días, sobre todo domingos. Esto hace que la serie temporal tenga una forma más caótica, dificultando la predicción de la misma. En la *Figura 4.19 (b)* se observa cierta periodicidad semanal, con picos de saldo los lunes y viernes (aunque no siempre se cumple) y caídas de saldo los sábados y domingos con

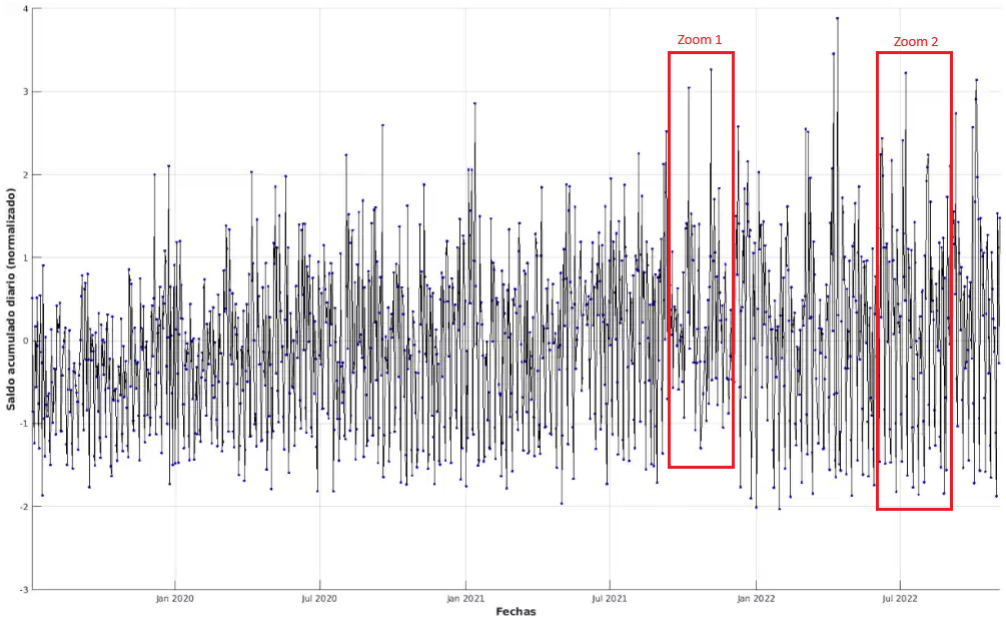
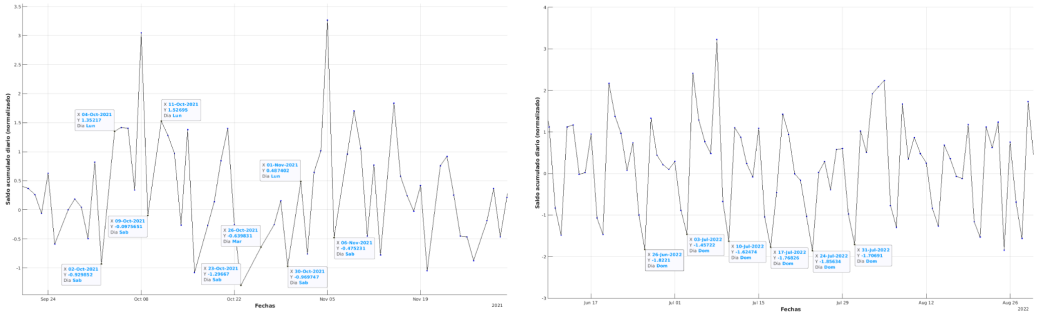


Figura 4.18 Saldo acumulado diario de las buzoneras de una sucursal concreta.



(a) Zoom 1 . **(b)** Zoom 2 .

Figura 4.19 Serie temporal en rangos de tiempo más reducidos.

respecto a los días laborables.

Por tanto, resumiendo:

- Faltan datos de transacciones en 29 días de un total de 1218 días (2.38 %), sobre todo son domingos. Será necesario aplicar alguna solución de "rellenado".
- La serie temporal presenta cierta periodicidad semanal. Será necesario realizar un breve análisis para definir una variable "tipo de día".

Se comienza realizando un análisis de los días de la semana para determinar el tipo de día. Para ello se utilizan dos gráficas distintas. En la *Figura 4.20* se muestra para cada día de la semana el número de transacciones y el saldo acumulado medio para ese día.

En la *Figura 4.21* se utilizan diagramas de caja para cada día de la semana. De esta forma se puede ver la distribución de los datos para los distintos días.

A partir de estas gráficas se definen los siguientes tipos de días:

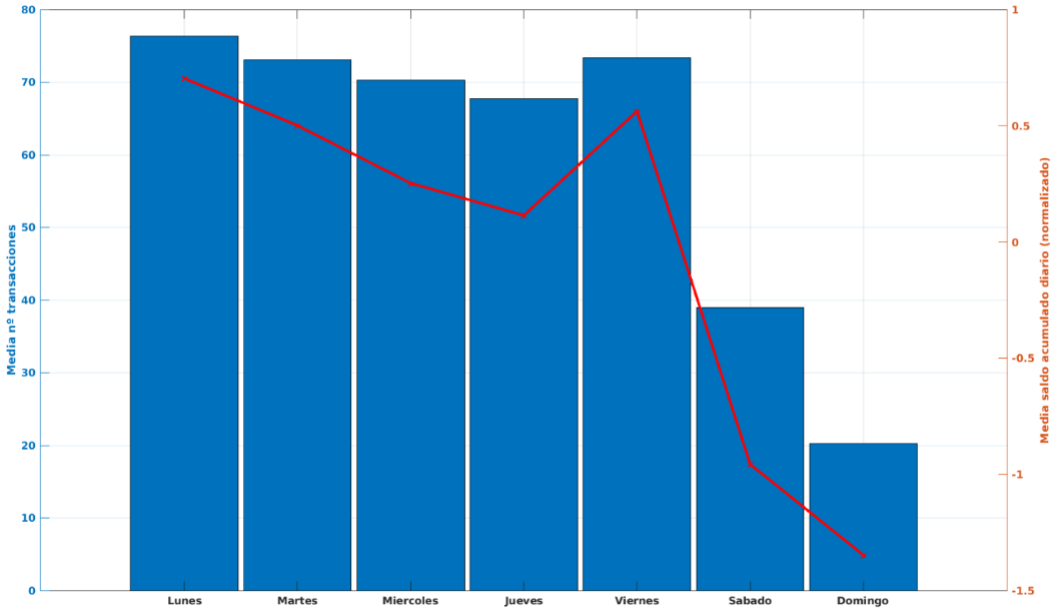


Figura 4.20 Número y saldo acumulado diario medios en los distintos días de la semana.

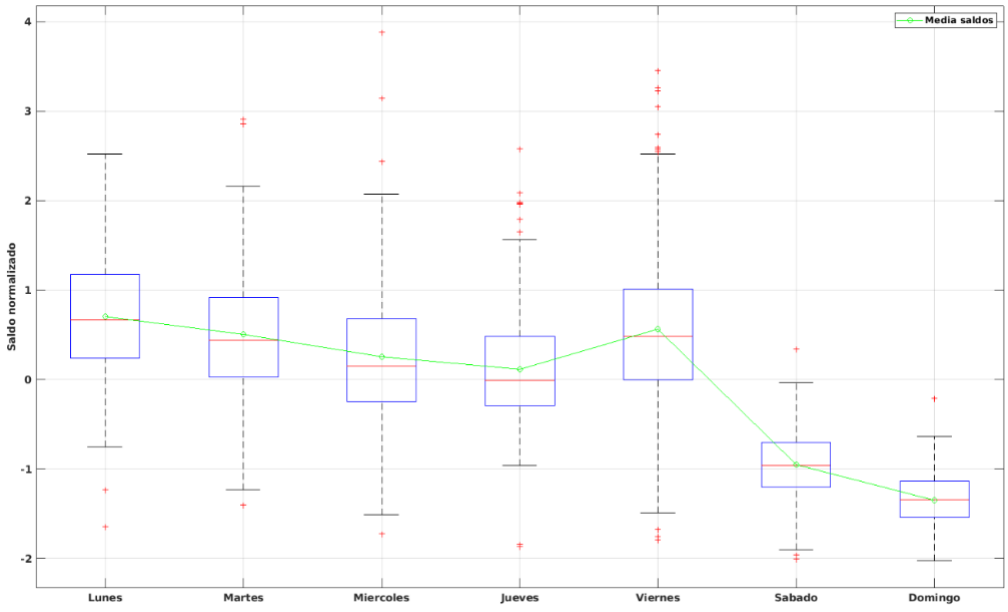


Figura 4.21 Diagramas de caja para el saldo acumulado de los diferentes días de la semana.

Tabla 4.12 Definición de los tipos de días.

Tipo de día	Días correspondientes	Valor asociado
1	Lunes, Viernes	0.8
2	Martes, Miércoles, Jueves	0.6
3	Sábado, Domingo	0.2

Los lunes y viernes se toman como mismo tipo de día, pues ambos presentan un incremento

de saldo con respecto al día anterior correspondiente, además de presentar los valores de saldo máximos. Los martes, miércoles y jueves tienen un saldo acumulado medio similar, considerándose como el mismo tipo de día. Por último, se define el tipo de día de fin de semana (sábado y domingo), que presentan un saldo acumulado mucho menor que el resto de días laborables. Los valores asociados definirán la variable "tipo de día" (TD) y normalmente se eligen de forma arbitraria, pero teniendo en cuenta la diferencia de saldo entre los distintos días de la semana (a mayores saldos, mayor será el valor asociado).

