Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN CIENCIA E ENXEÑARÍA DE DATOS

Análisis de datos y predicciones de series temporales para infraestructuras de saneamiento de aguas en áreas de precipitaciones intensas: Life Reseau

Estudante: Diego Martínez Muiño

Dirección: María González Taboada

Dirección: Saúl Díaz García

A Coruña, junio de 2024.

A mis padres, mi hermana Marta y mi amigo Adrián

Agradecimientos

En primer lugar, agradecer a mi familia por apoyarme y acompañarme durante todas las etapas de mi vida. De la misma forma, a mi amigo Adrián por su apoyo incondicional.

Por otro lado, agradecer a María González Taboada y a Saúl Díaz García por guiarme durante la elaboración de este proyecto y brindarme el soporte necesario para su correcta realización. También agradecer al Instituto Tecnológico de Galicia por permitirme colaborar y desarrollar este proyecto.

Por último, gracias a todos aquellos amigos, compañeros, familiares y profesores que me han acompañado durante estos años.

Resumen

La iniciativa Life Reseau surge para el estudio y actuación ante los efectos del cambio climático en infraestructuras de aguas residuales urbanas, concretamente en zonas que experimentan elevadas precipitaciones anuales, sobre las cuales recae el tratamiento de grandes volúmenes de agua de diversas fuentes.

Durante el desarrollo de este Trabajo de Fin de Grado (TFG), se llevarán a cabo diversas tareas para el tratamiento y procesamiento de los datos recolectados en las Estaciones Depuradoras de Aguas Residuales (EDARs) en relación con los caudales asumidos por estas estructuras. Esto permitirá a las empresas colaboradoras conocer, optimizar y planificar los recursos hídricos en estas áreas, así como reducir el impacto y el volumen de residuos producidos en estos procesos.

A continuación, usando la información previamente recopilada y debidamente procesada, se pretende desarrollar diferentes algoritmos y modelos de aprendizaje automático, los cuales nos permitirán analizar y realizar predicciones sobre series temporales relacionadas con la actividad de dichas estructuras, lo que será esencial para la futura elaboración de planes de actuación adecuados para afrontar los efectos del cambio climático en las zonas circundantes.

Abstract

The Life Reseau initiative aims to study and act on the effects of climate change on urban wastewater infrastructures, specifically in areas experiencing high annual rainfall, on which the treatment of large volumes of water from various sources relies.

During the development of this Final Degree Project, several tasks will be carried out for the treatment and processing of the data collected in the WWTP (Wastewater Treatment Plants) in relation to the flows assumed by these structures. This will allow the collaborating companies to know, optimise and plan the water resources in these areas, as well as to reduce the impact and volume of waste produced in these processes.

Then, using the information previously collected and duly processed, we intend to develop different algorithms and machine learning models, which will allow us to analyse and provide predictions on time series related to the activity of these structures, which will be essential for

the future development of appropriate action plans to deal with the effects of climate change in the surrounding areas.

Palabras clave:

- Series temporales
- Aprendizaje automático
- Cambio climático
- Predicciones
- Gestión eficiente del agua
- Ingeniería de datos

Keywords:

- Time Series
- Machine Learning
- Climate change
- Forecasting
- Efficient water management
- Data Engineering

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Como punto de partida, en esta primera sección del documento se abordarán con mayor profundidad los objetivos de este Trabajo de Fin de Grado (TFG), así como el contexto en que ha sido desarrollado. Además, se comentará la estructura de la memoria del proyecto.

1.1 Contexto del proyecto

Este TFG ha sido realizado en colaboración con el Instituto Tecnológico de Galicia (ITG) [12], reconocido por el Ministerio de Economía y Competitividad como Centro Tecnológico Nacional. ITG desarrolla su actividad en ámbitos como la Inteligencia Artificial y tecnologías asociadas como la realidad virtual y aumentada, visión artificial o el procesamiento del lenguaje natural. Cuenta también con los correspondientes departamentos centrados en el Ciclo Integral del Agua y construcción sostenible. En este caso, el proyecto en cuestión involucra tanto al grupo de Datos como al departamento de Agua, entre otros.

El proyecto en el que se integra la realización de este TFG trata precisamente de la iniciativa Life Reseau [13], enmarcada en el plan Life [14] de la Unión Europea (UE), dirigida a la investigación y control de los efectos del cambio climático que afecta gravemente a muchas zonas de nuestro continente, desestabilizando los periodos lluviosos y provocando una mayor precipitación anual y una mayor frecuencia de eventos lluviosos. Debido a esto, las infraestructuras de aguas residuales urbanas se ven afectadas, transportando por las redes de saneamiento unitarias una mayor cantidad de escorrentía, viéndose superadas y no pudiendo hacer un correcto análisis y procesado de estos caudales. Como consecuencia de estos sobreflujos, se generan descargas desde estos sistemas unitarios de recogida sobre las Estaciones Depuradoras de Aguas Residuales (EDARs), causando estos alivios de aguas no tratadas un severo impacto medioambiental y suponiendo un aumento en la contaminación de los medios acuáticos costeros y fluviales cercanos a estas estaciones.

LIFE RESEAU busca minimizar las descargas de estos sistemas de saneamiento en áreas de fuertes precipitaciones, donde estos efectos del cambio climático son más pronunciados, mediante el desarrollo y validación de novedosas soluciones para la actualización de las EDAR. Este proyecto se desarrolla y prueba en dos ubicaciones estratégicas: las instalaciones de la EDAR de Moaña, en España, y de la EDAR de la ciudad danesa *Søndersø* (se emplea su grafía española *Sonderso*). Estas estaciones de tratamiento de agua son gestionadas por diferentes empresas externas.

Este TFG estará específicamente destinado al trabajo desarrollado en la estación danesa, utilizando la EDAR de Moaña en secciones concretas de la memoria para analizar diferentes aspectos comparativos.

1.2 Motivación del TFG y objetivos concretos

En base al contexto del proyecto explicado en la Sección 1.1, el trabajo que se expone en este TFG tiene como motivación principal la búsqueda de posibles soluciones ante las problemáticas que aborda la iniciativa Life Reseau desde el punto de vista del análisis y la ingeniería de datos, centrándonos en la revisión analítica y procesamiento de los datos referentes a los caudales recibidos por dichas EDAR.

Como se mencionaba anteriormente, el proceso de canalización y tratamiento del agua de la red de saneamiento puede verse gravemente alterado en situaciones de climatología adversa, sobre todo en presencia de altas precipitaciones. En estos casos, el mayor volumen de agua entrante en la red provoca una situación crítica en las EDAR, incapaces de asumir este aumento de carga, y que se ven obligadas a liberar al medio acuático grandes volúmenes de agua que no ha sido debidamente tratada. Ante esta situación crítica nace la necesidad de la elaboración de este trabajo.

En esta ocasión, la finalidad última de este TFG consiste en aplicar diversas técnicas estadísticas y modelos de Aprendizaje Automático (AA) para la predicción de los aumentos de los caudales de la red, permitiendo preparar las EDAR adecuadamente ante estos acontecimientos y mejorar significativamente la eficiencia del proceso de tratamiento del agua.

Para ello, se seguirán una serie de pasos que permitirán obtener información de los datos con los que contamos así como obtener las predicciones finales:

- Extracción, análisis y procesado de información de datos enviados por el cliente y de servicios de monitorización meteorológica.

- Detección y tratamiento de datos anómalos y/o faltantes.
- Desarrollo y entrenamiento de diferentes modelos de AA para la predicción de las fluctuaciones del caudal en diferentes escenarios.

