Predicción de la desatención de demanda en España debido a la transición energética

Edward Alejandro Giraldo Gallón

Mafredy Acevedo Holguín

Monografía presentada para optar al título de Especialista en Analítica y Ciencia de Datos

Asesores

Walter Mauricio Villa Acevedo, Doctor (PhD)

Álvaro Jaramillo Duque, Doctor (PhD)

Universidad de Antioquia

Facultad de Ingeniería

Especialización en Analítica y Ciencia de Datos

Medellín, Antioquia, Colombia

2023

Cita	(Giraldo Gallón & Acevedo Holguín, 2023)
Referencia **Estilo APA 7 (2020)**	Giraldo Gallón, E.A, & Acevedo Holguín, M. (2023). *Predicción de la desatención de demanda en España debido a la transición energética* Trabajo de grado especialización. Universidad de Antioquia, Medellín, Colombia.

creative commons CC BY NC SA

Especialización en Analítica y Ciencia de Datos, Cohorte V.

Centro de Investigación Ambientales y de Ingeniería (CIA).

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia

Sistema de Bibliotecas

Centro de Documentación Ingeniería (CENDOI)

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

Rector: John Jairo Arboleda Céspedes.
Decano: Julio Cesar Saldarriaga Molina
Jefe departamento: Diego José Luis Botia Valderrama

El contenido de esta obra corresponde al derecho de expresión de los autores y no compromete el pensamiento institucional de la Universidad de Antioquia ni desata su responsabilidad frente a terceros. Los autores asumen la responsabilidad por los derechos de autor y conexos.

Dedicatoria

A nuestros queridos padres que, a través de sus enseñanzas y apoyo, han labrado el camino para cumplir esta meta.

A nuestros amados abuelos que, con sus historias, ternura y acompañamiento, nos han endulzado este arduo trayecto, convirtiéndolo en una travesía más llevadera.

A nuestras parejas, fuentes de constante de apoyo, paciencia, comprensión y, sobre todo, amor incondicional.

Por último, a nuestras adoradas mascotas, compañeros incansables en estas largas noches de trabajo.

A todos ustedes, nuestras familias, les dedicamos con gratitud y cariño el fruto de nuestros esfuerzos.

Agradecimientos

A la Universidad de Antioquia, le queremos expresar nuestro profundo agradecimiento por brindarnos la invaluable oportunidad de habernos acogido y permitirnos formar académicamente como especialistas.

Agradecemos especialmente a nuestros asesores de monografía, Walter Villa y Álvaro Jaramillo, cuya orientación, consejos y conocimientos han sido pilares fundamentales en el desarrollo de este proyecto.

A los profesores del programa en Analítica y Ciencia de datos, por sus valiosas enseñanzas, las cuales, han sido esenciales para nuestro aprendizaje.

Finalmente, nos agradecemos el uno al otro, ya que este viaje académico ha sido posible gracias a nuestra colaboración constante, compartiendo ideas, superando desafíos y celebrando logros juntos.

Tabla de contenido

Resumen 11

Abstract 12

1. Descripción del problema 13

1.1. Problemática 13

1.2. Aproximación desde la analítica de datos 13

1.3. Origen de los datos 14

1.4. Métricas de desempeño 14

2. Objetivos 18

2.1. Objetivo general 18

2.2. Objetivos específicos 18

3. Datos 19

3.1. Datos originales 19

3.2. Conjunto de datos 21

3.3. Analítica descriptiva 22

4. Proceso de analítica 34

4.1. Pipeline principal 34

4.2. Subprocesos 38

4.3. Modelos 42

4.3.1. OLS 42

4.3.2. Random Forest Regressor 43

4.3.3. Support Vector Regressor 43

4.3.4. Decisión Tree 44

4.3.5. Extreme Learning Machine 45

4.3.6. Extra Trees Regressor 45

4.3.7. Gradient Boosting Machine 46
4.3.8. Light Gradient Boosting Machine 46
4.3.9. Feed-Forward Neural Network 47
4.4. Métricas 48
5. Metodología 49
5.1. Herramientas 49
5.2. Línea Base (Baseline) 51
5.3. Iteraciones y evolución 57
5.3.1. Segunda iteración 57
5.3.2. Tercera iteración 60
5.3.3. Cuarta iteración 60
5.3.4. Quinta iteración 61
5.3.5. Sexta iteración 64
5.3.6. Séptima iteración 65
5.3.7. Octava iteración 66
6. Resultados y discusión 72
6.1. Métricas 72
6.2. Evaluación cualitativa 77
7. Conclusiones 78
8. Recomendaciones 81
Referencias 83
Anexos 85
Anexo A. Visualizaciones complementarias 85

Lista de tablas

Tabla 1. Características disponibles por ciudad 20

Tabla 2. Cantidad de observaciones y tipos de variables 22

Tabla 3. Códigos para condiciones climáticas (Weather Identification Codes) 29

Tabla 4. Parámetros de los modelos implementados en la línea base 52

Tabla 5. Parámetros del algoritmo Local Outlier Factor 53

Tabla 6. Resultados de las métricas de desempeño para los modelos base (primera iteración) 54

Tabla 7. Resultados de las métricas de desempeño para la segunda iteración. 59

Tabla 8. Resultados de las métricas de desempeño para la tercera iteración. 60

Tabla 9. Resultados de las métricas de desempeño para la cuarta iteración. 61

Tabla 10. Modelos de aprendizaje automático e hiperparámetros. 61

Tabla 11. Resultados de las métricas de desempeño para la quinta iteración. 63

Tabla 12. Resultados de las métricas de desempeño para la sexta iteración. 64

Tabla 13. Resultados de las métricas de desempeño para la séptima iteración. 66

Tabla 14. Desviación porcentual de las métricas RMSE obtenidas en la iteración 7 67

Tabla 15. Espacio de hiperparámetros para algoritmo RandomForestRegressor. 68

Tabla 16. Espacio de hiperparámetros para algoritmo ExtraTreesRegressor. 68

Tabla 17. Espacio de hiperparámetros para algoritmo LigthGradientBoostingMachine. 69

Tabla 18. Espacio de hiperparámetros para algoritmo AtificialNeuralNetwork. 69

Tabla 19. Mejores hiperparámetros para algoritmo RandomForestRegressor. 70

Tabla 20. Mejores hiperparámetros para algoritmo ExtraTreesRegressor. 70

Tabla 21. Mejores hiperparámetros para algoritmo LightGradientBoostingMachine. 70

Tabla 22. Mejores hiperparámetros para algoritmo AtificialNeuralNetwork. 71

Tabla 23. Resultados de las métricas de desempeño para la octava iteración.....71

Tabla 24. Desviación porcentual de las métricas RMSE obtenidas en la iteración 8. 71

Tabla 25. Resumen de las métricas para los mejores modelos por iteración. 72

Tabla 26. Variación porcentual de las métricas para los mejores modelos para cada iteración. .74

Lista de figuras

Figura 1. Histogramas de velocidad del viento en diferentes ciudades 24

Figura 2. Histogramas de ángulo del viento en diferentes ciudades 24

Figura 3. Histogramas de lluvia caída en una hora en diferentes ciudades.....24

Figura 4. Histogramas de lluvia caída en tres horas en diferentes ciudades 24

Figura 5. Histogramas de humedad en diferentes ciudades 25

Figura 6. Histogramas de nubosidad en diferentes ciudades 25

Figura 7. Histogramas de presión atmosférica en diferentes ciudades 25

Figura 8. Histogramas de nieve en diferentes ciudades 25

Figura 9. Histogramas de condición climática en diferentes ciudades 26

Figura 10. Histogramas de temperatura máxima en diferentes ciudades.....26

Figura 11. Histogramas de temperatura mínima en diferentes ciudades 27

Figura 12. Histogramas de temperatura promedio en diferentes ciudades 27

Figura 13. Gráfico de barras de variables categóricas 28

Figura 14. Histograma de variable de salida.....28

Figura 15. Covarianza entre variables del modelo.....30

Figura 16. Diagrama de pares de puntos para las variables de presión de la base de datos, con relación a la variable de salida. 31

Figura 17. Diagrama de pares de puntos para las variables de velocidad del viento de la base de datos, con relación a la variable de salida. 32

Figura 18. Diagrama de pares de puntos para las variables de humedad de la base de datos, con relación a la variable de salida. 32

Figura 19. Diagrama de pares de puntos para las variables de temperatura de la base de datos, con relación a la variable de salida.....32

Figura 20. Diagrama de pares de puntos para las variables de id del clima de la base de datos, con relación a la variable de salida.....33

Figura 21. Diagrama de pares de puntos para las variables nubes de la base de datos, con relación a la variable de salida. 33

Figura 22. Diagrama de pares de puntos para las variables lluvia de la base de datos, con relación a la variable de salida. 33

Figura 23. Vista general del Pipeline principal 34

Figura 24. Flujo de trabajo detallado 37

Figura 25. Funcionamiento de los algoritmos de búsqueda de hiperparámetros de un modelo de Machine Learning. (Cortinhas S, 2022) 41

Figura 26. Comparación visual entre predicciones y valores reales al usar regresión lineal. 55

Figura 27. Comparación visual entre predicciones y valores reales al usar Random Forest Regressor. 55

Figura 28. Nueva estructura de las variables "categoría del clima (weather_id)" 58

Figura 29. Diagrama de cajas de las características de presión disponibles en la Base de datos .58

Figura 30. Diagrama de cajas de las características de presión disponibles en la Base de datos, realizado el tratamiento de datos atípicos para la característica Barcelona_pressure. 59

Figura 31. Evolución del RMSE para los mejores modelos a lo largo de todas las iteraciones implementadas. 75

Figura 32. Evolución del MAE para los mejores modelos a lo largo de todas las iteraciones implementadas. 75

Figura 33. Comparación de la desatención de la demanda estimada y real obtenida mediante el modelo Random Forest luego del hypertuning. 76

Figura 34. Comparación de la desatención de la demanda estimada y real obtenida mediante el modelo Light Gradient Boosting Machine luego del hypertuning. 76

Figura 35. Comparación de la desatención de la demanda estimada y real obtenida mediante el modelo Artificial Neural Network Machine luego del hypertuning. 76

Siglas, acrónimos y abreviaturas

ANN	Artificial Neural Network
APA	American Psychological Association
corr	Correlación
CPU	Central Processing Unit
DT	Decisión Tree
ELM	Extreme Learning Machine
ETR	Extra Trees Regressor
GBM	Gradient Boosting Machine
GPU	Graphics Processing Unit
KNN	K-Nearest Neighbors
LightGBM	Light Gradient Boosting Machine
LOF	Local Outlier Factor
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
OLS	Ordinary Least Squares
PCA	Principal Components Analysis
PhD	Philosophie Doctor
R^2	Coeficiente de determinación
RF	Random Forest Regressor
RMSE	Root Mean Squared Error
SGD	Stochastic Gradient Descent
Std	Standard Scaler
SVM	Support Vector Machine
SVR	Support Vector Regressor
TPE	Tree-sctructured Parzen Estimator
VS	Visual Studio

Resumen

En respuesta a la creciente preocupación mundial por la sostenibilidad de las fuentes de energía, este documento aborda uno de los desafíos de la transición hacia una matriz energética más renovable en España. El enfoque se centra en encontrar un modelo predictivo basado en técnicas de aprendizaje automático para estimar la brecha entre la energía generada por combustibles fósiles y diversas fuentes renovables.

La propuesta de este estudio consiste en la utilización de características climáticas clave, tales como presión atmosférica, velocidad del viento y humedad, para prever la brecha de energía eléctrica. Los resultados obtenidos buscan, en primera instancia, explorar y analizar diversas técnicas de preparación y análisis de datos que se adapten a la complejidad inherente a la problemática planteada.

Las iteraciones de preparación de datos revelan diferentes enfoques, mientras que la exploración de múltiples modelos de aprendizaje automático ofrece distintos hallazgos encontrados al abordar este desafío desde la perspectiva de la ciencia de datos.

En última instancia, se proponen sugerencias y recomendaciones basadas en las experiencias adquiridas durante el análisis. Estos aportes tienen como objetivo brindar estrategias para enfrentar problemas similares de regresión, aprovechando las lecciones aprendidas durante el desarrollo de este estudio.

Palabras clave: déficit energético, fuentes de energía renovables, aprendizaje automático, optimización.

Los códigos relacionados a este trabajo pueden ser consultados en el siguiente repositorio de GitHub:

https://github.com/EACD-Prediccion-Desatencion-Demanda/Monografia

github.com/edwardg17

github.com/mafredy95

Abstract

In response to the growing worldwide concern about the sustainability of energy sources, this paper tackles a challenge tied to shifting towards a more renewable energy matrix in Spain. The main focus is on creating a predictive model, using Machine Learning techniques, to estimate the gap between energy generated from fossil fuels and various renewable sources.

This study suggests using key weather features like atmospheric pressure, wind speed, and humidity to predict electricity shortfall. The obtained results aim, initially, to explore and analyze various data preparation and analysis techniques that adapt to the inherent complexity of the problem presented.

The various steps of data preparation uncover different methods, and exploring multiple Machine Learning models brings out different findings while approaching this challenge from the perspective of data science.

Ultimately, suggestions and recommendations are given based on experiences gained during the analysis. The goal is to offer practical strategies for dealing with similar regression problems, drawing on the lessons learned throughout this study.

*Keywords***:** electricity shortfall, renewable energy sources, Machine Learning, optimization.

The scripts associated with this work can be accessed through the following GitHub repository:

https://github.com/EACD-Prediccion-Desatencion-Demanda/Monografia

github.com/edwardg17

github.com/mafredy95

1. Descripción del problema

1.1.Problemática

En los últimos años, la sostenibilidad de las fuentes de energía se ha convertido en una preocupación crítica a nivel mundial. La dependencia de fuentes basadas en combustible fósil para la generación de electricidad plantea desafíos para garantizar un suministro de energía constante y respetuoso con el medio ambiente. Como respuesta a esta problemática, los gobiernos están invirtiendo cada vez más en infraestructuras de energía renovable.

A medida que se acelera la transición energética, surgen problemáticas relacionadas con la generación de energía no convencional. Una de ellas es el déficit de suministro energético que puede ocurrir debido a la naturaleza aleatoria de los recursos energéticos renovables, como la disponibilidad intermitente del viento y el sol. En contraste, la generación de energía térmica basada en combustibles fósiles que se pretende reemplazar tiene una disponibilidad más fácil de estimar, siempre y cuando haya un suministro adecuado de combustibles. Este déficit energético es un desafío que debe abordarse en conjunto con la transición energética.

Siguiendo esta tendencia, el gobierno de España está considerando la expansión de sus recursos de energía renovable. Para respaldar este proceso de toma de decisiones, es esencial comprender las tendencias y patrones de las fuentes de energía renovable respecto a la generación de energía mediante combustibles fósiles en el país, y deben establecer la cantidad de energía eléctrica que puede quedar desatendida en el caso que la disponibilidad del recursos renovables se vean disminuidos y obligue al uso de plantas de generación que consumen combustibles fósiles u otros energéticos para producir energía eléctrica.

1.2. Aproximación desde la analítica de datos

Este estudio tiene como objetivo desarrollar un modelo predictivo que estime la brecha entre la energía generada por los combustibles fósiles y diversas fuentes de energía eléctrica usando recursos renovables en España. Mediante el análisis de características climáticas específicas de cada ciudad, como la presión atmosférica, la velocidad del viento y la humedad, se busca pronosticar el déficit diario de energía eléctrica. Los resultados de este estudio proporcionarán información valiosa a la hora de tomar decisiones informadas sobre la expansión de la

infraestructura de energía renovable y mejorar la sostenibilidad y confiabilidad general del suministro energético del país.

1.3. Origen de los datos

Se tomará como punto de partida la información suministrada para el desafío Kaggle realizado por Explore Data Science Academy llamado "Electricity Shortfall Challenge" (ExploreAI Academy, 2022).

Este reto se basa en un conjunto de datos que contiene información sobre las condiciones climáticas en varias ciudades españolas entre el 2015 al 2017. También, cuenta con los déficits de demanda eléctrica en ventanas de tres horas durante el mismo período de tiempo. En el contexto de este problema, el déficit de demanda eléctrica en períodos de tres horas es la diferencia entre la energía generada a través de tecnologías de generación de electricidad que usan combustibles fósiles y tecnologías de generación de energía que usan recursos renovables.

1.4. Métricas de desempeño

Para evaluar el desempeño de los modelos a emplear, se hace uso de diferentes métricas, especialmente aquellas que permitan evidenciar el desempeño en las mismas unidades de la variable de salida o como un porcentaje, de manera tal que se facilite su interpretación. A continuación, se detallan las principales métricas consideradas:

Coeficiente de determinación (R^2)

El coeficiente de determinación R^2 es una medida que indica la proporción de la variabilidad en la variable dependiente que puede explicarse mediante el modelo de regresión. En otras palabras, R^2 cuantifica cuánto del total de la variabilidad de los datos es capturado por el modelo. Un valor de R^2 cercano a 1 indica un buen ajuste del modelo, mientras que un valor cercano a 0 sugiere que el modelo que no es capaz de explicar la variabilidad en los datos.

Seguidamente se presenta la formulación de la métrica R^2:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Donde:

y: etiquetas o valores reales

ŷ: predicciones

Root Mean Squared Error (RMSE)

El RMSE, conocido como el Error de la Raíz Cuadrada de la Media en español, es una métrica ampliamente utilizada en estadística y en el campo del aprendizaje automático para evaluar la precisión de modelos de regresión y pronóstico. El RMSE se utiliza para medir la capacidad que tiene un modelo de predecir valores numéricos y cuánto se desvían los valores reales de los valores predichos.

El RMSE penaliza más los errores grandes que los pequeños, ya que los errores se elevan al cuadrado antes de promediarlos y tomar la raíz cuadrada. Cuanto menor sea el valor del RMSE, mejor será el rendimiento del modelo, ya que indica que las predicciones se ajustan más a los valores reales.

Según Chai y Draxler (2014): "el RMSE es más apropiado para representar el desempeño del modelo que el MAE cuando se espera que la distribución del error sea gaussiana" (p.7).

La siguiente formula corresponde al cálculo del RMSE:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2 - \hat{y}_i^2}$$

Donde:

n: número de muestras

y: etiquetas o valores reales

ŷ: predicciones

Mean Absolute Error (MAE)

En español, el Error Medio Absoluto, comúnmente conocido como MAE, se define en el contexto de la estadística como una medida de discrepancia entre dos observaciones que representan el mismo fenómeno. Su propósito principal es evaluar la precisión de un modelo de regresión o cuantificar la diferencia promedio entre las predicciones generadas por el modelo y los valores reales. El cálculo del MAE implica sumar los valores absolutos de todos los errores y dividir este total por el número de observaciones.

El MAE se distingue por su capacidad para no penalizar excesivamente los errores de gran magnitud, lo que lo hace especialmente robusto frente a valores atípicos. Además, destaca por su simplicidad y facilidad de comprensión, dado que su unidad de medida es la misma que la de la

variable objetivo. En términos prácticos, un valor más bajo de MAE indica un mejor desempeño del modelo, ya que sugiere que las predicciones se ajustan de manera más precisa a los valores reales.

Algunos análisis realizados en la literatura, como por ejemplo los llevados a cabo por Willmott y Matsuura (2005), indican que "el MAE es la medida más natural de la magnitud del error promedio, y que (a diferencia del RMSE) es una medida no ambigua" (p.30), sugiriendo ampliamente el uso de esta.

Posteriormente, se presenta la formulación de la métrica MAE:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

Donde:

n: número de muestras

y: etiquetas o valores reales

ŷ: predicciones

Mean Absolute Percentage Error (MAPE)

El MAPE, cuyas siglas representan el Error Porcentual Absoluto Medio (en inglés, Mean Absolute Percentage Error), es una métrica ampliamente empleada para cuantificar la discrepancia porcentual promedio entre las predicciones de un modelo y los valores reales. Su cálculo implica la obtención de la diferencia porcentual entre cada valor pronosticado por el modelo y su correspondiente valor real, seguido de la determinación del promedio de estas discrepancias.

El MAPE se expresa como un porcentaje, lo que facilita su comprensión y comunicación. Además, al evaluar los errores en términos porcentuales, el MAPE tiene en cuenta la magnitud relativa de las discrepancias en lugar de sus valores absolutos, lo cual resulta valioso en contextos donde la magnitud de los errores desempeña un papel crucial.

No obstante, es importante señalar que el MAPE puede mostrar sensibilidad a divisiones por valores pequeños cuando los valores reales son próximos a cero, lo que podría resultar en tasas de error extremadamente elevadas o inestables en tales situaciones. De hecho, de acuerdo con De Myttenaere et al. (2016), en contextos de aplicaciones prácticas, el MAPE es una elección común cuando se tiene la certeza de que la cantidad que se desea predecir se mantiene claramente por encima de cero.

Ahora, se presenta la formulación de la métrica MAPE:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

Criterio de desempeño

Como punto de referencia para evaluar la adaptabilidad de los modelos y seleccionar el de mejor desempeño, se empleará la mejor métrica publicada en el desafío Kaggle "Electricity Shortfall Challenge" (ExploreAI Academy, 2022), del cual se han extraído los datos para llevar a cabo el presente análisis. Estas métricas están detalladas en la tabla de clasificación (*LeaderBoard*) del desafío y corresponden a las consultadas puntualmente el 14 de noviembre de 2023.

Es esencial destacar que dicho desafío utiliza la métrica RMSE como criterio principal de evaluación. Aunque en este análisis se examinarán todas las métricas presentadas anteriormente con el objetivo de explorar el comportamiento de los modelos bajo diversos criterios, la evaluación definitiva del desempeño de los modelos se basará en el RMSE.

La mejor métrica reportada registró un valor de 3.262,18 unidades energéticas, y dicho valor se utilizará como referencia para medir el desempeño de los modelos evaluados y seleccionar el mejor entre ellos.

2. Objetivos

2.1.Objetivo general

Desarrollar un modelo predictivo utilizando técnicas de Machine Learning, que permita estimar con exactitud la brecha entre la energía generada por combustibles fósiles y diversas fuentes renovables en España, partiendo de condiciones climatológicas asociadas a una estampa de tiempo. Este modelo contribuirá a informar las decisiones de expansión de la infraestructura de energía renovable, mejorando la sostenibilidad y confiabilidad del suministro energético del país.

2.2.Objetivos específicos

- Realizar un proceso de limpieza y transformación de datos para asegurar la calidad y consistencia de los datos.
- Llevar a cabo un análisis exploratorio de los datos para comprender las tendencias, patrones y relaciones entre la generación de energía y las condiciones climáticas en diversas ciudades de España.
- Identificar posibles correlaciones entre los datos climáticos y la brecha energética entre fuentes renovables y combustibles fósiles.
- Experimentar con diversas técnicas de Machine Learning, como regresión u otras apropiadas, para desarrollar modelos predictivos de la brecha energética.
- Definir las métricas más adecuadas para evaluar el rendimiento de los modelos.
- Evaluar los diferentes modelos con las métricas definidas para encontrar los hiperparámetros más adecuados.
- Seleccionar el modelo de aprendizaje automático más apropiado para abordar el problema, junto con los hiperparámetros que maximizan su rendimiento.

3. Datos

3.1. Datos originales

El conjunto de datos original contiene cuarenta y siete (47) características y una (1) variable objetivo. Las características, están asociadas a mediciones de variables meteorológicas específicas de ciudades, por ejemplo, la velocidad del viento en Barcelona. En total, hay datos meteorológicos disponibles para 5 lugares, pero no todas las ciudades tienen información disponible para todas las categorías meteorológicas, por ejemplo, se puede tener datos de velocidad del viento para Barcelona, pero no datos de lluvia; mientras que se tiene información de lluvia y velocidad del viento para Valencia.

En resumen, se poseen datos meteorológicos para las siguientes ciudades de España:

- Madrid
- Valencia
- Sevilla
- Bilbao
- Barcelona

Las categorías meteorológicas en el conjunto de datos incluyen:

- **Fecha [time]:** Fecha (año-mes-día) y hora en la que fue tomada la observación.
- **Velocidad del viento [wind_speed]:** La velocidad del viento en un intervalo de tiempo específico.
- **Ángulo del viento [wind_degree]:** La dirección del viento en un intervalo de tiempo específico.
- **Lluvia (última hora) [rain_1h]:** Una métrica que expresa la cantidad de lluvia caída en la última hora en una ciudad en particular.
- **Lluvia (últimas 3 horas) [rain_3h]:** Una métrica que expresa la cantidad de lluvia caída en las últimas tres horas en una ciudad en particular.
- **Humedad [humidity]:** El nivel de humedad medido en el momento definido.
- **Nubosidad [clouds_all]:** El nivel de cobertura de nubes medido en el punto especificado en el tiempo.
- **Presión [pressure]:** La presión atmosférica en un intervalo de tiempo específico.

- **Nieve (periodo de 3 horas) [snow_3h]:** Una métrica que expresa la cantidad de nieve caída en las últimas tres horas.
- **Clasificación del clima [weather_id]:** Una métrica utilizada para explicar la condición climática de una ciudad específica en un momento determinado.
- **Temperatura máxima [temp_max]:** La temperatura máxima para una ciudad específica en un momento determinado.
- **Temperatura [temp]:** La temperatura promedio para una ciudad específica en un momento determinado.

Cada combinación de variable meteorológica y ciudad compone una columna de los datos, cuya denominación tiene el formato "*Ciudad_variable_meteorológica*"; por ejemplo, la velocidad del viento en Madrid estará descrita como "Madrid_wind_speed".

En la **Tabla *1*** se puede observar de manera más detallada la información que se posee por cada ciudad:

Tabla 1.
Características disponibles por ciudad

Categoría meteorológica	Madrid	Valencia	Sevilla	Bilbao	Barcelona
wind_speed	x	x	x	x	x
wind_deg		x		x	x
rain_1h	x		x	x	x
rain_3h			x		x
humidity	x	x	x		
clouds_all	x		x	x	
pressure	x	x	x	x	x
snow_3h		x		x	
weather_id	x		x	x	x
temp_max	x	x	x	x	x
temp_min	x	x	x	x	x
temp	x	x	x	x	x

Finalmente, para cada muestra se tiene un registro de la variable objetivo, denominada como "*load_shortfall_3h*", la cual se puede definir como la diferencia entre la energía generada

por fuentes de energía renovable (solar, eólica, geotérmica, etc.) y la energía generada con combustibles fósiles, entregada en intervalos de tiempo de tres horas.

Los registros están divididos en dos archivos: "*df_train.csv*" (Datos de entrenamiento) y "*df_test.csv*" (Datos de prueba).

Los datos de entrenamiento tienen un tamaño de aproximado 3,73 MB y están asociados a registros tomados cada 3 horas entre los años 2015 y 2017, conformando un total de 8.762 observaciones.

Los datos de prueba, tomados con la misma periodicidad, tienen un tamaño de aproximadamente 1,24 MB y contienen 2.920 muestras, correspondientes al año 2018. No obstante, estos datos de prueba están pensados para completar el desafío Kaggle del cual se ha tomado la información y por ende, no contienen la variable de salida "*load_shortfall_3h*" como parte del archivo.

Debido a lo anterior, para los objetivos a desarrollar en este documento, se toma el conjunto de datos "*df_train.csv*" (en adelante *"Datos completos"*) como única información de entrada y posteriormente, este será dividido tanto en datos de entrenamiento como prueba para la generación de los modelos.

3.2. Conjunto de datos

La información designada como *"Datos completos"* se dividirá en conjuntos de datos de entrenamiento y prueba en todas las evaluaciones. Esto se hace con el propósito de evaluar la efectividad de los modelos entrenados cuando se enfrentan a datos desconocidos y para prevenir el sobreajuste. Sin embargo, la forma en que se lleva a cabo esta partición dependerá de la etapa del análisis en cuestión.

En las primeras iteraciones, se realizará una partición en la que el 80% de los *"Datos completos"* se destinará al entrenamiento de los modelos, mientras que el 20% restante se utilizará para la prueba de los modelos. A medida que se realicen análisis más detallados con los modelos que mejor se adapten a los datos analizados, se recurrirá a técnicas de validación cruzada. En la mayoría de los casos, esta técnica utilizará el 90% de los datos para entrenamiento, dividiendo esta parte en 5 pliegues diferentes, y el 10% restante se reservará como datos de prueba. Si en algún momento se decide utilizar una partición diferente para un modelo en particular, se aclarará este punto durante el análisis correspondiente.

Para llevar a cabo estas tareas, se emplearán los métodos "train_test_split" y "cross-validation" de la biblioteca scikit-learn.

3.3. Analítica descriptiva

En este numeral, se presentan un análisis descriptivo del conjunto de datos a analizar, con el fin de proporcionar una visión general de las características y distribuciones de los datos. Este análisis será fundamental para comprender mejor la naturaleza de la información y destacar patrones relevantes.

