MÁSTER EN BIG DATA. TECNOLOGÍAS Y ANALÍTICA AVANZADA

TRABAJO FIN DE MÁSTER

Data Augmentation for Time Series

Autor: Juan Miguel Ramos Pugnaire

Director: Bernat Sopena Gilboy

Co-Director: Joaquín Gallego Moreno

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Data Augmentation for Time Series

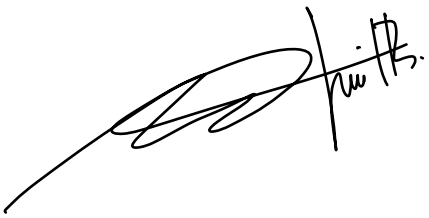
en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2022/23 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

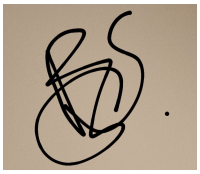
tomada de otros documentos está debidamente referenciada.



Fdo.: Juan Miguel Ramos Pugnaire Fecha: 20/ Junio/ 2023

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Bernat Sopena Gilboy Fecha: 20/ Junio/ 2023

Agradecimientos

Me gustaría agradecer a Bernat Sopena Gilboy y Joaquín Gallego Moreno, mis tutores en este proyecto, por la buena atención y ayuda recibida durante todo el desarrollo del trabajo.

Por otra parte, me gustaría agradecer a mi familia, en especial a mi abuelo, por ser un ejemplo para toda la familia. Allí donde estés, seguro que estás orgulloso de nosotros y nos estás cuidando.

DATA AUGMENTATION FOR TIME SERIES

Autor: Ramos Pugnaire, Juan Miguel.
Director: Sopena Gilboy, Bernat.
Entidad Colaboradora: Accenture, S.L.U.

RESUMEN

La aplicación de técnicas de Data Augmentation en series temporales ha ganado un creciente interés en el ámbito del forecasting. En este Trabajo de Fin de Máster (TFM), se explorará el uso de diferentes métodos de Data Augmentation para mejorar la calidad y el rendimiento de los modelos de predicción aplicados a series temporales.

El objetivo principal de este TFM es abordar el problema de la escasez de datos en series temporales, que a menudo dificulta la capacidad de los modelos de predicción para capturar patrones y realizar predicciones precisas. Data Augmentation, es decir, la generación de nuevos datos sintéticos a partir de los datos existentes se presenta como una estrategia prometedora para intentar solucionar este problema.

En este trabajo, se emplearán tanto métodos sencillos como métodos complejos de Data Augmentation. Entre los métodos sencillos, se incluirán técnicas como Time Warping, Add Noise, Pooling y Convolve, que han demostrado ser efectivas en la generación de datos aumentados para series temporales. Estas técnicas se basan en conceptos simples como la adición de ruido aleatorio o la convolución.

Además, se explorará el uso de un método más complejo, el Autoencoder, que se ha destacado en la literatura como una herramienta prometedora en Data Augmentation. El Autoencoder es una red neuronal que aprende a codificar y decodificar los datos de entrada, permitiendo así la generación de nuevos conjuntos de datos que conserven las características de los datos originales.

La elección de estos métodos se basa en el uso y estudio previo en la literatura. Sin embargo, el objetivo de este TFM no se limita únicamente a la aplicación de estos métodos, sino también en evaluar su efectividad y comparar su rendimiento utilizando distintos modelos de regresión, como el Random Forest y el XGBoost.

En resumen, este TFM tiene como objetivo principal la aplicación de métodos de Data Augmentation en series temporales, abarcando tanto métodos sencillos como complejos. Se busca mejorar la capacidad de predicción de los modelos de regresión utilizados en este contexto.

Palabras clave: Data Augmentation, Kaggle, M5 Forecasting, Tsaug, Time Warping, Add Noise, Pooling, Convolve, Autoencoder, Random Forest, XGBoost, Optuna, Backtesting.

ABSTRACT

The application of Data Augmentation techniques in time series has gained increasing interest in the field of forecasting. In this Master's Thesis, we will explore the use of different Data Augmentation methods to improve the quality and performance of prediction models applied to time series.

The main objective of this thesis is to address the issue of data scarcity in time series, which often hinders the ability of prediction models to capture patterns and make accurate predictions. Data Augmentation, i.e., the generation of synthetic data from existing data, emerges as a promising strategy to tackle this problem.

In this work, we will employ both simple and complex methods of Data Augmentation. Among the simple methods, techniques such as Time Warping, Add Noise, Pooling, and Convolve will be included, as they have demonstrated effectiveness in generating augmented data for time series. These techniques are based on simple concepts such as adding random noise or convolution.

Furthermore, we will explore the use of a more complex method, the Autoencoder, which has been highlighted in the literature as a promising tool in Data Augmentation. The Autoencoder is a neural network that learns to encode and decode input data, thus enabling the generation of new datasets that preserve the characteristics of the original data.

The selection of these methods is based on their usage and prior study in the literature. However, the objective of this TFM is not limited solely to the application of these methods, but also to evaluate their effectiveness and compare their performance using different regression models, such as Random Forest and XGBoost.

In summary, the main objective of this TFM is the application of Data Augmentation methods in time series, encompassing both simple and complex approaches. The aim is to enhance the prediction capacity of the regression models used in this context.

Keywords: Data Augmentation, Kaggle, M5 Forecasting, Tsaug, Time Warping, Add Noise, Pooling, Convolve, Autoencoder, Random Forest, XGBoost, Optuna, Backtesting.

Índice de la memoria

Capítulo 1. Introducción 4

1.1 Descripción del problema 4

1.2 Motivación 4

1.3 Objetivos del trabajo 5

Capítulo 2. Estado del Arte 6

Capítulo 3. Descripción de los Datos 8

Capítulo 4. Métodos Tradicionales de Data Augmentation 10

4.1 Time Warping ..... **Error! Bookmark not defined.**0

4.2 Add Noise ..... **Error! Bookmark not defined.**1

4.3 Pooling ..... **Error! Bookmark not defined.**2

4.4 Convolve ..... **Error! Bookmark not defined.**4

Capítulo 5. Método Avanzado de Data Augmentation 15

5.1 Autoencoder ..... **Error! Bookmark not defined.**5

*5.1.1 Arquitectura del Autoencoder ..... **Error! Bookmark not defined.**7*

*5.1.2 Elección de la Arquitectura ..... **Error! Bookmark not defined.**9*

Capítulo 6. Modelos de Evaluación 23

6.1 Random Forest 23

6.2 XGBoost ..... **Error! Bookmark not defined.**5

6.3 Implementación de los modelos ..... **Error! Bookmark not defined.**6

6.3.1 Optuna 27

6.3.2 Backtesting 29

Capítulo 7. Análisis de Resultados 311

Capítulo 8. Conclusiones y Trabajos Futuros 366

Capítulo 9. Bibliografía 38

Índice de figuras

Figura 1. Ejemplo: Time Warping [Elaboración Propia] 11
Figura 2. Ejemplo: Add Noise [Elaboración Propia] 12
Figura 3. Ejemplo: Pooling [Elaboración Propia] 13
Figura 4. Ejemplo: Pooling [Elaboración Propia] 14
Figura 5. Representación Autoencoder [6] 16
Figura 6. Ejemplo: Autoencoder [Elaboración Propia] 17
Figura 7. Ejemplo: Autoenc. Original [Elaboración Propia] 19
Figura 8. Ejemplo: Autoenc. R. Gaussiano=0.1 [Elaboración Propia] 20
Figura 9. Ejemplo: Autoenc. R. Gaussiano=0.2 [Elaboración Propia] 20
Figura 10. Ejemplo: Autoenc. R. Gaussiano=0.1+L2 [Elaboración Propia] 20
Figura 11. Ejemplo: Autoenc. R. Gaussiano=0.2+L2 [Elaboración Propia] 21
Figura 12. Ejemplo: Autoenc. R. Gaussiano=0.1+L2+DROP=0.1 [Elaboración Propia] ... 21
Figura 13. Ejemplo: Autoenc. R. Gaussiano=0.2+L2+DROP=0.1 [Elaboración Propia] ... 21
Figura 14. Diagrama Random Forest [7] 24
Figura 15. Diagrama XGBoost [8] 26
Figura 16. Diagrama Backtesting [9] 29

Índice de tablas

Tabla 1. Errores de Reconstrucción [Elaboración Propia] 22
Tabla 2. Resultados 20% [Elaboración Propia] 32
Tabla 3. Variaciones 20% [Elaboración Propia] 32
Tabla 4. Resultados 50% [Elaboración Propia] 33
Tabla 5. Variaciones 50% [Elaboración Propia] 34
Tabla 6. Resultados 100% [Elaboración Propia] 34
Tabla 7. Variaciones 100% [Elaboración Propia] 35

Capítulo 1. INTRODUCCIÓN

1.1 DESCRIPCIÓN DEL PROBLEMA

A lo largo de la historia, el número de productos lanzados diariamente ha aumentado significativamente en comparación con el pasado. Esta expansión del mercado ha dado lugar a la existencia de numerosos productos relativamente nuevos para los cuales se dispone de poca información histórica. Esta situación plantea el desafío de generar datos sintéticos que, combinados con los datos existentes, puedan ayudarnos a predecir la demanda de estos productos.

1.2 MOTIVACIÓN

En el actual contexto globalizado y tecnológicamente avanzado, se ha observado una transformación en prácticamente todos los sectores. Esta transformación ha generado la creación de una gran cantidad de productos, que a su vez se registran y digitalizan, lo que ha aumentado la disponibilidad de series temporales.

En este contexto, surge la técnica de Data Augmentation como una posible solución para mejorar tanto la calidad como la cantidad de las series temporales, con el objetivo de mejorar el entrenamiento de los modelos de predicción.

El problema anteriormente descrito, sumado al interés de un equipo especializado en forecasting, ha generado un entusiasmo por estudiar empíricamente cómo se comportarían determinados métodos en un escenario real.