1.3 Estructura de la memoria

La memoria de este proyecto recoge el contexto, objetivos, técnicas aplicadas y resultados obtenidos durante el proyecto. Está estructurada en varios capítulos que recogen estos diferentes bloques o tipos de trabajos realizados para llevar a cabo el TFG. En concreto, la memoria consta de los siguientes capítulos:

1. **Introducción (1)**: Se expone el contexto en que se desarrolla el trabajo, la motivación del mismo y sus objetivos concretos.
2. **Metodología y planificación (2)**: Se explica el método de trabajo que ha permitido el desarrollo eficiente del proyecto, así como los costos económicos humanos y materiales del mismo y una muestra de la planificación en cuánto a fechas que ha requerido.
3. **Herramientas utilizadas (3)**: Como en cualquier proyecto del ámbito tecnológico, es muy importante conocer las herramientas de las que se dispone para la realización de un proyecto. En esta sección se detallan aquellas que han sido esenciales para llevar a cabo este TFG y sus principales aplicaciones.
4. **Descripción del dominio (4)**: En este capítulo se detallan los fundamentos teóricos de las diferentes técnicas aplicadas durante el proyecto, como los modelos de AA, técnicas de tratamiento de datos o las métricas utilizadas para la comparación de los modelos. Además, se aporta información básica acerca de las EDAR u otros componentes del ciclo del agua que pueden ser esenciales para comprender otros detalles del proyecto.
5. **Estudio de los conjuntos de datos (5)**: Previo a su utilización, los datos han requerido de un proceso de limpieza, corrección e integración. En este capítulo se detallan las técnicas aplicadas para ello, así como un análisis de la información extraíble de los datos.
6. **Procesado de datos anómalos (6)**: En este sexto capítulo se expone el proceso de detección y corrección de los datos señalados como anómalos, que son frecuentes en mediciones del medio físico en entornos con gran volumen de agua en circulación.
7. **Modelado (7)**: Se muestran las diferentes ejecuciones de los modelos de AA expuestos anteriormente de forma gráfica y sus métricas.

8. **Conclusiones (8)**: Finalmente, se recopilan y comparan los resultados obtenidos, se exponen las conclusiones extraídas de este proyecto, los conocimientos adquiridos por el alumno y su relación con las competencias del grado en Ciencia e Ingeniería de Datos. Además, se presentarán los siguientes pasos en el desarrollo de este proyecto fuera del alcance del TFG.

9. **Acrónimos, glosario y bibliografía**: Capítulos recopilatorios donde se enumeran los diferentes acrónimos empleados en la memoria junto a su significado, así como un breve glosario de definiciones de términos empleados en distintas ocasiones en los que no se ha entrado en detalle. Finalmente, se recogen las relaciones bibliográficas o fuentes referenciadas en las diferentes secciones de la memoria.

Capítulo 2

Metodología y planificación

En este capítulo se detallarán cuestiones del ámbito de la planificación del proyecto y la metodología de trabajo empleada para su desarrollo, así como las herramientas necesarias para ello y una estimación de costes del mismo. Todo ello aportará información sobre el volumen de trabajo y los requerimientos técnicos y materiales para el desarrollo de un proyecto de estas características.

2.1 Método de trabajo adoptado

Dada la naturaleza y el contexto en que se desarrolla este proyecto, dentro de un equipo de trabajo de ITG y colaborando con empresas externas a modo de clientes, se han llevado a cabo reuniones diarias con el equipo de trabajo y reuniones semanales con los otros departamentos involucrados. Estas reuniones periódicas están orientadas a la planificación de las fases de trabajo a corto plazo, que son la fase principal de la metodología de trabajo ágil denominada *Scrum* [15], como se muestra en la Figura 2.1. En cada una de ellas, los objetivos parciales han podido ir variando y ajustándose a mejoras y nuevas especificaciones, sobre todo teniendo en cuenta el envío periódico de datos por parte de los clientes, lo que supone nuevos ajustes y mejoras. De hecho, son estas características (la mejora continua y la flexibilidad) algunas de las principales ventajas de esta metodología de trabajo.

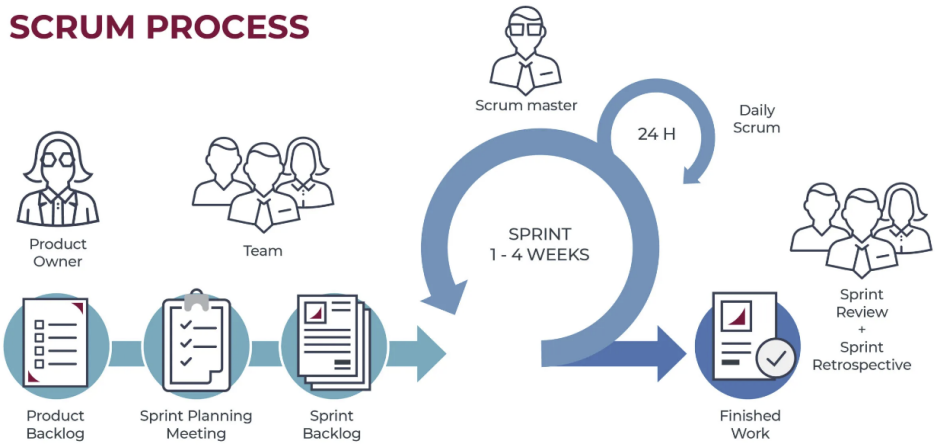


Figura 2.1: Fases típicas de la metodología SCRUM [1].

Como se indicaba en la Sección 1.2, el proyecto se ha dividido en tres tipos de trabajos o tareas principales, siguiendo cada una de ellas en sí misma un ciclo de trabajo o proceso Scrum:

1. **Extracción, análisis, procesado e integración de datos**: Al contar con distintas empresas externas participantes, cada una encargada de la gestión de una EDAR, con procesos internos específicos y formatos de datos diferentes, esta primera tarea será esencial para la continuidad del proyecto.

   De forma más o menos periódica, estas empresas han realizado envíos de datos procedentes de dichas estaciones que, como se explicará más adelante en esta memoria, han de ser tratados de forma específica y atendiendo a las indicaciones y particularidades de cada fuente. Esta actualización constante de la fuente de datos, así como el consiguiente procesado de los mismos, genera procesos Scrum en los que se toman decisiones acerca de la información extraída de los nuevos datos y de los nuevos objetivos en base a ellos.

2. **Detección y tratamiento de datos anómalos o faltantes**: Además del análisis exploratorio de los datos, en algunos casos será necesario el análisis pertinente para detectar y procesar debidamente las posibles mediciones anómalas de los datos. Como es frecuente en mediciones de grandes masas de agua en movimiento constante, la precisión de las herramientas de medida puede fluctuar inesperadamente debido a múltiples factores, pero puede ser corregido con las herramientas analíticas correspondientes.

   Además, en algunos casos, existen discontinuidades en las series temporales, generalmente debidas a interrupciones momentáneas del suministro eléctrico en las herramien-

tas de medición o al mantenimiento y trabajos realizados en las EDAR que interfieren en el muestreo continuo de algunas variables.

3. **Modelos de Aprendizaje Automático**: Haciendo uso de diferentes técnicas de Aprendizaje Automático se tratará de alcanzar el objetivo principal del proyecto, señalado en la Sección 1.2 de esta memoria, que se corresponde con la predicción de aumentos severos de los caudales de las EDAR ante eventos climáticos como temporadas de lluvias intensas para la actuación adecuada y gestión eficiente del agua.

Dada la variedad de modelos o algoritmos utilizados y la naturaleza de actualización constante de las fuentes de datos del proyecto, este proceso requiere de constantes revisiones por parte del equipo, toma de decisiones y nuevas iteraciones hasta obtener resultados concretos.