En cuanto a los datos faltantes se aplicará como solución utilizar el último dato conocido. Es importante mencionar que nunca se va a realizar una predicción para un día que no tenga datos, lo cual implica que para la salida no será necesario aplicar ninguna solución.

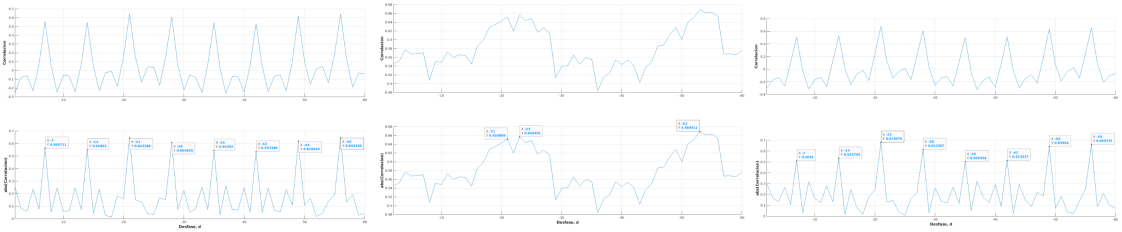
En la *Tabla 4.13* se muestra el conjunto de variables de entrada definidas:

Tabla 4.13 Conjunto de entradas potenciales definidas.

Descripción de la entrada potencial	Abreviatura	Tipo entrada	Rango desfases
Saldo acumulado diario utilizando el último valor conocido en caso de no tener dato	S	Autorregresiva	[-60 -2]
Saldo acumulado diario utilizando siempre saldos del mismo tipo de día que la salida	S_{damt}	Autorregresiva	[-60 -2]
Media de saldo acumulado de los últimos 5 días con dato	S_{mean5}	Autorregresiva	[-60 -2]
Mínimo valor de saldo acumulado en los últimos 5 días con dato	S_{min5}	Autorregresiva	[-60 -2]
Máximo valor de saldo acumulado en los últimos 5 días con dato	S_{max5}	Autorregresiva	[-60 -2]
Número de transacciones	nT	Exógena	[-60 -2]
Día de la semana	dS	Exógena	[7]
Día del mes	dM	Exógena	[7]
Mes	M	Exógena	[7]
Tipo de día	TD	Exógena	[7]
Seno del día de la semana	dS_{sen}	Exógena	[7]
Coseno del día de la semana	dS_{cos}	Exógena	[7]
Seno del día del mes	dM_{sen}	Exógena	[7]
Coseno del día del mes	dM_{cos}	Exógena	[7]
Seno del mes	M_{sen}	Exógena	[7]
Coseno del mes	M_{cos}	Exógena	[7]
Semana del mes	sM	Exógena	[7]
Seno de la semana del mes	sM_{sen}	Exógena	[7]
Coseno de la semana del mes	sM_{cos}	Exógena	[7]

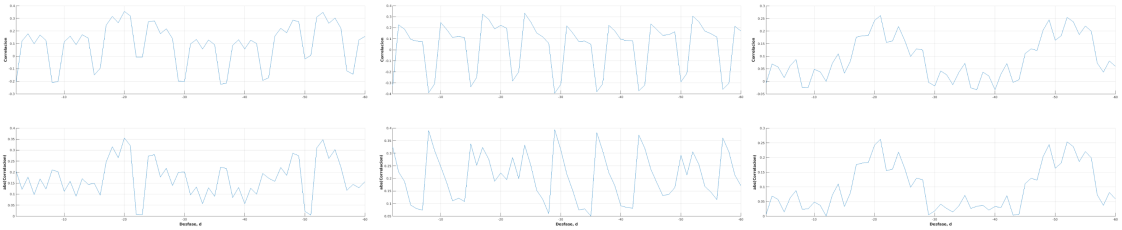
Con esto ya se tendrían los datos preparados. Antes de pasar a la creación de modelos, se muestra la gráfica de correlación de algunas variables de entrada para tener una idea de las correlaciones obtenidas en este problema y la diferencia con la serie temporal del servicio de atención al cliente.

Como se puede ver en las *Figuras 4.22, 4.23* los valores de correlación en este caso son significativamente menores que los que se obtenían con la serie de temporal del CAC, siendo estas



(a) $S(t+d)$. **(b)** $S_{damt}(t+d)$. **(c)** $nT(t+d)$.

Figura 4.22 Correlaciones de Pearson de la salida $S(t+7)$ con las diferentes variables de entrada que no aplican ninguna operación.



(a) $S_{mean5}(t+d)$. **(b)** $S_{min5}(t+d)$. **(c)** $S_{max5}(t+d)$.

Figura 4.23 Correlaciones de Pearson de la salida $S(t+7)$ con las diferentes variables de entrada que aplican una operación sobre los datos.

bastante menores para las variables que aplican una operación sobre los datos (*Figura 4.23*).

En las *Figuras 4.22 (a), 4.22 (c)* se comprueba la existencia de una periodicidad semanal, al ser las correlaciones para los desfases $d = -k \cdot 7$ mayores que el resto. En la *Figura 4.22 (b)*, sin embargo, se observa un comportamiento bastante distinto, con valores de correlación $\rho > 0.48$ y encontrando los valores mayores para desfases cercanos a las 3 semanas ($d = -21$) y a las 8 semanas ($d = -56$). Es decir, la entrada S_{damt} presenta valores de correlación generalmente elevados para cualquier desfase, mientras que las entradas S y nT presentan valores de correlación elevados sólo para determinados desfases, concretamente múltiplos de -7.

En las *Figuras 4.23 (a), 4.23 (b)* se aprecia cierta similitud en las correlaciones, mientras que la variable de entrada de la *Figura 4.23 (c)* tiene una gráfica de correlaciones completamente distinta.

Por último, se muestra en la *Tabla 4.14* los grupos de entradas definidos para las pruebas:

Tabla 4.14 Grupos de entradas definidos.

Grupo de entradas	Entradas incluidas
1	S
2	S_{damt}
3	S_{mean5}
4	S_{min5}
5	S_{max5}
6	nT
7	dS, dM, M, TD
8	dS_{sen}, dS_{cos}, dM_{sen}, dM_{cos}, M_{sen}, M_{cos}, sM, sM_{sen}, sM_{cos}

Con esto, se procede a la creación de modelos, comenzando con lo más simple: los modelos ARX. Puesto que ya se ha ilustrado este procedimiento, en este problema se presentan los resultados finales obtenidos.

Modelos ARX

Antes de presentar los mejores modelos ARX obtenidos, se definen los conjuntos de entradas que han dado lugar a modelos ARX destacados y se asocia un identificador para distinguirlos entre ellos. En la *Tabla 4.15* se muestran las entradas comunes a todos estos conjuntos.

Tabla 4.15 Entradas comunes de conjuntos de entrada destacados para modelos ARX.

Variables de entrada	Desfases (d)
S	-56, -49, -28, -7, -2, -12
S_{damt}	-23, -52, -26, -27
S_{mean5}	-19, -53, -46, -2, -55, -25
S_{min5}	-58, -17, -44, -53, -29
S_{max5}	-55

En la *Tabla 4.16* se muestran las entradas adicionales a las entradas comunes que presentan los diferentes conjuntos de entrada destacados.

Tabla 4.16 Conjuntos de entradas de modelos ARX destacados.

CX	Variable de entrada	Desfases (d)
CX 1	nT	-28
CX 2	nT	-12
CX 3	TD	7
	dS_{sen}	7
CX 4	TD	7
	dS_{sen}	7
	dS_{cos}	7
CX 5	TD	7
	dS_{sen}	7
	dM_{sen}	7
CX 6	TD	7
	dS_{sen}	7
	sM_{sen}	7

En la *Tabla 4.17* se muestran los modelos ARX destacados obtenidos para la predicción de $S(t+7)$.

En la *Figura 4.24* se representa gráficamente el rendimiento de los modelos presentados en la *Tabla 4.17*

Tabla 4.17 Modelos ARX con mayor rendimiento según diferentes métricas.

Conjunto de entradas	γ	δ	EAM_{CP}	ECM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
CX 1	1	10	0.4894	0.6851	0.2250	0.1178
CX 2	1	10	0.4916	0.6736	0.2329	0.1250
CX 3	1	10	0.4710	0.6777	0.2264	0.1148
CX 4	1	10	0.4720	0.6797	0.2262	0.1148
CX 5	1	10	0.4750	0.6801	0.2319	0.1174
CX 6	1	10	0.4707	0.6770	0.2290	0.1161

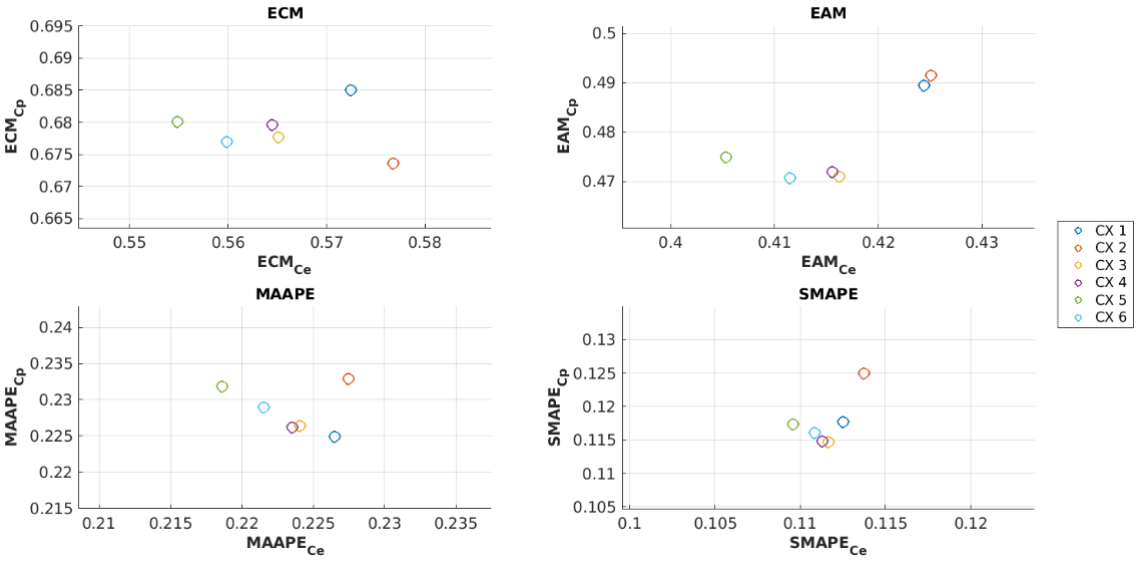


Figura 4.24 Representación gráfica del rendimiento de modelos ARX destacados.