La primera consideración importante consiste en determinar tanto la cantidad de datos faltantes, como la naturaleza (tipo) de las 47 características de entrada y la variable de salida. A continuación, se proporciona la información correspondiente:

Tabla 2.
Cantidad de observaciones y tipos de variables

Variable	Observaciones no nulas		Tipo de variable
	Cantidad	Porcentaje	
Madrid_wind_speed	8763	100%	numérica
Valencia_wind_deg	8763	100%	categórica
Bilbao_rain_1h	8763	100%	numérica
Valencia_wind_speed	8763	100%	numérica
Seville_humidity	8763	100%	numérica
Madrid_humidity	8763	100%	numérica
Bilbao_clouds_all	8763	100%	numérica
Bilbao_wind_speed	8763	100%	numérica
Seville_clouds_all	8763	100%	numérica
Bilbao_wind_deg	8763	100%	numérica
Barcelona_wind_speed	8763	100%	numérica
Barcelona_wind_deg	8763	100%	numérica
Madrid_clouds_all	8763	100%	numérica
Seville_wind_speed	8763	100%	numérica
Barcelona_rain_1h	8763	100%	numérica
Seville_pressure	8763	100%	categórica
Seville_rain_1h	8763	100%	numérica
Bilbao_snow_3h	8763	100%	numérica
Barcelona_pressure	8763	100%	numérica
Seville_rain_3h	8763	100%	numérica
Madrid_rain_1h	8763	100%	numérica

Variable	Observaciones no nulas		Tipo de variable
	Cantidad	Porcentaje	
Barcelona_rain_3h	8763	100%	numérica
Valencia_snow_3h	8763	100%	numérica
Madrid_weather_id	8763	100%	numérica
Barcelona_weather_id	8763	100%	numérica
Bilbao_pressure	8763	100%	numérica
Seville_weather_id	8763	100%	numérica
Valencia_pressure	**6695**	**76%**	numérica
Seville_temp_max	8763	100%	numérica
Madrid_pressure	8763	100%	numérica
Valencia_temp_max	8763	100%	numérica
Valencia_temp	8763	100%	numérica
Bilbao_weather_id	8763	100%	numérica
Seville_temp	8763	100%	numérica
Valencia_humidity	8763	100%	numérica
Valencia_temp_min	8763	100%	numérica
Barcelona_temp_max	8763	100%	numérica
Madrid_temp_max	8763	100%	numérica
Barcelona_temp	8763	100%	numérica
Bilbao_temp_min	8763	100%	numérica
Bilbao_temp	8763	100%	numérica
Barcelona_temp_min	8763	100%	numérica
Bilbao_temp_max	8763	100%	numérica
Seville_temp_min	8763	100%	numérica
Madrid_temp	8763	100%	numérica
Madrid_temp_min	8763	100%	numérica
load_shortfall_3h	8763	100%	numérica

De la **Tabla *2***, se destacan tres hallazgos de relevancia. En primer lugar, se evidencia la división de las variables en dos categorías distintas: numéricas (45) y categóricas (2). En segundo lugar, se resalta el hecho de que la variable "Valencia_pressure" presenta un 24% de datos faltantes. Por último, se identifica que la variable de salida "load_shortfall_3h" se clasifica como una variable numérica, lo que implica que los modelos a desarrollar serán de tipo regresión.

Siguiendo con el análisis de los datos, resulta fundamental comprender su distribución. Con el fin de lograr este objetivo, se han generado histogramas para las variables numéricas y gráficos de barras para las categóricas, los cuales se detallan desde la ***Figura 1*** hasta la ***Figura 14***.

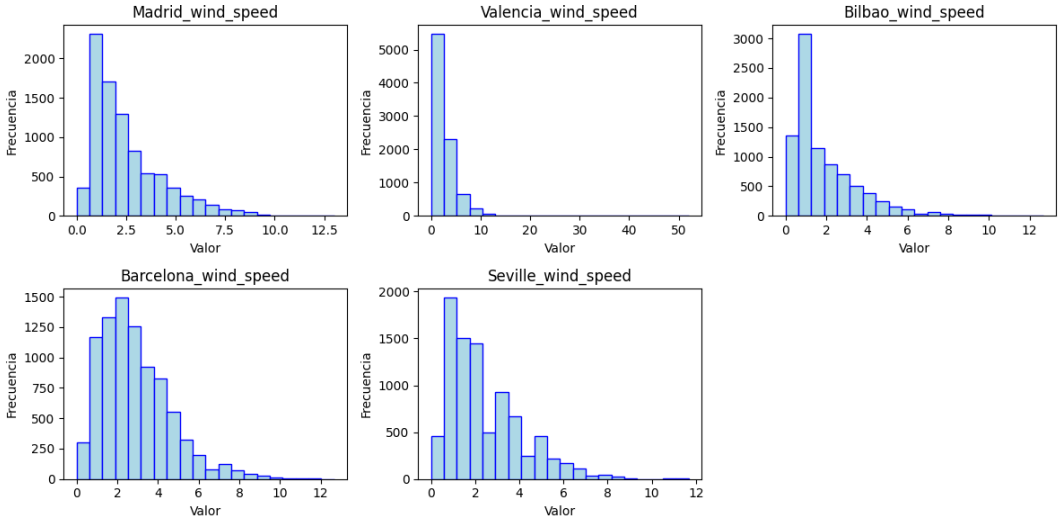


Figura 1. *Histogramas de velocidad del viento en diferentes ciudades*

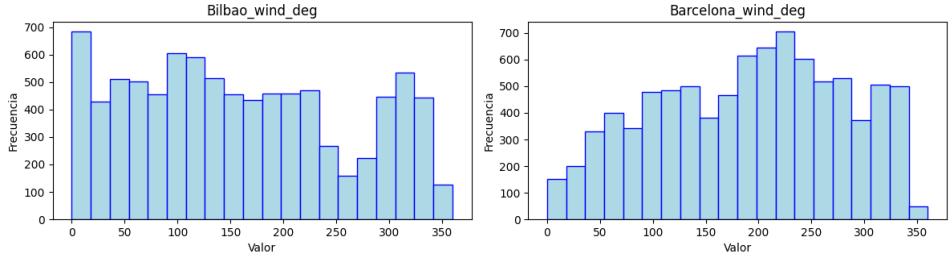


Figura 2. *Histogramas de ángulo del viento en diferentes ciudades*

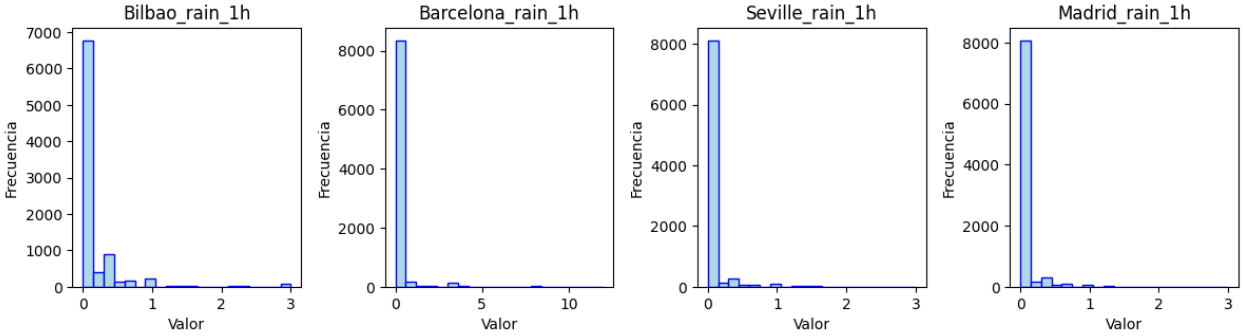


Figura 3. *Histogramas de lluvia caída en una hora en diferentes ciudades*

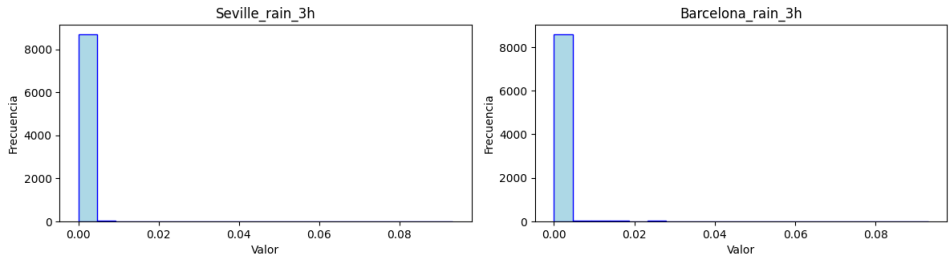


Figura 4. *Histogramas de lluvia caída en tres horas en diferentes ciudades*

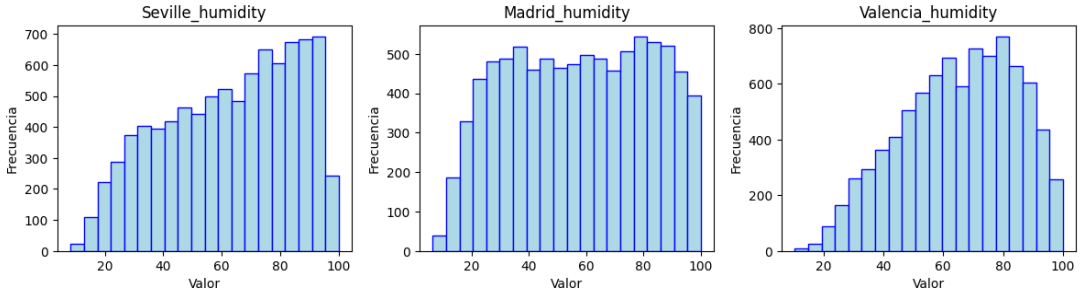


Figura 5. *Histogramas de humedad en diferentes ciudades*

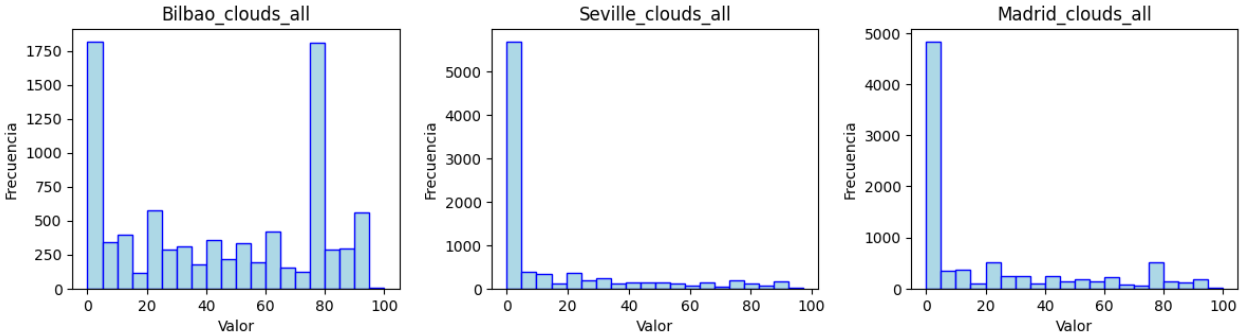


Figura 6. *Histogramas de nubosidad en diferentes ciudades*

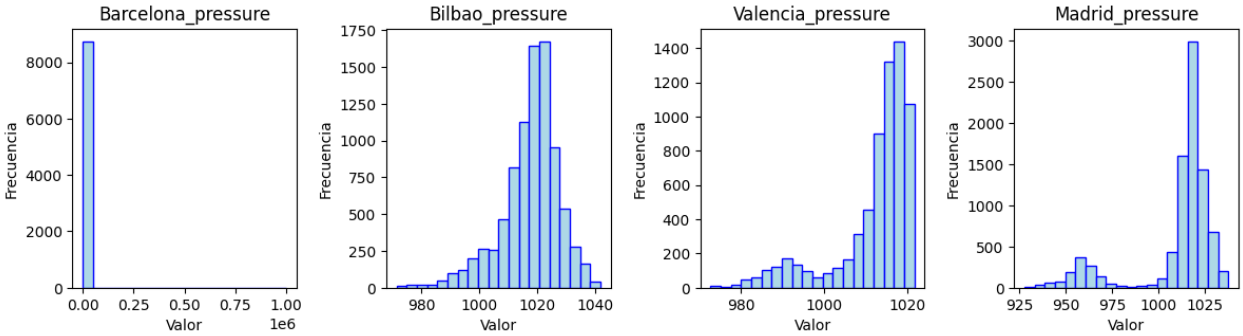


Figura 7. *Histogramas de presión atmosférica en diferentes ciudades*

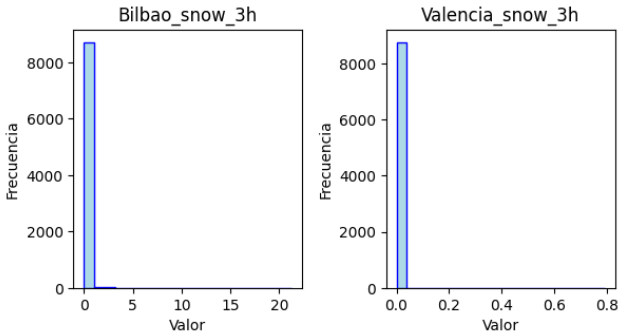


Figura 8. *Histogramas de nieve en diferentes ciudades*

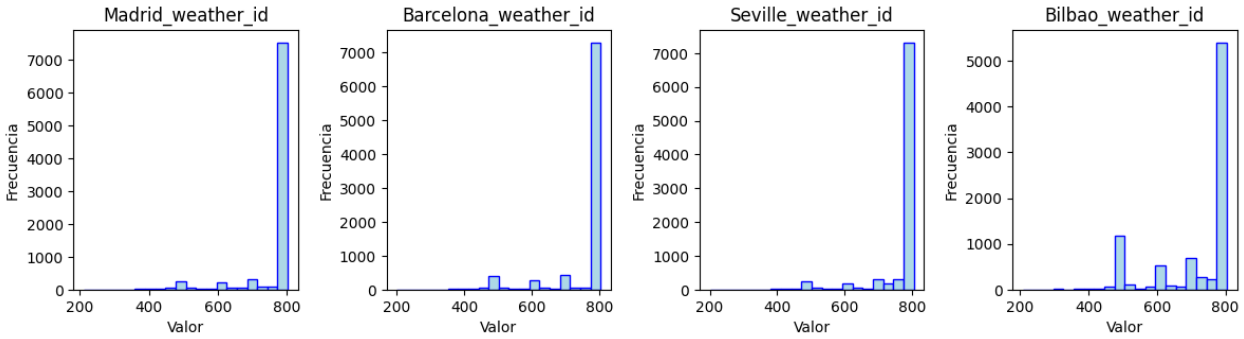


Figura 9. *Histogramas de condición climática en diferentes ciudades*

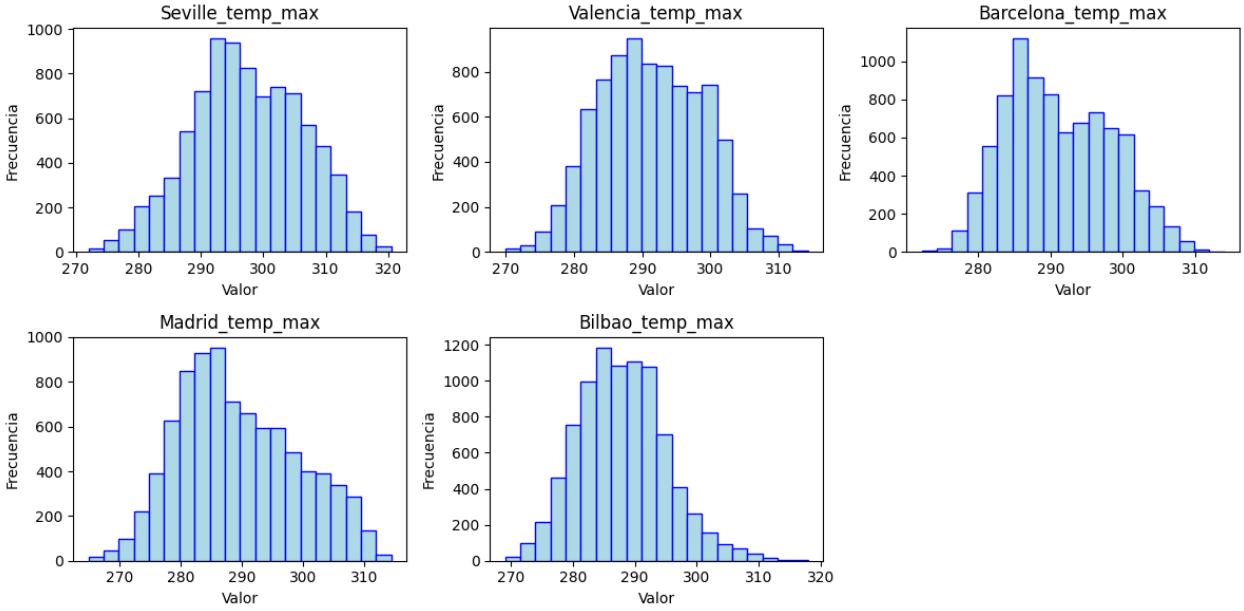


Figura 10. *Histogramas de temperatura máxima en diferentes ciudades*

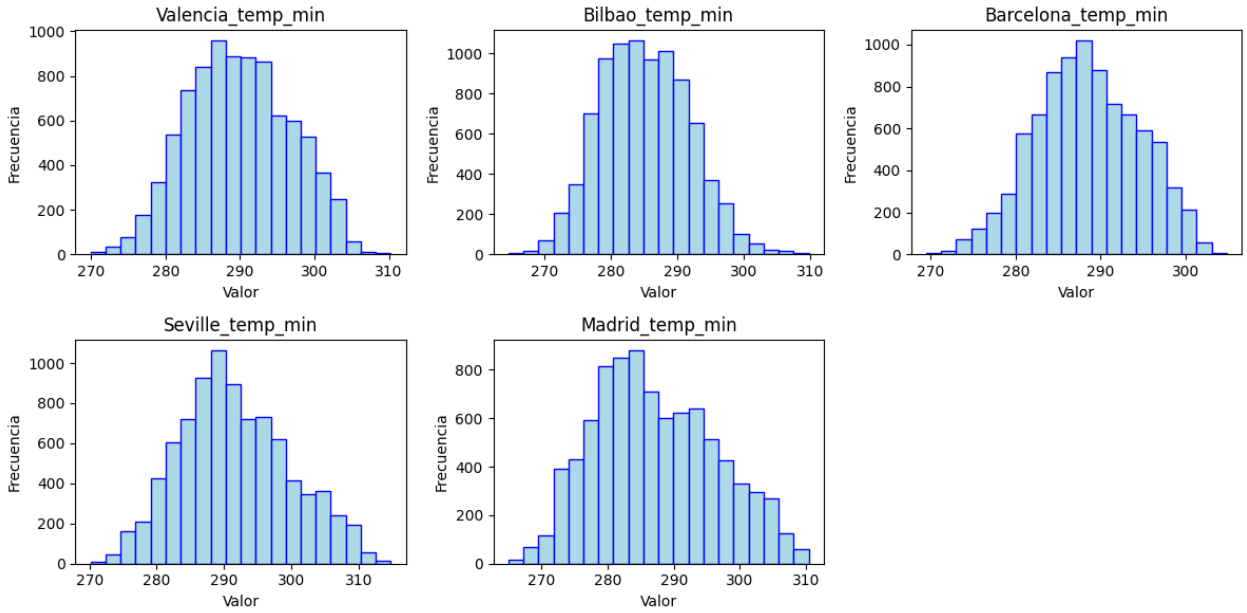


Figura 11. *Histogramas de temperatura mínima en diferentes ciudades*

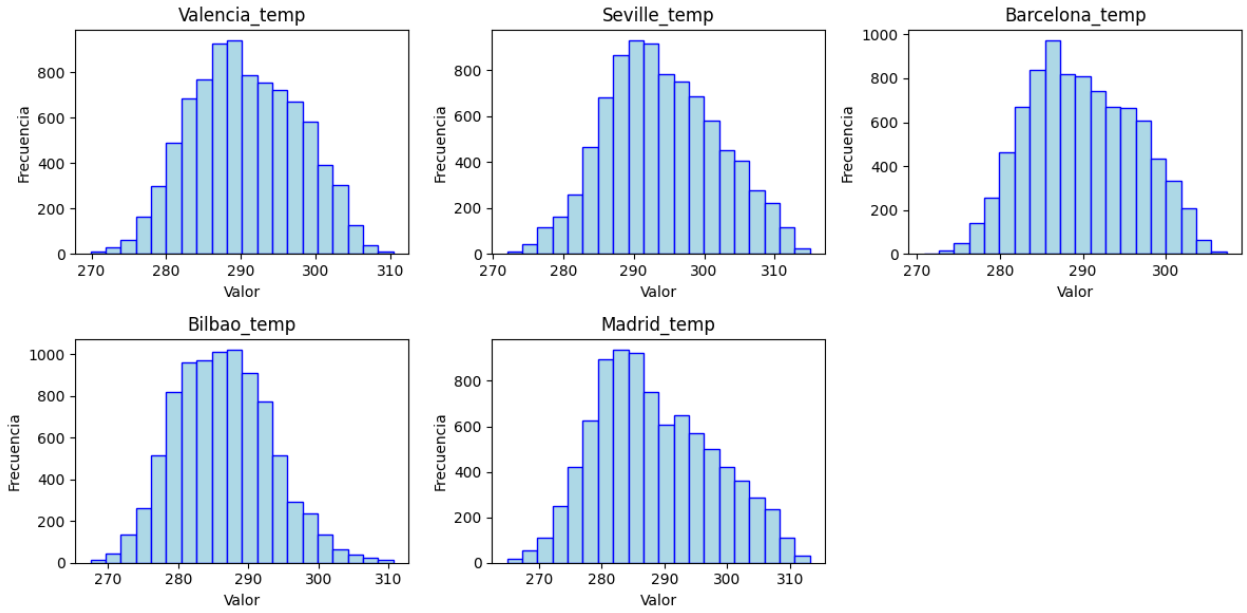


Figura 12. *Histogramas de temperatura promedio en diferentes ciudades*

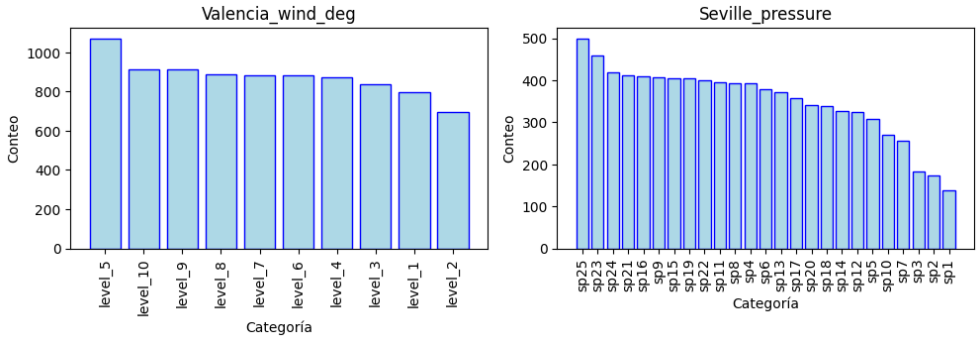


Figura 13. *Gráfico de barras de variables categóricas*

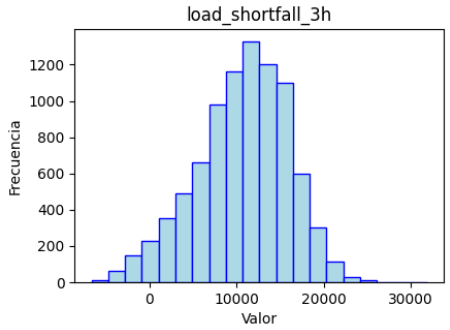


Figura 14. *Histograma de variable de salida*

Los gráficos de las figuras anteriores proporcionan una visión de la distribución de las variables numéricas en el conjunto de datos. La mayoría de estas variables siguen una distribución gaussiana, incluyendo la variable de salida. Sin embargo, las variables relacionadas con las condiciones meteorológicas, como la lluvia en tres horas y la nieve, muestran una acumulación de datos en el valor cero. Estas columnas tienen una varianza nula, lo que sugiere que podrían no aportar información relevante a los modelos que se pretenden desarrollar. La decisión de eliminar estas columnas se considerará en las iteraciones posteriores a medida que evolucionen los modelos.

Además, algunas otras características, como la lluvia en una hora y la nubosidad, presentan una gran concentración de registros en un valor único y una menor cantidad en un rango más amplio. Esto indica un desequilibrio en los datos de estas columnas. La importancia de estas caractersiticas, dependerá de cómo los modelos de aprendizaje automático respondan a ellas. Nuevamente, se evaluará la pertinencia de estas columnas como entrada para los modelos a medida que se avance en el proceso.

Por otro lado, se observa una concentración de datos alrededor del valor cero en la variable de presión en la ciudad de Barcelona. Sin embargo, al observar el rango de valores en el eje de la presión, se nota que estos llegan hasta 1 MhPa. Dado que los valores máximos de presión en otras ciudades son aproximadamente 1040 hPa y la máxima presión registrada en la Tierra es de aproximadamente 1084.8 hPa (World Meteorological Organization, 2012), es probable que esta variable requiera un tratamiento especial para abordar valores atípicos en iteraciones posteriores del análisis.

Respecto a las variables identificadas como categóricas, que incluyen la presión en Sevilla y el ángulo del viento en Valencia, surge una primera impresión: en otras ciudades, las mediciones de las mismas variables se expresan en valores numéricos. Esto plantea la necesidad de transformar estas variables de manera que se conviertan en magnitudes cuantificables que permitan asignar importancia a las categorías existentes hasta el momento. Este proceso se evaluará una vez que se inicie el análisis descriptivo.

Otra variable que ha requerido un análisis individual es "Weather_id" (Categoría del clima). Inicialmente, esta variable se identificó como numérica. Sin embargo, al revisar la definición de la variable en la sección 3.1, se sugiere que cada valor representa una categoría destinada a describir características específicas del clima en el momento de la toma del registro. No implica un orden numérico secuencial que denote una jerarquía o importancia entre las categorías (por ejemplo, que la categoría 300 sea inferior a la 800).

Tomando como base esta observación, se llevó a cabo una investigación para comprender mejor la interpretación de esta variable. De acuerdo con la información encontrada en el sitio web de OpenWeather (s.f.), las codificaciones utilizadas en la variable Weather_id en la base de datos corresponden a las siguientes:

Tabla 3.
Códigos para condiciones climáticas (Weather Identification Codes)

Condición	Grupo
Thunderstorm (Tormenta)	2XX
Drizzle (Llovizna)	3XX
Other (Categoría no descrita)	4XX
Rain (Lluvia)	5XX

Condición	Grupo
Snow (Nieve)	6XX
Atmosphere (Condición atmosférica)	7XX
Clear (Despejado)	800
Clouds (Nubosidad)	80X

Estas codificaciones se emplearán para llevar a cabo una transformación de la variable en categorías en etapas posteriores.

Adicionalmente, es de suma importancia comprender la covarianza existente entre las variables numéricas de entrada entre sí y con respecto a la variable de salida. En la **Figura *15*** se presenta la información usando un mapa de calor de covarianza:

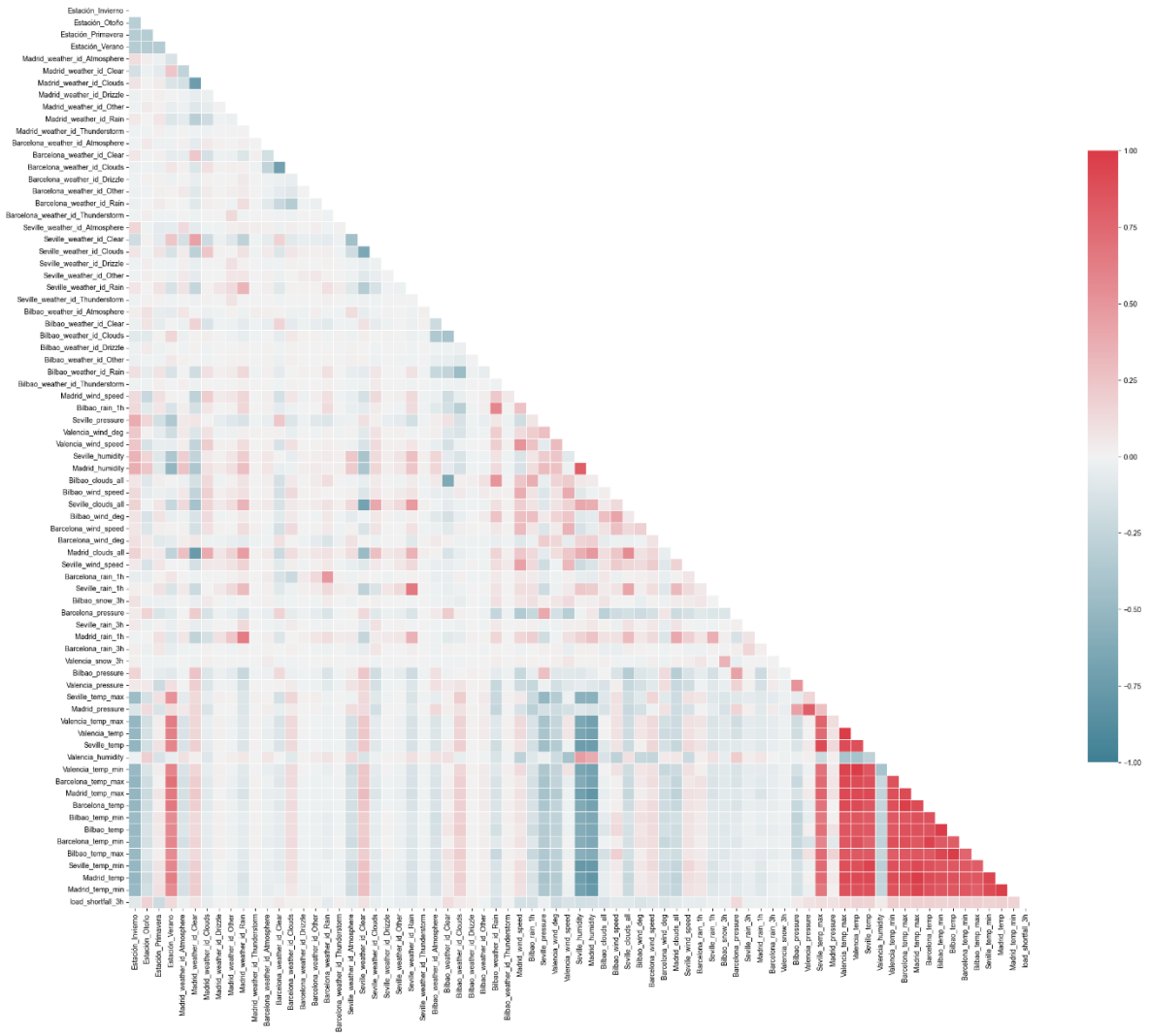


Figura 15. *Covarianza entre variables del modelo*

La **Figura *15*** revela tanto altas covarianzas positivas como negativas entre algunas de las variables de entrada en la base de datos, lo que sugiere la posibilidad de redundancia en la información proporcionada por estas variables. En etapas posteriores, se llevará a cabo un análisis exhaustivo para evaluar cómo la eliminación de dichas redundancias podría afectar el desempeño de los modelos.