2.2 Planificación y seguimiento

Además de las tareas principales para el desarrollo de este proyecto, como en todo TFG existen fases del trabajo orientadas a la elaboración de una documentación detallada y estructurada del trabajo desenvuelto, que terminará conformando esta memoria. Podría, por lo tanto, decirse que el trabajo se ha distribuido en diferentes fases, algunas de las cuales han sido explicadas en la Sección 2.1 de la memoria:

1. **Formalización del proyecto y establecimiento de objetivos**: Incluye el periodo de recopilación de información inicial y adaptación al equipo de trabajo. También se incluye en este bloque la elaboración del anteproyecto de TFG. El comienzo del mismo se corresponde con el inicio del contrato de prácticas extracurriculares en que se ha desarrollado, en la última semana del mes de enero de 2024.

2. **Extracción, análisis, procesado e integración de datos**: Tras cada nuevo envío de datos por parte de las empresas participantes, se ha ejecutado este proceso en varias ocasiones, dada la llegada de varios nuevos conjuntos de datos durante la duración del proyecto. Por simplicidad, esta fase será nombrada como EAPI en esta sección de la memoria.

3. **Detección y tratamiento de datos anómalos o faltantes**: Tras la llegada de datos del proceso anterior, se ha realizado este proceso en varias ocasiones.

4. **Modelos de Aprendizaje Automático**: Finalmente, los datos ya procesados son utilizados para la fase de modelado, los cuales requieren inherentemente múltiples ejecu-

ciones. Esta fase requiere de documentación e investigación previas, implementación de los modelos, entrenamiento de los mismos y evaluación de los resultados.

5. **Memoria del proyecto**: Documentación de todo el proceso llevado a cabo. Se corresponde con el periodo final del TFG.

Cabría incluir la subfase de *Evaluación de los resultados* en cualquiera de las fases principales, ya que está inherentemente incluida en todas las demás fases de desarrollo y forma parte de la metodología de trabajo de actualización y mejora constante seguida durante el mismo.

Podemos comprender de mejor manera el flujo de trabajo y su planificación mediante un Diagrama de Gantt [16], que se muestra en la Figura 2.2. Aunque en este diagrama se ha tratado de representar el flujo de trabajo de forma simplificada, debe comprenderse que el proyecto ha requerido de múltiples ejecuciones de cada fase, marcadas por las correspondientes reuniones y ciclos típicos de un proceso Scrum, como se muestra en la Figura 2.1. Se incluye también en la Tabla 2.1 una estimación del tiempo en horas necesario para realizar este trabajo en base al Diagrama de Gantt y teniendo en cuenta la jornada laboral establecida en el contrato de prácticas (5 horas diarias), aunque secciones del trabajo como la elaboración de la memoria se han desarrollado fuera del horario laboral.

Fase de trabajo	**Estimación de horas**
Formalización y establecimiento	70
EAPI datos	140
Anomalías	100
Modelos	275
Memoria	95
Subtotal desarrollo	515
Total estimado	680

Tabla 2.1: Estimación de horas requeridas para realizar el trabajo.

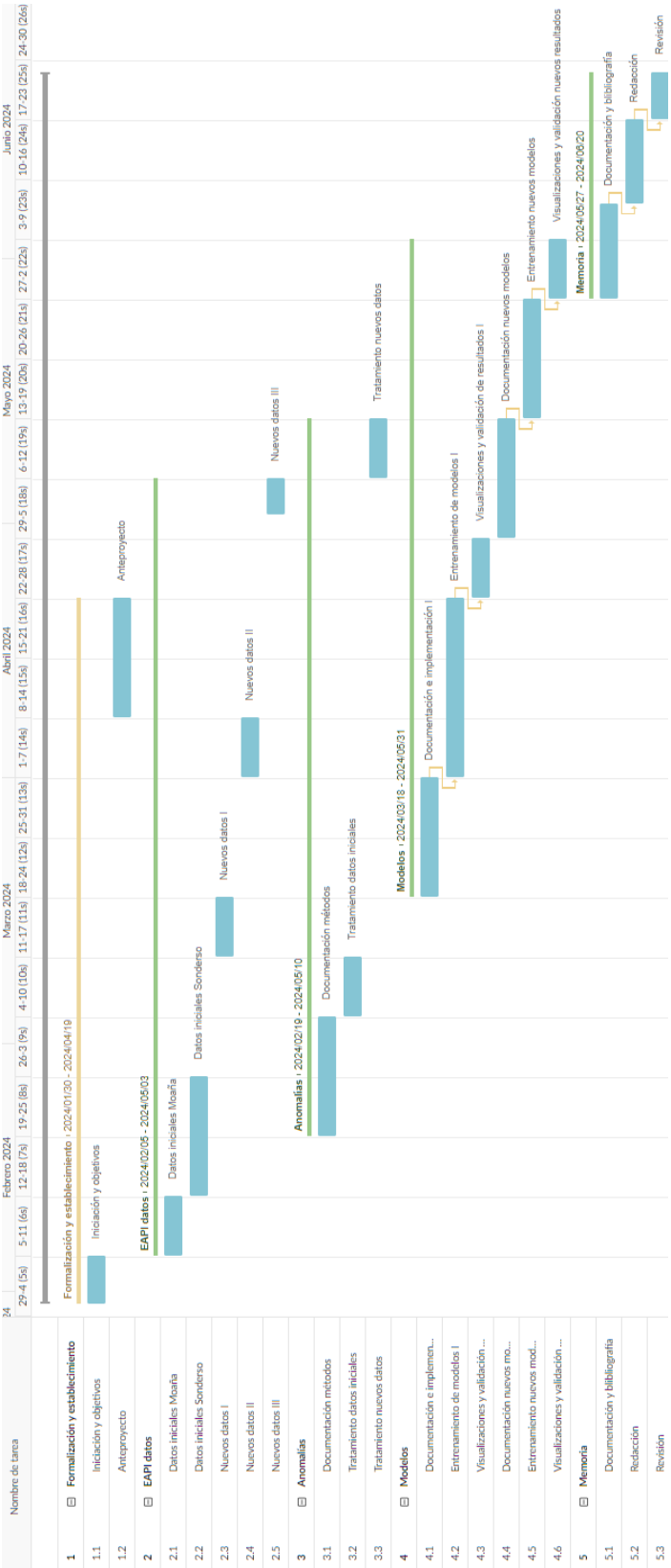


Figura 2.2: Diagrama de Gantt: planificación del proyecto.

2.3 Costes

Respecto al costo económico de la realización de un proyecto de estas características, se realiza una estimación aproximada teniendo en cuenta la estimación de horas prevista en la Tabla 2.1. Para el cálculo del coste humano del proyecto se han utilizado dos montos de referencia, por una parte el establecido en dicho contrato y por otra, el sueldo de un profesional junior del sector de la Ciencia de Datos en base a diversas fuentes [17, 18]. En base a esto, asumiendo la poca experiencia laboral del alumno, podríamos establecer como sueldo base unos 12 euros por hora de trabajo, mientras que la remuneración establecida como becario será de 6 euros por hora. Se resume el cálculo del coste humano total en la Tabla 2.2. De acuerdo a la realidad del proyecto, se tomará en cuenta este último para el cálculo del coste total del proyecto.

Fase de trabajo	**Horas**	**Coste/h**	**Total**
EAPI datos	140	6	840
Anomalías	100	6	600
Modelos	275	6	1650
Subtotal desarrollo	515	6	3090
Memoria	95	6	570
Formalización y establecimiento	70	6	420
Total estimado	680	6	4080

Tabla 2.2: Estimación del coste humano del proyecto.

Por otra parte, podría considerarse el material informático básico necesario para el desarrollo de un proyecto de estas características, como es un ordenador personal, monitor, teclado, etc. Aunque el precio total de compra de estos productos rondaría unos 2000 euros, si se ajusta este coste a la duración total del proyecto (menos de 6 meses) en proporción a la vida útil promedio de estos componentes de unos 4 años haciendo un uso intensivo de los mismos [19], podría deducirse un coste material real de 250 euros. Por lo tanto, la estimación de coste total del proyecto será de unos 4330 euros.