1.3 OBJETIVOS DEL TRABAJO

El objetivo principal de este Trabajo Fin de Máster es investigar y evaluar el impacto de las técnicas de Data Augmentation en los modelos de predicción de series temporales. Para alcanzar este objetivo, se plantean los siguientes objetivos específicos:

1. Analizar la literatura existente sobre Data Augmentation en el contexto de las series temporales.
2. Seleccionar y adaptar los métodos existentes al caso de estudio. Se identificarán y aplicarán aquellos métodos que sean prometedores y a la vez sencillos de implementar.
3. Implementar los métodos de Data Augmentation seleccionados en un conjunto de datos reales.
4. Evaluar los métodos utilizados y analizar los resultados obtenidos en la aplicación de modelos tradicionales de regresión.

Se intentarán alcanzar estos objetivos, y con ello, se espera contribuir al conocimiento en el campo de las series temporales y el Data Augmentation dentro del departamento de la compañía.

Capítulo 2. ESTADO DEL ARTE

El objetivo principal de este trabajo consiste en investigar y evaluar diversos métodos de Data Augmentation en la aplicación de modelos tradicionales de regresión.

La literatura existente sobre el uso de Data Augmentation en el caso específico de series temporales es limitada. No obstante, proporciona una base sólida para abordar este estudio. Investigaciones previas han destacado el uso de métodos tradicionales como Jittering, Scaling o Magnitude Warping como soluciones simples para abordar la escasez de datos. Aunque estos métodos brindan resultados acordes a su nivel de complejidad, se propone el empleo de métodos más avanzados y complejos, como Autoencoders y redes GANs, como soluciones más precisas que ofrecen mejores resultados en la utilización de modelos de regresión [1].

En cuanto al rendimiento de los métodos tradicionales, en [2] se sugiere la utilización de técnicas como Time Warping, Add Noise, Pooling o Convolve para mejorar el rendimiento de un modelo complejo, como una red neuronal LSTM (Long Short Term Memory). Sin embargo, este artículo demuestra que estos métodos no generan mejoras sustanciales en el rendimiento del modelo, y en el mejor de los casos, solo se obtiene un incremento del 0,2% en el accuracy del modelo.

Por otro lado, en [1] se destaca el rendimiento de los métodos avanzados, donde se logra una reducción del MAPE (Mean Absolute Percentage Error) del 4,52% utilizando un Variational Autoencoder y un modelo de evaluación 2NN (red neuronal de 2 capas). Al utilizar modelos generativos, se alcanza una mejora del 4,36% en RMSE (Root Mean Square Error) mediante el uso de 2NN.

A pesar de las mejoras en el rendimiento obtenidas por los métodos anteriores en el contexto de modelos complejos, este estudio propone la utilización de métodos tradicionales como Time Warping, Add Noise, Pooling y Convolve, junto con un modelo complejo como un

Autoencoder. Estos métodos serán evaluados utilizando modelos de regresión sencillos que se emplean rutinariamente en el departamento de la compañía, como Random Forest y XGBoost.

Capítulo 3. DESCRIPCIÓN DE LOS DATOS

En este capítulo se proporcionará una descripción detallada de la fuente de datos y el conjunto de datos utilizados en el presente trabajo.

La fuente de datos empleada es este proyecto proviene de Kaggle, una plataforma muy conocida en el ámbito de la ciencia de datos. El conjunto de datos utilizado, referenciado como [3], es de uso abierto y es el que se utiliza en la competición M5 Forecasting, organizada en colaboración con la empresa de retail Walmart.

La competición M5 Forecasting tiene como objetivo predecir con precisión las ventas diarias de diversos productos en diferentes ubicaciones de las tiendas de Walmart en Estados Unidos, concretamente en los estados de Texas, California y Wisconsin. Este conjunto de datos contiene un extenso historial de ventas diarias, información sobre precios, datos relacionados con promociones, datos de calendario y características específicas de cada producto y tienda.

Las características que hemos utilizado en el proyecto son:

- Fechas (Date): día del calendario, p.e: 2017-01-29.
- Identificador de tienda (Store_id): Identificador único para cada tienda, por ejemplo, CA_1 significa que ese dato corresponde a la tienda 1 del estado de California.
- Identificador de producto (Item_id): Identificador único para cada producto, por ejemplo, HOBBIES_1_001 significa que ese dato pertenece a la categoría de hobbies, a la subcategoría 1 y es el producto número 001.
- Ventas (Sales): Número total de ventas para cada combinación día-tienda-producto.

Para llevar a cabo este trabajo, se han seleccionado series temporales de igual duración, lo cual facilita la aplicación de los distintos métodos de Data Augmentation y, posteriormente, su evaluación. En este contexto, se ha trabajado con un conjunto de 282 series temporales distintas, cada una con una duración de 275 períodos.

En cuanto al preprocesamiento de los datos, se ha utilizado un módulo interno desarrollado por el departamento, el cual es de carácter confidencial y ha sido utilizado conforme a los objetivos específicos de este proyecto.

Por último, es importante mencionar que los datos utilizados en este trabajo están agrupados por semana, y la granularidad de cada dato se especifica mediante la estructura "item_id-store_id-week".

En resumen, la fuente de datos utilizada es este trabajo proviene de Kaggle y corresponde al conjunto de datos de la competición M5 Forecasting de Walmart. Se han utilizado las características: fecha, identificador de tienda, identificador de producto y el número de ventas. Se han utilizado 282 series temporales de igual duración, con un total de 275 períodos cada una. El preprocesamiento de los datos se ha realizado mediante un módulo confidencial del departamento, y los datos se encuentran agrupados por semana con una granularidad definida por "item_id-store_id-week".

Capítulo 4. MÉTODOS TRADICIONALES DE D.A.

En este capítulo, se presentarán y describirán los métodos tradicionales utilizados en este proyecto. El objetivo principal al seleccionar estos métodos ha sido asegurar que no alteren la característica esencial de las series temporales, es decir, su dimensión temporal. En consecuencia, se han elegido aquellos métodos que preserven la duración original de las series temporales. Tal y como se ha discutido en profundidad en el Capítulo 2, los métodos que vamos a estudiar son: Time Warping, Add Noise, Pooling y Convolve.

Para la implementación de los métodos mencionados, se ha empleado una librería de código abierta llamada Tsaug [4]. Esta herramienta, desarrollada por ARUNDO, ha sido desarrollada para facilitar y potenciar la aplicación de técnicas de Data Augmentation en el ámbito de las series temporales.

La librería Tsaug se caracteriza por su flexibilidad permitiendo la manipulación y transformación de series temporales de manera eficiente. Además de las técnicas que vamos a utilizar, proporciona otro número diverso de técnicas como Cropping (recortar las series temporales) o Resize (redimensionar las series temporales mediante interpolación lineal).

La elección de Tsaug como herramienta principal para la implementación de los métodos tradicionales se basa en su capacidad para simplificar y agilizar el proceso de aplicación de los distintos métodos de Data Augmentation.

4.1 TIME WARPING

La técnica de Time Warping en la librería Tsaug, es una técnica de Data Augmentation que se aplica a series temporales para introducir variaciones en la velocidad de cambio a lo largo del tiempo.

Esta técnica se aplica de manera aleatoria, de forma que consiste en modificar de manera controlada la velocidad de la línea de tiempo de la serie temporal. Esta modificación se

realiza mediante cambios en la velocidad a lo largo de la serie, lo que implica acelerar o desacelerar la progresión de los puntos temporales.

Esta técnica se controla mediante dos parámetros principales. El primero es el número de cambios de velocidad, que determina cuántas veces se producirán cambios en la velocidad a lo largo de la serie temporal. Cuántos más cambios de velocidad se realicen, mayor será la variabilidad de cambio de la serie. El segundo parámetro es la relación entre la velocidad máxima y la velocidad mínima permitida. Este parámetro controla la amplitud de los cambios de velocidad, asegurando que la serie temporal no experimente cambios extremos o poco realistas.

Un aspecto destacado de esta técnica es que se mantiene la duración original de la serie temporal. Esto significa que, aunque se introduzcan cambios en la velocidad, la longitud total de la serie no se altera. Preservar la duración original es esencial en muchas aplicaciones de series temporales, ya que permite mantener la estructura temporal y garantizar que los datos se interpreten correctamente.

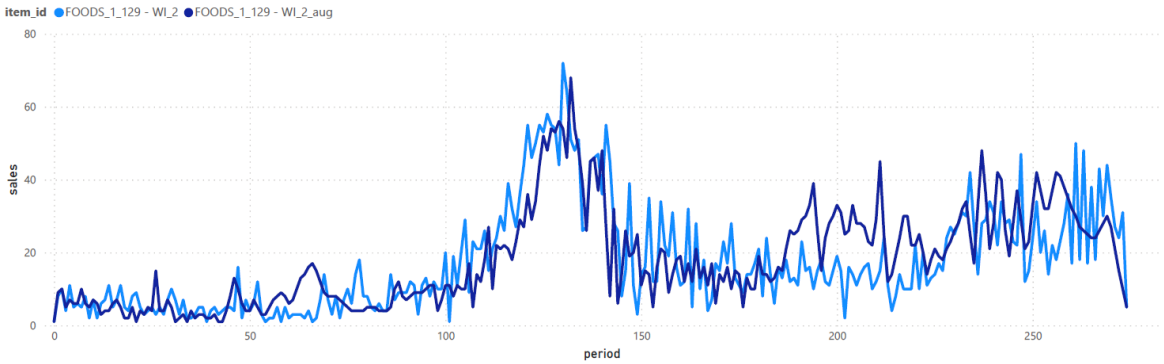


Figura 1. Ejemplo: Time Warping [Elaboración Propia]

4.2 ADD NOISE

La técnica de Add Noise de la librería Tsaug es una técnica de Data Augmentation que se basa en introducir ruido aleatorio en cada punto temporal de la serie.

En el contexto de Tsaug, el ruido aleatorio se aplica a cada punto de la serie temporal y se genera de forma independiente y se distribuye de manera idéntica en todos los puntos de la serie.

Al introducir ruido aleatorio, se introduce variabilidad en cada valor de la serie temporal. Este ruido aleatorio puede tener diferentes distribuciones: gaussiana, laplace o uniforme. En nuestro caso, hemos utilizado una distribución gaussiana, por tanto, tenemos que indicar la desviación estándar de la distribución.