Modelos RBF

Se parte de los siguientes valores de hiperparámetros iniciales:

- $nn = 125$
- $\sigma = 2$
- $\alpha = 0.001$
- $nPasadas = 50$
- $\gamma = 1$

En este caso destacan dos conjuntos de entradas. El primero (CX 7) presenta las siguientes entradas:

Tabla 4.18 Entradas comunes de conjuntos de entrada destacados para modelos RBF.

Variables de entrada	Desfases (d)
S	-7, -35
S_{damt}	-22, -53, -23, -49
S_{mean5}	-52, -55, -21, -49
nT	-42, -20
dS	7
M	7
TD	7
sM	7
dS_{cos}	7
dM_{cos}	7

El segundo conjunto de entradas (CX 8) es idéntico al CX 7 pero incluye una entrada adicional: $S(t-42)$.

En la *Tabla 4.19* se muestran los mejores modelos RBF obtenidos tras el ajuste de hiperparámetros mediante búsqueda en cuadrícula según las diferentes métricas.

Tabla 4.19 Modelos RBF con mayor rendimiento según diferentes métricas.

Conjunto de entradas	nn	σ	γ	α	$nPasadas$	EAM_{CP}	ECM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
CX 7	150	2	1	0.001	50	0.4618	0.6756	0.2045	0.1065
CX 8	150	2	1	0.001	50	0.4605	0.6678	0.2067	0.1072

Modelos LSTM

Se parte de la siguiente combinación inicial de hiperparámetros:

- $nn = 10$
- $\alpha = 0.05$
- $p_{Dropout} = 0.2$
- $miniBatchSize = 32$

- *maxPasadas* = 200

- *pacienciaCP* = 20

Para los modelos LSTM destaca el siguiente conjunto de entradas:

Tabla 4.20 Conjunto de entradas destacado con modelos LSTM.

Variables de entrada	Desfases (d)
S	-21, -49, -42, -2, -5, -12
nT	-5, -2

Como se puede ver, este conjunto destaca por utilizar únicamente ocho entradas de dos grupos de entradas. En este caso se comprueba la capacidad de las redes neuronales LSTM de extraer las características de la serie temporal empleando pocas entradas y disminuyendo así el trabajo de extracción de características.

En la *Tabla 4.21* se muestran los mejores modelos LSTM obtenidos con este conjunto de entradas tras el ajuste de hiperparámetros mediante búsqueda en cuadrícula según las diferentes métricas.

Tabla 4.21 Modelos LSTM con mayor rendimiento según diferentes métricas.

nn	α	$p_{Dropout}$	$miniBatchSize$	$maxPasadas$	$pacienciaCP$	EAM_{CP}	ECM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
42	0.05	0.3	32	200	20	0.4707	0.6597	0.2304	0.1153
40	0.05	0.3	32	200	20	0.4542	0.6705	0.2141	0.1068
46	0.05	0.2	32	200	20	0.4674	0.6891	0.2099	0.1076

Comparación de modelos

Primero, se comparan los resultados obtenidos por los diferentes modelos empleando dos gráficas. En la *Figura 4.25* se representa el rendimiento de los modelos en CP frente a CE.

En la *Tabla 4.22* se muestran los mejores resultados obtenidos con cada modelo según ECM en CP y en la *Figura 4.26* se representa la salida predicha frente a la salida real para dichos modelos.

Tabla 4.22 Mejores modelos de cada tipo según ECM en CP.

Modelo	EAM_{CP}	ECM_{CP}	$MAAPE_{CP}$	$SMAPE_{CP}$
ARX	0.4916	0.6736	0.2329	0.1250
RBF	0.4605	0.6678	0.2067	0.1072
LSTM	0.4707	0.6597	0.2304	0.1153

De nuevo, se obtienen diferentes conclusiones según la métrica y conjunto observados. En la *Tabla 4.23* se muestra un resumen de dichas conclusiones

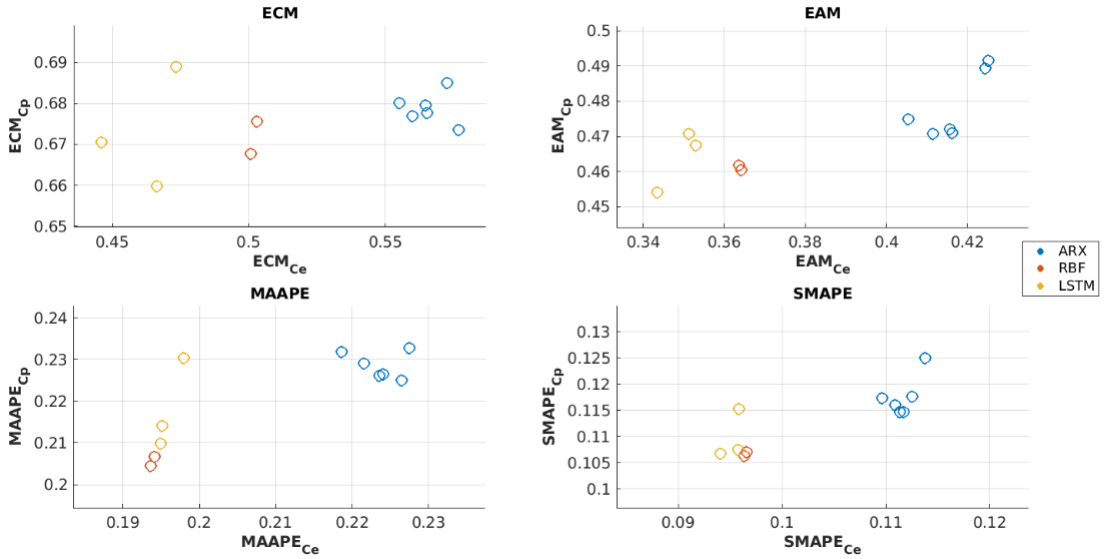
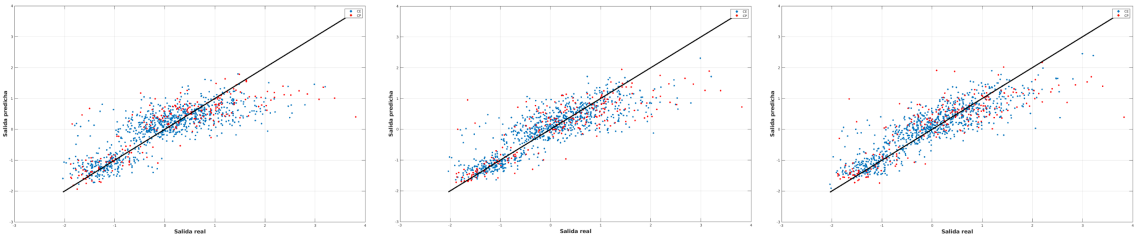


Figura 4.25 Comparación del rendimiento de los diferentes tipos de modelos para la predicción del saldo acumulado diario dentro de 7 días.



(a) Modelo ARX. **(b)** Modelo RBF. **(c)** Modelo LSTM.

Figura 4.26 Salida predicha frente a salida real en CE y CP para los diferentes modelos.

Tabla 4.23 Mejor modelo según la métrica y conjunto observados.

Conjunto / Métrica	CE	CP
ECM	LSTM	LSTM
EAM	LSTM	LSTM
MAAPE	RBF	RBF
SMAPE	LSTM	RBF

Se obtienen conclusiones parecidas a las obtenidas con la serie temporal anterior (*Tabla 4.11*). Al igual que antes, los modelos LSTM presentan los mejores resultados en los errores no porcentuales (EAM, ECM) para ambos conjuntos (CE, CP). En este caso, sin embargo, los modelos RBF igualan (e incluso superan) a los LSTM en los errores porcentuales (MAAPE, SMAPE), excepto en el SMAPE en CE.

4.4 Creación automatizada de modelos de predicción

En este apartado se parte del modelo de predicción LSTM y se aplica el método automatizado de creación de modelos de predicción a diferentes series temporales del conjunto de datos pertenecientes al banco iberoamericano para comprobar su eficacia.

4.4.1 Primera prueba: Comprobación del correcto funcionamiento del método

Se comienza aplicando la metodología de automatización a la misma serie temporal predicha en el apartado anterior para corroborar que los resultados obtenidos son válidos, así como comprobar si se obtienen conclusiones similares en cuanto a entradas e hiperparámetros seleccionados.