Capítulo 3

Herramientas utilizadas

En esta sección se comentarán las diferentes herramientas utilizadas, lenguaje de progamación y sus librerías.

3.1 Lenguaje de programación

Python [20] es un lenguaje de programación multiparadigma, interpretado y de alto nivel. Creado en el año 1991 por el científico y desarrollador Guido van Rossum [21], se ha convertido actualmente en uno de los lenguajes de programación más utilizados a nivel global, según índices como el de TIOBE [22]. Por su versatilidad, sintaxis legible y la gran cantidad de librerías disponibles, Python es una herramienta muy completa y adecuada para tareas de ciencia e ingeniería de datos. Durante este proyecto ha sido empleado en todas las fases desarrollo, desde el análisis inicial de los datos hasta la visualización final de los resultados.

3.1.1 Principales librerías

A continuación se hará una breve descripción de las principales librerías del lenguaje de programación Python empleadas en el proyecto:

- **Pandas** [23]: Es una librería esencial para la manipulación y análisis de datos en Python. Ofrece estructuras de datos como Series y DataFrames, que permiten realizar operaciones complejas de manera eficiente y sencilla. Es especialmente útil para trabajar con datos tabulares, limpieza de datos y análisis exploratorio.
- **Numpy** [24]: Proporciona soporte para grandes matrices y arrays multidimensionales, junto con una colección de funciones matemáticas de alto nivel para operar con estos arrays. Es fundamental para el cálculo científico y es la base de muchas otras librerías de ciencia de datos y aprendizaje automático en Python.

- **Dmi Open Data** [25]: Librería específica para acceder y trabajar con datos abiertos proporcionados por el Danish Meteorological Institute (DMI). Facilita la descarga y procesamiento de datos meteorológicos y climáticos, como la temperatura y humedad en el área circundante a las EDAR o las precipitaciones en la zona.
- **Math** [26]: Ofrece funciones matemáticas básicas como trigonometría, logaritmos, factoriales, entre otras, para realizar operaciones matemáticas generales.
- **Pytorch** [27]: Es un framework de aprendizaje profundo utilizado para construir y entrenar redes neuronales. Es conocido por su flexibilidad, dinamismo y facilidad de uso, lo que lo hace popular tanto en investigación como en producción. PyTorch proporciona dos características de alto nivel como son la computación de tensores (como NumPy) con una aceleración fuerte a través de unidades de procesamientos gráficos (GPU) y la construcción de redes neuronales profundas en un sistema de diferenciación automática de bases de datos.
- **Xgboost** [28]: Xgboost es una implementación de XGB para Python, optimizada para velocidad y rendimiento. Es ampliamente utilizada en competencias de machine learning y análisis de datos tabulares por su eficiencia y precisión.
- **Lightgbm** [29]: Es un framework de Light Gradient Boosting Machine (LGBM) eficiente y escalable. Está diseñado para ser más rápido y consumir menos memoria que otras implementaciones, lo que lo hace ideal para grandes volúmenes de datos.
- **Skforecast** [30]: Herramienta para la predicción de series temporales utilizando modelos de *Scikit-learn*. Facilita la creación, entrenamiento y evaluación de modelos de forecasting.
- **Sklearn** [31]: Scikit-learn es una biblioteca para aprendizaje automático que proporciona herramientas simples y eficientes para análisis de datos y minería de datos, incluyendo clasificación, regresión, *clustering* y reducción de dimensionalidad.
- **Matplotlib** [32]: Librería para crear visualizaciones estáticas, animadas e interactivas en Python. Es ampliamente utilizada para generar gráficos y visualizaciones 2D, como gráficos de líneas, barras, histogramas y dispersión.
- **Bokeh** [33]: Librería de visualización interactiva que permite crear gráficos interactivos y aplicaciones de visualización de datos para la web. Generalmente se utiliza para construir *dashboards* y herramientas visuales dinámicas.

- **Seaborn** [34]: Basada en Matplotlib, Seaborn proporciona una interfaz de alto nivel para graficación atractiva y complejos gráficos estadísticos de manera más sencilla. Facilita la creación de gráficos como mapas de calor, diagramas de violín y gráficos de pares.

3.2 Almacenamiento de datos

Minio [35] es una solución de almacenamiento de objetos de código abierto, compatible con Amazon S3, diseñada para manejar grandes volúmenes de datos no estructurados. Minio es ideal para crear infraestructuras de almacenamiento escalables y seguras, ya sea en la nube pública, privada o híbrida. Minio soporta la expansión horizontal, permitiendo agregar nodos de almacenamiento según sea necesario. También ofrece cifrado de extremo a extremo y control de acceso basado en políticas, garantizando que los datos estén protegidos.

Gracias a su implementación del servicio Amazon S3 de objetos en la nube será posible almacenar y consultar los conjuntos de datos desde cualquiera de las herramientas y dispositivos participantes en el proyecto, pudiendo ser cargados directamente desde esta herramienta a la hora de entrenar modelos.

3.3 Neural Network Intelligence (NNI)

NNI [36] es una plataforma de código abierto desarrollada por Microsoft para la automatización y optimización de experimentos de AA, permite la búsqueda automatizada de hiperparámetros, arquitectura de modelos y selección de características [37]. Proporciona además una interfaz gráfica donde visualizar dicha búsqueda de parámetros, detalles sobre cada configuración probada y obtener comparaciones entre ellas.

Para la realización de este proyecto, se utilizará esta herramienta tomando como entrada un espacio de búsqueda de parámetros, es decir, un conjunto de hiperparámetros sobre los que se desea obtener la configuración óptima y un intervalo de valores entre los cuales irá ajustando cada hiperparámetro. En base a ellos, la herramienta realizará múltiples entrenamientos sobre el modelo, variando en cada caso dichos parámetros y comparando las métricas resultantes de cada intento. De igual manera, seleccionará la arquitectura óptima en concepto de número de capas ocultas y unidades por capa. En la Figura 3.1 se muestra un ejemplo de entrenamiento con NNI, donde se aprecian los diferentes hiperparámetros que está ajustando. Las configuraciones que minimizan en mayor medida la métrica correspondiente se muestran en verde y las peores configuraciones, en rojo.

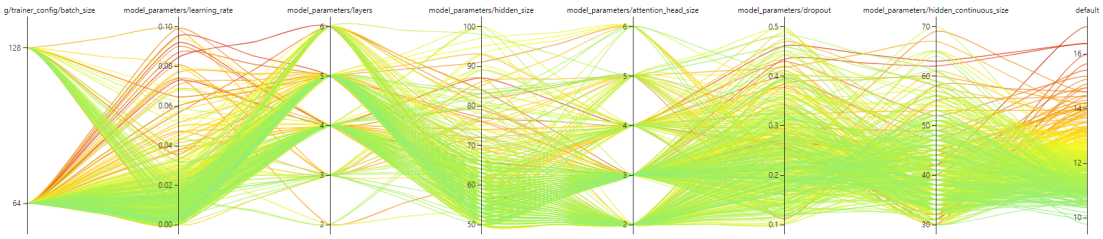


Figura 3.1: Ejemplo de ajuste de hiperparámetros con NNI.

3.4 AiSystemFramework

Se trata de un framework basado en PyTorch (véase la Subsección 3.1.1), desarrollado por ITG y de uso privado. Este implementa modelos de AA como Temporal Fusion Transformer (TFT), en el que profundizaremos más adelante en esta memoria. Utilizando este framework, que toma como entrada una serie de parámetros iniciales y el conjunto de datos, podemos completar entrenamientos del modelo TFT optimizando sus hiperparámetros y encontrar las mejores configuraciones de red neuronal para un problema concreto mediante la herramienta NNI (véase la Sección 3.3).

3.5 Máquina Virtual y recursos dedicados

Una máquina virtual es, esencialmente, una versión virtual de un ordenador físico o una instancia virtualizada de este, que puede realizar funciones como la ejecución de programas [38]. Toda máquina virtual debe estar alojada en una máquina física y no puede interactuar de forma directa con el sistema físico. El software encargado de la coordinación de recursos entre la máquina virtual y la máquina física se denomina hipervisor [39].