Al igual que ocurre con la técnica de Time Warping, la técnica de Add Noise mantiene la duración original de la serie temporal. Esto significa que, aunque se agregue ruido a cada punto de la serie, la longitud total de la serie no se altera.

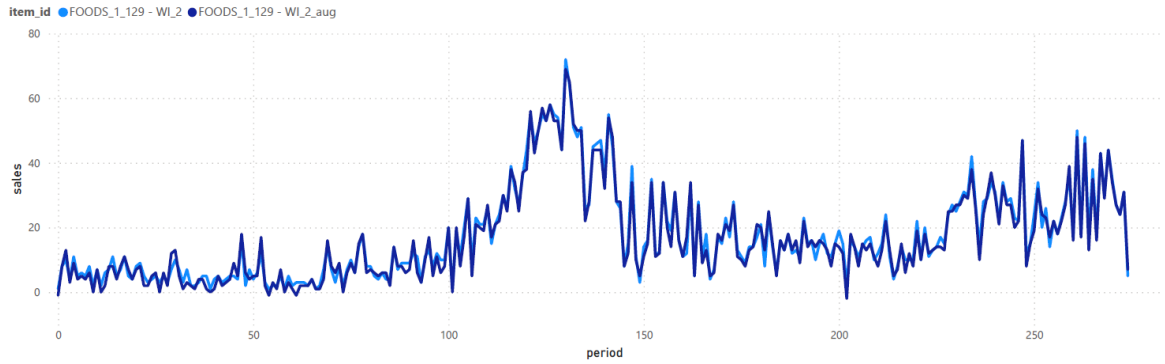


Figura 2. Ejemplo: Add Noise [Elaboración Propia]

4.3 POOLING

La técnica de Pooling de la librería Tsaug es una técnica de Data Augmentation que se basa en reducir la resolución temporal de la serie. Su objetivo principal es disminuir la frecuencia de muestreo de la serie sin alterar su longitud original.

En el contexto de Tsaug, el Pooling consiste en agrupar los puntos temporales de la serie en intervalos más grandes, lo que resulta en una reducción de la resolución temporal. Por ejemplo, si se aplica un factor de agrupación de 2, se tomarán cada 2 puntos temporales la métrica definida para obtener un único punto en el nuevo intervalo.

La técnica de Pooling es especialmente útil cuando se trabaja con series temporales con una gran cantidad de puntos y se desea reducir la complejidad y el ruido de los datos sin cambiar la duración total de la serie. Al reducir la resolución temporal, se logra una representación más generalizada de la serie, lo que puede ser beneficioso para el análisis y la predicción.

La técnica de Pooling de la librería de Tsaug ofrece la flexibilidad para ajustar el factor de agrupación que permite controlar la reducción de resolución temporal de acuerdo con las necesidades específicas del problema. Además, ofrece tres métricas distintas para aplicar en el intervalo de agrupación, estas son:

- Máximo (max): Consiste en elegir el valor máximo de dentro de cada intervalo. Esto implica que, en cada intervalo de agrupación, se toma el valor más alto y se descartan los demás.
- Mínimo (min): Consiste en elegir el valor mínimo de dentro de cada intervalo. Esto implica que, en cada intervalo de agrupación, se toma el valor más bajo y se descartan los demás.
- Media (ave): Consiste en elegir el valor medio de los puntos de dentro de cada intervalo. Esto implica que, en cada intervalo de agrupación, se toma el valor medio y se descartan los demás.

En nuestro caso, hemos utilizado la métrica de media en los intervalos de agrupación, ya que así se obtiene una representación más generalizada de la serie temporal.

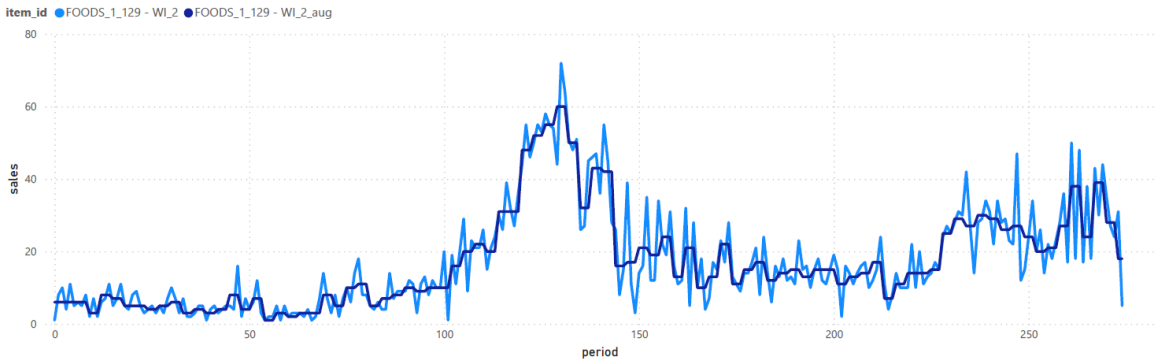


Figura 3. Ejemplo: Pooling [Elaboración Propia]

4.4 CONVOLVE

La técnica de Convolve de la librería Tsaug realiza la técnica de convolución de una serie temporal con una ventana de kernel. La convolución es una operación matemática que combina dos funciones para producir una tercera función que representa cómo una de las funciones influye en la otra a medida que se superponen. En el contexto de series temporales, la convolución se aplica para resaltar ciertas características o patrones presentes en los datos.

En el contexto de Tsaug, se puede especificar la longitud de las ventanas de convolución mediante el parámetro '*size'*. Esto controla el número de puntos de la serie que se incluyen en cada cálculo de convolución. Una ventana de longitud más grande implicará un mayor rango de influencia en la serie temporal, mientras que una ventana de longitud menor se centrará en regiones más locales.

La técnica de Convolve de la librería de Tsaug ofrece la flexibilidad para elegir el tipo de ventana que se va a aplicar en la convolución, las ventanas que ofrece son las que ofrece SciPy [5]. En particular, se han utilizado las ventanas '*hann'* y '*hamming'*. Estas ventanas son funciones matemáticas que asignan pesos a los puntos dentro de la ventana, con el fin de dar más influencia a los puntos centrales y menos a los puntos pegados a los bordes.

Al igual que en las anteriores técnicas, esta técnica también mantiene la duración original de la serie temporal. Esto significa que el número total de puntos de la serie no cambia, pero los valores de los puntos pueden ser alterados por la convolución.

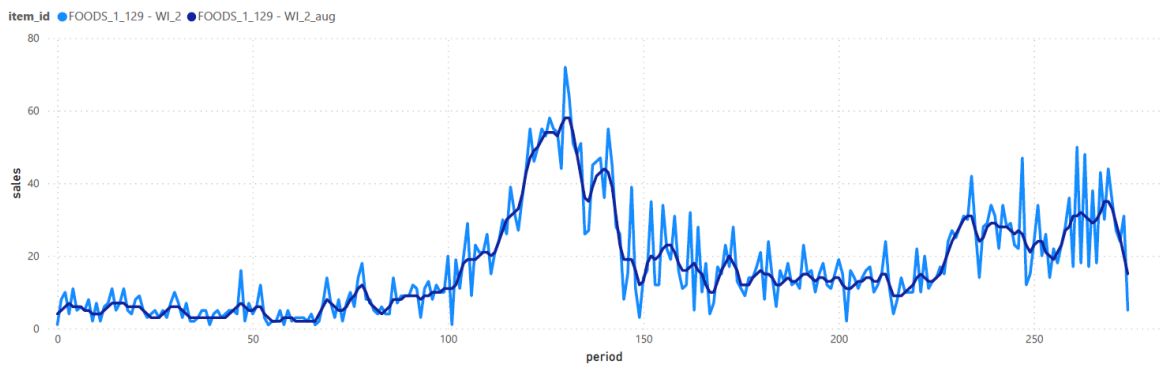


Figura 4. Ejemplo: Convolve [Elaboración Propia]

Capítulo 5. MÉTODO AVANZADO DE D.A.

En este capítulo se presentará y describirá el método avanzado de Data Augmentation utilizado en este trabajo.

Como hemos visto en el Capítulo 2, la literatura destaca que los modelos complejos más eficaces para este problema se encuentran relacionados con los autoencoders y modelos generativos. En nuestro caso, nos centraremos en estudiar el funcionamiento de un autoencoder aplicado al problema específico de Data Augmentation en series temporales. La elección de este modelo se fundamenta en su capacidad de proporcionar resultados comparables a los modelos generativos, a la vez que se caracteriza por ser más sencillo y de aplicación más sencilla.

5.1 AUTOENCODER

Un autoencoder es una arquitectura de red neuronal que se utiliza comúnmente en el aprendizaje no supervisado y la compresión de datos. Este tipo de red neuronal nace con el mismo objetivo que tienen las Componentes Principales (PCAs), cuya función principal es reducir la dimensionalidad de un conjunto de datos, conservando al mismo tiempo la mayor cantidad de información posible. Si el objetivo de las PCAs fuera el de comprimir al máximo la información y después volver a reconstruir la información utilizando esa información comprimida, estaríamos hablando del mismo concepto que aplican este tipo de redes neuronales.

Un autoencoder está diseñado para que no sea capaz de replicar a la perfección los datos de entrada. Un "buen" autoencoder es aquel que es capaz de codificar mejor la información, por eso es importante que la capa oculta tenga un número inferior de neuronas a las variables de entrada, ya que, si no, se produciría la función lineal, es decir, no hacer nada.

En este contexto, el objetivo principal de un autoencoder es aprender una representación compacta y significativa de los datos entrada, que luego se utiliza para reconstruir la entrada original lo más precisamente posible. Consiste en dos componentes principales: el codificador y el decodificador.

El codificador toma una muestra de datos de entrada y lo transforma en una representación de menor dimensión llamada "espacio latente". Este proceso de codificación implica la aplicación de varias capas de neuronas que reducen gradualmente la dimensión de los datos. A medida que la información se comprime en capas sucesivas, las características más relevantes y distintivas del conjunto de datos se preservan en el espacio latente.

Una vez que los datos se han codificado en el espacio latente, el decodificador realiza el proceso inverso. Recibe la codificación del conjunto de datos de entrada y lo transforma en una reconstrucción de la entrada original. El objetivo del decodificador es generar una salida lo más similar posible a la entrada original, utilizando capas de neuronas que aumentan gradualmente la dimensión hasta alcanzar el tamaño original del conjunto de datos de entrada.