Selección de entradas

Se han empleado los siguientes argumentos de entrada para el algoritmo de selección automática de entradas:

- Grupos de entrada ya utilizados para la última serie temporal (*Tabla 4.14*).
- Número de entradas a filtrar: $N = 20$.
- Paciencia en la selección hacia adelante: $P = 5$ iteraciones.
- Combinación de hiperparámetros idéntica a la empleada en la aplicación manual para poder comparar resultados:
  - nn = 10
  - α = 0.05
  - $p_{Dropout}$ = 0.2
  - $miniBatchSize$ = 32
  - $maxPasadas$ = 200
  - $pacienciaCP$ = 20

Se comienza ilustrando el método de selección automática de entradas utilizando como ejemplo el primer grupo de entradas: variable autorregresiva S y rango de desfases [-2, -60], teniendo un total de 59 entradas.

Las entradas resultantes del filtrado por correlación de Pearson son las siguientes: $S(t-56), S(t-21), S(t-8), S(t-7), S(t-14), S(t-35), S(t-42), S(t-37), S(t-33), S(t-2), S(t-40), S(t-9), S(t-12), S(t-44), S(t-5), S(t-16), S(t-47), S(t-30), S(t-58)$.

En la *Figura 4.27* se ilustra un resumen del método de selección hacia adelante para el primer grupo de entradas tras el filtrado (se representa el mínimo error obtenido de entre todos los modelos LSTM creados para cada iteración).

En la *Tabla 4.24* se muestran las entradas seleccionadas por el método en cada iteración.

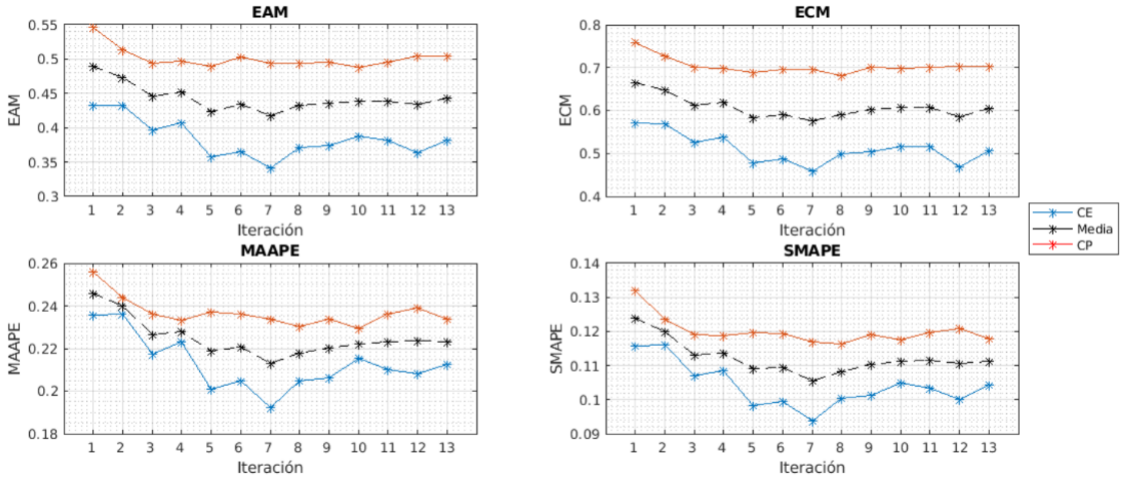


Figura 4.27 Ilustración del proceso de selección automática hacia adelante para el primer grupo de entradas.

Tabla 4.24 Evolución del conjunto base durante la selección automática hacia adelante para el primer grupo de entradas.

Iteración	Conjunto base	ECM_{CP}	Paciencia
1	$S(t-21)$	0.7584	5
2	$S(t-21), S(t-47)$	0.7265	5
3	$S(t-21), S(t-47), S(t-28)$	0.7005	5
4	$S(t-21), S(t-47), S(t-28), S(t-7)$	0.6989	5
5	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33)$	0.6898	5
6	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40)$	0.6954	4
7	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49)$	0.6947	3
8	$\mathbf{S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49), S(t-42)}$	**0.6824**	**5**
9	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49), S(t-42), S(t-16)$	0.7001	4
10	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49), S(t-42), S(t-16), S(t-12)$	0.6989	3
11	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49), S(t-42), S(t-16), S(t-12), S(t-56)$	0.7002	2
12	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49), S(t-42), S(t-16), S(t-12), S(t-56), S(t-9)$	0.7039	1
13	$S(t-21), S(t-47), S(t-28), S(t-7), S(t-33), S(t-40), S(t-49), S(t-42), S(t-16), S(t-12), S(t-56), S(t-9), S(t-2)$	0.7025	0

En esta tabla se comprueba que el método continúa iterando mientras el ECM en CP decrece y hasta pasadas $P = 5$ iteraciones tras alcanzar un mínimo absoluto en este criterio.

El método continúa iterando para el resto de grupos de entradas. En la *Figura 4.28* se representa el mínimo ECM en CP obtenido de entre todos los conjuntos de entradas de prueba creados para cada grupo de entradas.

Por último, se compara el mejor conjunto de entradas seleccionadas de forma automática y de forma manual tomando como criterio el ECM en CP (*Tabla 4.25*). Se puede comprobar que comparten algunas entradas, pero en general son conjuntos muy distintos.

En la *Tabla 4.26* se muestra el mejor modelo obtenido tras la selección automatizada de entradas y el mejor modelo obtenido tras la selección manual de entradas a modo de comparación.

Tabla 4.26 Evaluación del mejor modelo tras selección automatizada de entradas y comparación con el método manual.

Método - Fase	Nº entradas	EAM_{CE}	EAM_{CP}	ECM_{CE}	ECM_{CP}	$MAAPE_{CE}$	$MAAPE_{CP}$	$SMAPE_{CE}$	$SMAPE_{CP}$
Manual - Selección de entradas	8	0.3092	0.4586	0.4088	0.6654	0.1765	0.2174	0.0863	0.1098
Automatizado - Selección de entradas	20	0.2977	0.4575	0.4000	0.6483	0.1723	0.2284	0.0846	0.1141

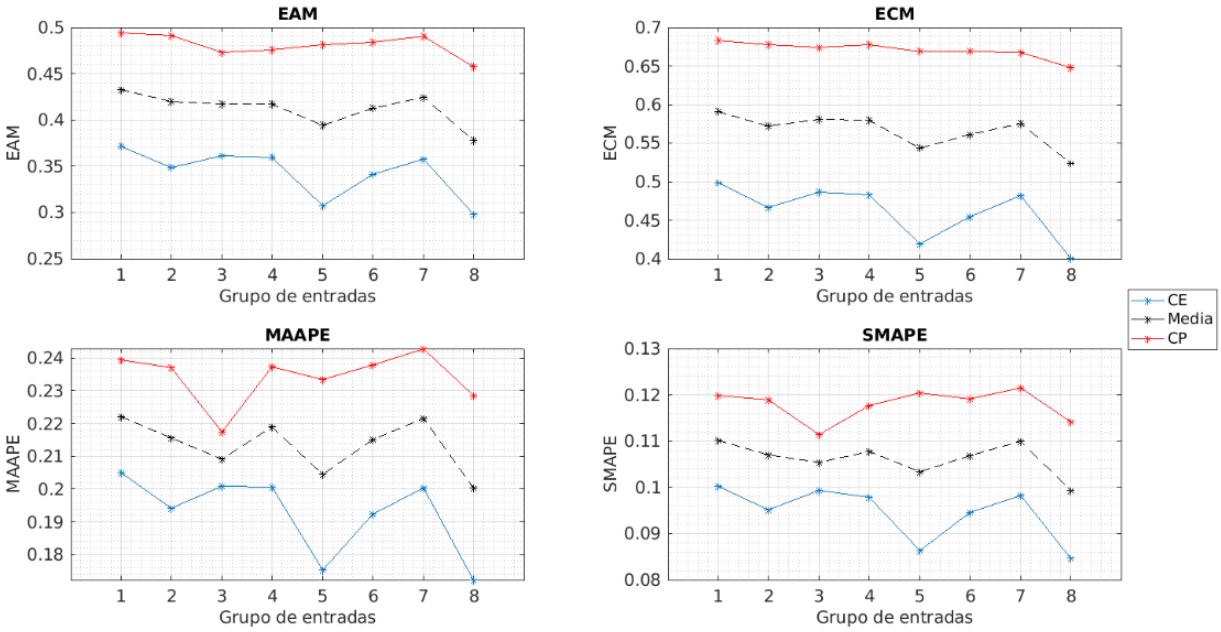


Figura 4.28 Resumen del proceso de selección automática hacia adelante para todos los grupos de entradas.

Tabla 4.25 Comparación de las entradas seleccionadas en las metodologías automatizada y manual según ECM en CP.

Variable	Entradas - Automático	Entradas - Manual
S	-21, -47, -28, -7, -33, -40, -49, -42	-21, -49, -42, -2, -5, -12
S_{damt}	-23	No
S_{mean7}	-36, -53, -17, -54	No
S_{min7}	No	No
S_{max7}	-53, -25	No
nT	No	-5, -2
dS	7	No
dM	No	No
M	No	No
TD	No	No
dS_{sen}	7	No
dS_{cos}	No	No
dM_{sen}	No	No
dM_{cos}	7	No
M_{sen}	7	No
M_{cos}	No	No
sM	No	No
sM_{sen}	No	No
sM_{cos}	7	No

Se comprueba que los errores obtenidos son en general similares entre ambos métodos, obteniendo un menor ECM en CP mediante el método automatizado al tratarse del criterio empleado para la selección de entradas. Otro punto a destacar es la diferencia entre el número de entradas seleccionadas por ambos métodos.