En el contexto de este proyecto, se hará uso de una máquina virtual alojada en un sistema dedicado a la computación, el cual dispone de gran capacidad de almacenaje y cómputo. Durante todo el proyecto se habrán entrenado múltiples modelos de AA, los cuales requieren continuas ejecuciones, que pueden tomar horas, almacenamiento específico y gran parte de las capacidades de cómputo de un portátil. Para evitar saturar el computador personal, algunos de los modelos serán ejecutados utilizando dicha máquina virtual proporcionada por ITG, aprovechando sus capacidades de computación.

Capítulo 4

Descripción del dominio

En este capítulo de la memoria se explicarán algunos conceptos teóricos que serán esenciales para la comprensión de este proyecto, su contexto y sus objetivos. Este fragmento de la memoria mostrará los fundamentos de las distintas técnicas de AA, conceptos relacionados con el ciclo del tratamiento de aguas residuales, métricas utilizadas para evaluar la calidad de las predicciones, entre otras. También, revisaremos el estado de la cuestión que trata este trabajo, visitando estudios similares o proyectos que hayan afrontado estas tareas. El desarrollo práctico llevado a cabo y sus resultados se explicarán en los siguientes capítulos de la memoria.

4.1 Estado del arte

En las Secciones 1.1 y 1.2 se ha explicado tanto el contexto en que se desarrolla este proyecto como sus ojetivos principales. También se ha señalado que Life Reseau (proyecto dentro del cual está integrado este TFG) pertenece al programa Life de la UE. Dentro de este programa han existido múltiples proyectos relacionados con el tratamiento de aguas residuales, como PRISTINE o BIODAPH2O, los cuales han afrontado el problema del tratamiento y gestión eficientes del ciclo del agua residual desde el punto de vista de la mejora de técnicas biotecnológicas aplicadas en el proceso. Life Reseau combina la mejora en dichas técnicas con la aplicación de técnicas basadas en la Ingeniería de Datos, como el AA. Es esta aplicación de técnicas de AA concreta la que recoge este TFG.

A lo largo de este proyecto, se tendrán que procesar constantemente series temporales que, según IBM [40], son un conjunto de observaciones que se obtiene midiendo una variable única de manera regular a lo largo de un período de tiempo. Estos conjuntos de mediciones se agrupan, dentro de un mismo espacio temporal, con las mediciones de otras variables relacionadas formando el conjunto de datos del que dispondremos. En otro artículo publicado por

esta misma fuente, IBM [41], podemos conocer algunas de las técnicas más frecuentes para la exploración de nuestro conjunto de datos. En concreto en este documento se destacan los análisis univariantes y multivariantes, en ambas de sus versiones gráficas y numéricas.

Para la tarea de detección y tratamiento de datos anómalos podemos hacer referencia a trabajos como el de Wei [42], en el cual se propone la utilización de la técnica DBSCAN como forma de agrupar los datos de entrada de forma que se obtengan grupos específicos de datos posiblemente anómalos en base a su distancia al resto de muestras. Además, Wei resalta la importancia de la correcta selección de los hiperparámetros de este algoritmo, como son la selección del número correcto de *clusters* o la distancia umbral que determina la pertenencia de un punto a un grupo determinado.

Además de la detección de los puntos anómalos, debemos procesar los anteriormente existentes datos faltantes así como los datos eliminados en la primera parte de la fase de detección de anomalías. En uno de los artículos referenciados, el de Abulkhair [43], se exponen algunas de las técnicas de imputación más utilizadas trabajando con series temporales. Abulkhair utiliza y compara diferentes algoritmos de imputación sobre una serie que contiene datos faltantes, como podría ser el caso de este proyecto, destacando las ventajas que puede aportar la técnica STL, que tiene como objetivo preservar la estructura general de la serie temporal, en particular sus componentes tendenciales y estacionales, al completar los valores faltantes, siendo más probable que los valores imputados reflejen la dinámica real de la serie temporal, lo que da lugar a modelos más precisos.

Por otra parte, existen gran variedad de técnicas de AA dedicadas a la generación de predicciones sobre series temporales. Como se explica en la Sección 4.4, uno de los modelos que utilizaremos será TFT, del que podemos encontrar referencias en trabajos relacionados con las predicciones de series temporales como el de Lim *et al.* [44] donde se propone el uso de TFT como una arquitectura DNN basada en la atención para pronósticos multihorizontes que logra un alto rendimiento al tiempo que permite nuevas formas de interpretabilidad y obtiene mejoras significativas en el rendimiento con respecto a los puntos de referencia de última generación.

Siguiendo con el hilo de las series temporales, podemos encontrar en la literatura gran cantidad de referencias al uso de modelos de la familia ARIMA para la modelización y predicción sobre serie temporales, como en uno de los artículos de Amat *et al.* [7]. En este extenso documento podemos encontrar información detallada sobre estos modelos y ejemplos prácticos de uso, como la predicción de una serie de tiempo que se corresponde con el consumo de combustible en España. Destacamos de este artículo la presentación del método SARIMAX, que se

utilizará como una ampliación del método ARIMA en aquellos casos en que exista estacionalidad en los datos y necesitemos incorporar variables exógenas para mejorar la precisión del modelo.

En el terreno de los modelos TFT y ARIMA, podemos encontrar referencias de literatura en la que se explora la idea de combinar ambos modelos, como es el caso de Linardatos *et al.* [45], utilizado en un estudio que busca predecir la concentración de dióxido de carbono en ciertos lugares. Los investigadores al cargo de este artículo destacan la efectividad de este método híbrido que combina las salidas de ambos modelos y aseguran haber obtenido resultados significativamente mejores de esta manera.

Por otra parte, en el contexto de modelos predictivos se encuentran múltiples estudios y artículos en los que se aplican modelos basados en el concepto de GB para obtener predicciones sobre series temporales. En esta ocasión, haremos referencia a la investigación llevada a cabo por Amat [46], donde se explica que los métodos basados en árboles se han convertido en un referente dentro del ámbito del AA por los buenos resultados que generan en problemas muy diversos, y se destacan algunas de las ventajas que aportan este tipo de modelos, como el ser menos susceptibles a la influencia de valores atípicos.

Por último, al igual que propone Dancker [5], exploraremos el funcionamiento del modelo NHITS para realizar predicciones sobre nuestros conjuntos de datos. Haciendo referencia al autor, este modelo se ha dado a conocer por la aplicación de la interpolación jerárquica, que permite combinar diferentes pronósticos en distintas escalas temporales y, tras el remuestreo y la interpolación, reducir la cantidad de parámetros que se pueden aprender, obteniendo modelos mucho más ligeros y con tiempos de entrenamiento cortos.

4.2 Infraestructuras

4.2.1 Funcionamiento de una red de saneamiento

Todo este proyecto se desarrolla en torno a estas estructuras, por lo que esta sección de la memoria tendrá como finalidad explicar los conceptos básicos acerca del funcionamiento de las mismas. Una Estación Depuradora de Aguas Residuales (EDAR) es una instalación que tiene como objetivo general el tratamiento de las aguas residuales mediante diferentes procedimientos biológicos, químicos y tecnológicos, a fin de emitir agua en condiciones de ser vertida al medio natural de nuevo, generalmente a ríos o mares. En general, estas estaciones tratan agua de origen local, que procede del consumo doméstico, la escorrentía superficial del

drenaje de las zonas urbanas, o cualquier otro tipo de agua influente que haya sido recogida en los diferentes puntos de su red de tuberías.

Tan solo en España, existían en el año 2012 más de 2100 EDAR, de las cuales, 120 se encontraban en Galicia [47]. En la Figura 4.1 [2] se muestran todas las estructuras de este tipo, dinstiguidas por colores según los procedimientos que realizan.