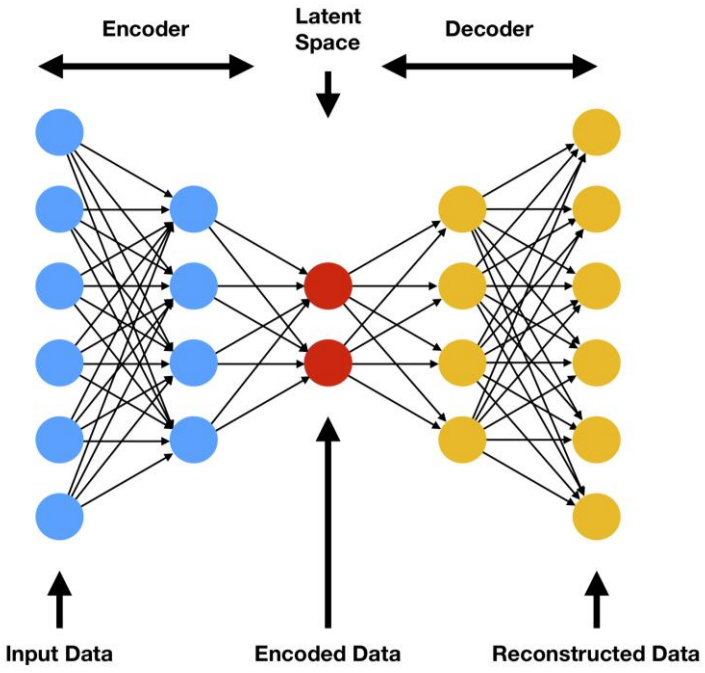


Figura 5. Representación Autoencoder [6]

En el contexto de Data Augmentation para series temporales, el autoencoder se utiliza como una técnica avanzada para generar nuevas series de manera sintética que se asemejen a las series originales, y así, aumentar el conjunto de datos original.

El proceso de generación de muestras sintéticas implicar tomar todas y cada una de las series que componen el conjunto de datos original, pasarlas a través del codificador para obtener su codificación en el espacio latente y luego, decodificar esta representación para las muestras sintéticas. Estas muestras sintéticas se añaden al conjunto de datos original para así, aumentar dicho conjunto de datos y obtener el conjunto de datos aumentado.

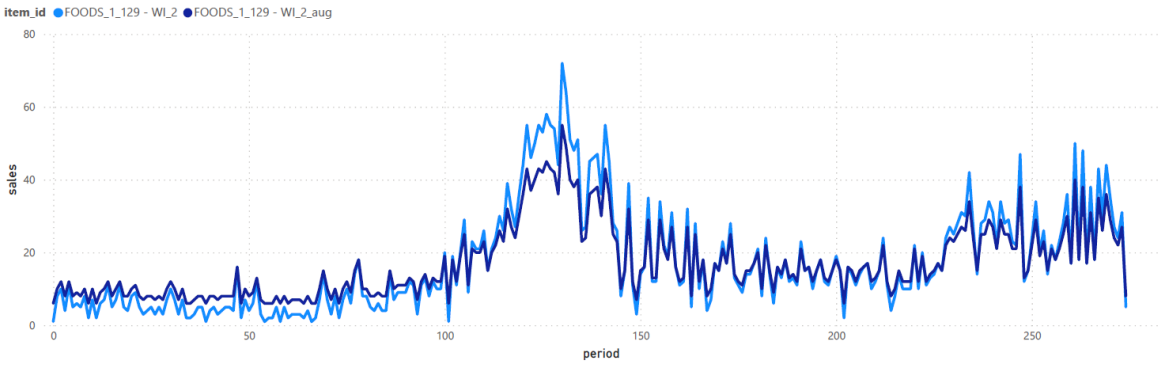


Figura 6. Ejemplo: Autoencoder [Elaboración Propia]

5.1.1 ARQUITECTURA DEL AUTOENCODER

En primer lugar, se ha realizado una reducción en la dimensión de las series temporales como entrada al modelo, originalmente teníamos 282 series temporales, cada una con una duración de 275 períodos. Sin embargo, se ha limitado la duración de las series a 11 períodos, lo que implica que ahora se tienen 25 "mini" series por cada serie temporal original. Por lo tanto, la dimensión del conjunto de datos de entrada del modelo ahora es 7050 series con una duración de 11 períodos.

Dado que el modelo requiere una entrada en forma de tensor de 3 dimensiones, finalmente se tiene un conjunto de datos de entrada con una dimensión de (7050,11,1). Esta representación en forma de tensor refleja la estructura tridimensional de los datos, donde cada elemento del tensor corresponde a un valor en una posición específica.

En cuanto a la elección de la arquitectura del autoencoder, se ha realizado un análisis exhaustivo de diferentes tipos de arquitecturas. Todas las arquitecturas consideradas tenían un base común, que consistía en:

- Codificador:
  - 1ª Capa: Dense de 11 neuronas con función de activación ReLu.
  - 2ª Capa: Dense de 8 neuronas con función de activación ReLu.
- Espacio latente (codificación):
  - 1ª Capa: Dense de 5 neuronas con función de activación ReLu.
- Decodificador:
  - 1ª Capa: Dense de 8 neuronas con función de activación ReLu.
  - 2ª Capa: Dense de 11 neuronas con función de activación ReLu.

En resumen, la arquitectura del autoencoder optimizada utiliza capas densas (totalmente conectadas) con función de activación ReLu. Esta función de activación activa una neurona si su entrada es mayor que cero y la desactiva si es menor o igual a cero. El codificador reduce gradualmente la dimensión del conjunto de series temporales de entrada hasta llegar al espacio latente, mientras que el decodificador aumenta la dimensión codificada hasta alcanzar el tamaño original de los datos.

Con relación a las diferentes modificaciones de la base del autoencoder, se ha llevado a cabo un análisis exhaustivo considerando las siguientes configuraciones:

1. Original: Se utilizó la base del autoencoder sin realizar ninguna modificación adicional.
2. Ruido Gaussiano = 0.1: Para dificultar la decodificación, se incluyó una capa de ruido gaussiano después de la codificación con una desviación estándar de 0.1.
3. Ruido Gaussiano = 0.2: Similar al escenario anterior, se agregó una capa de ruido gaussiano tras la codificación, pero con una desviación estándar de 0.2.
4. Ruido Gaussiano = 0.1 + L2: En este caso, se mantuvo la desviación estándar del ruido gaussiano en 0.1 y se agregó regularización L2 en cada una de las capas densas. Esta regularización ayuda a controlar la complejidad del modelo y a prevenir el

sobreajuste al penalizar los pesos grandes, lo cual conduce a soluciones más suaves y generalizadas.

5. Ruido Gaussiano = 0.2 + L2: Similar al escenario anterior, pero aumentando la desviación estándar del ruido gaussiano a 0.2.
6. Ruido Gaussiano = 0.1 + L2 + DROPOUT = 0.1: Se mantuvo la regularización L2 y la desviación estándar del ruido gaussiano en 0.1, y se agregó una capa de dropout de 0.1 tras las capas del codificador y del decodificador. El dropout es una técnica que apaga selectivamente un porcentaje de las neuronas durante el entrenamiento, lo cual limita la capacidad de aprendizaje del autoencoder.
7. Ruido Gaussiano = 0.2 + L2 + DROPOUT = 0.1: Similar al escenario anterior, pero aumentado la desviación estándar del ruido gaussiano a 0.2.

5.1.2 ELECCIÓN DE LA ARQUITECTURA

Para determinar la arquitectura final del autoencoder, se analizará el error de reconstrucción de cada modelo utilizando el MAE (Mean Absolute Error) como métrica. Además, se representarán las series sintéticas sobre las originales, lo que permitirá evaluar visualmente la calidad de la reconstrucción. Este análisis detallado nos ayudará a seleccionar la configuración óptima para el autoencoder y, así, obtener resultados confiables y precisos en el proceso de Data Augmentation para las series temporales en cuestión.

- Representación de las series:

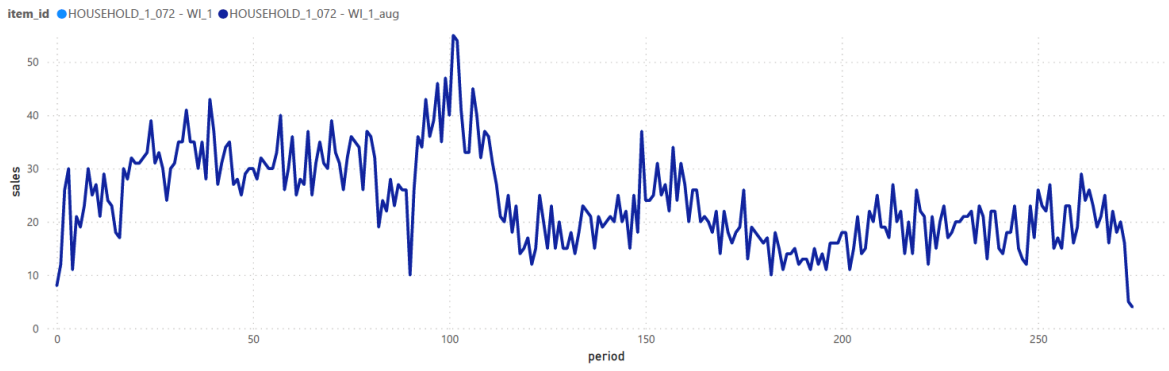


Figura 7. Ejemplo: Autoenc. Original [Elaboración Propia]