Ajuste de hiperparámetros

Se realizan las siguientes búsquedas en cuadrícula o barridos:

- $nn - miniBatchSize$ (*Figura 4.29*)

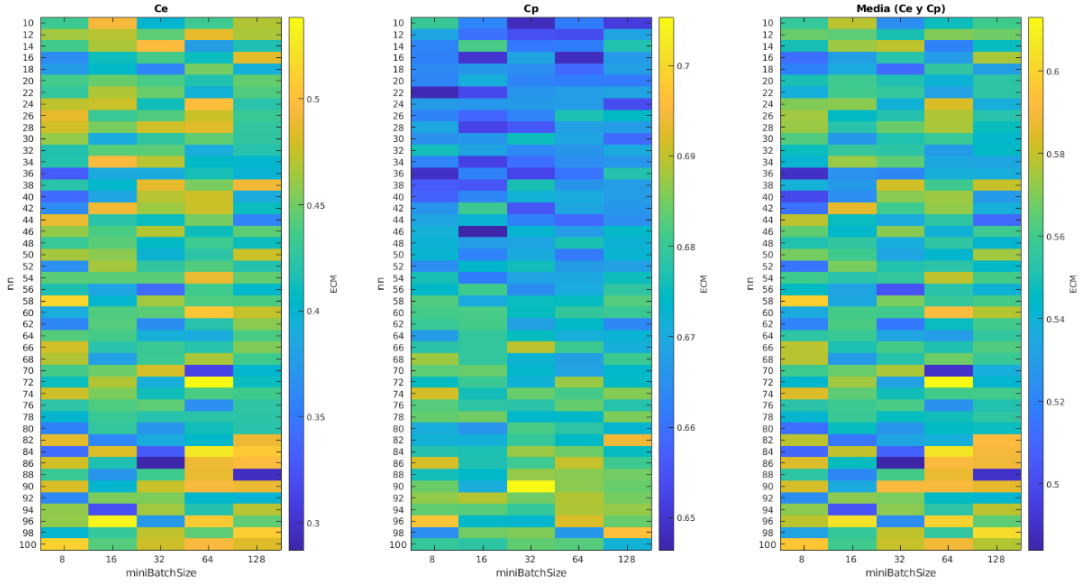


Figura 4.29 Búsqueda en cuadrícula de los hiperparámetros *nn* y *miniBatchSize*.

Se actualizan los hiperparámetros por defecto:

  - $nn = 10 \rightarrow 46$
  - $miniBatchSize = 32 \rightarrow 16$

- $p_{Dropout}$ (*Figura 4.30*)

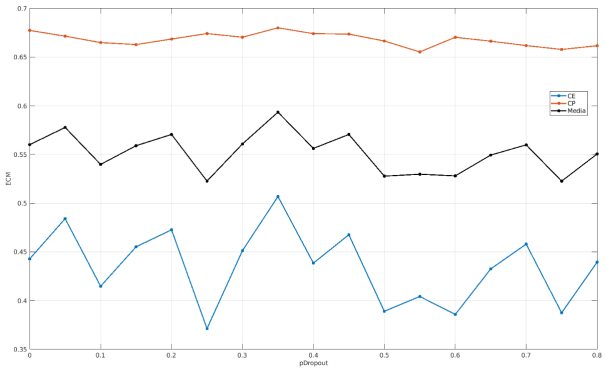


Figura 4.30 Búsqueda en cuadrícula del hiperparámetro *pDropout*.

Se actualizan los hiperparámetros por defecto:

- $pDropout = 0.2 \rightarrow 0.55$

- $\alpha - pacienciaCP$ (*Figura 4.31*)

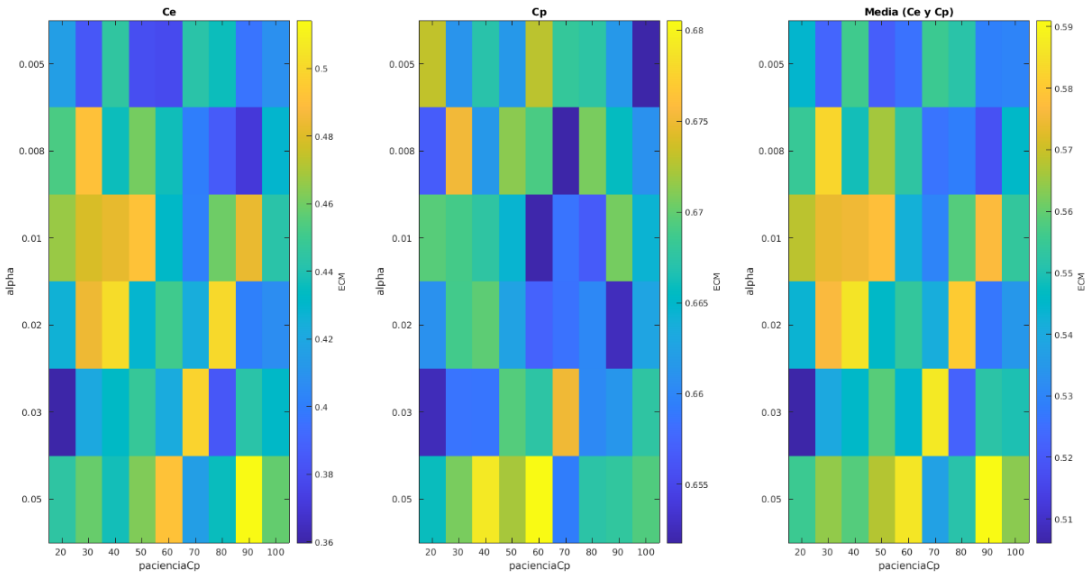


Figura 4.31 Búsqueda en cuadrícula de los hiperparámetros $alpha$ y $pacienciaCP$.

Se actualizan los hiperparámetros por defecto:

- $alpha = 0.05 \rightarrow 0.008$

- $pacienciaCP = 20 \rightarrow 70$

En la *Tabla 4.27* se compara el mejor modelo obtenido tras el ajuste automatizado de hiperparámetros y el mejor modelo tras el ajuste manual de hiperparámetros.

Tabla 4.27 Mejor modelo LSTM según ECM en CP tras el ajuste de hiperparámetros y comparación con el método manual.

Método - Fase	nn	$miniBatchSize$	$p_{Dropout}$	α	$pacienciaCP$	$\boldsymbol{EAM_{CP}}$	$\boldsymbol{ECM_{CP}}$	$\boldsymbol{MAAPE_{CP}}$	$\boldsymbol{SMAPE_{CP}}$
Manual - Ajuste de hiperparámetros	42	32	0.3	0.05	20	0.4707	0.6597	0.2304	0.1153
Automatizado - Ajuste de hiperparámetros	46	16	0.55	0.008	70	0.4783	0.6518	0.2439	0.1182

De nuevo, se observan resultados similares entre ambos métodos, con un mejor resultado en la metodología automatizada según ECM en CP por ser el criterio de ajuste.

Resumen y comparación de resultados

A continuación, se presenta en la *Tabla 4.28* un resumen y comparación de resultados entre los distintos métodos y fases de la metodología aplicada, indicando en negrita el menor error obtenido para cada métrica.

Tabla 4.28 Resumen de resultados en los diferentes métodos y fases de la metodología aplicada.

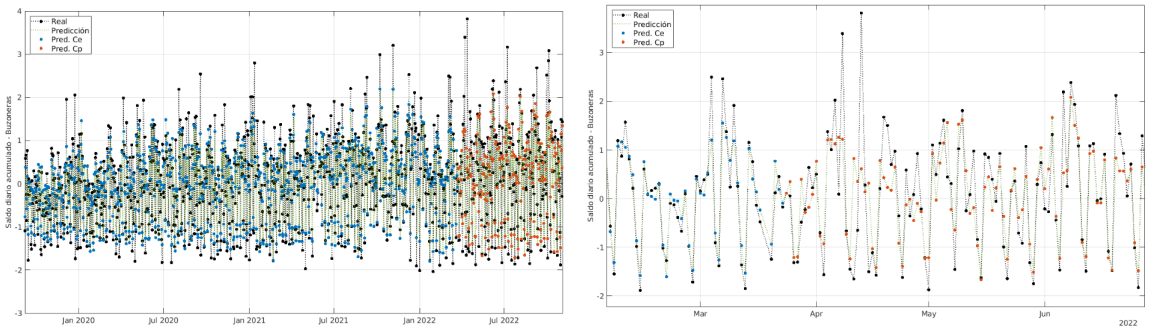
Método - Fase	EAM_{CE}	EAM_{CP}	ECM_{CE}	ECM_{CP}	$MAAPE_{CE}$	$MAAPE_{CP}$	$SMAPE_{CE}$	$SMAPE_{CP}$
Manual - Selección de entradas	0.3092	0.4586	0.4088	0.6654	0.1765	**0.2174**	0.0863	**0.1098**
Automatizado - Selección de entradas	0.2977	**0.4575**	**0.4000**	**0.6483**	0.1723	0.2284	0.0846	0.1141
Manual - Ajuste de hiperparámetros	0.3512	0.4707	0.4661	0.6597	0.1979	0.2304	0.0958	0.1153
Automatizado - Ajuste de hiperparámetros	**0.2918**	0.4743	0.4003	0.6518	**0.1643**	0.2270	**0.0809**	0.1145

Se comprueba que los errores obtenidos mediante ambos métodos (manual y automatizado) según las diferentes métricas son similares y de igual orden. En general, se obtienen menores errores mediante el método automatizado, excepto en las métricas porcentuales (MAAPE, SMAPE) en el conjunto de prueba (CP). Con esto, queda demostrada la efectividad y validez del método automatizado.