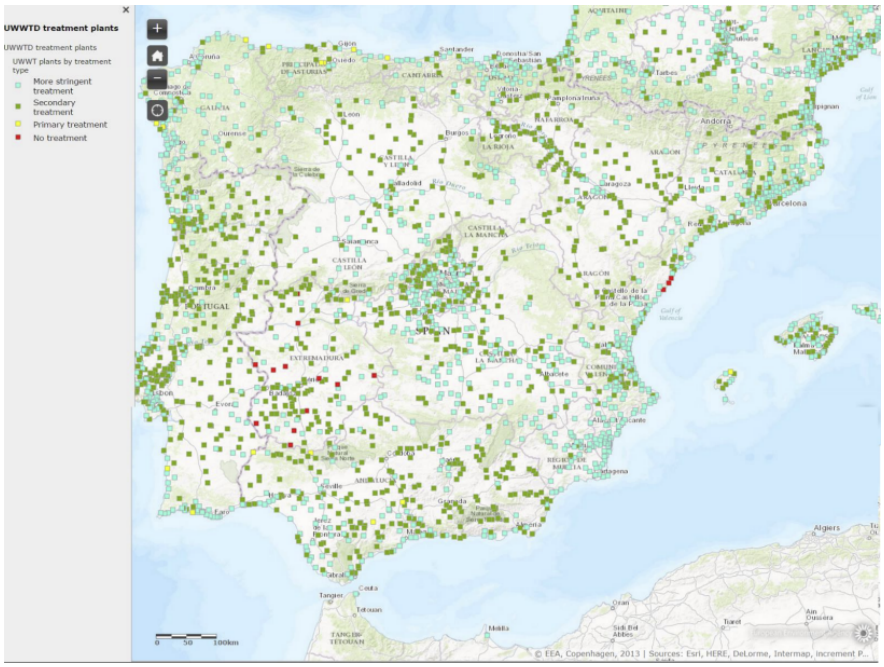


Figura 4.1: EDARs en la Península Ibérica en el año 2013 [2].

Por lo general, este tipo de estructuras son el punto final del trayecto de recogida de aguas de la red, la cual se extiende por las áreas urbanizadas circundantes. Además de la estación principal, las redes de recogida de aguas residuales cuentan con distintas Estaciones de Bombeo de Aguas Residuales (EBARs), encargadas de elevar las aguas residuales en los puntos en los que su transporte no se puede realizar por gravedad, permitiendo así su transporte a las EDAR. En una EBAR las aguas residuales se introducen y almacenan en un pozo, comúnmente conocido como pozo húmedo, que está equipado con instrumentación sensórica para detectar el nivel de aguas residuales presentes. Cuando el nivel de las aguas residuales se eleva a un punto determinado, se pondrá en marcha una bomba para elevarlas a través de un sistema de tuberías presurizadas y transportarlas a una EDAR.

La red de saneamiento en su conjunto puede entenderse como un grafo como el de la Figura 4.2, donde el nodo raíz será la EDAR, el resto de nodos las diferentes EBAR, y las aristas que unen los nodos serán las tuberías y canales comunicantes por los que fluye el agua.

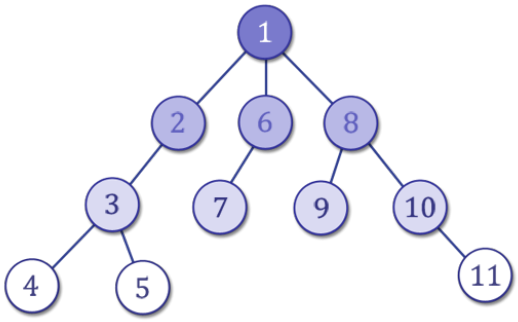


Figura 4.2: Grafo que representa una red cualquiera de transporte de aguas residuales

En las redes de recogida de aguas, además de los caudales y otras variables que están cuantificadas regularmente, existen muchos factores externos y ambientales que influyen en el funcionamiento de las mismas y que habrá que tener en cuenta a la hora de procesar los datos. Los más importantes son:

- **Infiltraciones y exfiltraciones**: las tuberías pueden verse afectadas por el paso del tiempo y las condiciones meteorológicas, haciendo que se quiebren o se introduzcan en ellas ramas de árboles cercanos. Se conoce como infiltración a la entrada de agua desde el medio al sistema, bien sea a través de grietas, condensación de humedad, porosidad de las tuberías o juntas mal selladas. Por el contrario, pueden producirse exfiltraciones por las mismas causas, pero en este caso el agua del sistema estaría vertiéndose al medio.
- **Recursos hídricos cercanos**: algunas EDAR tienen contacto directo con lagos, ríos o incluso el mar, como es el caso de Moaña. En muchas ocasiones, el desbordamiento de los ríos o las mareas altas pueden introducir agua al sistema y causar aumentos del caudal.
- **Industrias cercanas**: cuando una gran industria realiza vertidos al sistema de aguas residuales, puede generar datos anómalos o moldear la forma de la serie temporal del caudal en función de su actividad. Esto podría ser un problema para el proceso de análisis de datos cuando esto se haga de forma irregular, en cantidades masivas, que realmente estarán generando datos inesperados.

En este caso en concreto, trabajaremos con dos EDAR diferentes, que se detallan en las siguientes secciones, cada una con sus estaciones de bombeo y características particulares.

4.2.2 EDAR de Sonderso

En la Figura 4.3 se muestra la red básica de canalización de aguas residuales de la ciudad danesa de Sonderso. Esta estación fue diseñada con una capacidad de 20.000 h.e. Recibe una

gran cantidad de carga orgánica, incrementando la necesidad de aireación y eliminación de nutrientes. Anualmente se estiman cuatro eventos de descarga del sistema de 6.000 m^3 de agua sin tratar. En la Figura 4.3 se señalan en rojo las distintas EBAR que componen los puntos de bombeo de la red, en amarillo, la ubicación de la EDAR de Sonderso, y en verde, el sistema de alcantarillado que las comunica. Esta red podría dividirse en dos secciones principales, como se explicará en la Sección 5.2. En esta estación, tendremos información sobre el caudal de agua que recibe la depuradora, así como de los caudales que atraviesan las diferentes estaciones de bombeo.

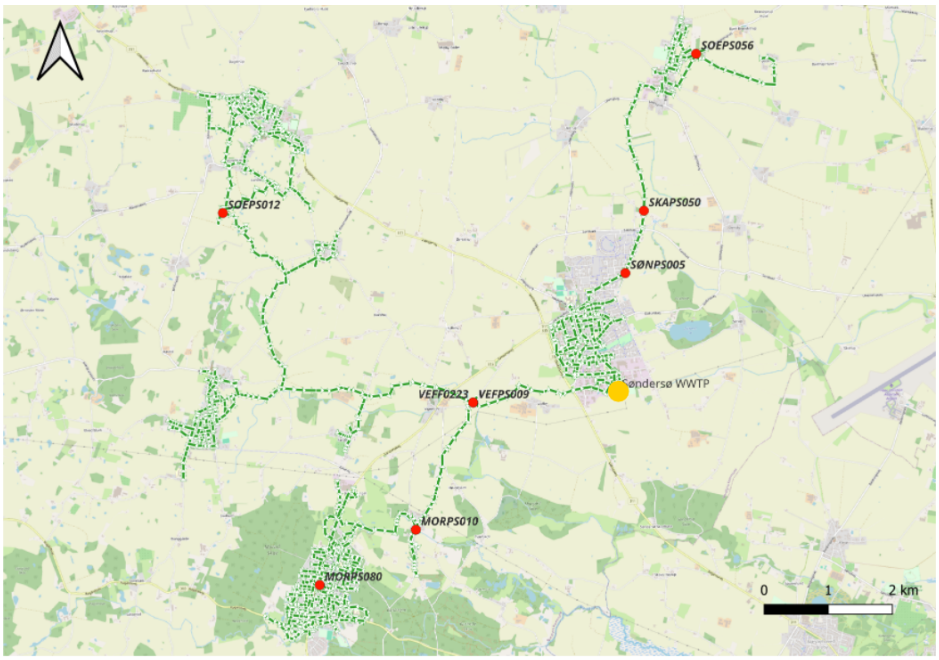


Figura 4.3: Red de la estación de Sonderso.