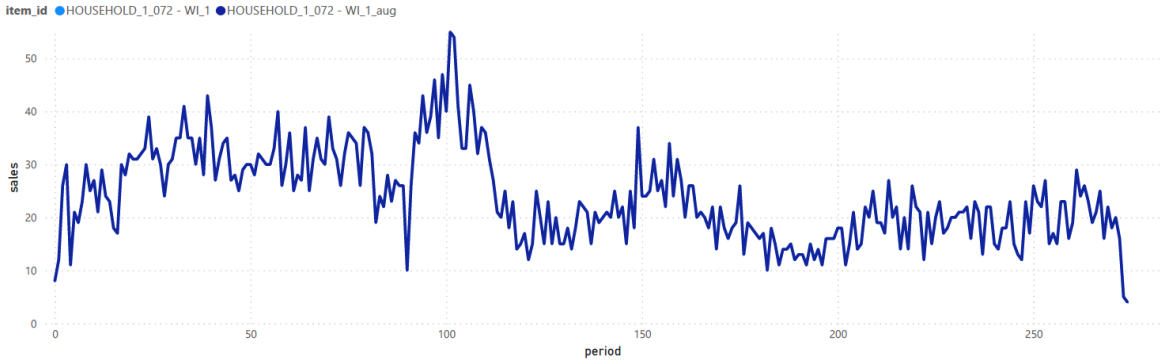


Figura 8. Ejemplo: Autoenc. R. Gaussiano=0.1 [Elaboración Propia]

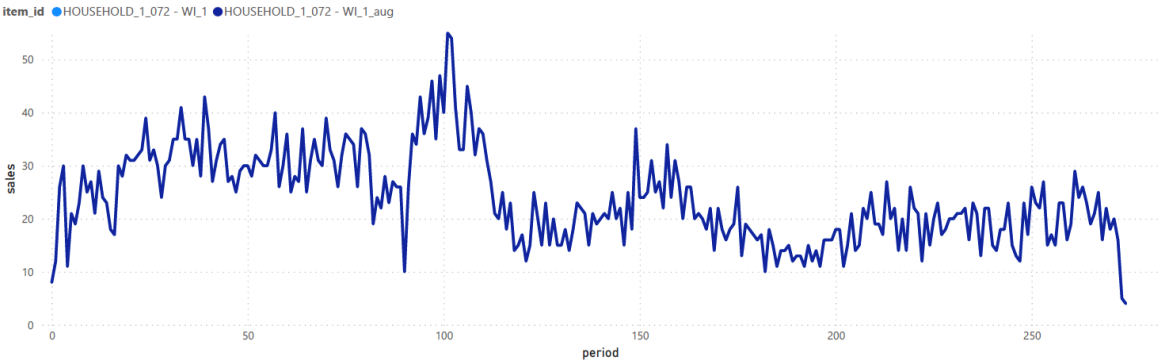


Figura 9. Ejemplo: Autoenc. R. Gaussiano=0.2 [Elaboración Propia]

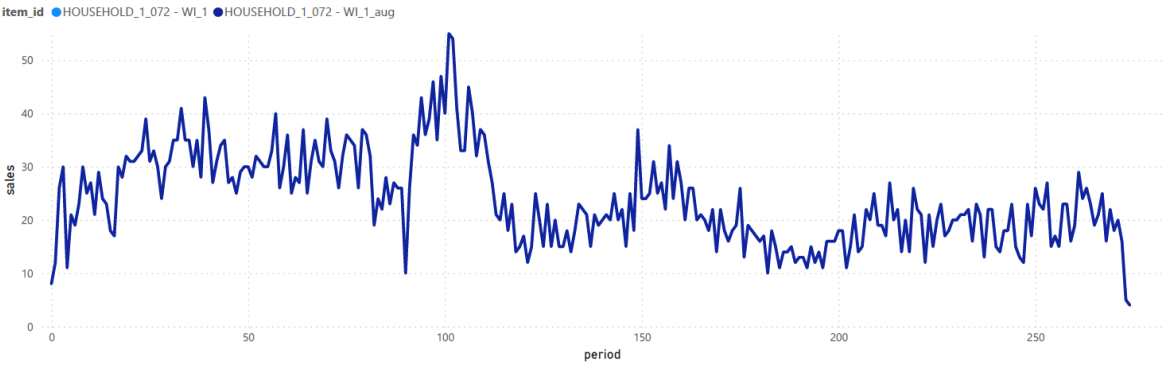


Figura 10. Ejemplo: Autoenc. R. Gaussiano=0.1+L2 [Elaboración Propia]

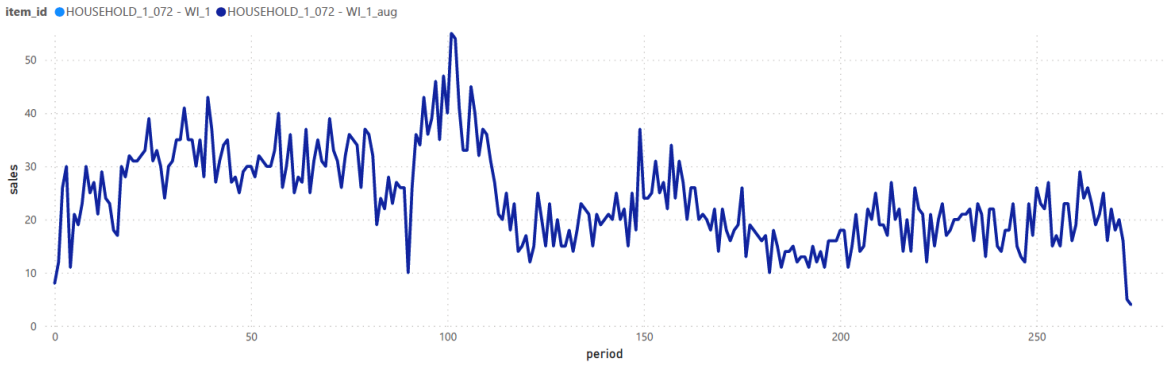


Figura 11. Ejemplo: Autoenc. R. Gaussiano=0.2+L2 [Elaboración Propia]

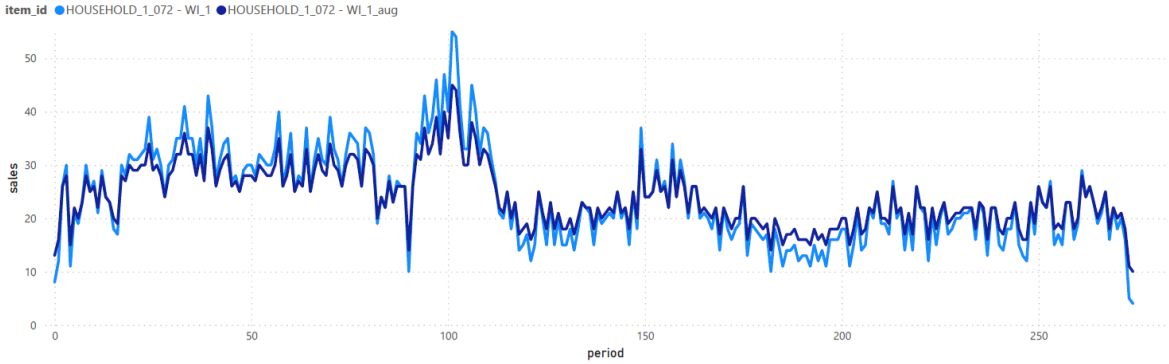


Figura 12. Ejemplo: Autoenc. R. Gaussiano=0.1+L2+DROP=0.1 [Elaboración Propia]

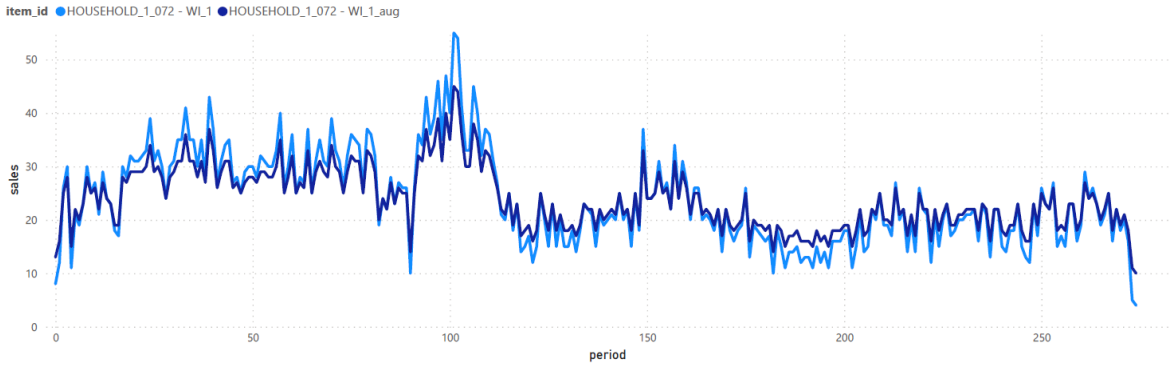


Figura 13. Ejemplo: Autoenc. R. Gaussiano=0.2+L2+DROP=0.1 [Elaboración Propia]

- Errores de reconstrucción:

Re. Err	***Original***	***R.G.=0.1***	***R.G.=0.2***	***R.G.=0.1+L2***
MAE	0.01	0.04	0.11	0.09

	R.G.=0.2+L2	***R.G.=0.1+L2+DROP=0.1***	***R.G.=0.2+L2+DROP=0.2***
MAE	0.04	5.35	5.42

Tabla 1. Errores de Reconstrucción [Elaboración Propia]

Al analizar las representaciones de las series temporales generadas sintéticamente por cada autoencoder, se observa que las configuraciones que incluyen dropout son las que presentan mayores dificultades de aprendizaje para el autoencoder. Esto se refleja también en los errores de reconstrucción, donde la inclusión del dropout contribuye a un mayor error.

En nuestro caso, buscamos que el error no sea demasiado pequeño para que las series generadas sean distintas a las originales, lo que nos permite aumentar el conjunto de datos con series temporales diferentes.

Por lo tanto, considerando las representaciones de las series temporales generadas sintéticamente y sus respectivos errores de reconstrucción, hemos seleccionado la configuración que incluye ruido gaussiano con una desviación estándar de 0.1, junto con regularización L2 y dropout, apagando un 10% de las neuronas en la capa correspondiente.

Capítulo 6. MODELOS DE EVALUACIÓN

En este capítulo se presentarán y describirán los modelos de predicción utilizados para evaluar el rendimiento de los distintos métodos de Data Augmentation implementados.