Adicionalmente, se puede notar que la covarianza entre la variable de salida y las variables de entrada alcanza, en el mejor de los casos, un valor de 0,21. Esto anticipa que los cambios en las variables de entrada no están fuertemente relacionados con los cambios en la variable de salida, lo que indica una correlación lineal bastante baja entre las entradas y la variable objetivo.

Finalmente, se lleva a cabo un análisis de la distribución de la información mediante la creación de un gráfico de pares de puntos para las variables de entrada. El objetivo es identificar la distribución de los datos con respecto a la variable de salida, con el fin de verificar visualmente la relación entre ambos datos. Dicha información se presenta de la **Figura *16*** a la **Figura *22***.

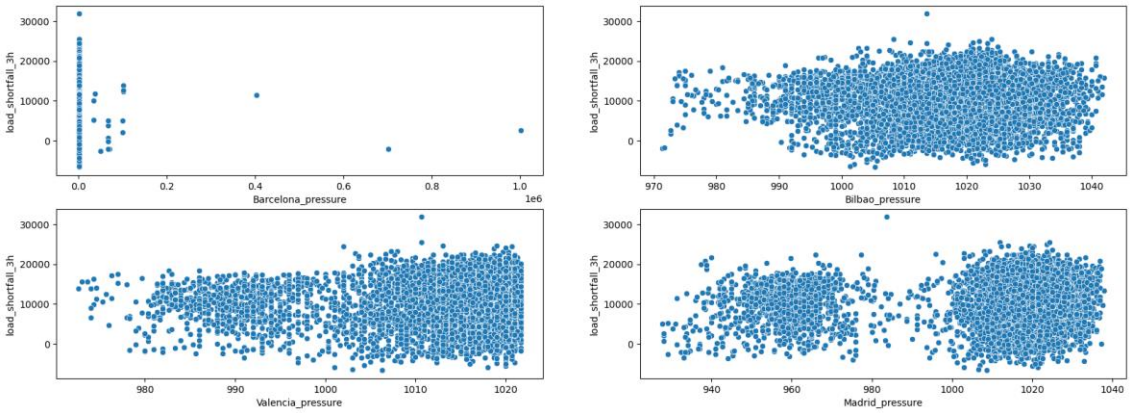


Figura 16. *Diagrama de pares de puntos para las variables de presión de la base de datos, con relación a la variable de salida.*

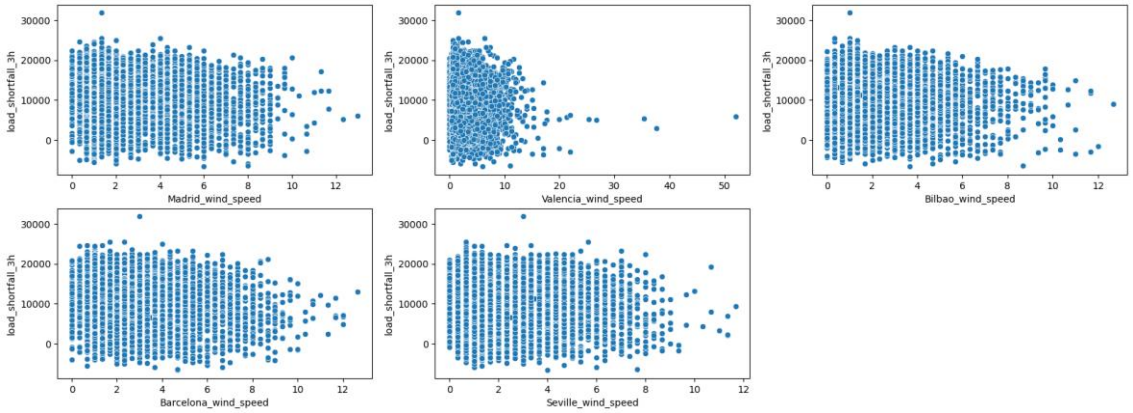


Figura 17. *Diagrama de pares de puntos para las variables de velocidad del viento de la base de datos, con relación a la variable de salida.*

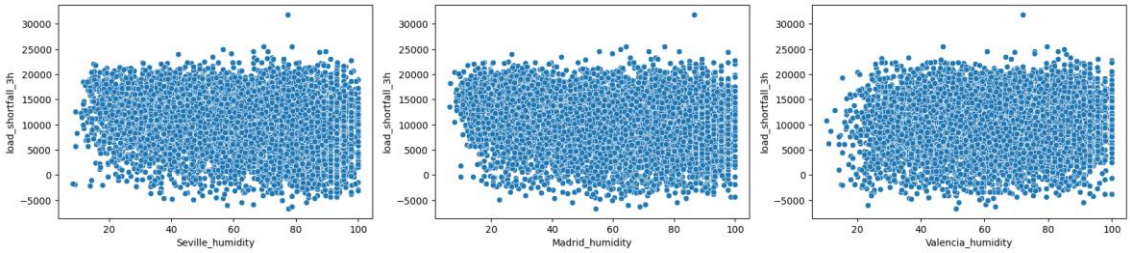


Figura 18. *Diagrama de pares de puntos para las variables de humedad de la base de datos, con relación a la variable de salida.*

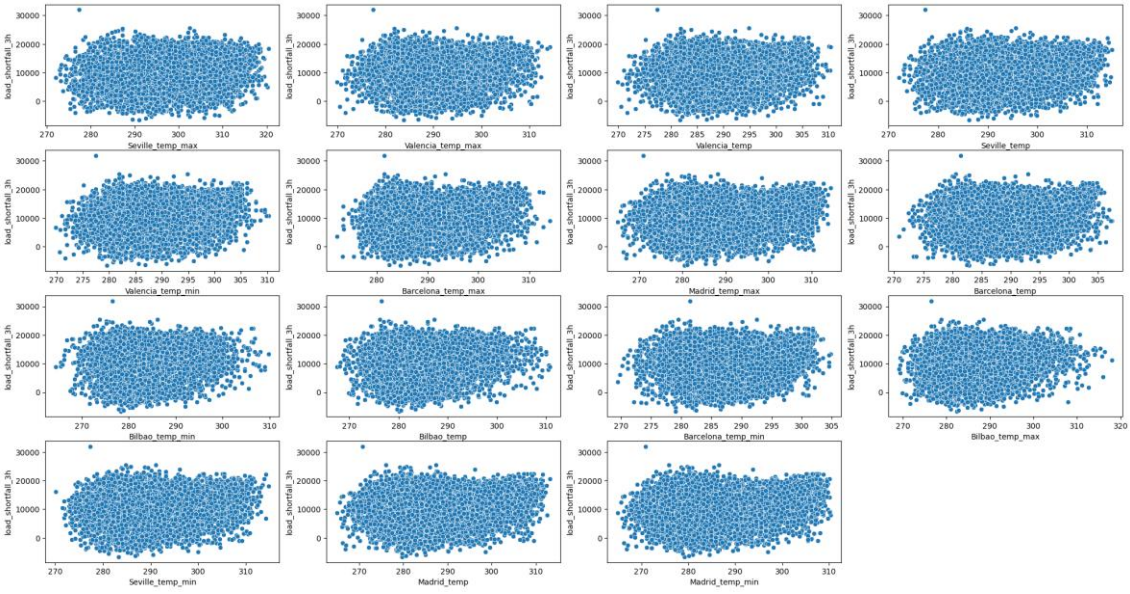


Figura 19. *Diagrama de pares de puntos para las variables de temperatura de la base de datos, con relación a la variable de salida.*

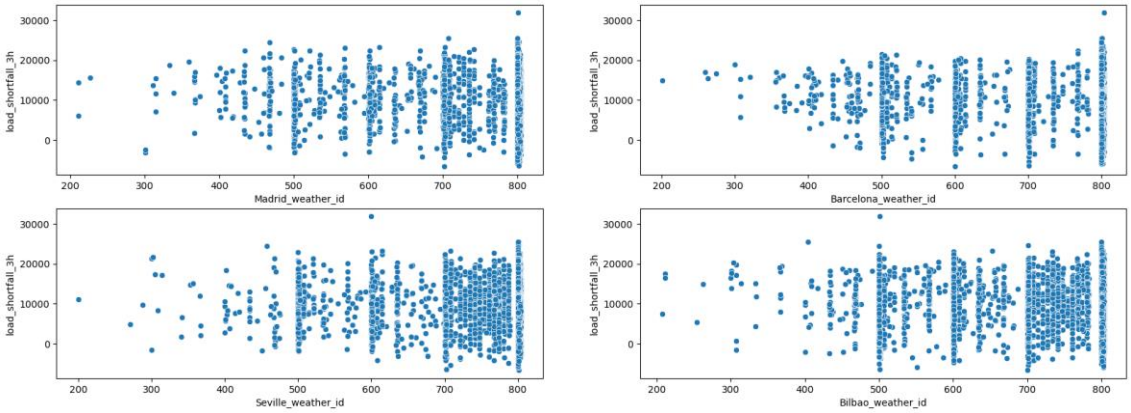


Figura 20. *Diagrama de pares de puntos para las variables de id del clima de la base de datos, con relación a la variable de salida.*

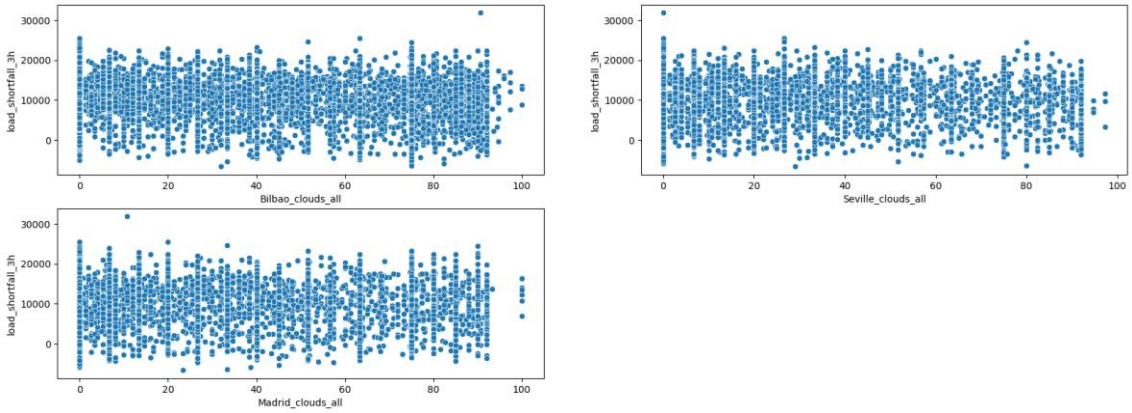


Figura 21. *Diagrama de pares de puntos para las variables nubes de la base de datos, con relación a la variable de salida.*

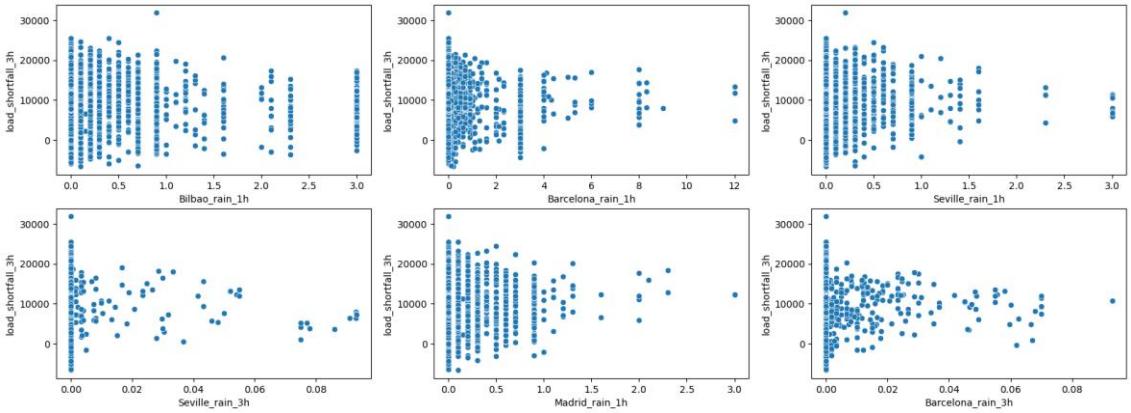


Figura 22. *Diagrama de pares de puntos para las variables lluvia de la base de datos, con relación a la variable de salida.*

De las figuras anteriores se evidenció que ninguna de las características numéricas del conjunto de datos presentan una relación lineal con respecto a la variable de salida, esto permite establecer una segunda evidencia de que los datos se comportan como un problema no lineal.

Es fundamental destacar que la falta de una relación lineal no necesariamente implica que las variables de entrada no sean útiles para predecir la variable de salida. En algunos casos, las relaciones pueden ser no lineales o estar enmascaradas, y modelos más avanzados podrían capturar eficazmente estas relaciones. Por lo tanto, es crucial llevar a cabo un análisis exhaustivo y considerar múltiples enfoques antes de tomar decisiones basadas exclusivamente en la covarianza o la correlación lineal.

4. Proceso de analítica

4.1.Pipeline principal

La analítica de datos implica la ejecución de varios pasos y procesos. Estos abarcan desde la identificación de las fuentes de información, la ingesta de datos, su almacenamiento, hasta la exploración, tratamiento y transformación de los datos. Este enfoque se realiza con el propósito de dotar a la información de las mejores características para abordar cada problema planteado y facilitar su aplicación.

En la **Figura *23***, se presenta un diagrama de flujo básico que ilustra un proceso de analítica centrado en la creación y utilización de algoritmos de aprendizaje de máquina.

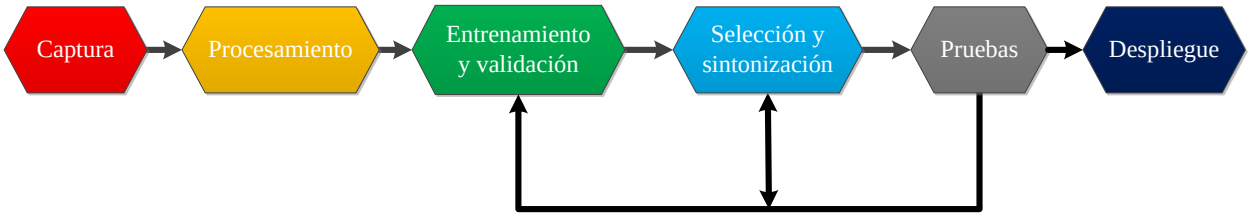


Figura 23. *Vista general del Pipeline principal*

La estructura del flujo de trabajo para cualquier aplicación requiere de una expansión, que evidencie cada uno de los subprocesos utilizados en las actividades generales de procesamiento, modelamiento, selección, validación, pruebas y despliegue, requeridos para los diferentes algoritmos de aprendizaje automático a implementar. A continuación, se describen los procesos

del *pipeline* principal utilizados para *la predicción de la desatención de demanda en España debido a la transición energética*:

- **Captura y adquisición:** consiste en la selección de la fuente desde la cual se pretende extraer la información, así como los métodos empleados para obtener los datos de dicha fuente, ya sea mediante procesos manuales o en línea mediante una serie de microservicios o algoritmos que facilitan la recolección automática de información.

  En el caso del conjunto de datos utilizados para prever la desatención de la demanda, la obtención se llevó a cabo de manera manual mediante la descarga directa de los archivos disponibles en el repositorio del desafío Kaggle *EDSA Individual | Electricity Shortfall Challenge* (ExploreAI Academy, 2022), tal como se describe en el capítulo 1.
- **Exploración de datos:** este constituye el paso inicial en cualquier análisis de datos, fundamentándose en el empleo de herramientas de visualización, análisis descriptivo y técnicas estadísticas. Estas se centran en obtener una descripción integral de la información, abarcando aspectos como la dimensión del conjunto de datos, su volumen, distribución, comportamiento de las características, relaciones entre variables, estructura de la información, así como la detección de datos atípicos y/o faltantes.

  La exploración de los datos se llevó a cabo mediante el uso del lenguaje de programación Python y una serie de módulos especializados en el manejo de la información (Pandas), la visualización de datos (Matplotlib y Seaborn) y el tratamiento de la información (Scikit-learn). Para obtener información más detallada sobre las herramientas empleadas, se remite a la sección 5.1.
- **Tratamiento de los datos:** en este proceso se busca modificar aquellos datos que suponen un inconveniente para los procesos posteriores en el flujo de trabajo, como lo pueden ser los datos faltantes, los datos nulos, datos atípicos, problemas de escala, colinealidades y características de baja importancia, entre otros. También se buscan implementar técnicas que permitan mejorar la información de entrada, de tal forma que estas resulten más beneficiosas para los modelos, como la extracción de características, la normalización y/o estandarización, el aumento de información, las transformaciones de características, entre otras. El tratamiento de los datos utilizó como base el lenguaje de programación Python y

el módulo de análisis de datos (Scikit-learn), en la sección 5.1 se presenta una mayor información de las herramientas utilizadas.

- **Creación y validación de modelos:** este proceso consiste en la creación de diferentes modelos de aprendizaje de máquina, para los cuales se valida su arquitectura, sus parámetros disponibles, las funciones de pérdida (métricas) disponibles y los métodos de optimización enfocados en la minimización de la pérdida, donde posteriormente se evalúa su predicción, mediante los resultados obtenidos a partir del análisis de un número variados de métricas de desempeño. Para la creación de los modelos se utilizaron librerías como Scikit-learn, lightgbm, Scikit-ELM, Tensorflow y Keras disponibles para el lenguaje de programación Python, en la sección 5.1 se presenta una mayor información de las herramientas utilizadas.
- **Evaluación de métricas:** este proceso consiste en el análisis y revisión de los resultados de las diferentes métricas para cada uno de los modelos implementados, a través de cada una de las iteraciones realizadas. Para este proceso se hizo uso de los módulos Pandas para almacenar los resultados de las métricas y los módulos utilizados en el proceso anterior de creación y validación de modelos.
- **Selección de mejores modelos:** en este proceso se seleccionan los modelos que mejor predicen la desatención de la demanda, basados en los resultados de las métricas de desempeño.
- ***Hipertuning*:** también conocido como sintonización, este proceso busca mejorar el desempeño de los modelos elegidos, mediante una serie de técnicas de optimización de la función de perdida elegida (métrica), a través de la selección de los mejores parámetros disponibles en el modelo. Para este proceso se exploraron las estrategias de la búsqueda aleatoria (*Randomized Search*), la búsqueda de rejilla (*Grid Search*) y la búsqueda bayesiana (*Bayesian Search*), disponibles en las librerías hyperopt, Scikit-learn, Tensorflow y Keras, para el lenguaje de programación Python, en la sección 5.1 se presenta una mayor información de las herramientas utilizadas.
- **Despliegue o modelos finales:** se obtienen los mejores 3 modelos para la predicción de la desatención de la demanda.

En la **Figura *24*** se presenta el *Pipeline* principal detallado para el proceso de analítica implementado.

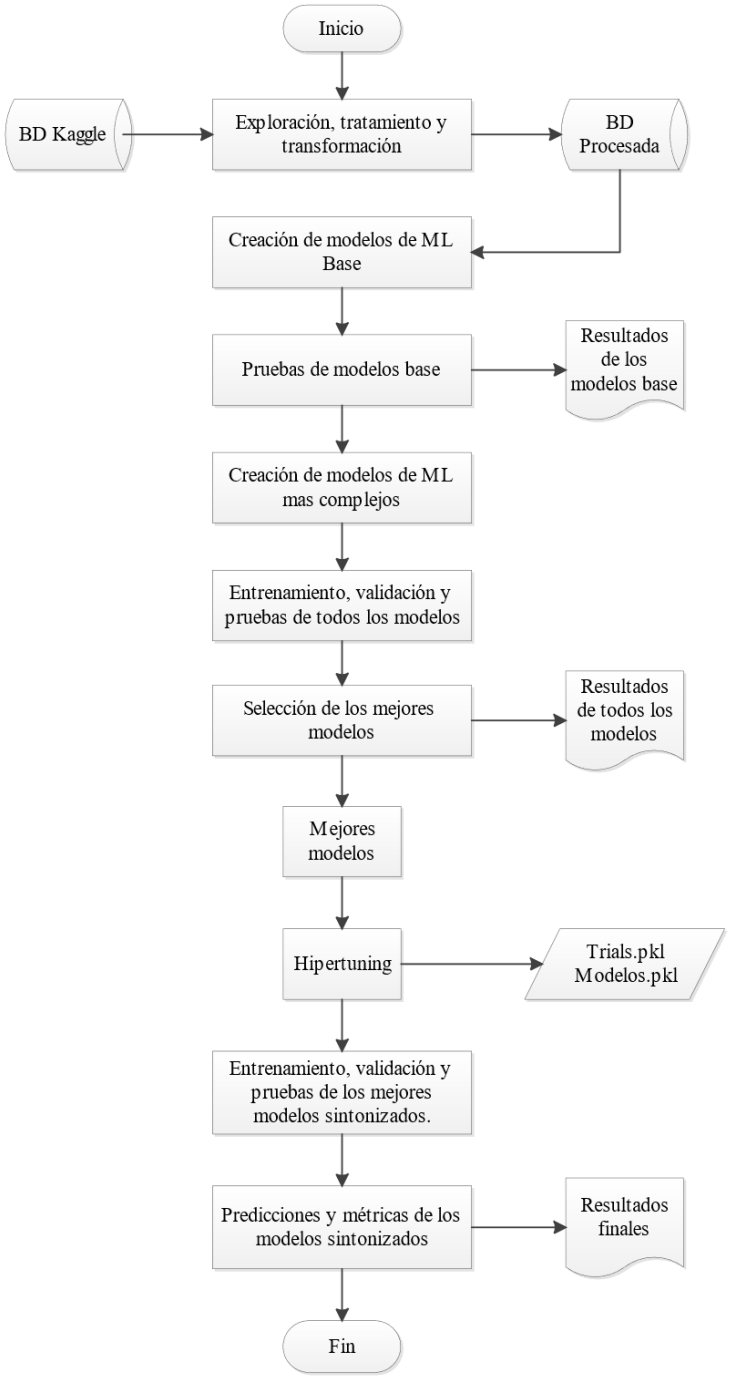


Figura 24. *Flujo de trabajo detallado*

4.2.Subprocesos

Durante los diferentes procesos desarrollados en el *Pipeline* principal, se realizaron una serie de subprocesos de gran importancia para todo el flujo de trabajo, esto debido a los diferentes hallazgos encontrados en la Analítica descriptiva, a las estrategias implementadas en el tratamiento de la información y en su impacto en los resultados de las métricas de los modelos. Estos subprocesos se detallan a continuación:

- **Transformación de variables categóricas:** en este proceso se modificaron las variables que inicialmente se representaban mediante clases (categorías), a variables de carácter numérico, debido a la naturaleza que representa la variable y el contexto de la información presente en la base de datos, como por ejemplo características de presión, temperatura, y velocidad.
- **Tratamiento de datos faltantes:** este proceso consiste en el reemplazo de los datos faltantes de la base de datos, mediante registros que cumplan con un criterio de similitud o relación con respecto a la característica que pertenecen. En este subproceso se aplicó la siguiente técnica:
    - **KNN imputer:** este algoritmo permite reemplazar los datos faltantes con valores estimados del mismo conjunto de datos, mediante la valoración de su proximidad, coincidencia y/o la importancia de sus datos vecinos.
- **Tratamiento de atípicos puntuales:** en este proceso se buscó eliminar los valores atípicos en una sola característica de la base de datos que incumplía un criterio de verosimilitud en la misma información, estos criterios de verosimilitud tienen en consideración la propia característica, los datos que la representan y las limitaciones reales de la propia variable, como ejemplo, lecturas de 0 o varios amperios dados por sensores 4-20mA, temperaturas de miles de grados centígrados en las ciudades, o presiones por encima de cualquier máximo histórico registrado; de tal forma que los algoritmos de tratamiento de datos atípicos y los mismos modelos de aprendizaje de maquina no deban lidiar con estos valores, mejorando así su rendimiento.
- **Tratamiento de datos atípicos:** en este proceso se busca eliminar los datos atípicos de la base de datos, considerando todas las características presentes en la base de datos. En este subproceso se aplicaron las siguientes técnicas:

- **Test Z:** este algoritmo permite detectar los datos atípicos mediante el análisis de la distribución de los datos, se obtiene la media y la desviación estándar para luego crear una distribución Z, los datos que se encuentren por fuera del umbral Z definido son catalogados como datos atípicos.
- **LOF:** este algoritmo permite detectar los datos atípicos mediante un análisis de densidad del conjunto de datos, para los cuales se calcula una puntuación que indica que tan inusual es el dato con respecto a sus vecinos. Una puntuación alta indica que la muestra es menos densa en su localidad en comparación con sus vecinos. Los datos que se encuentren por fuera del umbral definido son catalogados como datos atípicos.

- **Reducción de dimensionalidad:** en este proceso se busca reducir la cantidad de características a utilizar como información de entrada para la creación de los modelos de aprendizaje de máquina, descartando aquellas características que presenten alta colinealidad, correlación o un menor aporte en la representación total de los datos (varianza explicada). En este subproceso se aplicaron las siguientes técnicas:
  - **Correlaciones:** en este proceso se busca identificar las variables más correlacionadas entre sí y su correlación con la variable objetivo. Se reducen las características que cuenten con un valor de correlación mayor al umbral seleccionado. Se analiza la correlación de las variables de entrada con respecto a la de salida, para determinar la importancia de cada una de ellas (en este análisis no se elimina ninguna característica)
  - **Análisis de componentes principales (PCA):** este algoritmo se centra en reducir las dimensiones del conjunto de datos, mediante la selección de las componentes principales de la matriz de covarianza, en donde se seleccionan las componentes que explican la mayor varianza del conjunto de datos.
- **Estacionalidad:** A pesar de que se ha determinado que el problema actual no sigue una estructura de serie temporal, es fundamental tener en cuenta la estacionalidad asociada con el contexto en el que se están llevando a cabo el presente análisis. En el caso de España, el país experimenta cambios estacionales claramente definidos a lo largo del año, que se manifiestan en las cuatro estaciones: primavera, verano, otoño e invierno.

Esta información acerca de las estaciones puede ser un recurso valioso a considerar en la construcción de modelos predictivos y durante las iteraciones subsiguientes. Las estaciones del año pueden influir en una amplia gama de fenómenos y variables, como el clima (disponibilidad del recurso renovable), el comportamiento del consumidor (demanda), y por ende, estar directamente ligada al déficit energético que se busca predecir. Por lo tanto, al incorporar esta estacionalidad en los modelos, se puede potencialmente mejorar la precisión de las predicciones.