4.2.3 EDAR de Moaña

En la Figura 4.4 se muestra la estructura de la red de saneamiento de Moaña, una localidad española situada al norte de Vigo y de la ría homónima. Esta tiene la misma capacidad que la de Sonderso (véase la Sección 4.2.2), unos 20.000 , pero se estima que realiza una descarga de 15.000 m^3 de agua sin tratar anualmente. En esta ocasión, se señala en la Figura 4.4 en amarillo la depuradora principal, la EDAR de Moaña, y en rojo las estaciones de bombeo circundantes.

En el caso particular de Moaña, a diferencia de Sonderso, se debe tener en cuenta la posible influencia del mar en los movimientos de agua en la red. Además, al trabajar con esta estación (por indicaciones de la empresa que la gestiona) solo habrá que hacer uso de las referencias

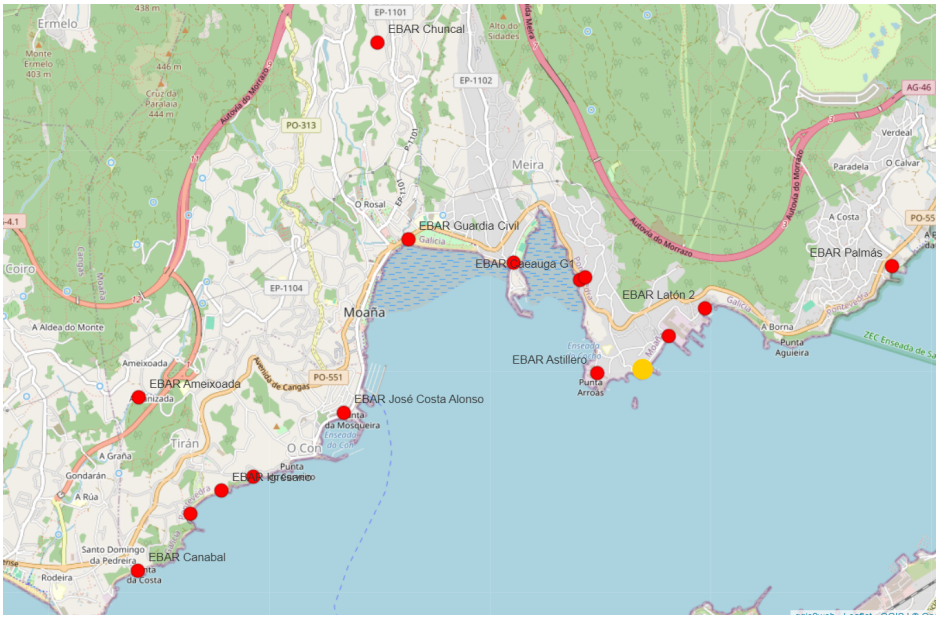


Figura 4.4: Red de la estación de Moaña.

aportadas por la EBAR de San Bartolomeu, que se muestra en detalle en la Figura 4.5. En este caso, para las estaciones de bombeo conoceremos datos como el nivel del agua de su tanque húmedo, en lugar del caudal de agua que han movilizado. Estas bombas solo se activan cuando dicho nivel supere una marca preestablecida, conocida como *consigna*.

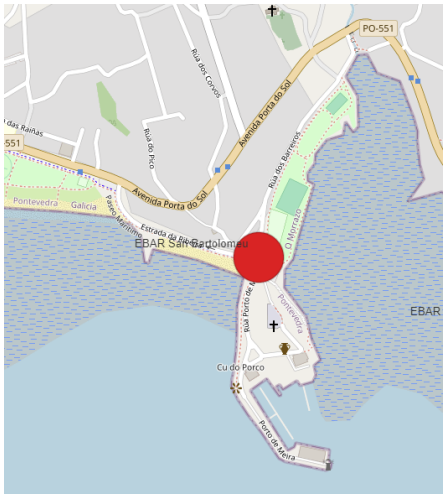


Figura 4.5: Localización de la EBAR de San Bartolomeu, Moaña.

4.3 Series temporales y sus componentes

Se conoce como Series Temporales a los conjuntos de observaciones que se obtiene midiendo una variable única de manera regular a lo largo de un período de tiempo [48, 49]. Los intervalos de tiempo en que se miden son constantes y ordenados cronológicamente. En cualquier situación cotidiana se pueden generar estas series de mediciones, observando la evolución del precio de algún producto, el crecimiento mensual de un humano, la variación diaria de la temperatura media, la evolución del número de espectadores de un canal de televisión minuto a minuto, etc. A lo largo de este proyecto se utilizarán series de tiempo compuestas por mediciones de los diferentes parámetros disponibles en una EDAR, como son el caudal de la misma, las precipitaciones que recibe u otras variables.

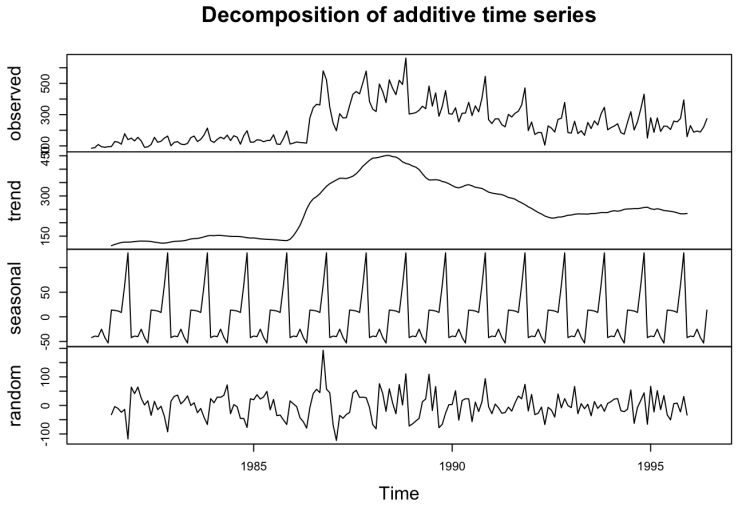


Figura 4.6: Muestra de las componentes básicas de una serie temporal [3].

Estas estructuras de datos vienen definidas por un conjunto de características, conocidas como componentes, en las cuales la serie de tiempo puede ser descompuesta para estudiar el comportamiento a largo plazo de la misma. Las componentes básicas de una serie de tiempo se explican brevemente a continuación [50, 51], y que se muestran en la Figura 4.6:

- **Tendencia (*trend*)**: Movimiento general de la variable a lo largo del tiempo, similar gráficamente a la media móvil de la variable.
- **Componente cíclica o estacional (*seasonal*)**: Componente periódico de la serie que determina la frecuencia con la cual un patrón o patrones se repiten.

- **Componente irregular (*random*)**: También llamado ruido, recoge las alteraciones de la serie sin una pauta periódica ni tendencia reconocibles.