Como hemos visto en el Capítulo 2, la literatura suele hacer uso de modelos complejos para evaluar el rendimiento de los métodos de Data Augmentation. No obstante, en nuestro trabajo hemos optado por utilizar modelos de regresión ampliamente conocidos y utilizados en el departamento de la compañía, además de estar ampliamente reconocidos en el ámbito de la Ciencia de Datos. Estos modelos, en particular, son el Random Forest y el XGBoost.

Ambos modelos, el Random Forest y el XGBoost, son globales en el sentido de que se entrenan utilizando todas las series temporales juntas, en lugar de entrenarse individualmente serie por serie. Esto significa que los modelos pueden capturar patrones y relaciones más complejas que podrían pasar desapercibidos al entrenarlos de forma individual. Además, al tener un conjunto de entrenamiento más grande, es más probable que los modelos capturen la variabilidad de los datos de manera más efectiva.

6.1 RANDOM FOREST

El Random Forest es un algoritmo de aprendizaje supervisado utilizado para problemas de regresión y clasificación. Es un modelo ensemble que combina la predicción de varios árboles de decisión para obtener una predicción final robusta y precisa.

En el contexto de regresión, el objetivo del Random Forest es predecir un valor numérico. Para lograrlo, se construye un número de árboles de decisión, cada uno de los cuales se entrena utilizando una parte aleatoria del conjunto de datos de entrenamiento.

El algoritmo del Random Forest utiliza la técnica de Bagging (Boostrap Aggregation) para reducir la varianza del modelo. La técnica de Bagging implica crear múltiples muestras, del mismo tamaño que la muestra original, mediante el muestreo con reemplazamiento de la

muestra original. Cada árbol de decisión se entrena con una de estas muestras y luego se promedian las predicciones de todos los árboles para obtener una predicción final.

Además de Bagging, el Random Forest introduce "regularización" al limitar el número de variables que cada árbol puede considerar al tomar decisiones en cada nodo. Esta limitación se conoce como "muestreo de características" (feature sampling). Al restringir las variables disponibles en cada árbol, se evita que todas las variables importantes sean seleccionadas por todos los árboles, lo que ayuda a reducir la correlación entre los árboles y mejorar la capacidad de generalización del modelo.

Al combinar los resultados de múltiples árboles de decisión entrenados con diferentes muestras y limitaciones de variables, el Random Forest puede capturar una mayor variedad de patrones y reducir el efecto del ruido en los datos. Esto permite obtener predicciones más precisas y robustas en problemas de regresión.

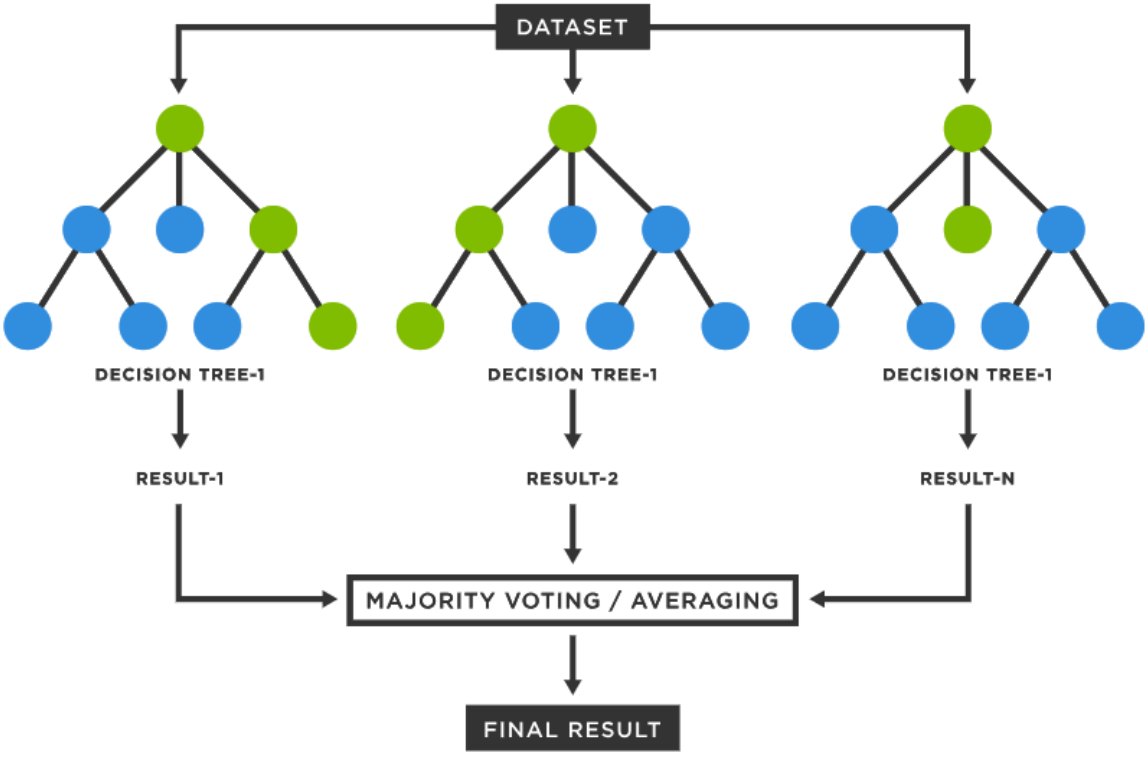


Figura 14. Diagrama Random Forest [7]

6.2 XGBOOST

El XGBoost (Extreme Gradient Boosting) es un algoritmo de aprendizaje supervisado utilizado para problemas de regresión y clasificación. Es una implementación mejorada del algoritmo de Gradient Boosting, que combina múltiples modelos débiles en un modelo más fuerte.

En el contexto de regresión, el objetivo del XGBoost es predecir un valor numérico. El proceso de construcción del modelo implica la generación de un conjunto de árboles de regresión simples. Cada árbol se entrena utilizando el conjunto de datos original y se ajusta con respecto a los residuos del árbol anterior.

El algoritmo del XGBoost utiliza la técnica de Boosting, que consiste en combinar las salidas de los modelos simples para obtener un modelo final más robusto y preciso. En cada iteración del proceso, se ajusta un nuevo árbol simple para reducir el error residual del modelo anterior.

Una característica distintiva del XGBoost es su capacidad para controlar la complejidad del modelo y evitar el sobreajuste mediante la incorporación de términos de regularización en la función de pérdida. Estos términos de regularización ayudan a penalizar los modelos con alta complejidad y favorecen la generalización del modelo.

Además, el XGBoost utiliza un enfoque de pruning (poda) durante la construcción de los árboles simples. Esta técnica consiste en eliminar las ramas del árbol que no contribuyen significativamente a la mejora de la predicción, lo que ayuda a reducir la complejidad del modelo y mejorar su eficiencia.

El proceso de entrenamiento del XGBoost se realiza mediante un algoritmo de optimización eficiente que busca encontrar los mejores valores para los parámetros del modelo. Estos parámetros incluyen la tasa de aprendizaje, que controla la contribución de cada árbol al modelo final, y la profundidad máxima de los árboles, que limita la complejidad del modelo.

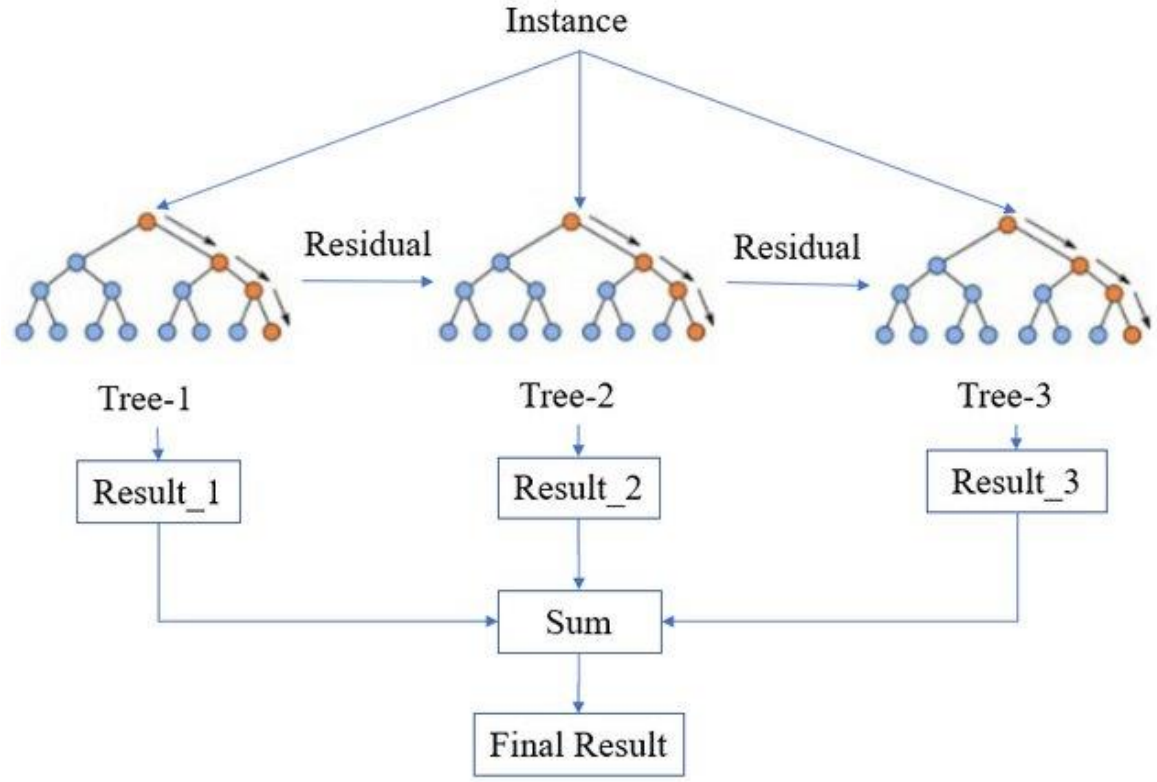


Figura 15. Diagrama XGBoost [8]

6.3 IMPLEMENTACIÓN DE LOS MODELOS

Los modelos descritos anteriormente se han aplicado a cada uno de los métodos de Data Augmentation descritos en los capítulos previos. Cada uno de los modelos se ha optimizado con el conjunto de datos original y después se ha evaluado comparando los resultados obtenidos utilizando el conjunto de datos original con los resultados obtenidos utilizando cada conjunto de datos aumentado.