- **Hypertuning:** Se exploraron 3 estrategias para buscar los mejores parámetros de los diferentes modelos de Machine Learning implementados para la predicción, con el objetivo de minimizar la función de pérdida del modelo y por ende obtener un mejor desempeño y/o ajuste del modelo con respecto a la variable de prueba/salida. Los parámetros son los atributos configurables con los que cuentan cada uno de los algoritmos de Machine Learning que no son obtenidos de las bases de datos ni de los procesos de entrenamiento. Las estrategias exploradas fueron las siguientes técnicas:
    - **Búsqueda de rejilla (*Grid Search*):** Este algoritmo se basa en generar una serie hiperparámetros a evaluar en el modelo, el algoritmo creara, entrenara y evaluara el modelo con todas las posibles combinaciones de hiperparámetros, por lo cual es considerado como un algoritmo de fuerza bruta por la búsqueda exhaustiva que se realiza (Yang A. & Shami, A, 2020). Posteriormente se selecciona el modelo y los hiperparámetros que obtuvieron el mejor resultado de las métricas de desempeño.
    - **Búsqueda aleatoria (*Randomized Search*):** Este algoritmo cuenta con características similares al algoritmo de búsqueda de rejilla, para el cual se dispone de un espacio de posibles hiperparámetros a seleccionar para el modelo, los hiperparámetros se seleccionan de manera aleatoria para cada iteración hasta que el número de iteraciones definidas sean completadas. Posteriormente se selecciona el modelo y los hiperparámetros que obtuvieron el mejor resultado de las métricas de desempeño. Este algoritmo presenta una mejora significativa en rendimiento con respeto al algoritmo de búsqueda de rejilla cuando se evalúan espacios de hiperparámetros de grandes dimensiones, sin embargo, requiere de un gran número de iteraciones para poder encontrar mínimos globales y se presenta una pérdida de

tiempo en la evaluación de regiones de bajo rendimiento (Yang A. & Shami, A, 2020).

- **Búsqueda bayesiana (*Bayesian Search*):** Este algoritmo se basa en generar un espacio de variadas características que conformara la región de la cual seleccionar los hiperparámetros para un modelo, cada parámetro puede tener un espacio uniformemente distribuido, un espacio de valores discretos, o una serie de valores fijos seleccionables. El algoritmo seleccionara de manera aleatoria un primer conjunto de hiperparámetros con los que creara, entrenara y evaluara el modelo, la selección del próximo conjunto de hiperparámetros considera el resultado de la anterior evaluación, proceso que se realiza mediante dos factores fundamentales, una función de adquisición y un modelo probabilístico sustituto, el modelo sustituto ajustara los valores observados en la función objetivo, para obtener la distribución predictiva del modelo sustituto, con lo cual inferir la región en la cual es más probable la obtención del optimo global (Yang A. & Shami, A, 2020).

En la **Figura *25*** se presentan tres gráficos a través de los cuales se puede visualizar el funcionamiento de las tres técnicas de hypertuning mencionadas anteriormente.

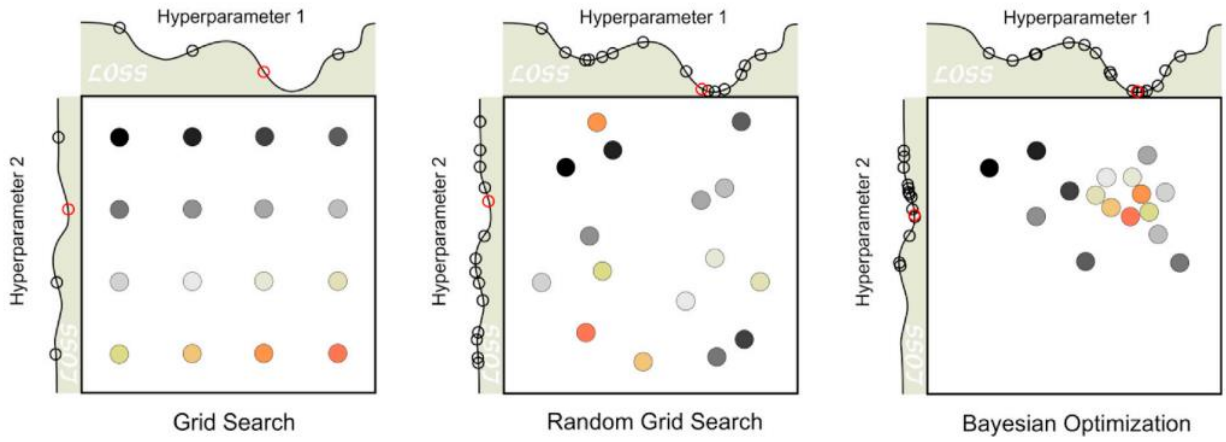


Figura 25. *Funcionamiento de los algoritmos de búsqueda de hiperparámetros de un modelo de Machine Learning. (Cortinhas S, 2022)*

- **Normalización:** Se refiere a la transformación de las características o variables de entrada (atributos) de un conjunto de datos para que tengan una escala y una distribución comunes. El objetivo principal de la normalización es mejorar el rendimiento y la convergencia de

los algoritmos de aprendizaje automático, especialmente aquellos que son sensibles a las diferencias de escala entre las características. Para el presente desarrollo se emplearon las siguientes técnicas:

- **MinMax:** es una técnica de transformación de datos utilizada en el preprocesamiento de datos en Machine Learning para escalar y ajustar los valores de una característica o variable en un rango específico. Esta técnica transforma los valores de una característica de manera que queden dentro de un intervalo predeterminado, generalmente entre 0 y 1, aunque el rango puede ser personalizado según las necesidades del problema. La normalización Min-Max es especialmente útil cuando se desea que todas las características tengan la misma escala y no se quiere perder la relación proporcional entre los valores originales.

$$x_{\text{scaled}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- **Standard Scaler:** La normalización estándar, también conocida como estandarización, es una técnica de transformación de datos utilizada en el preprocesamiento de datos en Machine Learning para escalar las características de manera que tengan una media igual a cero y una desviación estándar igual a uno. Esta técnica es particularmente útil cuando se trabaja con algoritmos sensibles a las diferencias de escala entre las características y cuando se desea que los datos tengan una distribución normal estándar.

$$z = \frac{x_i - \mu}{\sigma}$$

4.3.Modelos

En las siguientes secciones, se describen los diferentes modelos de aprendizaje automático empleados como parte del desarrollo del presente documento:

4.3.1. OLS

El modelo OLS (*Ordinary Least Squares*) es un enfoque comúnmente utilizado en regresión lineal para ajustar una línea recta a los datos con el objetivo de minimizar la suma de los cuadrados de las diferencias entre los valores observados y los valores predichos. La idea principal detrás del

modelo OLS es encontrar los coeficientes que mejor se ajusten a los datos y minimicen el error cuadrático.

El modelo OLS en Scikit-Learn se implementa a través de la clase LinearRegression y, por lo general, no requiere una afinación de hiperparámetros compleja, ya que el objetivo principal es ajustar la mejor línea recta a los datos en función de los coeficientes de regresión y el término de intercepción.

4.3.2. Random Forest Regressor

El modelo Random Forest Regressor es un algoritmo de aprendizaje automático basado en ensambles que se utiliza para tareas de regresión. Utiliza múltiples árboles de decisión para hacer predicciones y promedia sus resultados para obtener una estimación más precisa.

Cada árbol se ramifica en diferentes direcciones hasta llegar a su punto final, conocido como "hoja". La hoja representa la predicción final del árbol de decisión. Cada nodo de decisión se determina según si una de las características de entrada supera un valor específico (Keller y Evans, 2019).

Los principales hiperparámetros en la librería Scikit-Learn que afectan las predicciones son:

- **n_estimators (int):** Indica el número de árboles que se utilizarán en el ensamble.
- **max_depth (int o None):** Controla la profundidad máxima de cada árbol.
- **min_samples_split (int o float):** Representa el número mínimo de muestras requeridas para dividir un nodo interno del árbol.
- **min_samples_leaf (int o float):** Es el número mínimo de muestras que debe haber en una hoja del árbol.
- **max_features (int, float o {"auto", "sqrt", "log2"}):** Controla el número máximo de características a considerar en cada división de nodo.

4.3.3. Support Vector Regressor

El problema de regresión se puede ver como una extensión del problema de clasificación, en el que el modelo proporciona una salida continua en lugar de una salida que pertenece a un conjunto discreto. En términos simples, un modelo de regresión se utiliza para estimar una función multivariada con valores continuos. Las SVM pueden ser especialmente adecuadas para resolver

problemas de regresión (SVR) gracias a su capacidad para encontrar soluciones dispersas y su habilidad para generalizar eficazmente (Awad y Khana, 2015).

Este algoritmo de aprendizaje automático para regresión utiliza vectores de soporte para encontrar una función que se ajuste a los datos de manera óptima. Funciona tratando de encontrar una línea (o superficie en dimensiones más altas) que maximice el margen entre los puntos de datos y la función de regresión. El SVR es efectivo para manejar relaciones no lineales y puede proporcionar regresiones precisas en datos complejos.

Los principales hiperparámetros en la librería Scikit-Learn que afectan las predicciones son:

- **kernel (str):** Especifica el tipo de kernel utilizado en el modelo, como "linear" para regresión lineal, "rbf" para una función de base radial, o "poly" para un kernel polinómico.
- **C (float):** Controla el término de regularización. Un valor más bajo permitirá un mayor margen de error, mientras que un valor más alto penalizará fuertemente las violaciones del margen.
- **epsilon (float):** Establece la tolerancia para los errores, lo que significa que las predicciones dentro de un margen de epsilon se consideran aceptables y no contribuyen a la función de costo.
- **degree (int):** Aplicable solo para kernels polinómicos. Indica el grado del polinomio a utilizar.
- **gamma (float o "scale" o "auto"):** Controla la amplitud del kernel rbf. Un valor más bajo da un efecto más suave, mientras que un valor más alto produce un efecto más puntiagudo.

4.3.4. Decisión Tree

Los Árboles de Decisión son algoritmos de aprendizaje automático que dividen los datos en ramas basadas en preguntas sobre las características de entrada, hasta llegar a una predicción. Los hiperparámetros clave en Scikit-Learn son:

- **max_depth (int):** Controla la profundidad máxima del árbol.
- **min_samples_split (int):** Establece el número mínimo de muestras requeridas para dividir un nodo.
- **min_samples_leaf (int):** Define el número mínimo de muestras en una hoja.

- **max_features (int):** Controla el número máximo de características a considerar en cada división.

4.3.5. Extreme Learning Machine

Extreme Learning Machine (ELM) es un algoritmo de aprendizaje automático diseñado para redes neuronales de una sola capa oculta (SLFN). Se destaca por su rápida convergencia en comparación con los enfoques tradicionales y ha demostrado un rendimiento considerable (Wang et al., 2022).

En Scikit-elm, los principales hiperparámetros que afectan las predicciones en ELM son:

- **n_hidden (int):** Controla el número de neuronas en la capa oculta.
- **activation (str):** Especifica la función de activación utilizada en la capa oculta, como "sigmoid" o "relu."
- **alpha (float):** Controla la regularización para evitar el sobreajuste.
- **random_state (int o None):** Establece una semilla aleatoria para reproducibilidad.

4.3.6. Extra Trees Regressor

Similar a los algoritmos de árboles aleatorios (Random Forest), el algoritmo Extra Trees Regressor (ETR) consiste en un algoritmo de ensamblaje de un variado número de árboles de decisión, los cuales son posteriormente agrupados para generar una predicción. Su mayor diferencia con respecto a los modelos de Random Forest, consiste en el uso de todas las muestras para entrenar los distintos arboles de decisión a diferencia del Random Forest que utiliza solamente una porción (Bagging); y en la selección de manera aleatoria de los valores para los cuales una característica es dividida para generar nuevos nodos. Este tipo de modelos, suelen tener un menor costo computacional con respecto a los algoritmos de ensamblaje similares (Camana M, et al., 2020).

Los hiperparámetros clave en Scikit-Learn son:

- **n_estimators (int):** Indica el número de árboles que se utilizarán en el ensamble.
- **criterion:** Función con la cual se mide la calidad de cada división de los nodos.
- **min_samples_split (int o float):** Representa el número mínimo de muestras requeridas para dividir un nodo interno del árbol.

- **min_samples_leaf (int o float):** Es el número mínimo de muestras que debe haber en una hoja del árbol.

4.3.7. Gradient Boosting Machine

Gradient Boosting Machine (GBM) es un algoritmo de ensamblaje de amplia implementación para la resolución de problemas de clasificación y regresión, se fundamenta en el uso de predictores débiles que realicen pocas predicciones de la información, como los árboles de decisión o los regresores lineales, los cuales se irán añadiendo al modelo para crear predictores más robustos que minimicen la función objetivo. En cada iteración se computa el gradiente descendiente de la función de pérdida, con este se ajusta un nuevo predictor débil y se actualiza todo el modelo, a medida que se añaden mejores predictores se espera que se minimice el error general de la predicción. (Konstantinov, A & Utkin L, 2021)

Los hiperparámetros clave en Scikit-Learn son:

- **loss:** La función objetivo a ser optimizada.
- **learning_rate (float):** Tasa de aprendizaje con la cual se disminuye la contribución de cada predictor base. Presenta una situación de ganancia-pérdida con el parámetro n_estimators
- **n_estimators (int):** Indica el número de etapas de impulso a realizar.
- **criterion:** Función con la cual se mide la calidad de cada división.
- **max_features:** Indica el número de características a considerar cuando se busca la mejor división

4.3.8. Light Gradient Boosting Machine

Light Gradient Boosting Machine (LightGBM) es un algoritmo basado en el aumento o impulso de los gradientes que utiliza como predictor base el árbol de decisión, el cual crece de manera vertical mediante un algoritmo de división de hojas (leaf-wise splitting), la hoja que reduzca en mayor medida la función de pérdida es elegida para generar una división, a su vez, implementa un algoritmo de muestreo basado en gradientes de un solo lado (Gradient-based One-Side Sampling - GOSS), el cual computa solamente las muestras de la información con mayor gradiente e ignora aquellas con gradientes menos significativos (Alzamzami F et al., 2020). LightGBM es rápido y eficiente, especialmente en conjuntos de datos grandes (Taha A & Malebary S, 2020).

En LightGBM, los principales hiperparámetros que afectan las predicciones en LightGBM son:

- **boosting_type:** Tipo de predictores base a utilizar.
- **num_leaves (float):** máximo número de hojas para los predictores base.
- **max_depth (int):** Profundidad máxima de los árboles.
- **learning_rate (float):** Tasa de aprendizaje con la cual se disminuye la contribución de cada predictor base. Presenta una situación de ganancia-pérdida con el parámetro n_estimators.
- **n_estimators (int):** Indica el número de etapas de impulso a realizar.
- **reg_alpha (float):** Termino regularizador de los pesos – L1
- **reg_lambda (float):** Termino regularizador de los pesos – L2

4.3.9. Feed-Forward Neural Network

Una red neuronal es un modelo computacional inspirado en el funcionamiento del cerebro humano. Es un sistema que se utiliza para resolver problemas y tomar decisiones al aprender de los datos. Está compuesto por "neuronas artificiales" interconectadas que procesan información y la utilizan para realizar tareas como clasificación, predicción o reconocimiento.

En el campo de las redes neuronales, hay dos categorías fundamentales de arquitecturas que dependen del tipo de conexiones entre las neuronas, estas son conocidas como "redes neuronales artificial" y "redes neuronales recurrentes". La distinción clave radica en la presencia o ausencia de "retroalimentación" desde las salidas de las neuronas hacia las entradas a lo largo de la red. Cuando no hay dicho proceso de "retroalimentación", la red se clasifica como una "red neuronal artificial" (Sazli, 2006). Para los objetivos del presente desarrollo, se empleará principalmente las "redes neuronales prealimentadas".

La red neuronal artificial (ANN por sus siglas en inglés “Artificial Neural Network”) fue la primera y más sencilla forma de red neuronal artificial ideada. En esta red, la información se mueve en una única dirección: adelante. De los nodos de entrada, a través de los nodos escondidos (si los hay) hacia los nodos de salida. No hay ningún ciclo o bucle en estas redes.

- **Número de capas y neuronas:** Controla la arquitectura de la red, incluyendo el número de capas ocultas y neuronas en cada capa.

- **Funciones de activación:** Define las funciones (como ReLU o sigmoide) que determinan la salida de cada neurona.
- **Tasa de aprendizaje (learning rate):** Ajusta la velocidad de actualización de los pesos de la red durante el entrenamiento.
- **Número de épocas (epochs):** Indica cuántas veces la red pasa por el conjunto de entrenamiento durante el entrenamiento.
- **Batch size:** Especifica el número de ejemplos de entrenamiento en cada lote para actualizar los pesos.
- **Regularización:** Controla la penalización de pesos para evitar el sobreajuste, como L1 o L2.
- **Dropout:** Establece la proporción de neuronas que se desactivan aleatoriamente durante el entrenamiento para reducir el sobreajuste.
- **Función de pérdida (loss function):** Define la métrica utilizada para medir la diferencia entre las predicciones y los valores reales.
- **Optimizador:** Determina el algoritmo de optimización utilizado para ajustar los pesos de la red, como Adam o SGD.
- **Activación de salida:** Especifica la función de activación en la capa de salida, dependiendo del tipo de problema (regresión, clasificación, etc.).

4.4.Métricas

Para cuantificar la calidad de las predicciones hechas por los modelos, se han empleado métricas apropiadas para los diferentes modelos implementados y la información asociada a la variable a predecir. En general, se emplean las métricas contenidas en el módulo “metrics” de la librería scikit-learn.

A continuación, se describen las mencionan las métricas empleadas y las funciones de scikit-learn usadas para cada una de ellas:

- **R2:** sklearn.metrics.r2_score
- **MAE:** sklearn.metrics.mean_absolute_error
- **RMSE**: sklearn.metrics.mean_squared_error(squared=False)
- **MAPE:** sklearn.metrics.mean_absolute_percentage_error

5. Metodología

5.1.Herramientas

Python

Python es un lenguaje de programación versátil, de alto nivel e interpretabilidad que se destaca por su legibilidad y facilidad de uso. En palabras de Van Rossum (2003):

> La sintaxis elegante de Python y su tipificación dinámica, junto con su naturaleza interpretable, lo convierten en un lenguaje ideal para la creación de scripts y el desarrollo rápido de aplicaciones en muchas áreas y en la mayoría de las plataformas. (p.3)

En el ámbito de la ciencia y la analítica de datos, Python se ha convertido en un lenguaje esencial. Bibliotecas como NumPy y Pandas facilitan la manipulación y el análisis de datos, mientras que matplotlib y Seaborn son ideales para la visualización. Además, herramientas como SciPy y scikit-learn son fundamentales en la investigación científica y el aprendizaje automático. Python es una opción de primer orden para el desarrollo de aplicaciones de análisis de datos, desde minería y procesamiento hasta la creación de modelos predictivos.

Numpy

NumPy es una librería de Python diseñada para la manipulación eficiente de arreglos y matrices multidimensionales, facilitando su manejo de manera intuitiva. Además, ofrece una extensa variedad de funciones matemáticas prácticas, que abarcan desde operaciones de álgebra lineal hasta transformación de Fourier y procedimientos de generación de números aleatorios (Idris, 2015).

En ciencia de datos, NumPy es esencial, ya que proporciona estructuras de datos para almacenar y manipular datos de manera eficaz, lo que facilita cálculos numéricos y operaciones estadísticas en conjuntos de datos de gran tamaño. Además, es fundamental para muchas otras bibliotecas de análisis de datos en Python, como Pandas y Scikit-learn.

Pandas

Pandas es una biblioteca de código abierto en Python que amplía las capacidades del lenguaje para trabajar con datos que se asemejan a hojas de cálculo, permitiendo la carga, manipulación, alineación y combinación de datos de manera eficiente, entre otras funcionalidades. Para lograr esto, Pandas introduce dos nuevos tipos de datos en Python: Series y DataFrame.

Mientras que DataFrame representa la totalidad de una hoja de cálculo o datos en formato rectangular, Series corresponde a una única columna dentro de un DataFrame. En esencia, un DataFrame de Pandas puede ser conceptualizado como un diccionario o una colección de objetos Serie (Chen, 2017).

En ciencia de datos, Pandas es ampliamente empleado para cargar, limpiar, transformar y analizar datos antes de realizar tareas más avanzadas, como modelado estadístico, visualización y aprendizaje automático.

Matplotlib

Es un paquete de Python diseñado para crear gráficos en 2D de alta calidad. Gracias a su amplia gama de formatos de salida, diversos tipos de gráficos y la capacidad de funcionar tanto en entornos interactivos como no interactivos. Matplotlib resulta versátil y útil en una variedad de contextos diferentes (Tosi, 2009).

Seaborn

Seaborn es una biblioteca de visualización de datos en Python que simplifica la creación de gráficos. Resulta ser una herramienta bastante versátil para tareas de visualización en estadística y ciencia de datos. De acuerdo con Waskorn (2021):

Cuando se proporciona un conjunto de datos y una especificación para crear un gráfico, Seaborn mapea automáticamente los valores de los datos en atributos visuales como color, tamaño o estilo, realiza transformaciones estadísticas internas y decora el gráfico con etiquetas informativas en los ejes y una leyenda. Muchas de las funciones de Seaborn pueden generar figuras con múltiples paneles que permiten comparar subconjuntos condicionales de datos o diferentes combinaciones de variables en un conjunto de datos. Seaborn está diseñado para ser útil en todas las etapas de un proyecto científico. Seaborn puede ser empleado para exploración de datos, análisis de distribuciones, relaciones y datos categóricos.

Scikit-learn

Scikit-learn (también conocido como sklearn) es una extensión de Python que reúne una amplia gama de algoritmos avanzados de aprendizaje automático supervisado y no supervidado, para abordar problemas de mediana escala. Este paquete tiene como objetivo hacer que el aprendizaje automático sea accesible para aquellos que no son especialistas en el campo, utilizando a la vez un lenguaje de alto nivel (Pedregosa et al., 2011).

Es empleado en para tareas de: regresión, clasificación, reducción de dimensionalidad, aprendizaje no supervisado, evaluación de modelos, procesamiento de lenguaje natural, preprocesamiento de datos y aplicaciones de investigación científica, abarcando una amplia gama de aplicaciones clave en el análisis de datos y el aprendizaje automático.

Tensorflow

Es una librería de código abierto desarrollada y mantenida por Google, diseñada para realizar cálculos numéricos de manera eficiente. Aunque su API principal está destinada para Python, también ofrece acceso a su API subyacente en C++. A diferencia de otras bibliotecas numéricas utilizadas en Deep Learning, TensorFlow es versátil, utilizada tanto en investigación como en sistemas de producción, incluyendo aplicaciones como RankBrain en la búsqueda de Google y proyectos creativos como DeepDream. Además, es compatible con una variedad de plataformas, desde CPUs y GPUs hasta dispositivos móviles y sistemas distribuidos a gran escala (Brownlee, 2016).

Visual Studio Code + Jupyter Notebooks

Visual Studio Code (VS Code) es un popular entorno de desarrollo integrado de código abierto desarrollado por Microsoft. Está diseñado para ser altamente personalizable, eficiente y adecuado para una amplia gama de lenguajes de programación.

La extensión de Jupyter Notebooks para VS Code permite a los usuarios trabajar con documentos de Jupyter de manera efectiva, lo que es esencial en la programación científica, análisis de datos y tareas de investigación.

Google Colaboratory

Más conocido como Google Colab, es un entorno de Jupyter Notebooks gratuito que opera en la nube, sin necesidad de configuración previa. Con Google Colab, es posible escribir y ejecutar código, guardar y compartir análisis, y aprovechar recursos informáticos potentes, todo desde el navegador, sin costos asociados (Gunawan et al., 2020).

5.2. Línea Base (Baseline)

Para establecer la línea base de las técnicas de Machine Learning se optó por identificar el tipo de comportamiento que muestran las diferentes características de la base de datos. ¿El problema de la predicción de la desatención de la demanda sigue un comportamiento líneal o no líneal? Se implementaron dos modelos de aprendizaje de maquina básicos que permitieran

esclarecer las características del problema, a su vez, se realizó un proceso de analítica base para el entendimiento, análisis y tratamiento de los datos presentes.

Se seleccionó un regresor lineal básico (OLS) de la librería Scikit-learn y un algoritmo de bosques aleatorios (Random Forest) de la misma librería, el primero enfocado en problemas de regresión lineal y el segundo en problemas de regresión no lineal. En ambos se seleccionaron parámetros por defectos. Los parámetros seleccionados, su descripción y el valor asignados se consigan en la **Tabla 4**.

Tabla 4.

Parámetros de los modelos implementados en la línea base

Modelo	Parámetros	Descripción	Valor
LinearRegression (OLS)	fit_intercept	Permite calcular el intercepto del modelo	True
	copy_X	Crear una copia de la información de entrada o sobrescribirla	True
	n_jobs	Numero de procesadores a utilizar (None =1)	None
	positive	Forzar los coeficientes a tomar valores solamente positivos	False
Random Forest Regressor **(RF)**	n_estimators	Numero de árboles en el bosque	100
	criteron	Función que permite medir la calidad de la división del árbol/bosque. Friedman_mse usa el error medio cuadrático mejorado	friedman_mse
	max_depth	Profundidad máxima del árbol	5
	max_features	Numero de características a considerar para la mejor división	auto
	oob_score	Si deseas utilizar las muestras fuera de bolsa para estimar la generalización del modelo	False
	n_jobs	Numero de procesadores a utilizar (-1 = todos los procesadores, físicos y virtuales)	-1
	Random_state	Control de la aleatoriedad del muestreo a la hora de construir los arboles	1234

El proceso de analítica realizado al conjunto de datos y del cual se obtuvo la información de entrada para los modelos de aprendizaje de máquina de la línea, se conformó con los siguientes análisis:

- **Identificación de la información de entrada:**
    - Las variables "*Time*," "*Valencia_wind_deg*", "*Seville_pressure*", y todas las variables "*Weather_id*" se han identificado como variables categóricas durante el desarrollo de la analítica descriptiva (sección 3.3) y, como tal, se han excluido de la información de entrada para los modelos previamente descritos. Esta decisión se basa en la necesidad de un procesamiento previo de estas variables antes de poder utilizarlas como información de entrada en los modelos. Dicho procesamiento se llevará a cabo en iteraciones posteriores.
    - Las demás características se identificaron como variables numéricas. Estas conformaron la información de entrada para los modelos.
- **Tratamiento de datos faltantes y datos atípicos:** se reemplazaron los datos faltantes de la base de datos mediante el algoritmo de imputación simple, usando como criterio de reemplazo el valor de la media de la característica correspondiente a la ubicación del dato faltante. Como se identificó previamente en el numeral 3.3, se tiene un total de 2.068 datos faltantes para toda la base de datos, todos pertenecientes a la característica *Valencia_pressure*, que corresponde a un 23,6 % del total de datos de la característica.
    - Para la eliminación de datos atípicos se optó por el algoritmo LOF, con el cual se encontraron un total de 20 filas con dichos valores. Los parámetros utilizados para este algoritmo se consignan en la **Tabla 5**:

Tabla 5.

Parámetros del algoritmo Local Outlier Factor

Modelo	Parámetros	Descripción	Valor
Local Outlier Factor (LOF)	n_neighbors	Número de vecinos a usar	5
	algorithm	Algortimo a utilizar para computar los vecinos más cercanos. En Auto se selecciona el mejor algoritmo basado en los valores enviados al método fit	auto
	contamination	Valor de la contaminación esperada en la base de datos, también conocida como la proporción de atípicos presente en la base de datos	auto
	metric	Métrica para computar el cálculo de la distancia de los vecinos más cercanos	euclidean

A partir del resultado obtenido se plantea la hipótesis de una mayor existencia de atípicos en la base de datos. En iteraciones posteriores se realiza un análisis más profundo de este apartado.

Se optó por dividir la base de datos asignando un 80% como datos de entrenamiento y un 20% como datos de prueba. Para lograr esto, se utilizó el método "train_test_split" de la librería Scikit Learn. Se usó el atributo "shuffle" para obtener datos con posiciones aleatorias y, además, se empleó el parámetro "random_state" de la misma función. Este último se utilizó con el propósito de asegurar la comparabilidad entre los modelos instanciados en esta etapa y las iteraciones subsiguientes, garantizando así el uso constante del mismo conjunto de datos de prueba y entrenamiento.

Una vez implementados y entrenados los modelos, se procedió a revisar los resultados obtenidos métricas de desempeño, usando como base los datos de prueba. En primera instancia se utilizan todas las métricas definidas en el numeral 1.4. Los resultados se presentan en la **Tabla *6***.

Tabla 6.

Resultados de las métricas de desempeño para los modelos base (primera iteración)

Modelo	R^2	RMSE	MAE	MAPE
LinearRegression	0.14	4889	3883	1.99
Random Forest Regressor	0.16	4836	3836	2.03

Adicionalmente, en las **Figura *26*** y **Figura *27*** se proporciona una comparación visual entre las predicciones generadas por ambos algoritmos al emplear el conjunto de prueba, contrastándolas con los valores reales.