Además de esto, hay que destacar las ventajas que ofrece la automatización propuesta:

- Eficiencia de trabajo. La automatización del proceso de creación de modelos de predicción resulta en la ejecución en segundo plano de esta tarea, permitiendo la realización de otras tareas en paralelo.
- Consistencia en los resultados. La automatización de una metodología predefinida y la toma de decisiones reduce las probabilidades de error humano, dando lugar a una creación normalizada de modelos.
- Simplificación del proceso, permitiendo a usuarios poco experimentados realizar esta tarea.
- Facilidad de aplicación a diferentes series temporales una vez realizado el paso previo de preparación de los datos.

Por último, se representa en la *Figura 4.32* la salida predicha junto a la salida real del modelo que ha presentado el menor ECM en CP (método automatizado tras selección de entradas), además de un zoom en el punto de inflexión entre CE y CP para observar las predicciones realizadas por el modelo en ambos conjuntos, y en la *Figura 4.33* la salida predicha frente a la real.



(a) Serie temporal original . **(b)** Zoom en la transición CE → CP .

Figura 4.32 Salida real junto a salida predicha por el mejor modelo LSTM obtenido mediante el método automatizado.

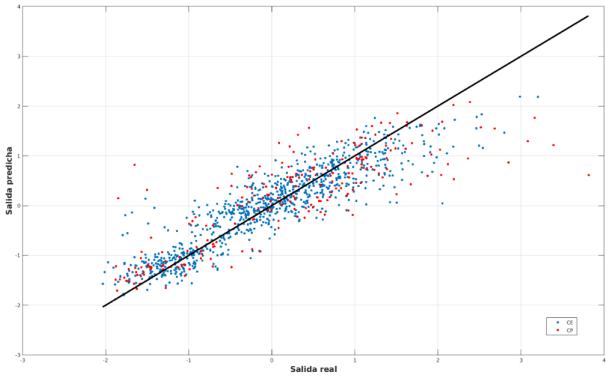


Figura 4.33 Salida real frente a salida predicha por el mejor modelo LSTM obtenido mediante el método automatizado.

4.4.2 Segunda prueba: Comprobación de la capacidad de generalización del método

Para finalizar, se realiza una última prueba de predicción automatizada con una serie temporal distinta, aunque pertenciente al mismo conjunto de datos, para comprobar que el método se puede aplicar a cualquier serie temporal.

Se comienza representando la serie temporal en la *Figura 4.34*.

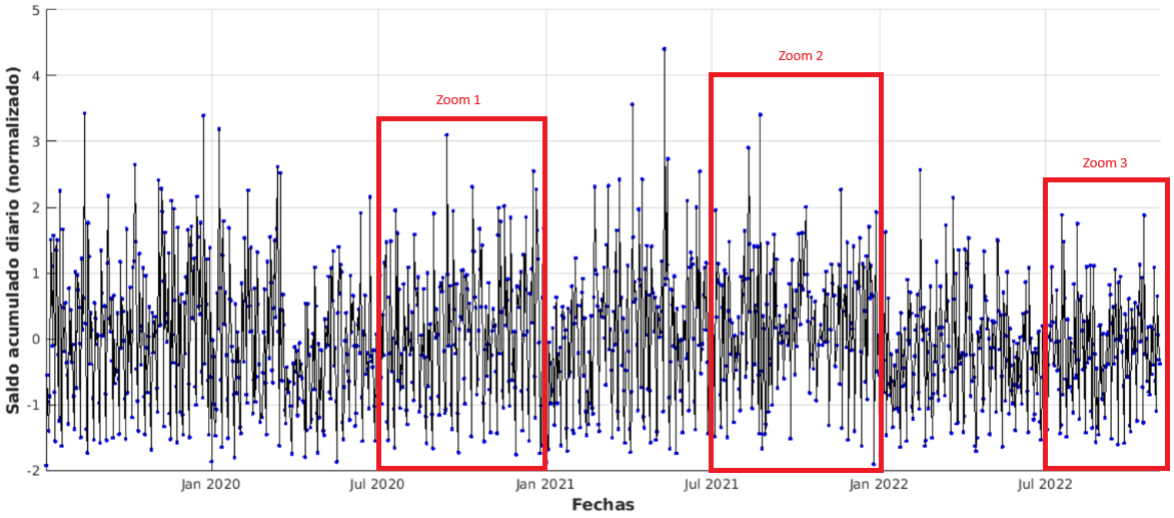
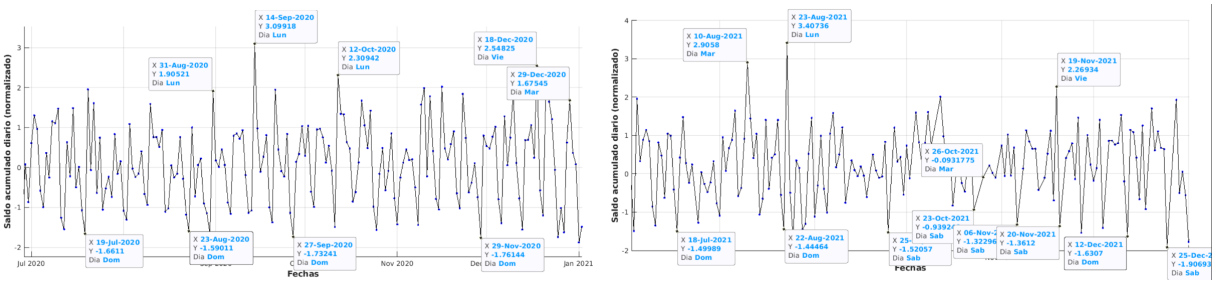


Figura 4.34 Saldo acumulado diario de las buzoneras de una sucursal concreta.

En las *Figuras 4.35, 4.36* se representan distintos zooms realizados sobre la serie temporal.



(a) Zoom 1 . **(b)** Zoom 2 .

Figura 4.35 Serie temporal en rangos de tiempo más reducidos.

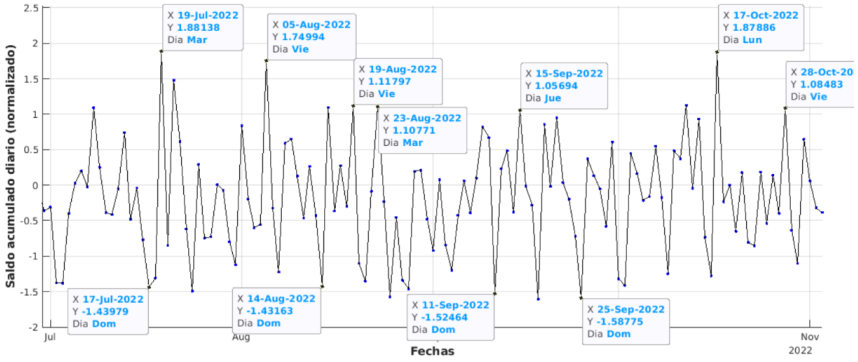


Figura 4.36 Serie temporal en un rango de tiempo más reducido y reciente.

Se obtienen conclusiones parecidas a las obtenidas con la serie temporal anterior, siendo (en general) los lunes los días de mayor saldo y los domingos los de menor saldo. También se encuentra la misma falta de datos (*Figura 4.35 (b)*), representada anteriormente en la *Figura 4.19 (a)*. A continuación, se realiza el análisis de tipo de día:

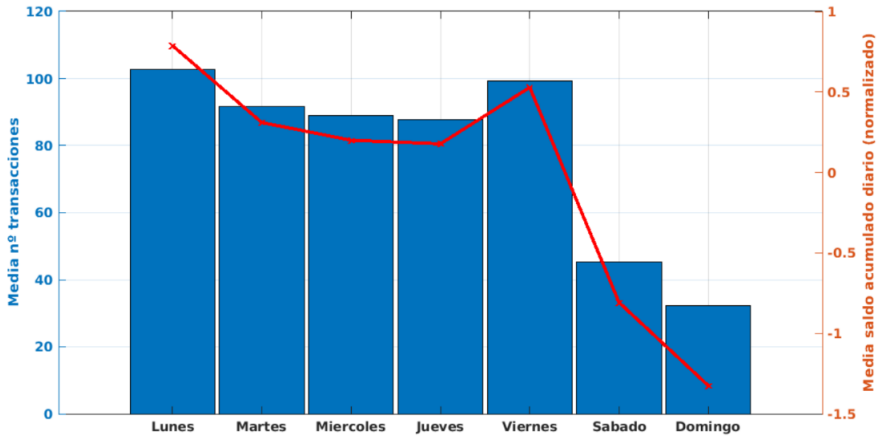


Figura 4.37 Número y saldo acumulado diario medios en los distintos días de la semana.

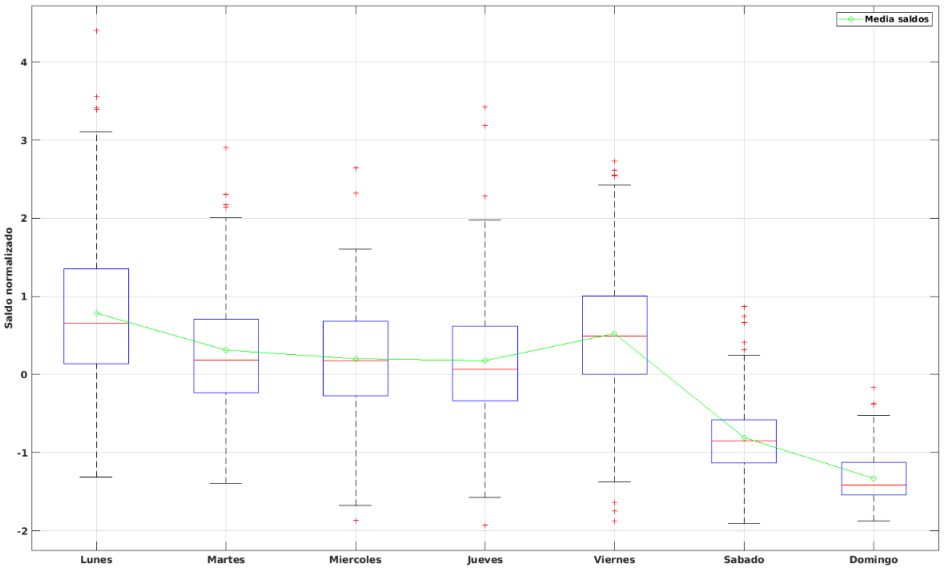


Figura 4.38 Diagramas de caja para el saldo acumulado diario de los diferentes días de la semana.