En primer lugar, tanto para el entrenamiento como para la predicción de los modelos, se ha utilizado un conjunto de datos enriquecido con 30 lags. Esto implica agregar 30 pasos de tiempo anteriores como características adicionales en cada instancia de los datos. Esta inclusión de lags tiene el propósito de capturar y utilizar la información histórica de las series temporales para mejorar la capacidad predictiva de los modelos. Al proporcionar a los

modelos una mayor perspectiva temporal, se espera que sean capaces de capturar patrones a largo plazo y tendencias en los datos, lo que puede resultar en una mejor predicción de los valores futuros.

El uso de múltiples "lags" permite a los modelos capturar la dependencia temporal de los datos y tener en cuenta la evolución de las variables a lo largo del tiempo. Cada lag representa una observación pasada en el tiempo, lo que permite al modelo tener en cuenta la información histórica relevante para la predicción actual. Al incluir múltiples lags, se proporciona al modelo una mayor capacidad para capturar la dinámica temporal y las interacciones entre las variables a lo largo del tiempo.

6.3.1 OPTUNA

La optimización de parámetros es un proceso clave en el desarrollo de modelos de aprendizaje automático, ya que permite encontrar la combinación óptima de valores para los parámetros del modelo.

Optuna es un framework de optimización de hiperparámetros. Su enfoque se basa en algoritmos de búsqueda en árboles y métodos de estimación basados en modelos probabilísticos.

El proceso de optimización de parámetros mediante Optuna se lleva a cabo en varias etapas. En primer lugar, se define una función objetivo que evalúa el rendimiento del modelo para una determinada combinación de parámetros, en nuestro caso hemos utilizado el MAE (Mean Absolute Error).

A continuación, se definen los parámetros del modelo que se desean optimizar y se establecen los rangos de valores posibles para cada uno de ellos.

Optuna genera de forma automática muestras dentro de los rangos especificados para explorar diferentes combinaciones de parámetros.

Durante el proceso de optimización, Optuna ajusta dinámicamente las propuestas de parámetros en función de las iteraciones anteriores. Esto permite que el algoritmo de

búsqueda se enfoque en las regiones más prometedoras del espacio de búsqueda y descarte las combinaciones menos prometedoras. A medida que avanza el proceso, Optuna converge hacia una configuración óptima de los parámetros del modelo.

Una vez finalizado el proceso de optimización, se obtiene la mejor combinación de parámetros encontrada por Optuna. Esta combinación se utiliza para entrenar el modelo final utilizando el conjunto de datos completo de entrenamiento. Posteriormente, se evalúa el rendimiento del modelo final utilizando un conjunto de datos de prueba o validación para obtener una estimación realista de su capacidad predictiva.

En el contexto de este proyecto, los parámetros que hemos optimizado han sido:

- Random Forest:
  - Número de estimadores (n_estimators).
  - Número de variables (max_features).
  - Máxima profundidad por árbol (max_depth).
  - Mínimo número de muestras en nodo (min_samples_split): Especifica el número mínimo de muestras requeridas para que un nodo se divida en dos nodos hijos.
  - Mínimo número de muestras en hoja (min_samples_leaf): Indica el número mínimo número de muestras requeridas en una hoja terminal.
  - Mínima disminución de entropía (min_impurity_decrese): Especifica el umbral mínimo de entropía necesario para que se realice un corte.
- XGBoost:
  - Número de estimadores (n_estimators).
  - Ratio de aprendizaje (learning_rate): Controla la magnitud de la actualización de los pesos en cada paso del entrenamiento. Un valor más bajo implica actualizaciones más pequeñas en cada paso, lo que puede llevar a un modelo más lento, pero más preciso. Por otro lado, un valor más alto implica actualizaciones más grandes, lo que puede llevar a un modelo más rápido, pero menos preciso.

- Máxima profundidad por árbol (max_depth).
- Mínimo pérdida en el corte (min_split_loss): Especifica cuanto debe reducirse como mínimo la función de pérdida tras la realización de un corte.

6.3.2 BACKTESTING

El *backtesting* es una técnica ampliamente utilizada en el ámbito del forecasting para evaluar el rendimiento de los modelos de predicción aplicados a datos secuenciales a lo largo del tiempo. Se basa en la simulación de la aplicación del modelo en datos históricos y la comparación de las predicciones generadas con los valores reales conocidos. Esta técnica nos permite analizar el rendimiento del modelo en situaciones pasadas y evaluar su capacidad de generalización en situaciones futuras.

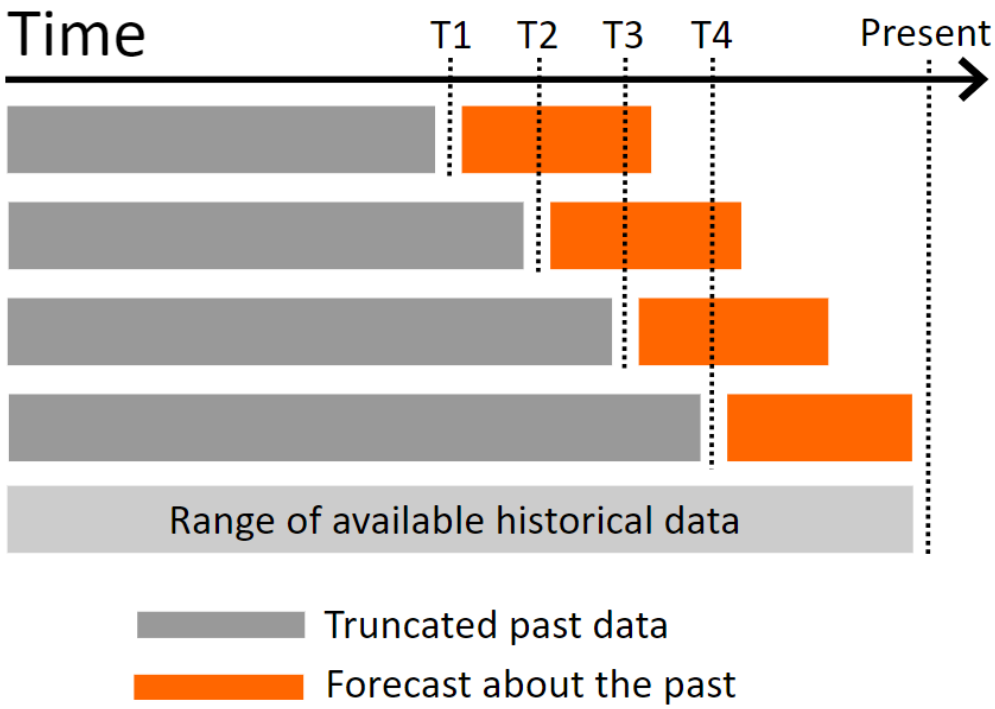


Figura 16. Diagrama Backtesting [9]

El proceso de backtesting para series temporales sigue los siguientes pasos:

1. Selección de datos históricos: En primer lugar, se selecciona un conjunto de datos históricos que abarque el período en el que se quiere evaluar el modelo.

2. División de los datos: A continuación, se divide el conjunto de datos históricos en dos partes: un conjunto de entrenamiento y un conjunto de prueba. El conjunto de entrenamiento se utiliza para ajustar el modelo, mientras que el conjunto de prueba se utiliza para evaluar su rendimiento.
3. Ajuste del modelo: En esta etapa, se ajusta el modelo de predicción, es decir, selección de variables a utiliza y optimización de los parámetros.
4. Generación de predicciones: Una vez ajustado el modelo, se generan las predicciones para el conjunto de prueba.
5. Evaluación del rendimiento: Por último, se compara los resultados obtenidos por el modelo con los valores reales del conjunto de prueba. Como métrica a evaluar nosotros hemos seleccionado: RMSE (Root Mean Square Error), MAE (Mean Absolute Error), MAPE (Mean Absolute Percentage Error)

Por último, para realizar una comparación rigurosa entre los errores obtenidos utilizando los conjuntos de datos aumentados y los errores obtenidos utilizando el conjunto de datos original, hemos aplicado el test de Wilcoxon. Esta prueba estadística se basa en la comparación de los errores generados por ambos conjuntos, es decir, el conjunto de datos aumentado y el conjunto de datos original.

El objetivo del test de Wilcoxon es determinar si existe una mejora significativa en los errores obtenidos al utilizar el conjunto de datos aumentado en comparación con el conjunto de datos original. El p-value resultante de esta prueba nos proporciona una medida de la significancia estadística de esta mejora.

Si el p-value es menor que 0.05, se considera que el rendimiento del modelo utilizando el conjunto de datos aumentados es estadísticamente mejor que el rendimiento del modelo utilizando el conjunto de datos original. Esto implica que los errores obtenidos con el conjunto de datos aumentado son significativamente menores a los obtenidos con el conjunto de datos original.

Capítulo 7. ANÁLISIS DE RESULTADOS

En este capítulo, se expondrán los resultados obtenidos a partir de la implementación de los modelos en los distintos conjuntos de datos aumentados, así como en el conjunto de datos original. Para evaluar el rendimiento de los modelos, se utilizarán métricas de evaluación establecidas, como el RMSE (Root Mean Square Error), el MAE (Mean Absolute Error) y el MAPE (Mean Absolute Percentage Error). Estas métricas nos permitirán medir la precisión y el rendimiento de los modelos.

Además, se realizará una comparación directa entre las predicciones generadas por los modelos utilizando los conjuntos de datos aumentados y las predicciones obtenidas utilizando el conjunto de datos original. Esta comparación nos permitirá determinar si los conjuntos de datos aumentados han logrado mejorar la precisión de las predicciones en comparación con el conjunto de datos original.