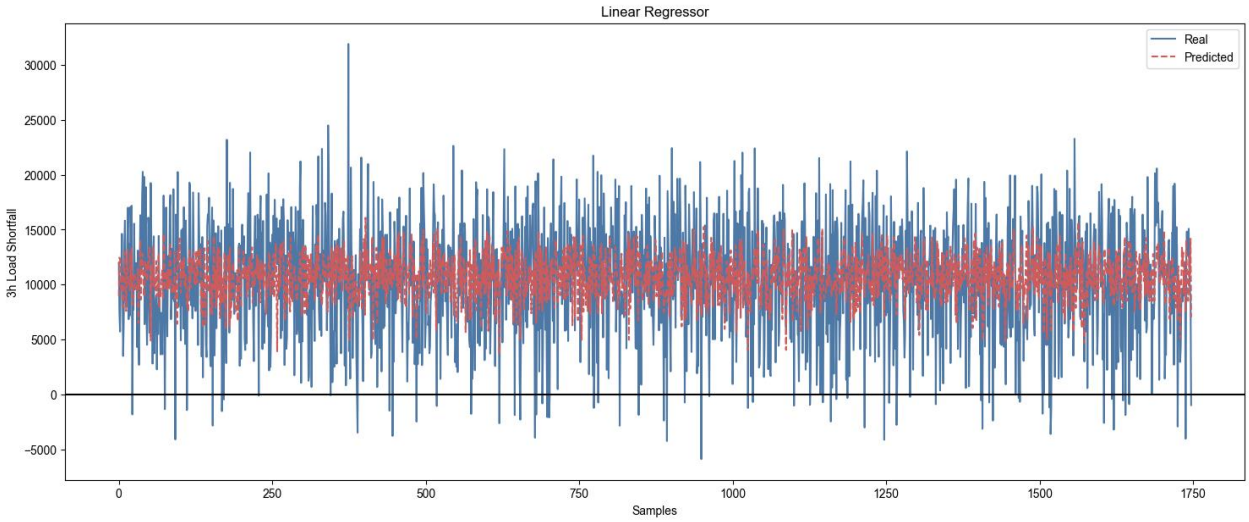


Figura 26. *Comparación visual entre predicciones y valores reales al usar regresión lineal.*

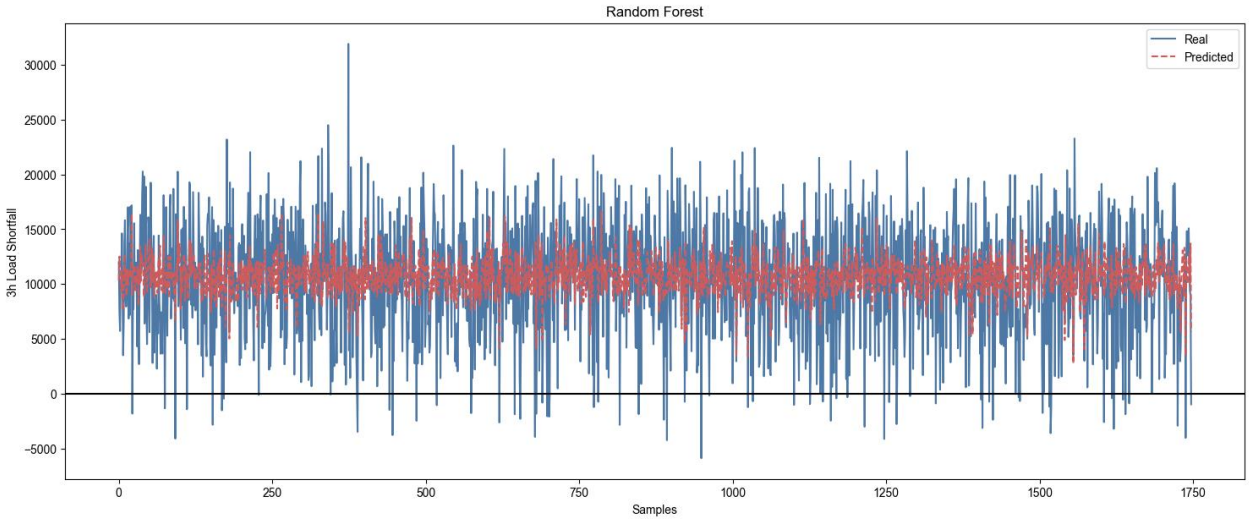


Figura 27. *Comparación visual entre predicciones y valores reales al usar Random Forest Regressor.*

De los resultados obtenidos, se observa un bajo rendimiento (ajuste débil) en ambos modelos, como se refleja en las métricas y las visualizaciones. Las **Figura *26*** y **Figura *27*** muestran claramente la incapacidad de ambos modelos para seguir de cerca las marcadas variaciones en la variable de salida, con el regresor lineal fluctuando más que el Random Forest, que mantiene una estabilidad alrededor de los valores centrales.

Es importante destacar que el modelo Random Forest, diseñado para abordar problemas no lineales, arrojó métricas de mejor desempeño, con un aumento en el coeficiente de determinación

(R^2) y una disminución en las métricas RMSE y MAE. Estos resultados respaldan la hipótesis planteada en la sección de Analítica descriptiva de que estamos tratando con un problema de regresión no lineal.

Por otro lado, al analizar las métricas obtenidas, se observan hallazgos específicos relacionados tanto con el problema en cuestión como con las métricas mismas:

- En cuanto al MAPE, no se encuentran grandes variaciones entre los dos modelos, con valores cercanos al 2 (un error del 200%). Sin embargo, las visualizaciones revelan muchos valores cercanos a cero, lo que sugiere inestabilidades numéricas. Esto lleva a desestimar el MAPE, ya que no es adecuado para los datos en cuestión debido a la naturaleza de la variable que se está prediciendo.
- El coeficiente de determinación (R^2) no proporciona una cuantificación del error en relación con la predicción, ya que es un valor adimensional que evalúa cómo se ajustan los datos predichos a los datos de prueba. Esto complica la interpretación del error en términos de unidades de energía, lo que resulta fundamental para comprender los resultados, por lo tanto, se decide descartarla.
- El RMSE y el MAE, que tienen unidades de la variable de salida (energía en este caso), ofrecen una mejor interpretación de los resultados. Los valores de error obtenidos con el MAE son menores que los obtenidos con el RMSE, ya que este último penaliza de manera más severa las fluctuaciones en la variable de salida. Esto sugiere que el MAE puede ser más adecuado para esta variable predicha, ya que no penaliza en exceso fluctuaciones que son inherentes a los datos y que no necesariamente indican valores atípicos.
- Al comparar el RMSE con la métrica referencial propuesta en la sección 1.4, se evidencia un incremento de aproximadamente el 50% en ambos modelos implementados, indicando la necesidad de mejoras ya sea en el procesamiento de los datos o en la selección y parametrización de los modelos. En esta instancia, se hará énfasis en mejorar la calidad de los datos.

Debido a los resultados anteriores, las iteraciones posteriores se centrarán en lo siguiente:

- Realizar un mejor tratamiento de los datos de entrada.
- Continuar empleando las métricas MAE y RMSE, ya que se ha verificado que muestran mayor interpretabilidad y son más adecuadas para un problema de regresión no lineal.

- Implementar nuevos modelos de Machine Learning con capacidad de tratar problemas de regresión no lineal.
- Implementar diferentes métodos de optimización de los hiperparámetros de los modelos implementados, con la finalidad de obtener mejores resultados en las métricas de desempeño.

5.3.Iteraciones y evolución

5.3.1. Segunda iteración

La segunda iteración se enfocó mejorar el análisis y tratamiento de la base de datos, con el objetivo de obtener mejores métricas de desempeño con respecto a la línea base. Para esta iteración se considera únicamente modelo Random Forest de acuerdo con los hallazgos encontrados en la línea base. Los procesos implementados fueron los siguientes:

- **Tratamiento de variables categóricas:** Las variables categóricas Seville_pressure y Valencia_wind_deg, de acuerdo con la analítica descriptiva llevada a cabo, requieren transformación que permita expresarlas como variables numéricas, al igual que las mismas variables medidas en otras ciudades.

  Para la presión de Sevilla (*Seville_preassure*) se observó la codificación 'SPXX' donde XX representa dos caracteres numéricos, se optó por mantener los caracteres numéricos y modificar los caracteres SP por 10, de tal forma que los datos representaran valores de miles, estructura semejante a los datos de presión de las demás ciudades presentes en la base de datos. El ángulo de viento de Valencia (*Valencia Wind Deg*) presenta una codificación por niveles 'Level X' donde X era un carácter numérico de 1 a 10, se optó por mantener únicamente los valores numéricos del 1 al 10, respetando la numeración como un indicador de la severidad del ángulo.
- **Tratamiento de variables "Weather_id":** Las categorías del clima para las diferentes ciudades en las que fue censada, de acuerdo con las necesidades descritas en la analítica descriptiva, requiere ser tratada como una variable categórica de acuerdo con las codificaciones descritas previamente. Una vez tratada esta variable, se obtiene la distribución de datos para las diferentes ciudades visualizada en la **Figura *28***.

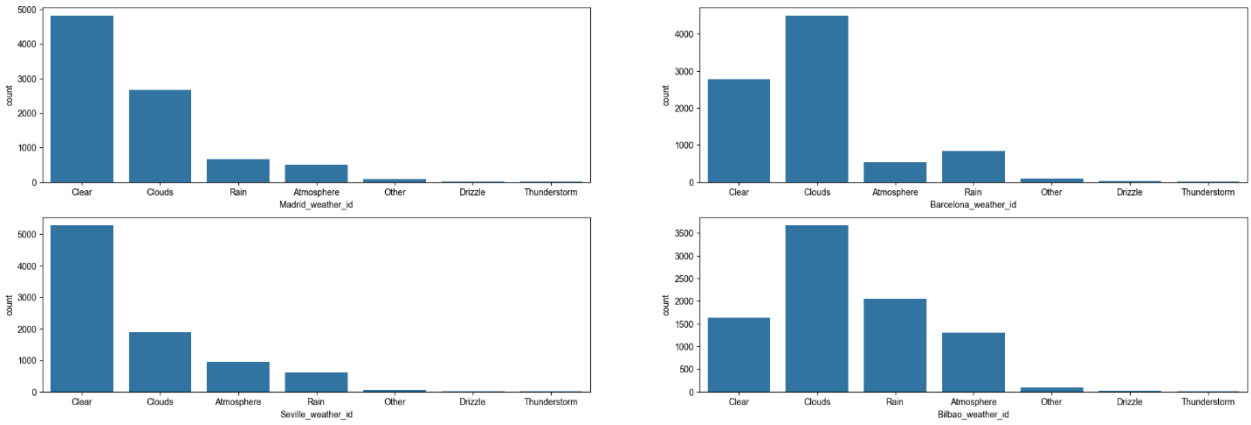


Figura 28. *Nueva estructura de las variables "categoría del clima (weather_id)"*

Además, con el objetivo de utilizar esta nueva representación de la variable "Weather_id" como información de entrada en cualquier modelo de regresión, se ha decidido aplicar la técnica de codificación OneHotEncoding. Esto permitirá obtener una nueva representación numérica de la variable que no introduzca una jerarquía numérica entre las categorías.

- **Eliminación puntual de datos atípicos:** En la característica *Barcelona*_pressure se utilizó el test Z, eliminando los valores superiores a una desviación estándar, con base en el hallazgo de valores de presión cercanos al millón de unidades de presión. Este criterio se basa en la no existencia de cuidades en la tierra con valores de presión atmosférica por encima de los 1084.8 hPa (World Meteorological Organization, 2012). En la **Figura *29*** se pueden observar la distribución de los datos de las variables numéricas asociadas a la presión y el impacto de los valores atípicos identificados para la característica *Barcelona_pressure.*

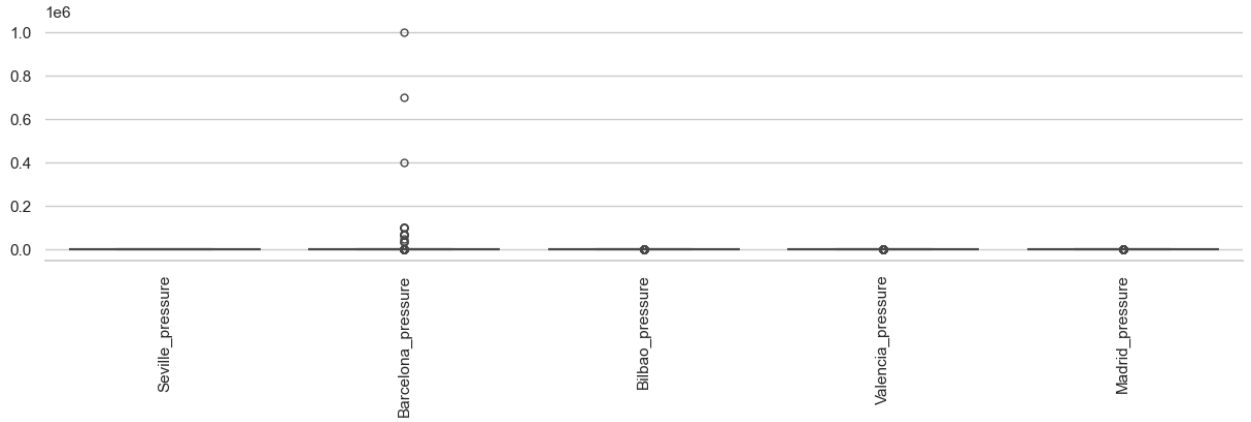


Figura 29. *Diagrama de cajas de las características de presión disponibles en la Base de datos*

Posteriormente se optó por eliminar los valores superiores al 20% del promedio de los datos de la presión de Barcelona, el comportamiento para las variables numéricas de presión una vez realizado este tratamiento de datos atípicos se observa en la **Figura 30**.

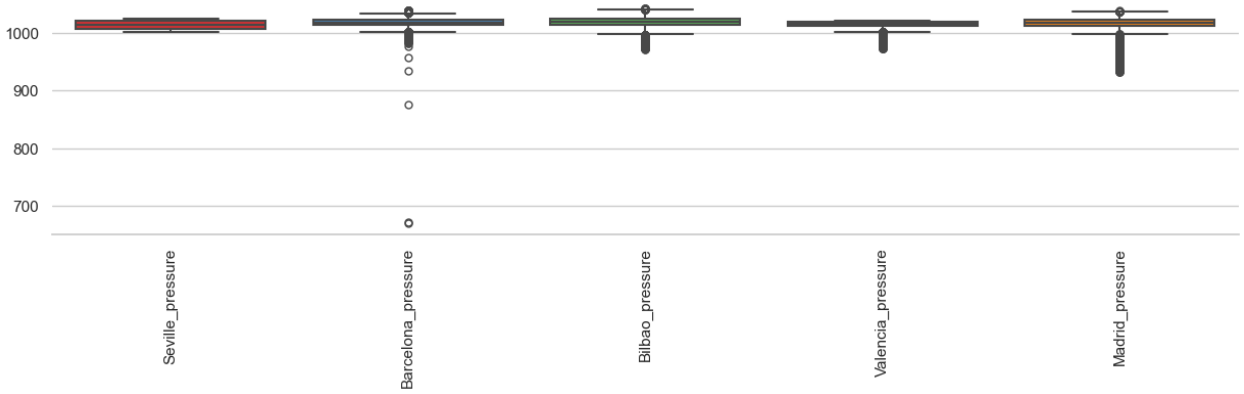


Figura 30. *Diagrama de cajas de las características de presión disponibles en la Base de datos, realizado el tratamiento de datos atípicos para la característica Barcelona_pressure.*

Luego de aplicar el tratamiento de datos mencionado anteriormente, se procedió a entrenar nuevamente el modelo RandomForest que se definió en la iteración previa. El objetivo es determinar si las opciones de tratamiento propuestas tienen un impacto positivo en el rendimiento del modelo. A continuación, se detallan los resultados obtenidos:

Tabla 7.
Resultados de las métricas de desempeño para la segunda iteración.

Modelo	Iteración	RMSE	MAE
Random Forest Regressor	1 (Anterior)	4836	3836
	2 (Actual)	4682 ↓	3745 ↓

Los resultados obtenidos muestran mejoras en ambas métricas consideradas, lo que permite concluir que las estrategias de tratamiento consideradas han sido efectivas. Sin embargo, los resultados de RMSE obtenidos siguen siendo superiores en un 43% respecto a la métrica referencial.

Con base en estos hallazgos, las iteraciones posteriores tomarán como información de entrada los datos consolidados hasta este punto.

5.3.2. Tercera iteración

En esta sección se lleva a cabo el análisis de estacionalidad propuesto en la sección 4.2. Con el propósito de alcanzar este objetivo, se ha utilizado la fecha de cada registro para agregar una nueva columna al conjunto de datos que clasifica cada entrada según la estación en la que se realizó la medición. A continuación, estas estaciones se someten a un proceso de transformación mediante la técnica de One-Hot Encoding, disponible en la biblioteca scikit-learn. Este procedimiento tiene como finalidad obtener una representación binaria de las categorías generadas, lo que resulta más apropiado para su utilización en algoritmos de aprendizaje automático.

Posteriormente, se procedió a entrenar el modelo RandomForest definido en la línea base con los nuevos datos agregados. A continuación, se detallan los resultados obtenidos:

Tabla 8.
Resultados de las métricas de desempeño para la tercera iteración.

Modelo	Iteración	RMSE	MAE
Random Forest Regressor	2 (Anterior)	4682	3745
	3 (Actual)	4458 ↓	3551 ↓

Los resultados obtenidos permiten identificar que la adición de la estación como categoría ha mejorado el desempeño del modelo. Por ende, iteraciones posteriores tomarán como información de entrada los datos consolidados hasta este punto.

Además, al continuar monitoreando el rendimiento de la métrica RMSE en relación con el criterio seleccionado, se observa que en este punto los valores obtenidos están un 37% por encima de dicho criterio.

5.3.3. Cuarta iteración

Otro hallazgo importante presentado en la Analítica descriptiva, que debemos considerar como parte del proceso de tratamiento de datos, es la eliminación de variables que muestran una acumulación de datos en un valor único. Concretamente, las variables que se deben eliminar son aquellas que registran datos de lluvia durante tres o una hora, nieve o nubosidad (a excepción de la nubosidad en Bilbao). Este criterio se aplica a todas estas variables en las diferentes ciudades.

Tabla 9.
Resultados de las métricas de desempeño para la cuarta iteración.

Modelo	Iteración	RMSE	MAE
Random Forest Regressor	3 (Anterior)	4458	3551
	4 (Actual)	4460	3553

Los cambios en las métricas encontradas luego de la eliminación de las variables mencionadas resultan ser insignificantes al ser comparados con los obtenidos en la iteración anterior. Sin embargo, aunque el desempeño del modelo no cambia con el tratamiento implementado, se logra verificar que estas variables no aportan información a los modelos y por lo tanto, se decide su eliminación con el objetivo de optimizar los modelos a instanciar en las siguientes iteraciones.

5.3.4. Quinta iteración

Hasta este punto, hemos preparado los datos utilizando una variedad de enfoques, que van desde el tratamiento de datos atípicos hasta las transformaciones destinadas a mejorar la estructura de los mismos para su posterior utilización en modelos de aprendizaje automático. Todos estos tratamientos han sido validados a través de un modelo RandomForest. En esta sección, se lleva a cabo los primeros ensayos con nuevos modelos de aprendizaje automático supervisado con el objetivo de identificar cuáles de ellos se adaptan de manera óptima y proporcionan predicciones más precisas.

En la **Tabla *10***, se presentan los modelos empleados y los principales hiperparámetros con los cuales han sido instanciados:

Tabla 10.
Modelos de aprendizaje automático e hiperparámetros.

Modelo	Hiperparámetros
Random Forest Regressor	n_estimators = 100, criterion= 'friedman_mse', max_depth= 5, max_features = 1.0, oob_score= False

Modelo	Hiperparámetros
Support Vector Regressor	kernel='rbf', degree=3, gamma='scale', coef0=0.0, tol=0.001, C=1.0, epsilon=0.1,
Decision Trees	criterion='squared_error', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1,
Extratrees Regressor	n_estimators=100, criterion='squared_error', max_depth=None, min_samples_split=2, min_samples_leaf=1,
Gradient Boosting Machine	loss='squared_error', learning_rate=0.1, n_estimators=100, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3,
Light Gradient Boosting Machine	boosting_type='gbdt', num_leaves=31, max_depth=-1, learning_rate=0.1, n_estimators=100,
Extreme Learning Machine	alpha=1e-07, batch_size=None, include_original_features=False, n_neurons=None, ufunc='tanh',

Modelo	Hiperparámetros
Deep Neural Network	**1 hidden layer:** units=100, activation='tanh', kernel_regularizer=l2(1e-6) **output layer:** units=1, activation='linear'

En la ***Tabla 11***, se presentan los resultados obtenidos. Los resultados gráficos pueden ser encontrados dentro del Anexo A.

Tabla 11.
Resultados de las métricas de desempeño para la quinta iteración.

Modelo	RMSE	MAE
Random Forest Regressor (RF)	4458	3552
Support Vector Regressor (SVR)	5029	4001
Decision Trees (DT)	5749	4371
Extratrees Regressor (ETR)	3915	3063
Gradient Boosting Machine (GBM)	4345	3438
Light Gradient Boosting Machine (LightGBM)	4067	3196
Extreme Learning Machine (ELM)	4802	3836
Atificial Neural Network (ANN)	9591	8501

Entre los modelos implementados, destacan inicialmente el rendimiento de dos modelos: el ExtraTrees Regressor y la Light Gradient Boosting Machine. Con respecto a las métricas RMSE y MAE, experimentaron una disminución aproximada del 13% y el 9%, respectivamente, en comparación con los resultados de referencia obtenidos hasta el momento con el modelo Random Forest Regressor.

Por otro lado, es esencial mencionar los resultados de los modelos de Máquina de Soporte Vectorial (SVM) y Redes Neuronales (RNN y ELM). Estos modelos presentaron métricas con un desempeño significativamente inferior, relacionado con un aumento de hasta un 140% en comparación con los valores obtenidos mediante los modelos Random Forest iniciales.

Sin embargo, tras una revisión de la literatura, se ha determinado que los tipos de modelos mencionados anteriormente requieren la normalización de los datos para garantizar su funcionamiento óptimo. En el caso de las redes neuronales, según Dawson y Wilby (1998), debido a la naturaleza de las funciones de activación, es prudente llevar a cabo la normalización de los

datos. Sin este procedimiento, variables con valores en escalas significativamente diferentes respecto a otras variables necesitarían pesos extremadamente pequeños, lo que podría dar lugar a inestabilidades en los modelos.

Por otro lado, según Dewi y Widiastuti (2020), en el contexto de las máquinas de soporte, la normalización de los datos contribuye a que estos cumplan de manera más precisa con las suposiciones establecidas por las funciones kernel.

En función de los descubrimientos previos, en la próxima iteración se llevará a cabo la evaluación de técnicas de normalización con el objetivo de potenciar el rendimiento de los modelos fundamentados en máquinas de soporte y redes neuronales. Al mismo tiempo, se procederá a analizar el desempeño de los demás modelos considerados cuando se aplican sobre datos normalizados.

5.3.5. Sexta iteración

En esta iteración, se examinan las respuestas de los diversos modelos definidos en la sección anterior cuando se utilizan datos normalizados. Concretamente, se han empleado los métodos MinMax y StandardScaler de la biblioteca Scikit Learn. A través de la **Tabla *12***, se muestran los resultados obtenidos, destacando que, con el fin de facilitar la interpretación de los datos, se han reconvertido los resultados a la escala original de la variable de salida.

Tabla 12.
Resultados de las métricas de desempeño para la sexta iteración.

Modelo	RMSE			MAE		
	Unscaled	**Std**	**MinMax**	**Unscaled**	**Std**	**MinMax**
Random Forest Regressor (RF)	4458	4457	4458	3552	3551	3552
Support Vector Regressor (SVR)	5029	4064	4285	4001	3120	3398
Decision Trees (DT)	5749	5712	5787	4371	4365	4384
Extratrees Regressor (ETR)	3915	3948	3937	3063	3092	3074
Gradient Boosting Machine (GBM)	4345	4344	4346	3438	3437	3437
Light Gradient Boosting Machine (LightGBM)	4067	4083	4063	3196	3222	3198
Extreme Learning Machine (ELM)	4802	4647	4528	3836	3564	3578
Artificial Neural Network (ANN)	9591	5884	4658	8501	4599	3594

Como se había anticipado, las normalizaciones efectuadas han demostrado ser beneficiosas para el rendimiento de los modelos SVR, ELM y ANN. La mejora es particularmente evidente en

el caso del modelo ANN, especialmente cuando se emplea el escalado MinMax. En cuanto a los otros modelos, no se observan diferencias significativas en las métricas al entrenar los modelos con datos escalados mediante los dos procedimientos considerados.

A pesar de las mejoras observadas en los modelos mencionados anteriormente, los modelos LightGBM y ETR continúan exhibiendo el mejor desempeño, siendo el ETR el que mejor se adapta hasta este punto a los datos. Es relevante señalar que el modelo RF, empleado como referencia desde la primera iteración, muestra en términos generales las terceras mejores métricas.

5.3.6. Séptima iteración

A continuación, se llevó a cabo la evaluación de técnicas de reducción de dimensionalidad, centrándose en preservar la máxima cantidad de información con el menor número posible de características de la base de datos. Las características resultantes se utilizaron para entrenar y probar los modelos de Machine Learning definidos en la iteración 5, permitiendo así la obtención de sus métricas de desempeño. Se emplearon dos técnicas para la reducción de la dimensión:

- **Por correlación:** se descartaron aquellas características que mostraran una correlación de al menos el 70% con respecto a otra característica, ya sea de manera positiva o negativa. Este proceso redujo las colinealidades en la base de datos, eliminando un total de 19 variables de entrada de 74 originales.
- **Análisis de componentes principales:** se implementó el algoritmo PCA de la biblioteca Scikit-learn, seleccionando el número de componentes con base en el criterio de obtener al menos 96% de la varianza total de la base de datos. Se obtuvieron 48 componentes principales a partir de los 74 originales, las cuales generaron un error promedio del 0.7% al comparar los datos originales con los datos reconstruidos mediante la transformada inversa de las componentes principales. Para la implementación de este modelo, se ha empleado la normalización de datos mediante el método StandardScaler. Esto se ha llevado a cabo con el propósito de permitir que el modelo identifique con precisión las direcciones de máxima varianza, sin verse afectado por las disparidades en la escala de las variables.

Los resultados obtenidos se detallan en la **Tabla *13***. Se debe interpretar las etiquetas "completa", "corr" y "PCA" como los resultados correspondientes a la base de datos completa, la

versión reducida por criterio de correlación, y la versión reducida mediante el método PCA, respectivamente.

Tabla 13.
Resultados de las métricas de desempeño para la séptima iteración.

Modelo	RMSE			MAE		
	Completa	**Corr**	**PCA**	**Completa**	**Corr**	**PCA**
Random Forest Regressor	4458	4541	4644	3552	3603	3713
Support Vector Regressor	5029	5029	5005	4001	4009	3989
Decision Trees	5749	6135	6222	4371	4705	4832
Extratrees Regressor	3915	4043	4270	3063	3165	3358
Gradient Boosting Machine	4345	4398	4509	3438	3477	3577
Light Gradient Boosting Machine	4067	4228	4427	3196	3329	3465
Extreme Learning Machine	4802	4861	4892	3836	3906	3713
Deep Neural Network	9591	5200	5013	8501	4241	4024

Se pudo observar que los modelos entrenados con datos de menor dimensionalidad presentaron métricas más altas en comparación con aquellos entrenados utilizando la base de datos completa, con la excepción de los resultados obtenidos para la ANN. Aunque las diferencias en las métricas no son excesivas en ninguno de los casos y podría considerarse mantener alguna de las reducciones, se ha decidido conservar la totalidad de los datos. Esta elección se fundamenta en la ausencia de mejoras significativas en los tiempos de procesamiento y, además, eliminar o reducir las características mediante PCA podría comprometer la interpretabilidad de la información de entrada.

5.3.7. Octava iteración

A partir de los resultados obtenidos en las anteriores iteraciones, se concluye hasta este punto que los modelos que mejor se adaptan a los datos luego de procesar estos con las diferentes técnicas de tratamiento propuestas, son Ligth Gradient Boosting Machine (LightGBM), Gradient Boosting Machine (GBM) y Extra Trees Regressor (ETR). Debido a la similitud entre los modelos LightGBM y GBM, se ha decidido conservar el método LightGBM debido a su eficiencia y desempeño respecto al GBM. Además, se explorará el siguiente mejor método, que corresponde a Random Forest (RF) y, por otro lado, se considerará una Red Neuronal Artificial (ANN), ya que esta última podría demostrar una mayor robustez en comparación con la configuración actual, al agregarle una mayor complejidad.

La ***Tabla 14*** resume para los cuatro modelos seleccionados, la desviación porcentual de los errores RMSE obtenidos en la iteración anterior, en base al criterio de referencia (3262 unidades energéticas).