De nuevo, se obtienen conclusiones similares a las obtenidas en las *Figuras 4.20, 4.21*. Por tanto, se definen los mismos tipos de días ya definidos en la *Tabla 4.12*.

De igual forma, se utilizan las mismas entradas potenciales definidas en la *Tabla 4.13* y los mismos grupos de entradas (*Tabla 4.14*). Se pasa a mostrar los resultados obtenidos tras aplicar los algoritmos de creación automatizada de modelos.

Los argumentos de entrada utilizados son los siguientes:

- Número de entradas a filtrar: $N = 20$.
- Paciencia en la selección hacia adelante: $P = 5$ iteraciones.
- Combinación de hiperparámetros:
  - $nn = 50$
  - $\alpha = 0.05$
  - $p_{Dropout} = 0.2$
  - $miniBatchSize = 16$
  - $maxPasadas = 200$
  - $pacienciaCP = 50$

Selección de entradas

Tras aplicar la selección automatizada de entradas se obtiene el siguiente mejor conjunto de entradas según ECM en CP:

Tabla 4.29 Entradas seleccionadas de forma automática según ECM en CP.

Variable	Entradas
S	-56, -33, -21, -49, -42, -12, -2, -7, -37, -30, -9, -28, -40
S_{damt}	-54, -18, -17, -53, -55, -24
S_{mean7}	-37, -20, -9, -15, -25
S_{max7}	-19, -48, -56, -22

Para este conjunto de entradas y combinación de hiperparámetros inicial se obtienen los siguientes errores:

Tabla 4.30 Evaluación del mejor modelo tras selección de entradas.

$\boldsymbol{EAM}_{CE}$	$\boldsymbol{EAM}_{CP}$	$\boldsymbol{ECM}_{CE}$	$\boldsymbol{ECM}_{CP}$	$\boldsymbol{MAAPE}_{CE}$	$\boldsymbol{MAAPE}_{CP}$	$\boldsymbol{SMAPE}_{CE}$	$\boldsymbol{SMAPE}_{CP}$
0.5623	0.4306	0.7448	0.5593	0.3093	0.2604	0.1552	0.1307

Ajuste de hiperparámetros

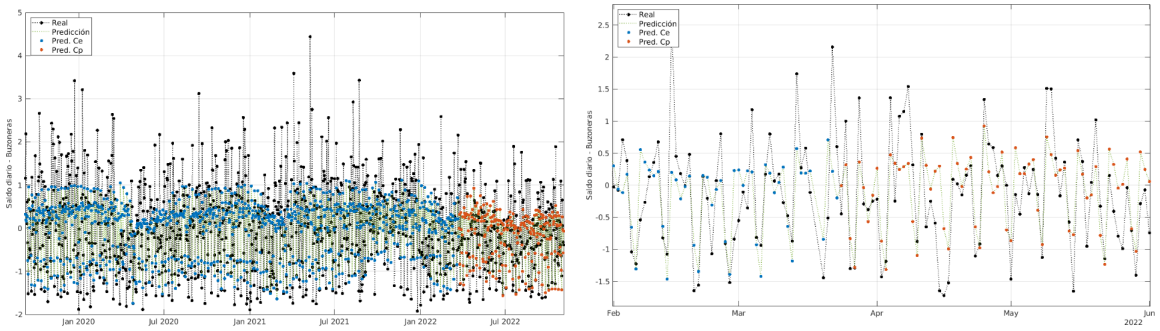
En la *Tabla 4.31* se muestra el mejor modelo obtenido tras la selección de entradas y el mejor modelo obtenido tras el ajuste de hiperparámetros.

Tabla 4.31 Mejor modelo LSTM según ECM en CP en cada fase del método automatizado.

Fase	nn	$miniBatchSize$	$p_{Dropout}$	α	$pacienciaCP$	$\boldsymbol{EAM}_{CP}$	$\boldsymbol{ECM}_{CP}$	$\boldsymbol{MAAPE}_{CP}$	$\boldsymbol{SMAPE}_{CP}$
Selección de entradas	50	16	0.2	0.05	50	0.4306	0.5593	0.2604	0.1307
Ajuste de hiperparámetros	52	32	0.2	0.03	70	0.4328	0.5667	0.2680	0.1327

Al igual que sucedía con la serie temporal anterior, no se consigue mejorar los resultados mediante la ajuste de hiperparámetros.

Finalmente, se muestran las gráficas de salida predicha y salida real para el mejor modelo obtenido.



(a) Serie temporal original . **(b)** Zoom en la transición CE → CP .

Figura 4.39 Salida predicha junto a salida real del mejor modelo LSTM obtenido por el método automatizado.

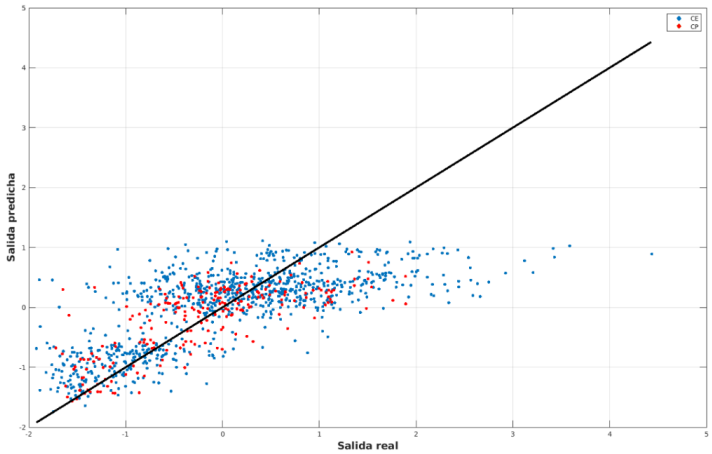


Figura 4.40 Salida predicha frente a salida real del mejor modelo LSTM obtenido por el método automatizado.

5 Conclusiones

Este trabajo se ha enfocado en la predicción de series temporales mediante modelos de predicción de diferentes características existentes en la literatura: ARX, RBF y LSTM, así como la automatización parcial del proceso, concretamente los pasos relativos a la creación de modelos: selección de entradas y ajuste de hiperparámetros, con el fin de facilitar y agilizar la obtención y optimización de modelos de predicción para diferentes series temporales.

Para el estudio se han utilizado dos conjuntos de datos de series temporales de diferente naturaleza que han permitido, por un lado, predecir el volumen de llamadas entrantes en un centro de atención al cliente y, por otro lado, predecir el saldo acumulado diario de transacciones realizadas en dispositivos de ingreso de efectivo de una gran entidad bancaria iberoamericana.

El primer paso ha sido la aplicación de una metodología manual con el objetivo de obtener unos primeros resultados de referencia y escoger un modelo como base para la automatización. Seguidamente, se han aplicado los algoritmos de automatización propuestos sobre dos series temporales del último conjunto de datos: primero, la serie temporal ya estudiada para comprobar que el método de automatización es válido y proporciona resultados útiles en la práctica y, segundo, una serie temporal diferente, aunque de idéntica naturaleza, para demostrar que el método se puede aplicar con facilidad a cualquier serie temporal una vez se han completado los pasos previos de preparación de los datos y elección del modelo.

En cuanto a la primera parte de este trabajo, se ha concluido que los modelos LSTM son superiores a los modelos lineales ARX y ligeramente superiores o similares a los modelos basados en redes neuronales RBF, por lo que se ha tomado el modelo LSTM como base para la automatización.

En la última parte del trabajo se ha demostrado la efectividad y validez del método automatizado propuesto, proporcionando resultados de predicción similares e incluso algo mejores que los obtenidos mediante la metodología manual. A esto hay que sumarle las ventajas que conlleva la automatización: mayor eficiencia de trabajo, consistencia en los resultados, simplificación del proceso, facilidad de aplicación a otras series temporales.

Bibliografía

[1] R.J. Hyndman and G. Athanasopoulos. *Forecasting: Principles and Practice*. 2 edition, 4 2018.

[2] Hooman H Rashidi, Nam K Tran, Elham Vali Betts, Lydia P Howell, and Ralph Green. Artificial intelligence and machine learning in pathology: The present landscape of supervised methods. *Academic pathology*, 6, 2019.

[3] Tom O'Haver. A pragmatic introduction to signal processing, 1993.

[4] Yanxia Yang, Pu Wang, and Xuejin Gao. A novel radial basis function neural network with high generalization performance for nonlinear process modelling. *Processes*, 10(1), 2022.

[5] Thorir Mar Ingolfsson. Insights into lstm architecture. Accedido el 9 de junio, 2023, desde *https://thorirmar.com/post/insight_into_lstm/*.

[6] Laerd Statistics. Pearson's product moment correlation. Statistical tutorials and software guides, 2020. Retrieved June 5, 2023, from *https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php*.