Para cada uno de los métodos de Data Augmentation implementados, se han realizado tres tipos de aumentaciones en relación al tamaño del conjunto de datos original: un 20% de las series, un 50% y un 100%. Para cada uno de los escenarios, el método que mejor rendimiento ha presentado ha sido:

- 20%:

MÉTODO	***RMSE***	***MAE***	***MAPE***
WARP	12.7	10.9	28.9
NOISE	12.6	10.9	28.9
POOL	12.6	10.9	28.9
CONVOLVE	12.6	10.9	28.8
AUTOENC	12.7	11.0	30.0

Tabla 2. Resultados 20% [Elaboración Propia]

Al aumentar el conjunto de datos original en un 20% mediante la inclusión de series sintéticas adicionales, donde el nuevo conjunto de datos aumentado consiste en el 100% de las series originales más un 20% de series aumentadas, se observa que no existen diferencias significativas entre los errores generados por los diferentes métodos en este escenario particular. Sin embargo, se destaca que el método que ha mostrado un mejor rendimiento en términos de precisión y ajuste ha sido el método Convolve.

A continuación, se procederá a realizar una comparación directa de los errores obtenidos con el método Convolve y el conjunto de datos original. Esta comparación permitirá evaluar el impacto de la aplicación de dicho método en la mejora o empeoramiento de los errores generados por los modelos.

METRIC	***VARIATION RF***	***VARIATION XGB***
RMSE %Δ	0.00	0.00
MAE %Δ	0.00	0.00
MAPE %Δ	0.00	0.00
p-value	1.00	0.90

Tabla 3. Variaciones 20% [Elaboración Propia]

Tras analizar la tabla anterior, se puede concluir que los errores obtenidos mediante el uso del conjunto de datos aumentado no presentan ninguna mejora en comparación con los errores obtenidos utilizando el conjunto de datos original. Este hecho se sustenta en el valor del p-value obtenido en la prueba de Wilcoxon, el cual indica que, en ninguno de los modelos, los errores obtenidos utilizando el conjunto de datos aumentado ha sido significativamente menor a los errores obtenidos utilizando el conjunto de datos original.

- 50%:

MÉTODO	***RMSE***	***MAE***	***MAPE***

WARP	12.7	11.0	28.9
NOISE	12.7	10.9	29.0
POOL	12.7	11.0	29.0
CONVOLVE	12.7	10.9	28.9
AUTOENC	13.1	11.3	31.3

Tabla 4. Resultados 50% [Elaboración Propia]

Al aumentar el conjunto de datos original en un 50% mediante la inclusión de series sintéticas adicionales, donde el nuevo conjunto de datos aumentado consiste en el 100% de las series originales más un 50% de series aumentadas, se observa que al igual que en el caso anterior, no existen diferencias significativas entre los errores generados por los diferentes métodos en este escenario particular. Sin embargo, se destaca que el método que ha mostrado un mejor rendimiento, al igual que en el caso anterior, ha sido el método Convolve.

A continuación, se procederá a realizar una comparación directa de los errores obtenidos con el método Convolve y el conjunto de datos original. Esta comparación permitirá evaluar el impacto de la aplicación de dicho método en la mejora o empeoramiento de los errores generados por los modelos.

METRIC	***VARIATION RF***	***VARIATION XGB***
RMSE %Δ	0.01	0.01
MAE %Δ	0.02	0.01
MAPE %Δ	0.01	0.00
p-value	1.00	1.00

Tabla 5. Variaciones 50% [Elaboración Propia]

Al igual que en el caso anterior, rras analizar la tabla anterior, se puede concluir que los errores obtenidos mediante el uso del conjunto de datos aumentado no presentan ninguna

mejora en comparación con los errores obtenidos utilizando el conjunto de datos original, en concreto, en alguna métrica hasta es un 2% mayor. Este hecho se sustenta en el valor del p-value obtenido en la prueba de Wilcoxon, el cual indica que, en ninguno de los modelos, los errores obtenidos utilizando el conjunto de datos aumentado ha sido significativamente menor a los errores obtenidos utilizando el conjunto de datos original.

- 100%:

MÉTODO	***RMSE***	***MAE***	***MAPE***
WARP	12.7	11.0	29.1
NOISE	12.6	10.9	28.9
POOL	12.8	11.1	29.1
CONVOLVE	12.8	11.0	29.0
AUTOENC	16.0	14.0	32.7

Tabla 6. Resultados 100% [Elaboración Propia]

Al aumentar el conjunto de datos original en un 100% mediante la inclusión de series sintéticas adicionales, donde el nuevo conjunto de datos aumentado consiste en el 100% de las series originales más un 100% de series aumentadas, es decir, este conjunto de datos aumentado se compone por el doble de series que el conjunto de datos original, se observa que, al igual que en los casos anteriores, no existen diferencias significativas entre los errores generados por los diferentes métodos en este escenario particular. Sin embargo, se destaca que el método que ha mostrado un mejor rendimiento en términos de precisión y ajuste ha sido el método Noise.

A continuación, se procederá a realizar una comparación directa de los errores obtenidos con el método Noise y el conjunto de datos original. Esta comparación permitirá evaluar el impacto de la aplicación de dicho método en la mejora o empeoramiento de los errores generados por los modelos.

MÉTRICA	***VARIACIÓN RF***	***VARIACIÓN XGB***
RMSE %Δ	0.00	0.00
MAE %Δ	0.00	0.00
MAPE %Δ	0.00	0.01
p-value	0.39	0.89

Tabla 7. Variaciones 100% [Elaboración Propia]

Por último, y al igual que en ambos casos anteriores, tras analizar la tabla anterior, se puede concluir que los errores obtenidos mediante el uso del conjunto de datos aumentado no presentan ninguna mejora en comparación con los errores obtenidos utilizando el conjunto de datos original. Este hecho se sustenta en el valor del p-value obtenido en la prueba de Wilcoxon, el cual indica que, en ninguno de los modelos, los errores obtenidos utilizando el conjunto de datos aumentado ha sido significativamente menor a los errores obtenidos utilizando el conjunto de datos original.

Capítulo 8. CONCLUSIONES Y TRABAJOS FUTUROS

En este capítulo, se expondrán las conclusiones derivadas de la aplicación de diferentes métodos de Data Augmentation sobre un conjunto de datos, así como la evaluación de su rendimiento mediante modelos de regresión.

En primer lugar, se ha observado que la aplicación de técnicas de Data Augmentation puede generar variaciones en los conjuntos de datos, enriqueciendo la diversidad y cantidad de muestras disponibles para el entrenamiento de los modelos de regresión. Sin embargo, los resultados han mostrado que no se obtiene una mejora significativa en el rendimiento de los modelos al utilizar conjuntos de datos aumentados en comparación con el conjunto de datos original. Esto indica que la aplicación de técnicas de Data Augmentation no garantiza necesariamente una mejora en la precisión de los modelos de regresión en este contexto específico.

Además, se ha observado que el rendimiento de los diferentes métodos simples ha sido prácticamente similar en cada uno de los 3 casos estudiados. Por otro lado, a diferencia de la literatura, el método complejo, en particular el autoencoder, no ha logrado mejorar los resultados en comparación con los métodos simples e incluso ha mostrado un rendimiento menor al de estos últimos. Por lo tanto, es crucial seleccionar cuidadosamente los métodos de Data Augmentation más apropiados para el conjunto de datos y el problema específico a abordar.

La diferencia entre nuestros resultados y las expectativas de la literatura pone de manifiesto la necesidad de realizar investigaciones adicionales para comprender mejor los factores que influyen en el rendimiento de los métodos de Data Augmentation. Esto implica examinar en mayor detalle las características del conjunto de datos, la estructura del problema y las propiedades de cada método de augmentación para identificar los factores clave para su estudio.

En cuanto a las futuras líneas de mejora, se plantean varias áreas de investigación. En primer lugar, se sugiere analizar la aplicación de técnicas de Data Augmentation más avanzadas y sofisticadas, como la generación de datos sintéticos mediante el uso de redes generativas adversarias (GANs). Estas técnicas podrían ofrecer una mayor diversidad y calidad en los datos aumentados, lo que potencialmente podría mejorar el rendimiento de los modelos de predicción.

Además, siguiendo las recomendaciones de la literatura, se sugiere evaluar los métodos de Data Augmentation utilizando modelos más complejos, como las redes neuronales. Aunque en este proyecto no se utilizaron debido a que no son comúnmente empleadas en el departamento de la compañía, resulta interesante estudiar y aplicar estos modelos para poder evaluar de manera exhaustiva el rendimiento real de los métodos de Data Augmentation.

Capítulo 9. BIBLIOGRAFÍA

[1] Demir, S., Mincev, K., Kok, K., & Paterakis, N. G. (2021). Data augmentation for time series regression: Applying transformations, autoencoders and adversarial networks to electricity price forecasting. Applied Energy, 304(117695), 117695. https://doi.org/10.1016/j.apenergy.2021.117695.

[2] Fons, E., Dawson, P., Zeng, X.-J., Keane, J., & Iosifidis, A. (2020). Evaluating data augmentation for financial time series classification. En arXiv [q-fin.ST]. http://arxiv.org/abs/2010.15111.

[3] M5 forecasting - accuracy. (s/f). Kaggle.com. Recuperado el 7 de junio de 2023, de https://www.kaggle.com/competitions/m5-forecasting-accuracy/data.

[4] Tsaug: An open-source python package for time series augmentation. (s/f). Arundo.com. Recuperado el 7 de junio de 2023, de https://www.arundo.com/articles/tsaug-an-open-source-python-package-for-time-series-augmentation.

[5] Scipy.Signal.Get_window — SciPy v1.10.1 manual. (s/f). Scipy.org. Recuperado el 7 de junio de 2023, de https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.get_window.html.

[6] Flores, S. (s/f). Variational autoencoders are beautiful. Compthree.com. Recuperado el 8 de junio de 2023, de https://www.compthree.com/blog/autoencoder/.

[7] (S/f). Tibco.com. Recuperado el 9 de junio de 2023, de https://www.tibco.com/reference-center/what-is-a-random-forest.

[8] (S/f-b). Researchgate.net. Recuperado el 9 de junio de 2023, de https://www.researchgate.net/publication/348025909_Predicting_the_Risk_of_Chronic_Kidney_Disease_CKD_Using_Machine_Learning_Algorithm.

[9] Backtesting. (2012). En Essential Mathematics for Market Risk Management (pp. 319–325). John Wiley & Sons, Ltd. https://www.lokad.com/backtesting-definition.