Tabla 14.
Desviación porcentual de las métricas RMSE obtenidas en la iteración 7

Modelo	Desviación respecto al RMSE referencial
Random Forest Regressor	+ 37%
Extratrees Regressor	+ 20%
Light Gradient Boosting Machine	+ 25%
Artificial Neural Network	+ 43%

Considerando que las métricas siguen siendo superiores al error referencial, la meta de esta iteración consistirá en mejorar sustancialmente el desempeño de los tres modelos seleccionados anteriormente. Este objetivo se llevará a cabo mediante la implementación de técnicas de HyperTuning. En este caso específico, y tomando como base las definiciones presentadas en la sección 4.2, se empleará la búsqueda bayesiana. Este enfoque se elige por ofrecer una búsqueda óptima de hiperparámetros en comparación con los otros dos métodos investigados, Randomized Search y Grid Search.

El algoritmo implementado para la optimización bayesiana corresponde al estimador de árboles estructurados de Parzen (Tree-sctructured Parzen estimator - TPE) disponible en la librería Hyperopt de Python. La optimización TPE consiste en la definición de dos funciones de densidad, que conforman los modelos generativos para todas las variables del hiperespacio, las observaciones se dividen en buenos y malos resultados mediante un percentil previamente definido y^*, los resultados son modelados mediante una ventana simple de Parzon, de los cuales se obtiene la mejora esperada en la función de adquisición y se ve reflejada en la relación de las funciones de densidad, la cuales se utilizan para determinan las nuevas configuraciones para la evaluación. (Yang L & Shami A, 2020)

La librería Hyperopt mediante la clase “hp”, permite definir el espacio de evaluación para cada uno de los parámetros que se desean optimizar, a partir de una serie de funciones que determinan las características de dicho espacio, entre las cuales se resaltan las funciones choice, quniform, uniform, cuyo comportamiento se detalla a continuación:

- **choice:** Esta función selecciona una de las opciones disponibles definidas previamente mediante una lista de posibles valores.
- **uniform (etiqueta, mínimo, máximo):** Esta función retorna un valor uniformemente distribuido en un intervalo previamente definido mediante un valor minino y un valor máximo.
- **quniform (etiqueta, mínimo, máximo, q):** Esta función retorna un valor de un espacio uniformemente distribuido el cual es discretizado a partir de la siguiente ecuación.

$$\text{val} = \text{round}\left(\frac{\text{uniform}(\text{mínimo}, \text{máximo})}{\text{q}}\right) * \text{q}$$

De la **Tabla *15*** a la **Tabla *18*** se informa el espacio de hiperparámetros explorado para cada uno de los algoritmos:

Tabla 15.
Espacio de hiperparámetros para algoritmo RandomForestRegressor.

Hiperparámetro	Configuración del espacio de búsqueda
n_estimators	choice('n_estimators',[10, 50, 100, 300, 750, 1200,1300,1500])
criterion	choice('criterion',['squared_error', 'absolute_error', 'friedman_mse'])
min_samples_split	quniform('min_samples_leaf', 2, 10, 1)
min_samples_leaf	quniform ('min_samples_split', 1, 10, 1)
max_features	choice('max_features', [1.0, 'sqrt','log2', None])

Tabla 16.
Espacio de hiperparámetros para algoritmo ExtraTreesRegressor.

Hiperparámetro	Configuración del espacio de búsqueda
n_estimators	choice('n_estimators',[10, 50, 100, 300, 750, 1200,1300,1500])
criterion	choice('criterion',['squared_error', 'absolute_error', 'friedman_mse'])
min_samples_split	uniform('min_samples_leaf', 2, 10)
min_samples_leaf	uniform ('min_samples_split', 1, 10)
max_features	choice('max_features', [1.0, 'sqrt','log2', None])

Tabla 17.
Espacio de hiperparámetros para algoritmo LigthGradientBoostingMachine.

Hiperparámetro	Configuración del espacio de búsqueda
boosting_type	choice('boosting_type',['gbdt','dart'])
num_leaves	quniform('num_leaves', 2, 100, 10)
learning_rate	choice('learning_rate', [0.01, 0.5, 0.9])
n_estimators	choice('n_estimators',[10, 50, 100, 300, 750, 1200,1300,1500,])
min_split_gain	choice('min_split_gain', [0.0, 0.5, 1.0])
min_child_samples	choice('min_child_samples', [10, 20, 50, 100])
reg_alpha	choice('reg_lambda', [0.0, 0.0001, 0.000001])
reg_lambda	choice('reg_lambda', [0.0, 0.0001, 0.000001])
importance_type	choice('importance_type', ['split', 'gain'])

Tabla 18.
Espacio de hiperparámetros para algoritmo AtificialNeuralNetwork.

Hiperparámetro	Configuración del espacio de búsqueda
learning_rate	Float('learning_rate', min_value=1e-4, max_value=1e-1)
drop out	Float('drop_out', min_value=0.1, max_value=0.5)
n_layers	Choice('n_layers', [1,2,3,4,5])
units	Int(f'layer_{x}_units', min_value=32, max_value=512)

Se diseñó una función objetivo para cada algoritmo de aprendizaje automático, encargada de evaluar el modelo correspondiente con datos de entrenamiento que abarcan el 90% de los datos totales. En el proceso de búsqueda bayesiana, esta función objetivo se evalúa en el espacio de hiperparámetros definido para el modelo, buscando identificar la configuración que minimice una métrica seleccionada.

Aunque la métrica seleccionada como criterio de desempeño es el RMSE, debido a ser la métrica especificada en el desafío Kaggle que sirve como base para este estudio, análisis previos han señalado que la métrica MAE puede ofrecer una descripción más precisa del comportamiento de los datos. Por esta razón, se ha decidido que la métrica a minimizar mediante la búsqueda bayesiana sea el MAE. En consecuencia, se ha definido que la función objetivo proporciona un promedio de métrica MAE a través de una validación cruzada de 5 pliegues.

Posteriormente, el modelo óptimo se pone a prueba con datos de prueba, que constituyen el 10% restante, para validar su desempeño.

De la **Tabla *19*** a la **Tabla *22*** se muestran los mejores hiperparámetros encontrados para cada algoritmo, mientras que la **Tabla *23*** presenta un resumen de las métricas RMSE y MAE obtenidas para cada modelo óptimo.

Tabla 19.
Mejores hiperparámetros para algoritmo RandomForestRegressor.

Hiperparámetro	Mejor valor
n_estimators	50
criterion	Friedman_mse
max_depth	None
min_samples_split	2
min_samples_leaf	2
max_features	None

Tabla 20.
Mejores hiperparámetros para algoritmo ExtraTreesRegressor.

Hiperparámetro	Mejor valor
n_estimators	1200
criterion	absolute_error
min_samples_split	5
min_samples_leaf	1
max_features	1.0

Tabla 21.
Mejores hiperparámetros para algoritmo LightGradientBoostingMachine.

Hiperparámetro	Mejor valor
boosting_type	dart
num_leaves	50
learning_rate	0.5
n_estimators	750
min_split_gain	0.0
min_child_weight	0.001
min_child_samples	20
reg_alpha	1e-6
reg_lambda	0.0001
importance_type	gain

Tabla 22.
Mejores hiperparámetros para algoritmo AtificialNeuralNetwork.

Hiperparámetro	Mejor valor
learning_rate	0.0004832
drop_out	0.1
n_layers	4
units (layer1)	481
units (layer1)	512
units (layer1)	512
units (layer1)	376

Tabla 23.
Resultados de las métricas de desempeño para la octava iteración.

Modelo	RMSE			MAE		
	Sin HT	Con HT	Reducción	Sin HT	Con HT	Reducción
Random Forest Regressor	4458	3892	13%	3552	3034	15%
Extratrees Regressor	3915	3880	1%	3063	3015	2%
Light Gradient Boosting Machine	4067	3757	8%	3196	2919	9%
Artificial Neural Network	4663	3665	21%	3619	2686	26%

HT: HyperTuning

Los resultados obtenidos revelan una mejora notable en el rendimiento de los modelos ANN y LightGBM optimizados, destacándose el algoritmo ANN. Este último logró una reducción del 21% al comparar el valor RMSE obtenido con respecto al valor sin la optimización mediante HyperTuning.

Respecto al criterio de desempeño, los valores RMSE de los modelos optimizados presentaron las siguientes disminuciones expresadas como porcentajes del mismo criterio en la **Tabla *24***.

Tabla 24.
Desviación porcentual de las métricas RMSE obtenidas en la iteración 8.

Modelo	Desviación respecto al RMSE referencial
Random Forest Regressor	+ 19%
Extratrees Regressor	+ 19%
Light Gradient Boosting Machine	+ 15%
Artificial Neural Network	+ 12%

Entre los cuatro modelos, se destaca nuevamente el desempeño de la Red Neuronal Artificial (ANN), la cual, obtuvo el resultado más cercano a la métrica referencial, mostrando un error RMSE de 3665 unidades energéticas, el cual es 12% superior al criterio de desempeño establecido de 3262 unidades energéticas. Este modelo ha sido seleccionado como la solución final para abordar el problema de análisis de datos planteado en el presente documento.

En el próximo apartado, profundizaremos en los resultados obtenidos a través de las diferentes iteraciones.

6. Resultados y discusión

6.1. Métricas

En la **Tabla *25*** se presenta el resumen de los resultados para el RMSE y el MAE obtenidos de las iteraciones implementadas, destacando únicamente los modelos Random Forest (RF), Light Gradient Boosting Machine (LightGBM) y Artificial Neural Network (ANN), que corresponden a los modelos con el mejor desempeño a lo largo de las diferentes evaluaciones. Los resultados para ANN y LightGBM se muestran a partir de la iteración 5, iteración en la cual se implementan por primera vez.

Tabla 25.
Resumen de las métricas para los mejores modelos por iteración.

	RMSE			MAE		
	RF	**ANN**	**LightGBM**	**RF**	**ANN**	**LightGBM**
Iteración 1	4836			3836		
Iteración 2	4682			3745		
Iteración 3	4458			3551		
Iteración 4	4460			3553		
Iteración 5	4458	9591	4067	3552	8501	3196
Iteración 6	4457	4658	4063	3551	3594	3196
Iteración 7	4458	5013	4067	3552	4024	3196
Iteración 8	3892	3665	3757	3034	2686	2919

Se muestra el mejor valor obtenido en cada iteración.

Los resultados del modelo Random Forest mostraron una mejora para el RMSE y el MAE desde la iteración 1 a la iteración 3, iteraciones enfocadas en el procesamiento de los datos de entrada, mediante las técnicas de tratamiento de datos faltantes, eliminación de atípicos, trasformación de variables categóricas y aumento de información, como respuesta al aumento de

la calidad de los datos debido su tratamiento. Para la iteración 4 se implementó la eliminación de variables con concentración de datos en valores de cero, dando como resultado una leve desmejora en las métricas, sin embargo, un aumento en la velocidad de procesamiento del algoritmo debido a la supresión de dichas características.

La iteración 5 se basó en la implementación de los modelos: SVR, Decision Trees, Extratrees Regressor, Gradient Boostring Machine, Light Gradient Boosting Machine, Extreme Learning Machine, y la Artificial Neural Network, todos con sus hiperparámetros por defecto. De los resultados se destacaron los modelos Extratrees Regressor y Light Gradient Boosting Machine, al obtener una mejora significativa con respecto a los resultados del RF del 13% y 9% respectivamente, se debe resaltar los bajos resultados obtenidos por la SVR y la ANN, las cuales mostraron una desmejora de hasta un 140% con respecto al RF, lo cual se debió a la falta de normalización de la información de entrada.

Los resultados de la iteración 6 evidenciaron que los algoritmos enfocados en la normalización de los datos, ya sea mediante la estandarización o el escalamiento, ayudaron a mejorar las métricas de los modelos cuya arquitectura y construcción requieren de este tipo de transformaciones (SVR, ELM y ANN en este caso), sin embargo, poco efecto tuvo en las métricas de los mejores modelos como se muestra en la **Tabla *25***.

Los resultados de la iteración 7 no mostraron ningún beneficio para el RMSE y el MAE con relación a la iteración 5. En esta iteración se implementó la reducción de las dimensiones de la información de entrada mediante un análisis de correlación y el algoritmo PCA, con los cuales se redujo la información de entrada en 19 y 26 características respectivamente. La relación costo-beneficio de reducir las dimensiones de la información de entrada para el entrenamiento y ajuste de los modelos y la perdida de la capacidad explicativa de las características de entrada con respecto a la variable de salida no representó mayor ventaja con relación al uso de la información completa. Los modelos RF, ANN y LightGBM fueron seleccionados como los mejores modelos.

En la octava iteración, se incorporó la sintonización de hiperparámetros de los mejores modelos mediante la optimización bayesiana, utilizando el algoritmo TPE y otras funcionalidades de hypertuning disponibles en la biblioteca Hyperopt de Python. Los resultados revelaron mejoras notables, con una reducción del 13% en el RMSE para Random Forest (RF), un 21% para la Red

Neuronal Artificial (ANN) y un 8% para LightGBM. Asimismo, se observó una mejora del 15% en el MAE para RF, un 26% para ANN y un 9% para LightGBM.

Estos hallazgos indican la utilidad significativa del tratamiento previo de los datos realizado en iteraciones anteriores, lo cual facilitó la selección de los modelos óptimos y, por ende, la optimización de los estimadores principales. Además, el empleo de la optimización bayesiana permitió identificar los mejores hiperparámetros de estos modelos al minimizar la función objetivo, resultando en una mejora general en la capacidad predictiva.

Aunque las métricas obtenidas se aproximan al criterio de desempeño establecido, destacando el caso particular de la ANN que logró un resultado apenas 1.12 veces superior a dicho criterio, no alcanzaron niveles inferiores al valor máximo esperado para el RMSE. Esto indica que existe margen para la mejora, ya sea mediante el refinamiento en el procesamiento de datos, la afinación de modelos o en ambos aspectos.

En la **Tabla *26*** se presentan las variaciones porcentuales del RMSE y el MAE para cada iteración respecto a la iteración anterior para los mejores modelos RF, ANN y LightGBM.

Tabla 26.

Variación porcentual de las métricas para los mejores modelos para cada iteración.

	RMSE [%]			**MAE [%]**		
	RF	**ANN**	**LightGBM**	**RF**	**ANN**	**LightGBM**
Iteración 2	-3.184			-2.372		
Iteración 3	-4.784			-5.180		
Iteración 4	0.045			0.056		
Iteración 5	-0.045			-0.028		
Iteración 6	-0.022	-51.434	-0.098	-0.028	-57.723	0.000
Iteración 7	0.022	7.621	0.098	0.028	11.964	0.000
Iteración 8	-12.696	-26.890	-7.622	-14.583	-33.250	-8.667

No se muestra la iteración 1 o línea base al ser el punto de partida

En la **Figura 31** y la **Figura 32** se resumen los resultados previamente descritos para el RMSE y el MAE con los tres mejores modelos: RF, ANN y LightGBM.

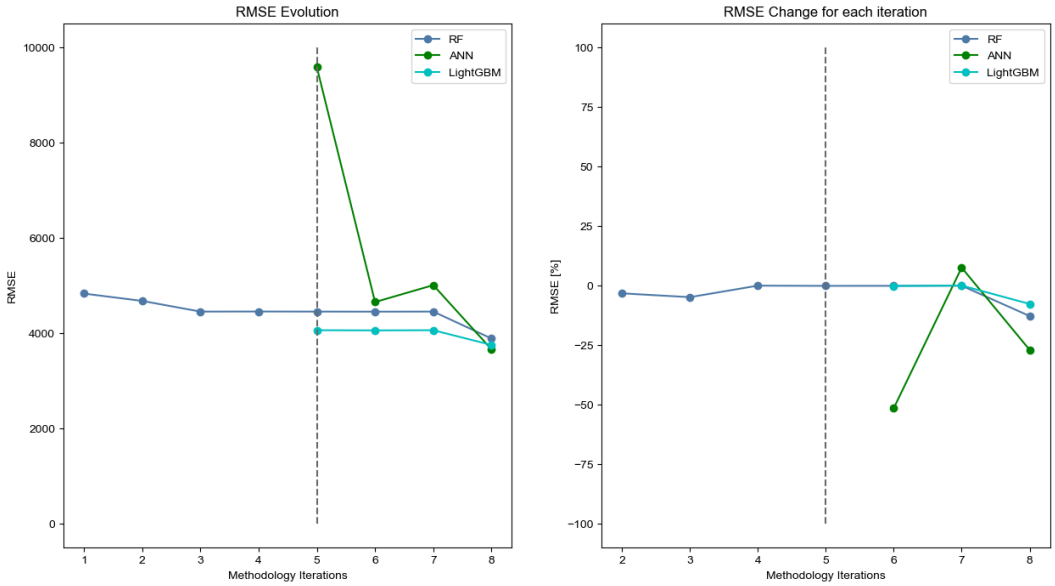


Figura 31. *Evolución del RMSE para los mejores modelos a lo largo de todas las iteraciones implementadas.*

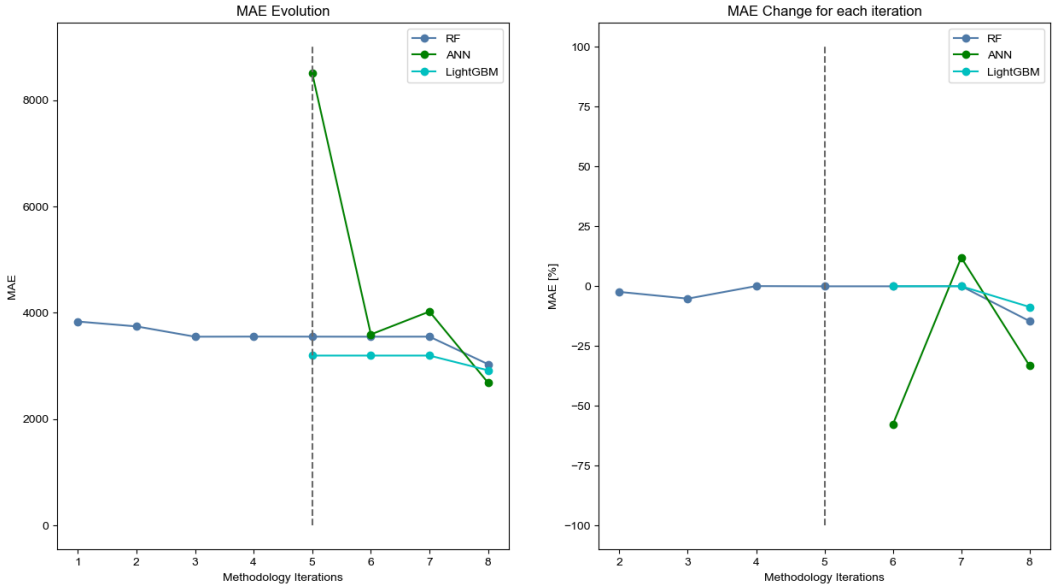


Figura 32. *Evolución del MAE para los mejores modelos a lo largo de todas las iteraciones implementadas.*

Y en la **Figura 33**, la **Figura 34** y la **Figura 35** se presenta la comparación de la desatención de la demanda estimada y real para los modelos RF, ANN y LightGBM respectivamente. Estos

resultados fueron obtenidos a partir de 852 muestras seleccionadas previamente para prueba de manera aleatoria mediante el algoritmo train_test_split de la librería Scikit Learn.

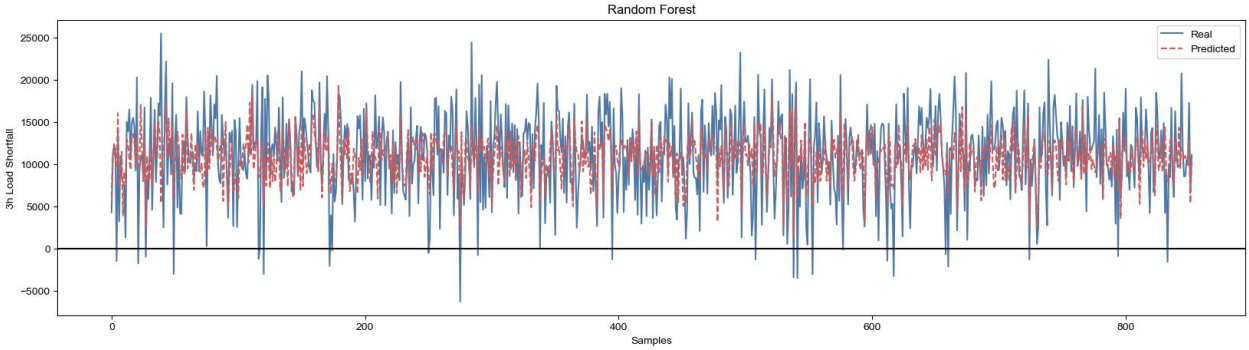


Figura 33. *Comparación de la desatención de la demanda estimada y real obtenida mediante el modelo Random Forest luego del hypertuning.*

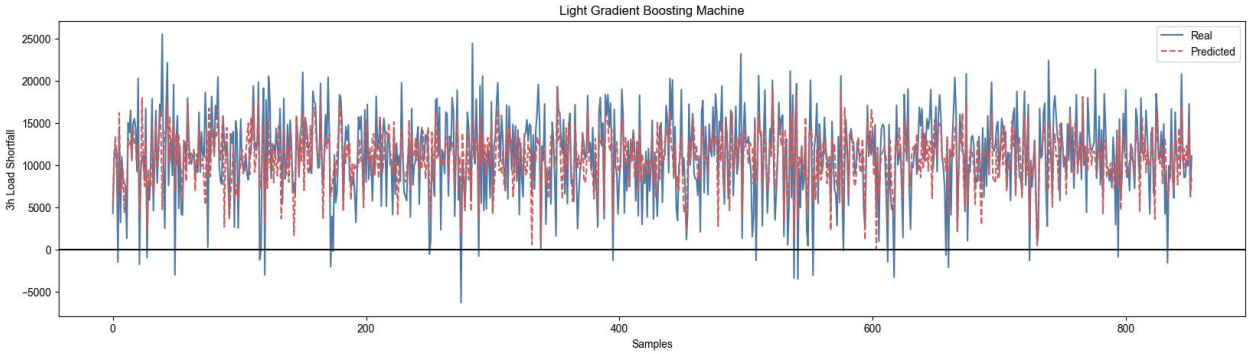


Figura 34. *Comparación de la desatención de la demanda estimada y real obtenida mediante el modelo Light Gradient Boosting Machine luego del hypertuning.*

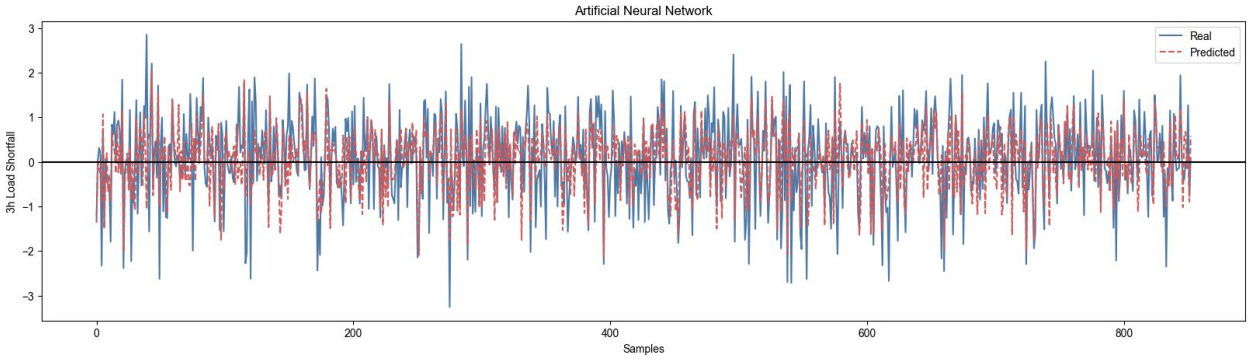


Figura 35. *Comparación de la desatención de la demanda estimada y real obtenida mediante el modelo Artificial Neural Network Machine luego del hypertuning.*

6.2. Evaluación cualitativa

Como se pudo observar, se lograron mejores resultados mediante la implementación de modelos más complejos, con arquitecturas más especializadas, así como a la selección de los mejores hiperparámetros para cada uno de los mismos, a su vez, el tratamiento de datos realizado para la base de datos permitió que los modelos tuvieran un mejor desempeño, al garantizar que las características de las variables de entrada contaran con las especificaciones requeridas, sin embargo no fue posible obtener una métrica superior al valor de referencia tomado por los resultados expuestos en el desafío Kaggle.

A partir de los resultados no se evidencia un sobreajuste en la predicción realizada por los ninguno de los modelos, lo cual era esperado, debido a la implementación de técnicas que minimizan la aparición de este fenómeno, como la división de la base de datos, en segmentos de entrenamiento y prueba y la validación cruzada.

Los resultados dados por el MAE y el RMSE sugieren que los modelos presentan un subajuste, sin embargo, al considerar el comportamiento de los mismos datos por el análisis de componentes principales, la eliminación de características de baja importancia o con datos concentrados en valores de cero y el análisis de correlación, dan muestra que los modelos contaban con las características más adecuadas como información de entrada disponible en el conjunto de datos. El subajuste puede deberse a la baja correlación de las variables de entrada con respecto a la variable de salida, a la falta de un mayor número de muestras en la información de entrada y/o a la regularización implementada que pudo limitar la captura de la información de manera más adecuada.

No se descarta la presencia de datos atípicos en la información con la cual se entrenaron y probaron los modelos, debido a las limitaciones dadas por el algoritmo LOF, el cual requiere de la coincidencia de datos atípicos en todas las características para garantizar su eliminación, no obstante, su impacto debe ser evaluado con más detalle ya que puede comprometerse el desempeño de los modelos al reducir de manera drástica el número de muestras disponibles con base en las técnicas de tratamiento de datos atípicos implementadas.

Todos los procesos implementados para la construcción del mejor modelo de Machine Learning para la predicción de la desatención de la demanda nos permitieron obtener un MAE de

2686 y un RMSE de 3665 unidades energéticas, donde este último es superior al criterio de desempeño establecido de 3262 unidades energéticas.

La diferencia entre las métricas obtenidas y el criterio desempeño podría poner en entredicho la utilidad de los resultados obtenidos, sin embargo, el resultado solo es un 12% superior a la métrica objetivo.

Por lo anterior, este trabajo podrá considerarse como una base de la cual pueden darse más y mejores resultados. con base en la construcción de mejores modelos de Machine Learning los cuales estén más acordes al problema, a la exploración con mayor profundidad y entendimiento la base de datos a la vez que se utilizan técnicas más especializadas para el tratamiento de los mismos.

7. Conclusiones

Este trabajo propuso la búsqueda de un modelo de regresión de Machine Learning con la capacidad de predecir la desatención de la demanda en España, dada por la brecha entre la energía producida a partir de combustibles fósiles y las fuentes de energía basadas en recursos renovables, debido a las condiciones climatológicas. Para alcanzar este objetivo, se implementaron diversas técnicas de preparación de datos y se evaluaron hasta ocho algoritmos de Machine Learning: Logistic Regressor, Random Forest, SVR, Decision Trees, Extratrees Regressor, Gradient Boosting Machine, Extreme Learning Machine y Artificial Neural Network.

En el proceso de preparación de los datos, se adoptaron diversos enfoques, entre los cuales se incluyen la imputación de datos, la eliminación de valores atípicos, la transformación de los datos, la aumentación de datos y reducciones de dimensionalidad. Como es común en cualquier procedimiento de tratamiento de datos, la imputación de datos faltantes y el manejo de valores atípicos desempeñaron un papel crucial en las etapas iniciales, ya que contribuyeron significativamente a mejorar la eficacia de los procesos subsiguientes, como las transformaciones, reducciones de dimensionalidad y la evaluación de diversos modelos de Machine Learning.

En el ámbito de las transformaciones, se destaca la importancia de comprender la naturaleza de cada variable de entrada del modelo. En este contexto específico, se resalta la transformación de variables categóricas que tenían más sentido al tratarse como numéricas para los modelos. Además, se abordó la transformación de variables originalmente numéricas, pero conceptualmente representativas de categorías, sin un orden secuencial lógico. Dada la naturaleza de un problema

de regresión, la codificación de variables categóricas fue esencial para la arquitectura de modelos como SVR, ELM y ANN. La elección de la técnica de codificación más adecuada, en este caso, generalmente correspondió a OneHotEncoding, que binariza cada categoría de las variables categóricas, evitando imponer un orden secuencial, que era el desafío inicial.

Adicionalmente, se identificó la relevancia de la normalización de datos, especialmente para modelos como SVR, ELM y ANN, que son sensibles a las disparidades en la escala de las variables. En términos generales, la estandarización de datos se identificó como la técnica óptima para la normalización. Cabe destacar que los demás modelos no mostraron variaciones significativas en sus métricas de desempeño al utilizar datos estandarizados. Aunque este hallazgo no representó una mejora directa para esos modelos, permitió la consistencia al emplear la misma data de entrada en todas las evaluaciones.

Por otro lado, la aumentación de datos a través de la utilización de la temporalidad original de los registros ha contribuido a la reducción del margen de error en los modelos implementados. En este contexto, se ha aprovechado la estacionalidad inherente al clima en España para crear una nueva variable categórica con cuatro posibles valores: primavera, verano, otoño e invierno. Esta variable adicional ha enriquecido la información durante el entrenamiento de los modelos, permitiéndoles establecer asociaciones más precisas entre cada registro y una temporada específica. Nuevamente, al tratarse de una variable categórica, se ha procedido a su codificación mediante OneHotEncoding.