[7] Diagrama de caja y bigotes (boxplot). Accedido el 24 de junio, 2023, desde *https://www.probabilidadyestadistica.net/diagrama-de-caja-y-bigotes-boxplot/*.

[8] G. Udny Yule. On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 226:267–298, 1927.

[9] G. Walker. On periodicity in series of related terms. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 131:518–532, 6 1931.

[10] Stockholm University Department of Statistics. A brief history of time series analysis. Accedido el 17 de junio, 2023, desde *https://www.su.se/english/research/research-subjects/statistics/statistical-models-in-the-social-sciences/a-brief-history-of-time-series-analysis-1.612367*.

[11] H. L. H. L. S. A study in the analysis of stationary time series. by herman wold. [pp. 214 + viii. almqvist and wiksells boktryckeri-a.-b., uppsala. 1938. price kr. 6.]. *Journal of the Institute of Actuaries*, 70:113–115, 3 1939.

[12] Zoran Ivanovski, Ace Milenkovski, and Zoran Narasanov. Time series forecasting using a moving average model for extrapolation of number of tourist. pages 121–132, 01 2018.

[13] Jarosław Joostberens, Aurelia Rybak, Joachim Pielot, and Artur Dylong. Application of the arma model to describe and forecast the flotation feed solids flow rate. *Energies*, 14(24), 2021.

[14] George.E.P. Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1976.

[15] Paul Newbold. The principles of the box-jenkins approach, 1970.

[16] Charles C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 20(1):5–10, 2004.

[17] Peter R. Winters. Forecasting sales by exponentially weighted moving averages. *Management Science*, 6:324–342, 4 1960.

[18] Robert Goodell Brown. Statistical forecasting for inventory control. 1959.

[19] Ruey S. Tsay. Time series and forecasting: Brief history and future research. *Journal of the American Statistical Association*, 95(450):638–643, 2000.

[20] NCSS Statistical Software. Chapter 470 the box-jenkins method.

[21] Warren L. Young. The Box-Jenkins approach to time series analysis and forecasting : principles and applications. *RAIRO - Operations Research - Recherche Opérationnelle*, 11(2):129–143, 1977.

[22] J K Das. Business forecasting: Box-jenkins methodology. 01 2015.

[23] Farhan Mohammad Khan and Rajiv Gupta. Arima and nar based prediction model for time series analysis of covid-19 cases in india. *Journal of Safety Science and Resilience*, 1(1):12–18, 2020.

[24] Peng Chen, Aichen Niu, Duanyang Liu, Wei Jiang, and Bin Ma. Time series forecasting of temperatures using sarima: An example from nanjing. *IOP Conference Series: Materials Science and Engineering*, 394:052024, 08 2018.

[25] Nari Arunraj, Diane Ahrens, and Michael Fernandes. Application of sarimax model to forecast daily sales in food retail industry. *International Journal of Operations Research and Information Systems*, 7:1–21, 04 2016.

[26] Cathal Murray, Naomi Du Bois, Lynsey Hollywood, and Damien Coyle. State-of-the-art deep learning models are superior for time series forecasting and are applied optimally with iterative prediction methods of time series forecasting capabilities of eight such state-of-the-art.

[27] Christopher A. Sims. Macroeconomics and reality. *Econometrica*, 48(1):1–48, 1980.

[28] Vector autoregression. *International Encyclopedia of the Social Sciences*. Accedido el 18 de junio, 2023, desde *https://www.encyclopedia.com/social-sciences/applied-and-social-sciences-magazines/vector-autoregression*,.

[29] Robert F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica*, 50(4):987–1007, 1982.

[30] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986.

[31] Kevin Amate Vicente. Modelos arch y garch: Aplicación a series financieras, 6 2018.

[32] Ajay Dhamija and VK Bhalla. Financial time series forecasting : Comparison of various arch models. *Global Jounal of Finance and Management*, 2:159–172, 01 2010.

[33] Altaf Hossain. Comparison of garch and neural network methods in financial time series prediction. 12 2008.

[34] Houssainy El, Amal Mohamed, and Haitham Fawzy. Time series forecasting using tree based methods. *Journal of Statistics Applications & Probability*, 10:229, 03 2021.

[35] Evangelos Spiliotis. Decision Trees for Time-Series Forecasting. *Foresight: The International Journal of Applied Forecasting*, (64):30–44, Q1 2022.

[36] Nicholas I. Sapankevych and Ravi Sankar. Time series prediction using support vector machines: A survey. *IEEE Computational Intelligence Magazine*, 4(2):24–38, 2009.

[37] U Thissen, R Brakel, A.P Weijer, W.J Melssen, and Lutgarde Buydens. Using support vector machines for time series prediction. *Chemometrics and Intelligent Laboratory Systems*, 69:35–49, 11 2003.

[38] John Moody and Christian J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[39] Md. Shiblee, P. K. Kalra, and B. Chandra. Time series prediction with multilayer perceptron (mlp): A new generalized error based approach. In Mario Köppen, Nikola Kasabov, and George Coghill, editors, *Advances in Neuro-Information Processing*, pages 37–44, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[40] J. Park and I. W. Sandberg. Universal Approximation Using Radial-Basis-Function Networks. *Neural Computation*, 3(2):246–257, 06 1991.

[41] Mohammed Awad, Hector Pomares, Ignacio Ruiz, Osama Salameh, and Mai Hamdon. Prediction of time series using rbf neural networks: A new approach of clustering. *Int. Arab J. Inf. Technol.*, 6:138–143, 04 2009.

[42] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[43] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18:1527–54, 08 2006.

[44] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427, jan 2021.

[45] Dezdemona Gjylapi and Eljona Proko. Recurrent neural networks in time series prediction. 07 2017.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[47] Steven Elsworth and Stefan Güttel. Time series forecasting using lstm networks: A symbolic approach, 2020.

[48] Felix Gers, Douglas Eck, and Jürgen Schmidhuber. Applying lstm to time series predictable through time-window approaches. pages 669–676, 08 2001.

[49] Mohammad Masum, Hossain Shahriar, Hisham M. Haddad, and Md. Shafiul Alam. r-lstm: Time series forecasting for covid-19 confirmed cases with lstmbased framework. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1374–1379, 2020.

[50] Yupeng Wang, Shibing Zhu, and Changqing Li. Research on multistep time series prediction based on lstm. In *2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE)*, pages 1155–1159, 2019.

[51] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[52] Shiv Kumar Verma, Aman Gupta, and Ankita Jyoti. Stack layer & bidirectional layer long short - term memory (lstm) time series model with intermediate variable for weather prediction. In *2021 International Conference on Computational Performance Evaluation (ComPE)*, pages 065–070, 2021.

[53] Dymitr Ruta, Ling Cen, and Quang Hieu Vu. Deep bi-directional lstm networks for device workload forecasting. In *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, pages 115–118, 2020.

[54] Alex Graves. Generating sequences with recurrent neural networks, 2014.

[55] XiaoFeng Wang and Ying Zhang. Multi-step-ahead time series prediction method with stacking lstm neural network. In *2020 3rd International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 51–55, 2020.

[56] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[57] Komal Saini and Sandeep Sharma. Gated recurrent unit (gru) in rnn for traffic forecasting based on time-series data. In *2022 2nd International Conference on Innovative Sustainable Computational Technologies (CISCT)*, pages 1–4, 2022.

[58] Umesh Saini, Rajesh Kumar, Vipin Jain, and M.U Krishnajith. Univariant time series forecasting of agriculture load by using lstm and gru rnns. In *2020 IEEE Students Conference on Engineering & Systems (SCES)*, pages 1–6, 2020.

[59] Jinah Kim and Jinah Kim. Bilstm model based on multivariate time series data in multiple field for forecasting trading area. *Journal of Ambient Intelligence and Humanized Computing*, 07 2019.

[60] Changchun Cai, Yuan Tao, Tianqi Zhu, and Zhixiang Deng. Short-term load forecasting based on deep learning bidirectional lstm neural network. *Applied Sciences*, 11(17), 2021.

[61] Frank Xiao. Time series forecasting with stacked long short-term memory networks, 2020.

[62] Z. Zainuddin, E. A. P. Akhir, and M. H. Hasan. Predicting machine failure using recurrent neural network-gated recurrent unit (rnn-gru) through time series data. *Bulletin of Electrical Engineering and Informatics*, 10:870–878, 2021.

[63] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[64] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms, 2012.

[65] Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms, 2013.

[66] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[67] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Auto-sklearn 2.0: Hands-free automl via meta-learning, 2022.

[68] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning, 2017.

[69] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search, 2018.

[70] Emannual C. Ifeacor and Barrie W. Jervis. *Digital signal processing: a practical approach*. Pearson Education, 2nd edition, 2002.

[71] H. Chen and B.R. Bakshi. 3.12 - linear approaches for nonlinear modeling. In Steven D. Brown, Romá Tauler, and Beata Walczak, editors, *Comprehensive Chemometrics*, pages 453–462. Elsevier, Oxford, 2009.

[72] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[73] Sungil Kim and Heeyoung Kim. A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting*, 32:669–679, 07 2016.

[74] Manuel R. Arahal, Manuel Berenguel, Eduardo F. Camacho, and Fernando Pavón. Selección de variables en la predicción de llamadas en un centro de atención telefónica. *Revista Iberoamericana de Automática e Informática industrial*, 6(1):94–104, ene. 2009.

[75] Manuel R. Arahal, P. Fernando Pavón, and Eduardo F. Camacho. Models for incoming calls forecasting in a customer attention center. *IFAC Proceedings Volumes*, 36(16):205–210, 2003. 13th IFAC Symposium on System Identification (SYSID 2003), Rotterdam, The Netherlands, 27-29 August, 2003.