La estrategia de seleccionar un modelo base y observar la evolución de los resultados de las predicciones al aplicar diversas técnicas de tratamiento de datos demostró ser altamente válida. Esto permitió identificar el impacto de cada técnica en la respuesta del modelo, dando lugar a la construcción de una ruta de optimización manual. En esta ruta, las técnicas que influyeron positivamente en la respuesta del modelo se conservaron para la siguiente iteración, mientras que aquellas con impacto negativo fueron descartadas.

Es importante destacar que, aunque el enfoque iterativo descrito fue beneficioso para este trabajo, puede no ser la opción más adecuada en entornos empresariales de producción. En tales casos, se suele requerir la finalización de casi la totalidad de un pipeline de analítica de datos y Machine Learning para evaluar el impacto de una técnica implementada, lo cual podría resultar

costoso, especialmente en situaciones con grandes volúmenes de datos, modelos complejos o recursos limitados, donde cada iteración cuenta.

Al analizar los resultados obtenidos a través de la base de datos preparada con las técnicas mencionadas, se observó que entre los diferentes modelos aplicados, la Red Neuronal Artificial (ANN) destacó por su rendimiento superior en las métricas de RMSE y MAE, seleccionadas para evaluar la calidad de las predicciones y el desempeño general de los modelos evaluados.

Los resultados indicaron que la ANN logró el mejor ajuste para la desatención de la demanda estimada, con un error RMSE de 3665 unidades energéticas, un 12% superior a la métrica de desempeño referencial de 3262 unidades energéticas.

El proceso de sintonización de hiperparámetros, utilizando la optimización bayesiana mediante el algoritmo del estimador de árboles estructurados de Parzen, en general, mejoró el rendimiento de los modelos sintonizados. Esto se tradujo en una reducción del 21% en el valor de RMSE y del 26% en el MAE para la ANN. Estos resultados subrayan la importancia de una selección adecuada de hiperparámetros en los modelos de Machine Learning y su impacto en las predicciones obtenidas.

En conclusión, aunque los resultados obtenidos no cumplen completamente con las expectativas, se acercan a la métrica referencial de 3262 unidades energéticas. Esto sugiere que el problema abordado presenta oportunidades de mejora, tanto en la conformación y preparación de la base de datos como en la optimización de los modelos de Machine Learning. Se proporcionarán recomendaciones detalladas sobre estos aspectos posteriormente.

8. Recomendaciones

Las predicciones enfocadas en la desatención de la demanda cobran mucho sentido cuando se comprende que con cada día que transcurre, las energías renovables presentan una mayor participación en las matrices energéticas en los países como reemplazo a las fuentes de energía convencionales, con base en el desarrollo de la transición energética. Sin embargo, los factores climatológicos se configuran como el talón de Aquiles de las energías no convencionales, debido a su volatilidad e incertidumbre, los cuales pueden llegar a ser más severos gracias al cambio climático.

Como una de las líneas de investigación futuras propuestas a partir de este trabajo, se encuentra la recolección de datos climatológicos que se encuentren en las regiones en las cuales se concentran las diferentes fuentes de energías no convencionales. Generar un estudio más amplio de la correlación de las variables y la generación de energía, debido a que la información de entrada, presente en la base de datos, mostraba una baja correlación con la desatención de la demanda por lo cual estos datos pudieron no ser los más representativos y adecuados para obtener las mejores predicciones. Los resultados de las predicciones pueden deberse a la baja relación entre las variables climatológicas locales (ciudades) y una desatención de la demanda de carácter más general (país), en el cual las condiciones climatológicas de los centros de generación puedan no correlacionarse a los datos analizados.

Como recomendación futura se propone la exploración de técnicas que consideren la temporalidad que presentan las variaciones climatológicas en un país europeo (España) que cuentan con estaciones climatológicas. Esto puede dar como resultado una variación cíclica de las variables climatológicas, de forma tal, que los algoritmos como el train_test_split que seleccionan de manera aleatoria de la información de entrenamiento y prueba pueden romper estos comportamientos, los cuales podrían resultar útiles para los diferentes algoritmos de Machine Learning.

Algoritmos basados en series de tiempo, para la división y tratamiento de la información de manera temporal pueden transformase en un paradigma capaz de obtener información la cual no fue posible con la aproximación empleada en este trabajo.

Implementar una sintonización de hiperparámetros para los modelos con mejores resultados, pero con un mayor número de iteraciones, algoritmos de parada que mejoren los

tiempos de entrenamiento, así como a la exploración de los términos regularizadores L1, L2 (ANN) alpha y lamda (LightGBM). Estos aspectos pueden permitir que el crecimiento de los pesos no se vea limitado, al punto de no ser capaz de obtener patrones importantes de la información, de tal forma que se puedan obtener mejores métricas, pero garantizado que el crecimiento no afecte el desempeño de los modelos o genere un sobreajuste de los mismos.

Utilizar técnicas más avanzadas de Machine Learning que permitan tratar con las no linealidades y alta volatilidad presentes tanto en las variables de entrada como en la variable de salida. Las redes neuronales recurrentes pueden conseguir mejores resultados que la red neuronal artificial implementada en este trabajo, considerando sus capacidades de memorización. A su vez, pueden tratar y aprender del comportamiento temporal que pueda extraerse del conjunto de datos. Por lo tanto, las redes neuronales recurrentes se configuran como excelente opción para afrontar la predicción de la desatención de la demanda y como modelos bases para las investigaciones futuras.

Referencias

Alzamzami, F. Hoda, M. & Saddik, E. A., (2020). Light Gradient Boosting Machine for General Sentiment Classification on Short Texts: A Comparative Evaluation. IEEE Access, vol. 8, pp. 101840-101858.

Awad, M., & Khanna, R. (2015). Support vector regression. Efficient learning machines: Theories, concepts, and applications for engineers and system designers, 67-80.

Brownlee, J. (2016). Deep learning with Python: develop deep learning models on Theano and TensorFlow using Keras. Introduction to TensorFlow (p.14). Machine Learning Mastery.

Camana, R. M, Ahmed, S. Garcia, E. C & Koo, I. (2020). Extremely Randomized Trees-Based Scheme for Stealthy Cyber-Attack Detection in Smart Grid Networks. IEEE Access, vol. 8, pp. 19921-19933

Chai, T. and Draxler, R. R.: Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature, Geosci. Model Dev., 7, 1247–1250, https://doi.org/10.5194/gmd-7-1247-2014, 2014.

Chen, D. Y. (2017). Pandas for everyone: Python data analysis. Pandas DataFrame Basics (Chapter 1). Addison-Wesley Professional.

Cortinhas, S. (2022). Tuning methods [Online image]. Advanced WandB: Hyper-parameter tuning (sweeps). https://www.kaggle.com/code/samuelcortinhas/advanced-wandb-hyper-parameter-tuning-sweeps

Dawson, C. W., & Wilby, R. (1998). An artificial neural network approach to rainfall-runoff modelling. Hydrological Sciences Journal, 43(1), 47-66.

De Myttenaere, A., Golden, B., Le Grand, B., & Rossi, F. (2016). Mean absolute percentage error for regression models. Neurocomputing, 192, 38-48.

Dewi, K. E., & Widiastuti, N. I. (2020). Support vector regression for GPA prediction. In IOP Conference Series: Materials Science and Engineering (Vol. 879, No. 1, p. 012112). IOP Publishing.

ExploreAI Academy. (2022). EDSA Individual | Electricity Shortfall Challenge. Kaggle. https://kaggle.com/competitions/edsa-individual-electricity-shortfall-challenge

Gunawan, T. S., Ashraf, A., Riza, B. S., Haryanto, E. V., Rosnelly, R., Kartiwi, M., & Janin, Z. (2020). Development of video-based emotion recognition using deep learning with Google Colab. TELKOMNIKA (Telecommunication Computing Electronics and Control), 18(5), 2463-2471.

Idris, I. (2015). NumPy: Beginner's Guide. Preface (p.ix). Packt Publishing Ltd.

Keller, C. A., & Evans, M. J. (2019). Application of random forest regression to the calculation of gas-phase chemistry within the GEOS-Chem chemistry model v10. Geoscientific Model Development, 12(3), 1209-1225.

Konstantinov, V. A., & Utkin, V. L., (2021). Interpretable Machine Learning with an ensemble of gradient boosting machines. Knowledge-Based Systems, 222, 106993.

OpenWeather. (s.f.). Weather Conditions. https://openweathermap.org/weather-conditions#Weather-Condition-Codes-2. Consultado el 5 de noviembre de 2023.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. the Journal of Machine Learning research, 12, 2825-2830.

Sazli, M. H. (2006). A brief review of feed-forward neural networks. Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering, 50(01).

Taha, A. A., & Malebary, J. S., (2020). An Intelligent Approach to Credit Card Fraud Detection Using an Optimized Light Gradient Boosting Machine. IEEE Access, vol. 8, pp. 25579-25587.

Tosi, S. (2009). Matplotlib for Python developers (Preface). Packt Publishing Ltd.

Van Rossum, G. (2003). An introduction to Python (p.3). F. L. Drake (Ed.). Bristol: Network Theory Ltd.

Wang, J., Lu, S., Wang, S. H., & Zhang, Y. D. (2022). A review on extreme learning machine. Multimedia Tools and Applications, 81(29), 41611-41660.

Waskom, M. L. (2021). Seaborn: statistical data visualization (p.1). Journal of Open Source Software, 6(60), 3021.

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate research, 30(1), 79-82.

World Meteorological Organization. (2012). World: Highest Sea Level Air Pressure Above 750 meters. https://web.archive.org/web/20121017130834/http://wmo.asu.edu/highest-sea-lvl-air-pressure-above-700m. Consultado el 5 de noviembre de 2023.

Yang, L., & Shami, A. (2020). On hyperparameter optimization of Machine Learning algorithms: Theory and practice. Elsevier Neurocomputing, no. 415, Nov. 2020. 295–316

Anexos

Anexo A. Visualizaciones complementarias

Iteración 6.

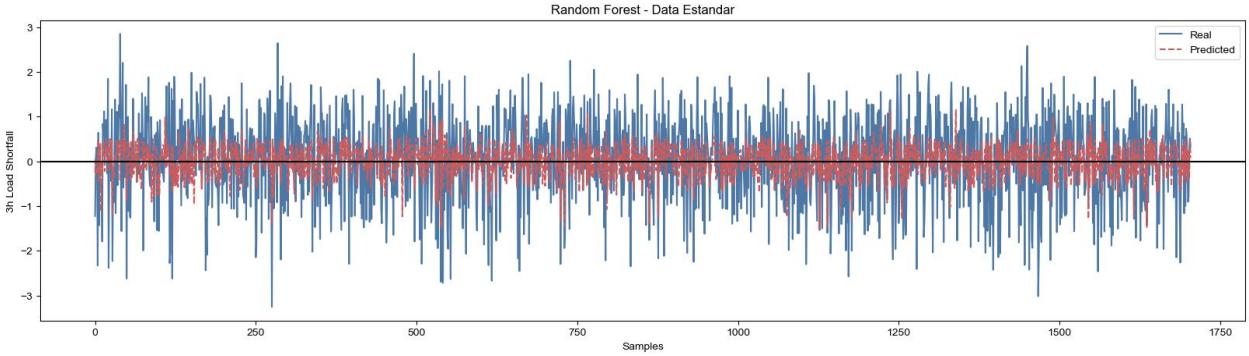


Figura A1. *Comparación visual entre predicciones y valores reales al usar Random Forest con datos normalizados mediante estandarización– iteración 6*

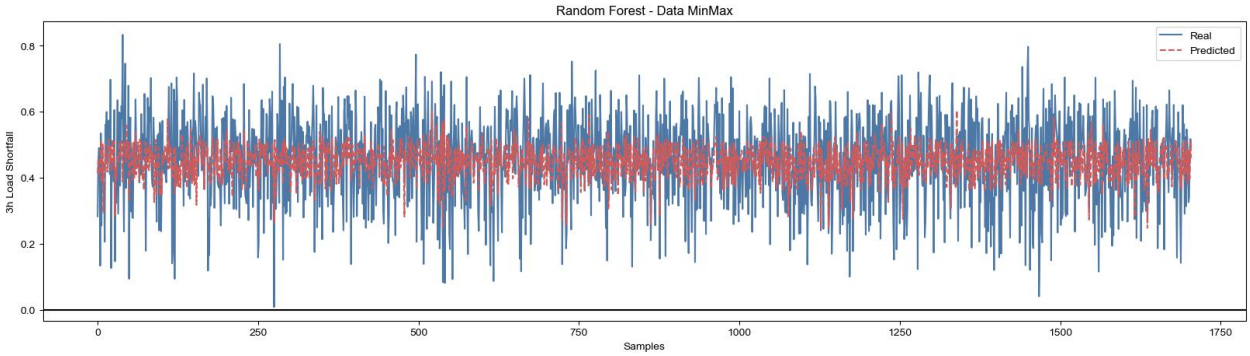


Figura A2. *Comparación visual entre predicciones y valores reales al usar Random Forest con datos normalizados mediante MinMax– iteración 6*

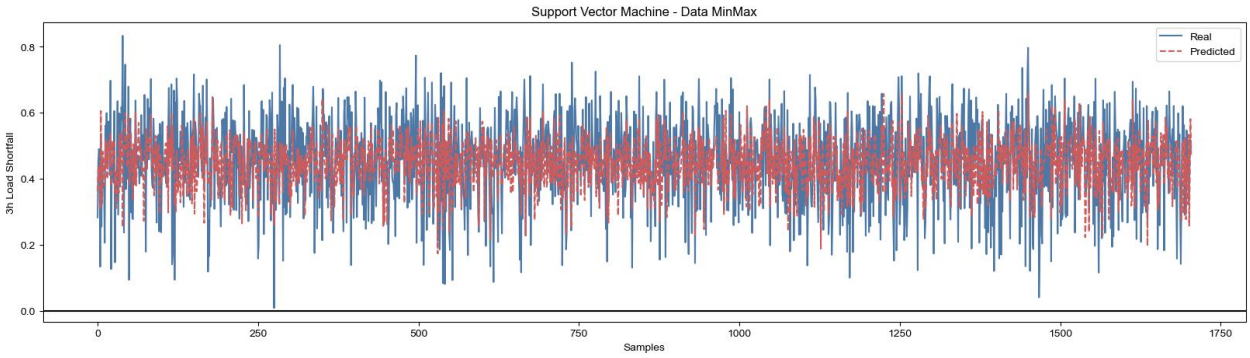


Figura A3. *Comparación visual entre predicciones y valores reales al usar SVR con datos normalizados mediante estandarización– iteración 6*

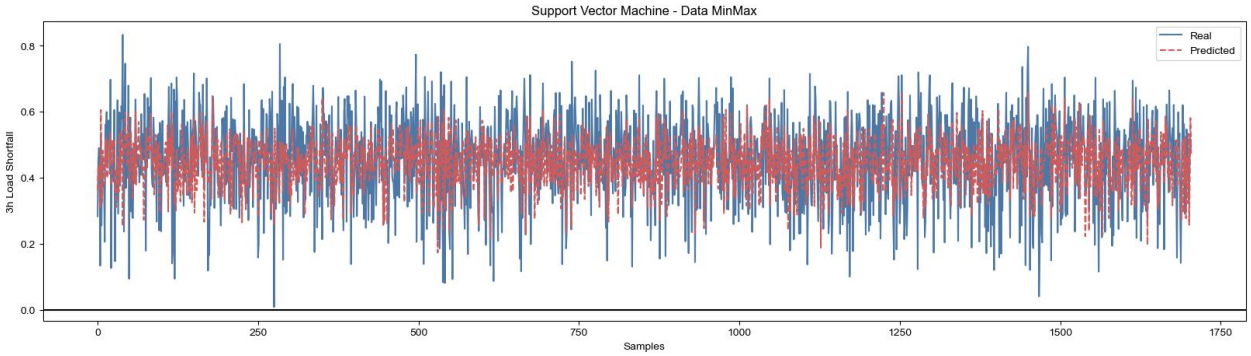


Figura A4. *Comparación visual entre predicciones y valores reales al usar SVR con datos normalizados mediante MinMax– iteración 6*

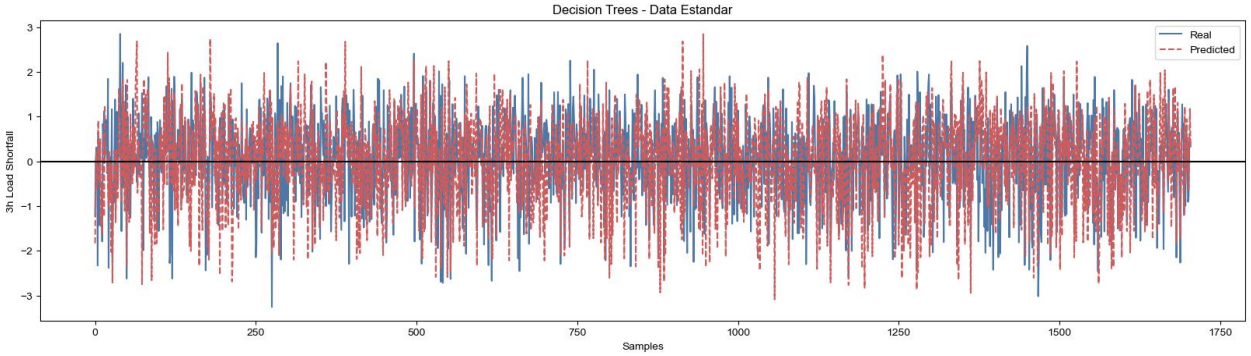


Figura A5. *Comparación visual entre predicciones y valores reales al usar DT con datos normalizados mediante estandarización– iteración 6*

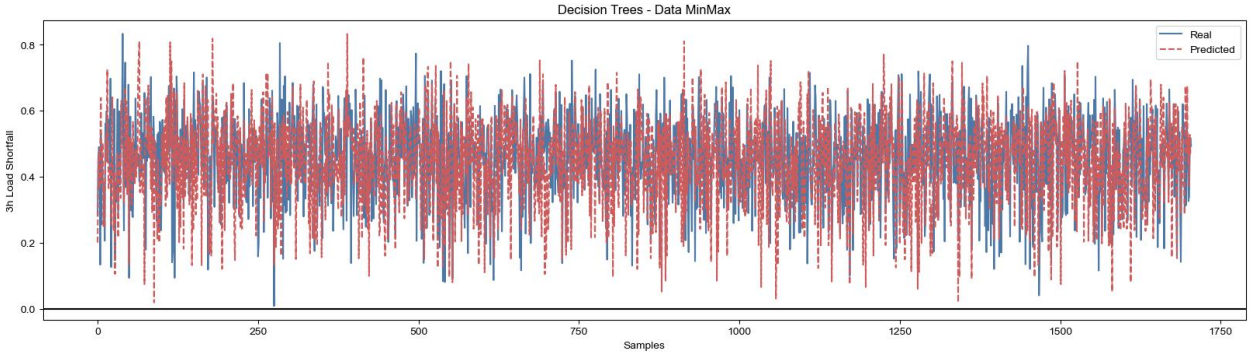


Figura A6. *Comparación visual entre predicciones y valores reales al usar DT con datos normalizados mediante MinMax– iteración 6*

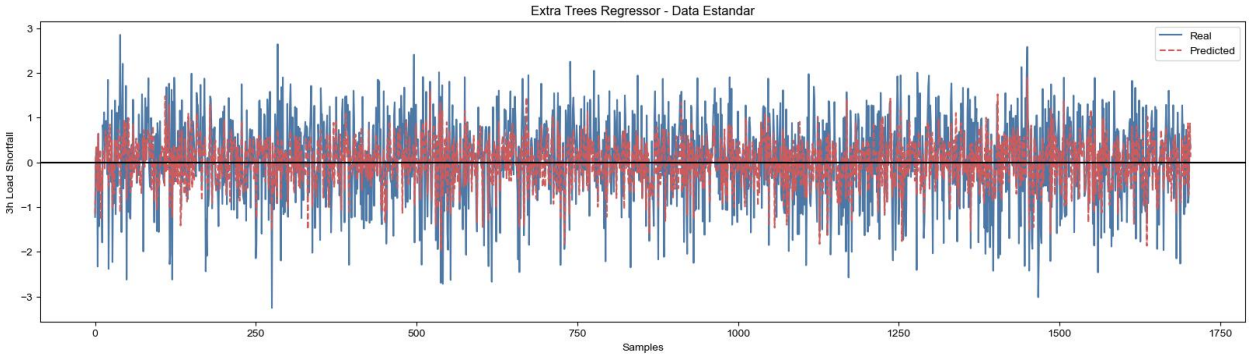


Figura A7. *Comparación visual entre predicciones y valores reales al usar ETR con datos normalizados mediante estandarización– iteración 6*

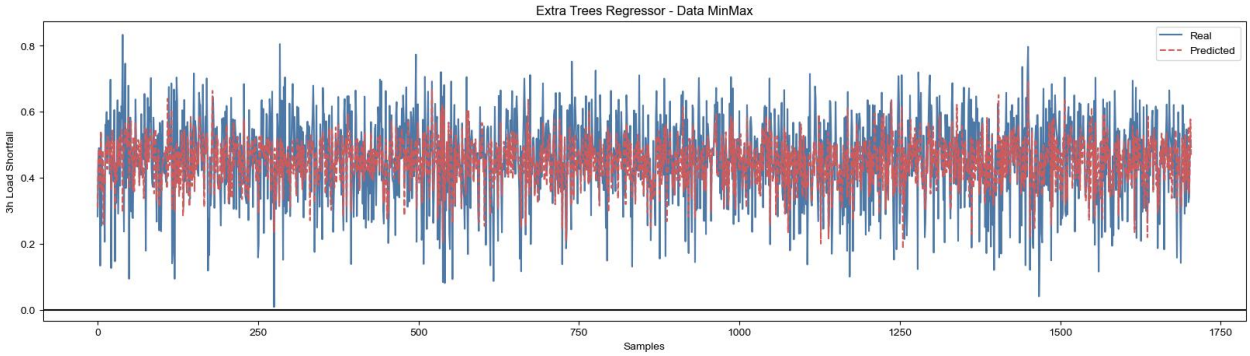


Figura A8. *Comparación visual entre predicciones y valores reales al usar ETR con datos normalizados mediante MinMax– iteración 6*

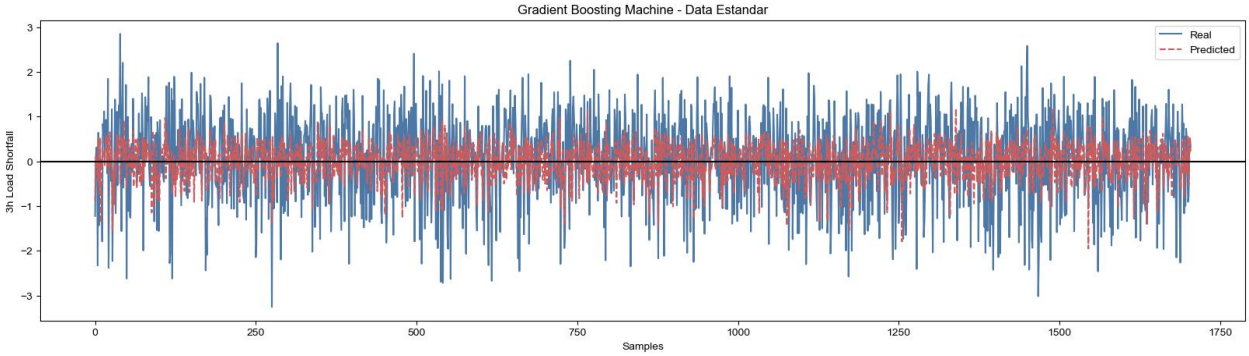


Figura A9. *Comparación visual entre predicciones y valores reales al usar GBM con datos normalizados mediante estandarización– iteración 6*

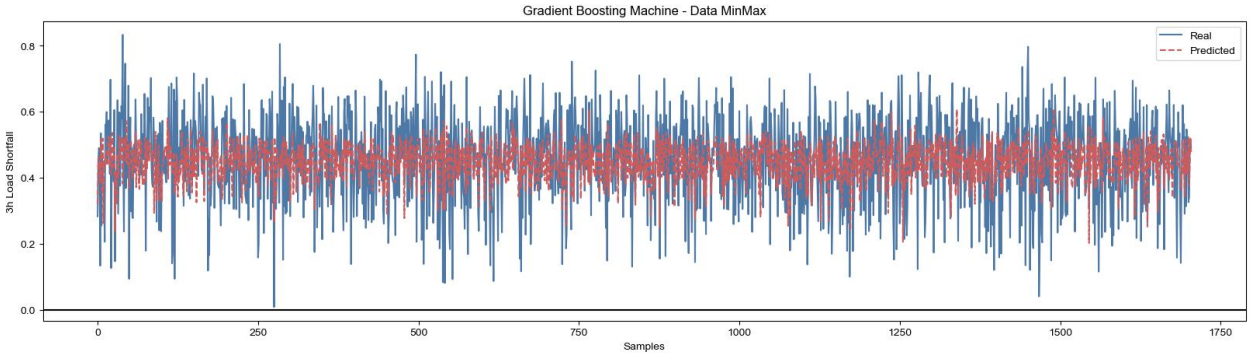


Figura A10. *Comparación visual entre predicciones y valores reales al usar GBM con datos normalizados mediante MinMax– iteración 6*

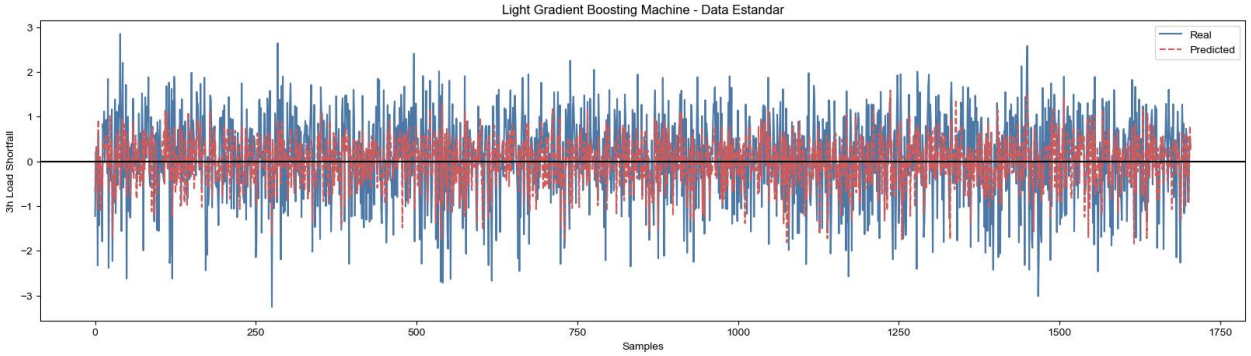


Figura A11. *Comparación visual entre predicciones y valores reales al usar LightGBM con datos normalizados mediante estandarización– iteración 6*

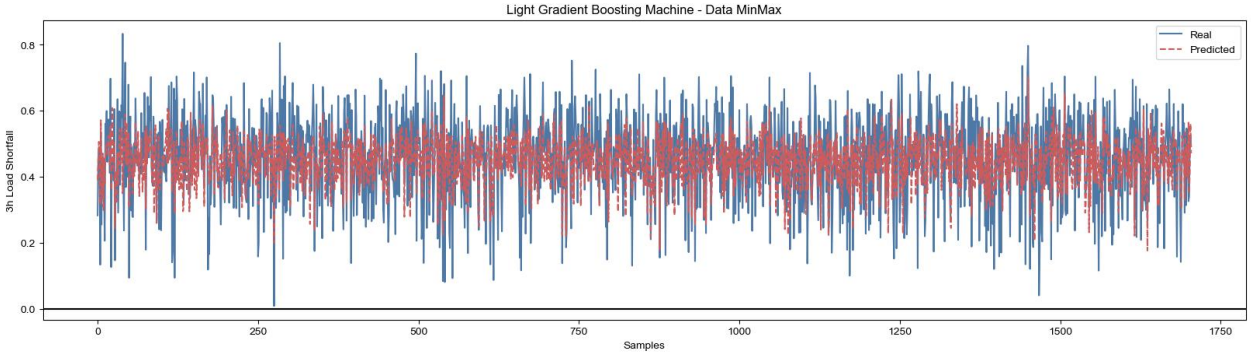


Figura A12. *Comparación visual entre predicciones y valores reales al usar LightGBM con datos normalizados mediante MinMax– iteración 6*

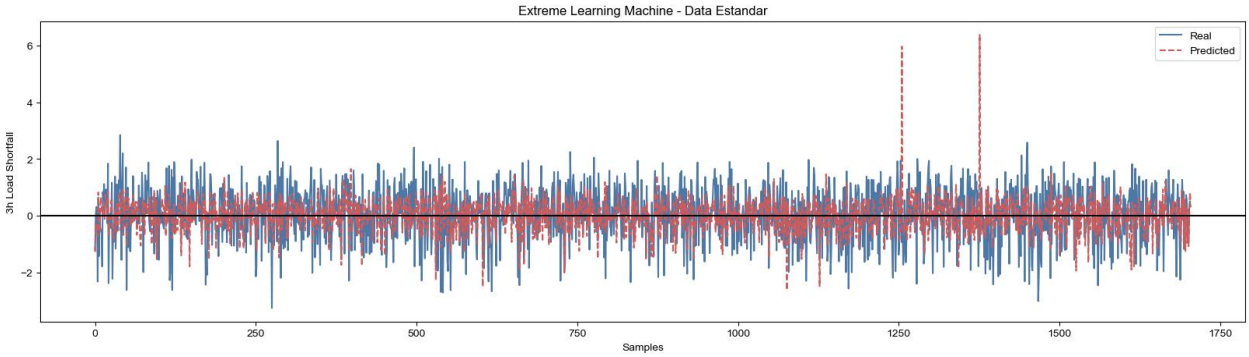


Figura A13. *Comparación visual entre predicciones y valores reales al usar ELM con datos normalizados mediante estandarización– iteración 6*

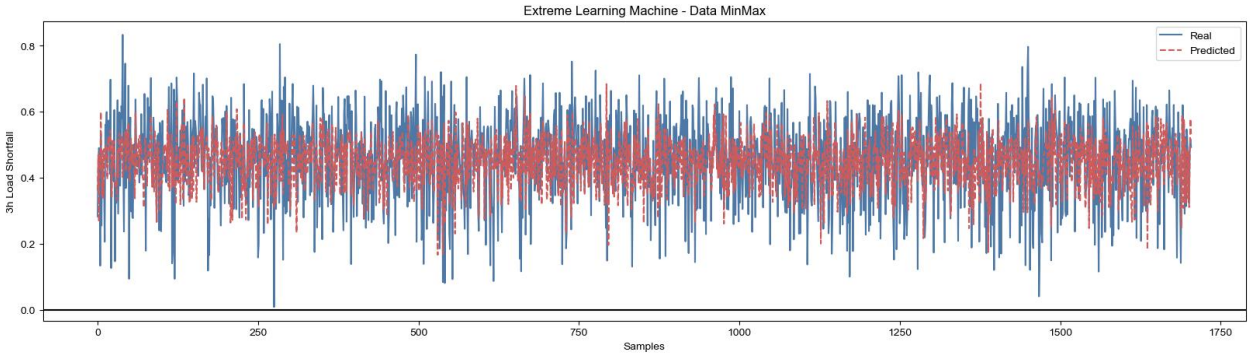


Figura A14. *Comparación visual entre predicciones y valores reales al usar ELM con datos normalizados mediante MinMax– iteración 6*

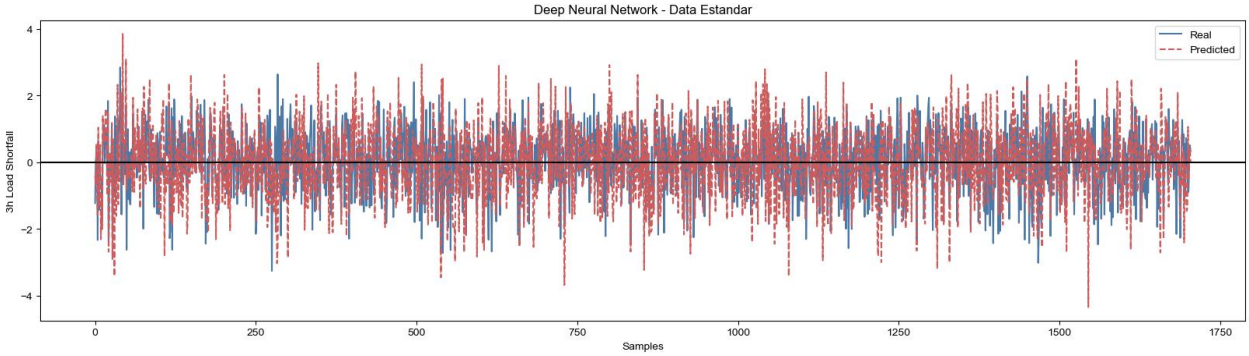


Figura A15. *Comparación visual entre predicciones y valores reales al usar ANN con datos normalizados mediante estandarización– iteración 6*

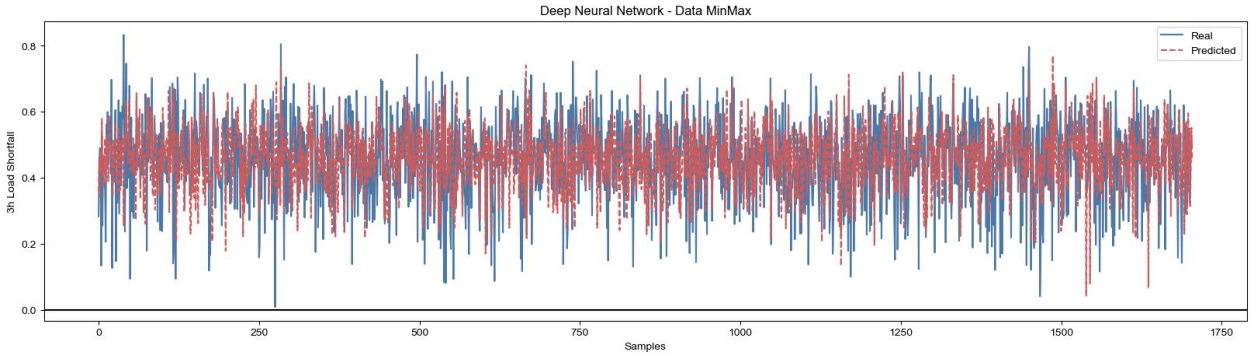


Figura A16. *Comparación visual entre predicciones y valores reales al usar ANN con datos normalizados mediante MinMax– iteración 6*

Iteración 7.

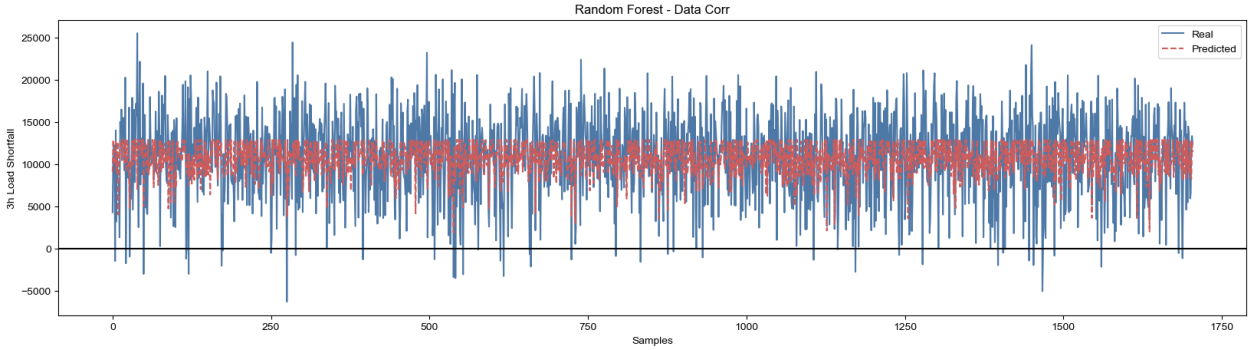


Figura A17. *Comparación visual entre predicciones y valores reales al usar Random Forest con datos reducidos por correlación – iteración 7*

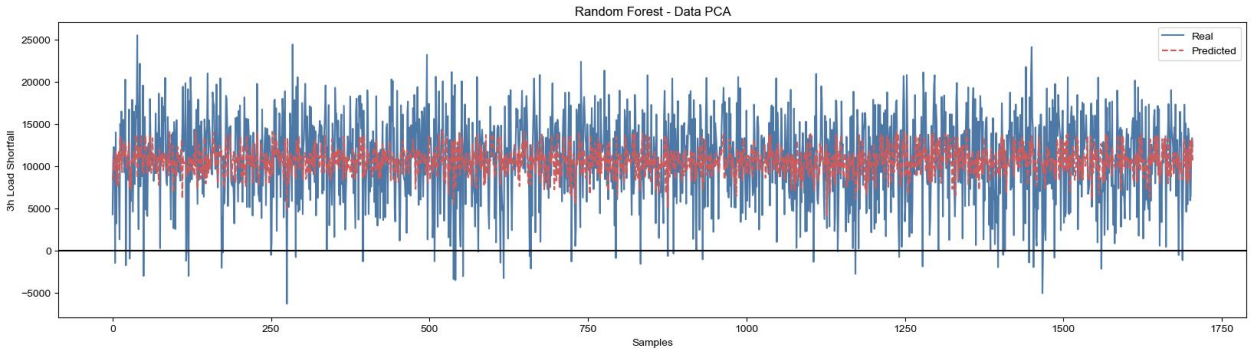


Figura A18. *Comparación visual entre predicciones y valores reales al usar Random Forest con datos reducidos mediante PCA – iteración 7*

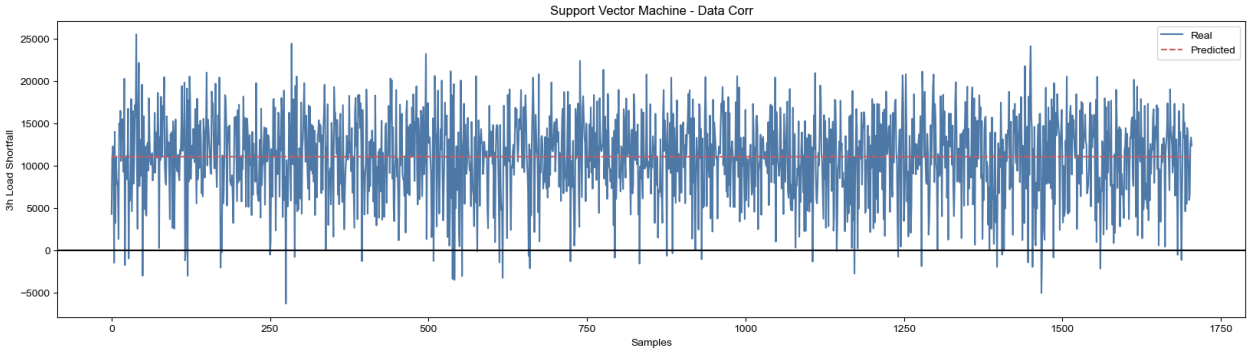


Figura A19. *Comparación visual entre predicciones y valores reales al usar SVR con datos reducidos por correlación – iteración 7*

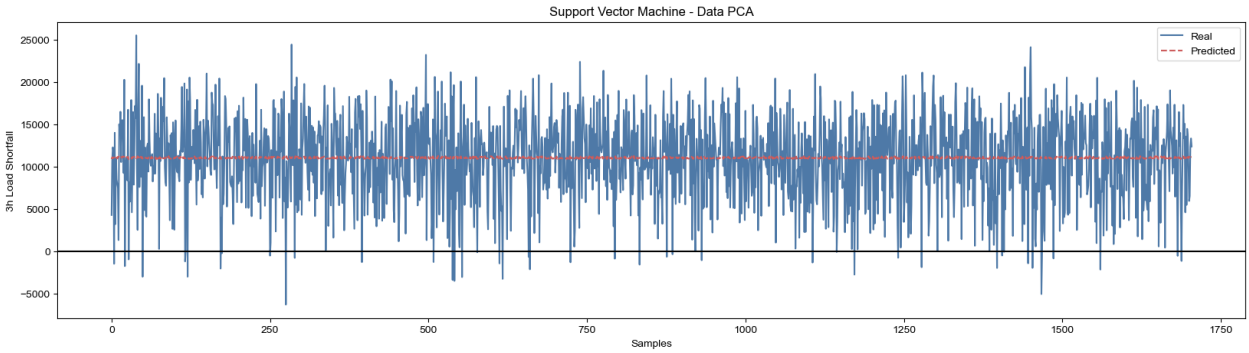


Figura A20. *Comparación visual entre predicciones y valores reales al usar SVR con datos reducidos mediante PCA – iteración 7*

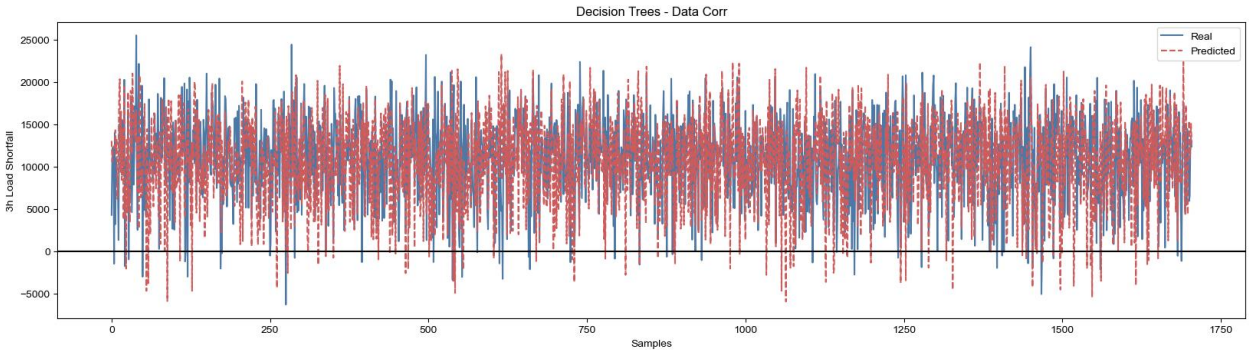


Figura A21. *Comparación visual entre predicciones y valores reales al usar DT con datos reducidos por correlación – iteración 7*

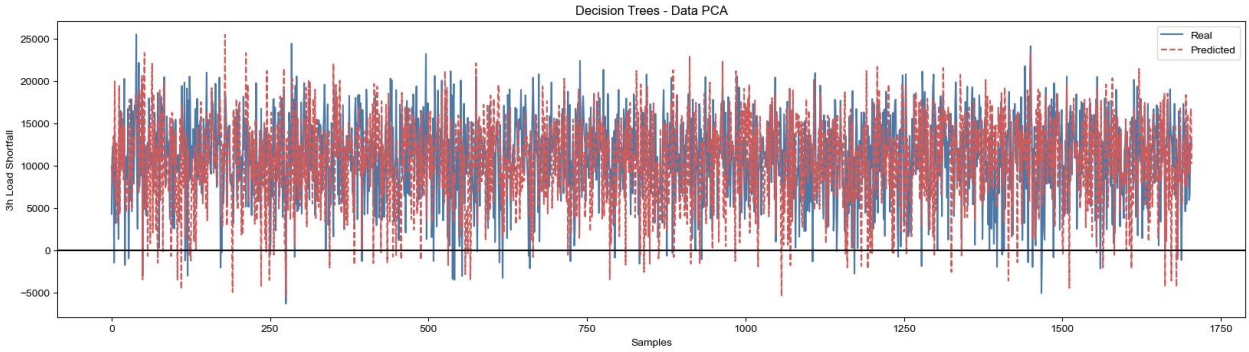


Figura A22. *Comparación visual entre predicciones y valores reales al usar DT con datos reducidos mediante PCA – iteración 7*

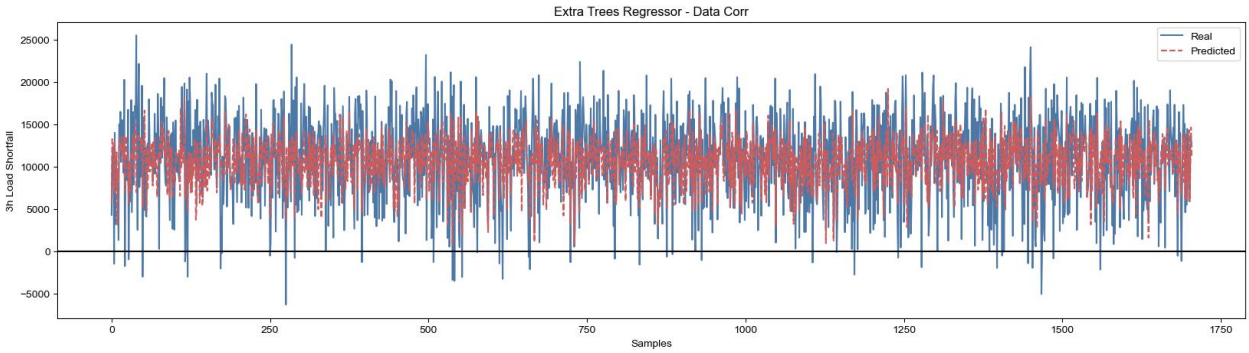


Figura A23. *Comparación visual entre predicciones y valores reales al usar ETR con datos reducidos por correlación – iteración 7*

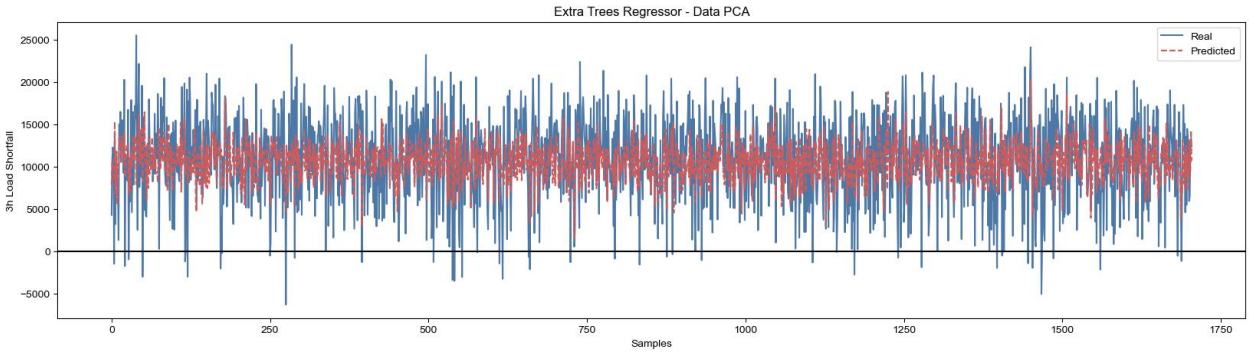


Figura A24. *Comparación visual entre predicciones y valores reales al usar ETR con datos reducidos mediante PCA – iteración 7*

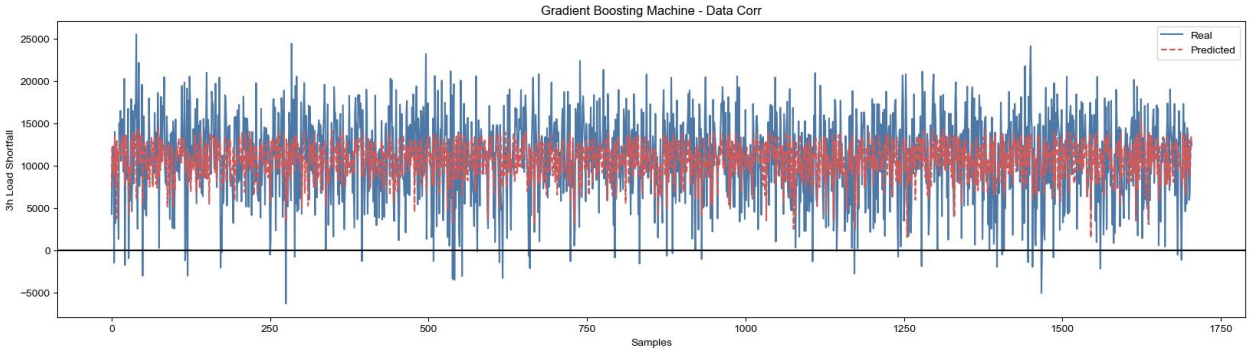


Figura A25. *Comparación visual entre predicciones y valores reales al usar GBM con datos reducidos por correlación – iteración 7*

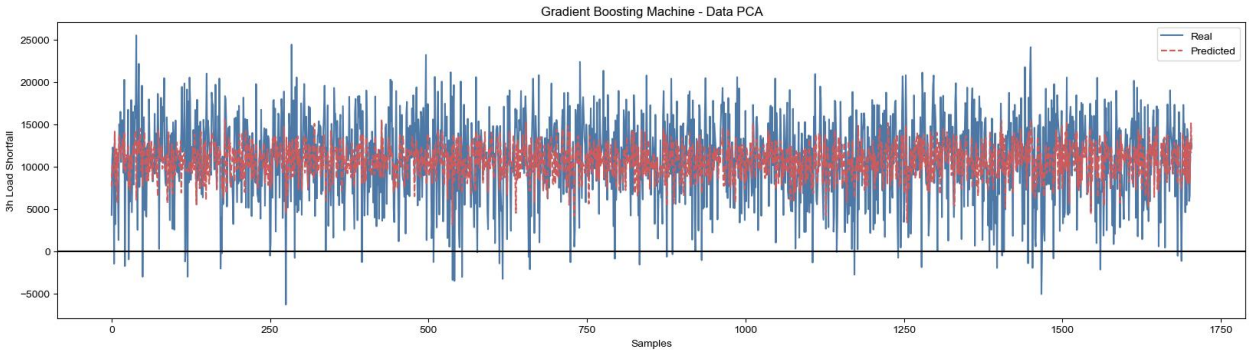


Figura A26. *Comparación visual entre predicciones y valores reales al usar GBM con datos reducidos mediante PCA – iteración 7*

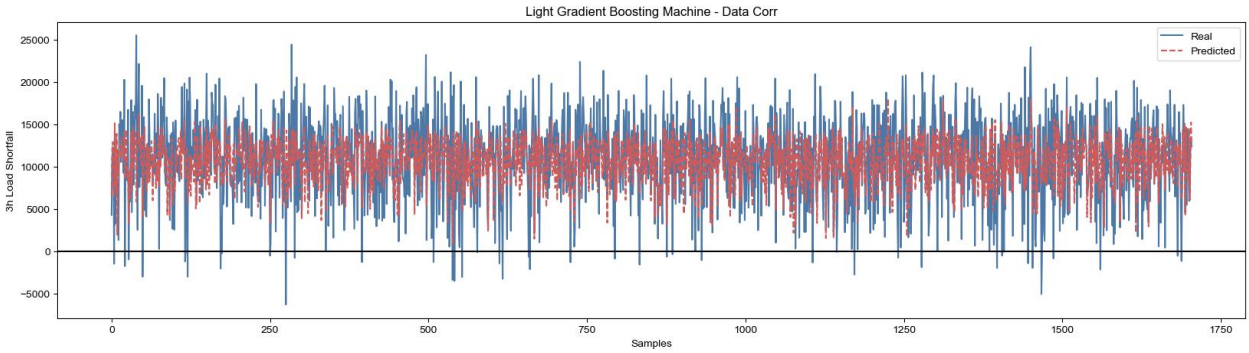


Figura A27. *Comparación visual entre predicciones y valores reales al usar LightGBM con datos reducidos por correlación – iteración 7*

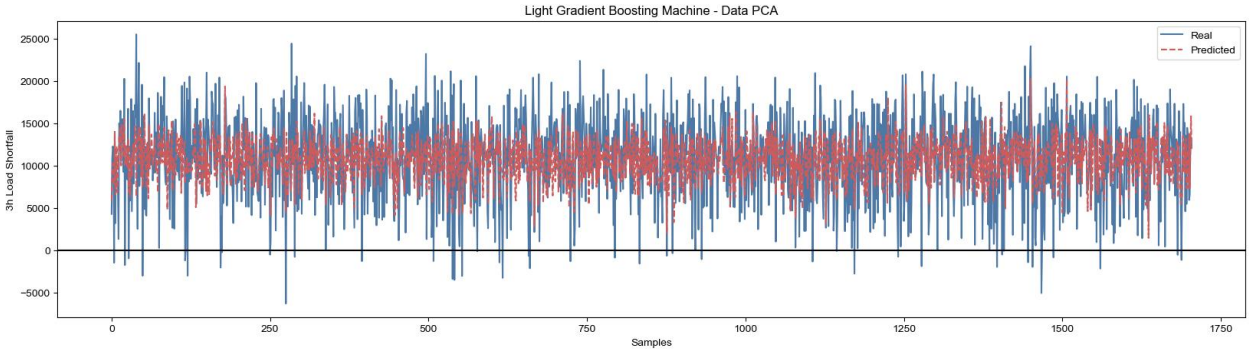


Figura A28. *Comparación visual entre predicciones y valores reales al usar LightGBM con datos reducidos mediante PCA – iteración 7*

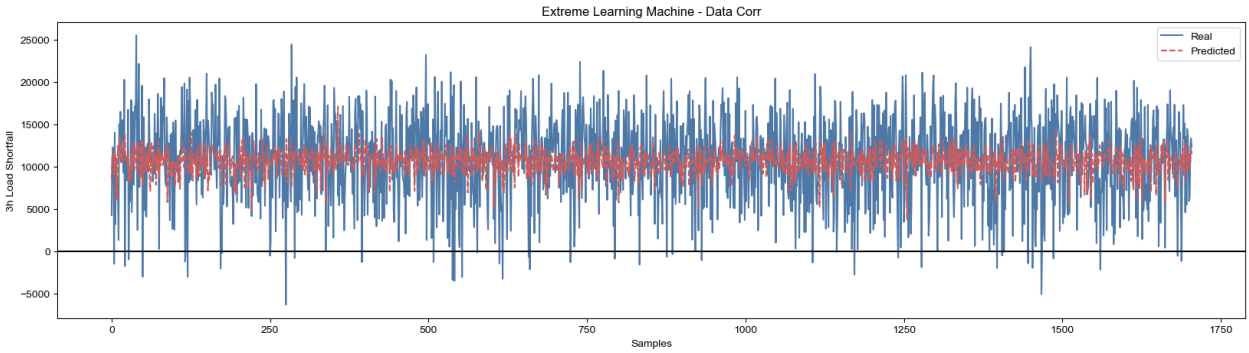


Figura A29. *Comparación visual entre predicciones y valores reales al usar ELM con datos reducidos por correlación – iteración 7*



Figura A30. *Comparación visual entre predicciones y valores reales al usar ELM con datos reducidos mediante PCA – iteración 7*

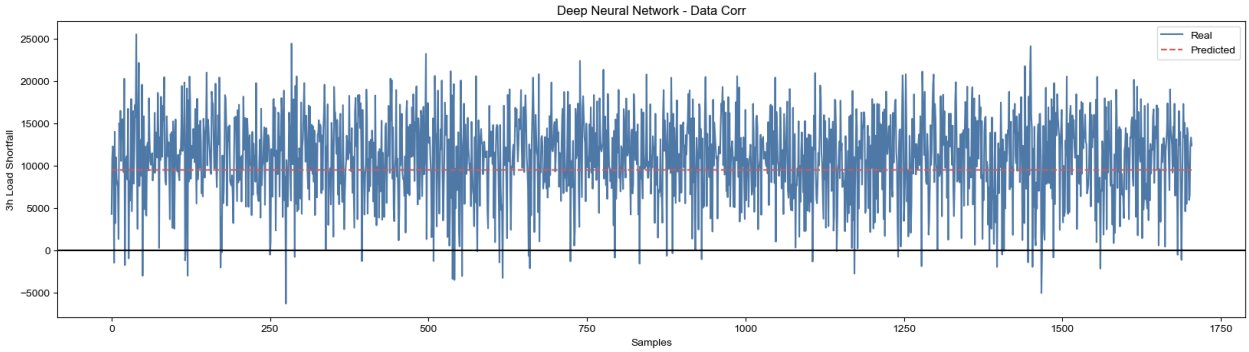


Figura A31. *Comparación visual entre predicciones y valores reales al usar ANN con datos reducidos por correlación – iteración 7*

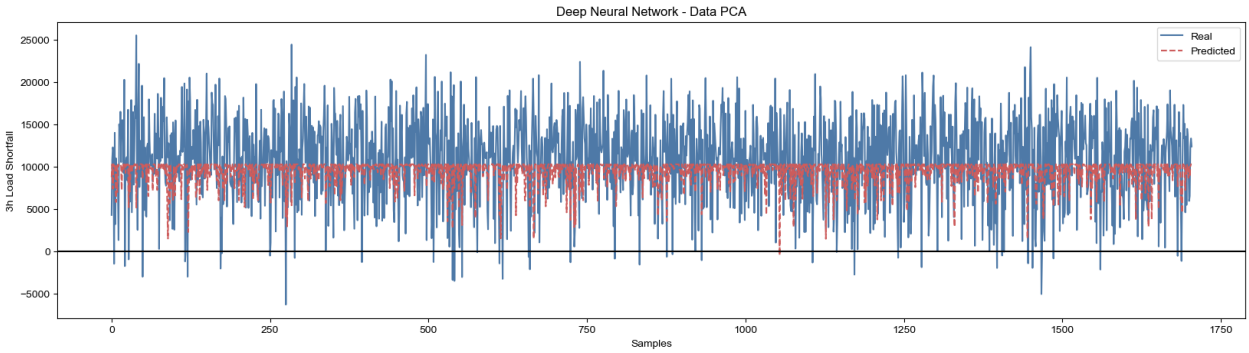


Figura A32. *Comparación visual entre predicciones y valores reales al usar ANN con datos reducidos mediante PCA – iteración 7*