4.3.1 Proceso estacionario

Además de las componentes citadas anteriormente, algunas series temporales presentan una característica llamada *estacionariedad*. Se dice que una serie es *estacionaria* cuando sus propiedades estadísticas (como la media, la varianza y la autocorrelación) no cambian con el tiempo [50]. Existen herramientas estadísticas cuya función es determinar si una serie temporal se corresponde con un proceso estacionario, como pueden ser:

- **Test de Dickey-Fuller Aumentado ADF**: Prueba la hipótesis nula de que existe una raíz unitaria para la serie temporal, es decir, la ecuación característica del proceso tiene raíz con valor 1. En caso de ser necesario, utilizaremos su implementación en Python mediante la función *adfuller* de la librería *statsmodels*. Esta prueba devolverá un p-valor que, en caso de ser menor que el nivel de significancia elegido, permitirá concluir que la serie temporal es estacionaria.
- **Test Kwiatkowski Phillips Schmidt and Shin (KPSS)**: A la inversa que la prueba anterior, KPSS considera la hipótesis de la existencia de una raíz unitaria como su hipótesis alternativa. En este caso, si su p-valor fuese significativo debería asumirse que la serie presenta una tendencia, es decir, no es estacionaria. Su implementación en Python está presente en la función *kpss* de *statsmodels*.

4.4 Conceptos del Aprendizaje Automático

El Machine Learning, Aprendizaje Automático (AA) o Aprendizaje Máquina es una rama de la Inteligencia Artificial (IA) que permite a las máquinas aprender de datos y tomar decisiones o hacer predicciones sin estar explícitamente programadas para realizar tareas específicas [52]. Podría decirse que se refiere a la capacidad de los sistemas informáticos para mejorar su rendimiento en una tarea a través de la experiencia en la exploración y análisis de los datos de entrada. Se basa en el uso de algoritmos y modelos estadísticos para analizar y extraer patrones de grandes volúmenes de datos. En la literatura pueden encontrarse miles de ejemplos de uso del AA, como son el marketing, las predicciones de ventas, la seguridad, la prevención de fraudes, la movilidad autónoma de distintos vehículos, etc. Los principales tipos de AA son los siguientes:

- **Aprendizaje Supervisado**: A partir de un conjunto de datos etiquetados (se conoce el valor de la variable de estudio para los datos), el algoritmo tratará de diseñar una función capaz de modelar el comportamiento de los datos, siendo capaz de etiquetar,

es decir, asignar un valor estimado para la variable objetivo a un nuevo conjunto de datos desconocidos en función de lo que haya aprendido del conjunto de entrenamiento. Generalmente, se utiliza en tareas de **clasificación** (asignar etiquetas categóricas, que simbolizan la pertenencia a determinados grupos de datos) y **regresión** (asignar valores continuos que estiman el valor de la variable objetivo para nuevos datos).

- **Aprendizaje NO Supervisado**: En el lado opuesto, este tipo de algoritmos no requieren etiquetas en los datos de entrada. Su tarea será la clasificación, agrupando estos datos en función de la similitud de sus características. Este tipo de mecanismos suelen utilizarse en el análisis exploratorio de datos, la clasificación de datos y la reducción de dimensionalidad de un conjunto de datos. Otros algoritmos que emplean aprendizaje no supervisado son el K-Medias y los métodos de agrupación probabilística.

- **Aprendizaje Semisupervisado**: A medio camino entre las dos clases anteriores, combina una pequeña colección de datos etiquetados con un gran conjunto de datos sin etiquetar. Suele utilizarse cuando el etiquetado de datos es costoso o no puede llevarse a cabo.

A diferencia de ellos, el **Aprendizaje por Refuerzo** busca modelar el algoritmo de forma que tome decisiones mediante la interacción directa con el entorno, sin un conjunto determinado de datos de entrada, y utilizando recompensas y penalizaciones como indicador de aprendizaje.

A lo largo de este proyecto, se exploran varios de los tipos de algoritmos que engloba el concepto del AA, como son:

- **Redes Neuronales**: Simulan el funcionamiento de un cerebro humano, con complejas estructuras formadas por capas de neuronas interconectadas, que realizan diferentes cálculos. Suelen utilizarse en el reconocimiento de audio y video, procesamiento de lenguaje natural y predicciones de series temporales.

- **Regresión**: Se utiliza este algoritmo para predecir valores numéricos, basándose en relaciones lineales entre valores, en el caso de la regresión lineal, o predecir respuestas categóricas, en la regresión logística.

- ***Clustering* o Agrupación**: Utiliza Aprendizaje no Supervisado para agrupar los puntos del conjunto de datos. Pueden encontrar relaciones complejas entre valores, incluso en alta dimensionalidad.

- **Árboles de decisión**: Tienen múltiples usos, desde la regresión a la clasificación. Utilizan una estrategia de ramificación de decisiones enlazadas, que pueden representarse

como un árbol, y que marcan el camino que el algoritmo tomará en cada paso, decidiendo entre ramas a las que moverse.

- **Bosques Aleatorios**: Se conoce así a los métodos de aprendizaje que utilizan una combinación de árboles de decisión para tomar decisiones más complejas.

A la hora de seleccionar los modelos de AA que se utilizarán habrá que tener en cuenta su capacidad para utilizar variables exógenas de referencia. En el caso que aborda este proyecto se utilizará la lluvia como variable exógena básica siempre que sea posible.

4.4.1 Optimización de hiperparámetros en el AA

A la hora de implementar y entrenar los distintos algoritmos y modelos utilizados, el ajuste de sus hiperparámetros de forma adecuada es esencial para su correcto funcionamiento y la obtención de resultados óptimos. Se conoce como hiperparámetros a las variables que definen la arquitectura o las funciones matemáticas que ejecuta el modelo, y que determinan su tamaño, la estrategia que seguirá a la hora de entrenarse o los pesos que otorga a cada variable [53, 54]. Como se ha mencionado, existen múltiples herramientas para la automatización de esta tarea, como NNI o herramientas incorporadas en librerías de Python. Algunos de los principales hiperparámetros que se mencionan a lo largo de esta memoria son:

- **Tasa de aprendizaje (*learning rate)***: Define la tasa de actualización de los pesos de las redes neuronales en las distintas etapas de su entrenamiento. Es, probablemente, el parámetro más importante y determinante. Tomar un valor muy alto hará que los cambios en los pesos sean muy grandes de una iteración a otra, pudiendo saltarse el punto óptimo, pero si es demasiado pequeño el modelo no se entrenará lo suficiente.
- **Tamaño de lote (*Batch size)***: Determina el tamaño del subconjunto de datos que el modelo recibe en cada iteración del entrenamiento.
- **Número de capas y unidades por capa**: En una red neuronal, determina el número de capas que conforman su estructura y el número de neuronas que componen cada capa. La combinación óptima de profundidad (más capas) y anchura (número de neuronas) dependerá de cada modelo y de sus datos.
- ***Dropout***: Define la probabilidad de que, tras cada iteración del entrenamiento, se desactiven neuronas aleatoriamente. Esto contribuye a reducir el sobreajuste de modelos y mejora su capacidad de generalizar.

4.4.2 El problema del *garbage in, garbage out*

El concepto del GIGO es un principio fundamental en la informática y el análisis de datos, que destaca la importancia de la calidad de los datos de entrada en los resultados generados por un sistema o algoritmo. Básicamente, representa la idea de que si se introducen datos de mala calidad, incorrectos o irrelevantes, el sistema o modelo será igualmente malo e incorrecto. En otras palabras, la calidad de los resultados de cualquier proceso de Ingeniería de Datos es directamente proporcional a la calidad de sus datos de entrada.

Este concepto es especialmente importante en el AA, y más en el contexto de un proyecto como este en el que se busca modelar el comportamiento de la red de saneamiento para obtener predicciones fiables. Los algoritmos de AA aprenden patrones y toman decisiones basadas en los datos proporcionados, si estos datos son incorrectos, incompletos o sesgados, el modelo aprenderá patrones incorrectos y tomará decisiones equivocadas, lo que puede llevar a resultados desfavorables o incluso peligrosos en aplicaciones críticas. En el contexto del proyecto, podría suponer el mal funcionamiento del proceso de tratamiento de aguas o la no anticipación ante eventos de desbordamientos. Este concepto estará presente a lo largo del proyecto, y se detallará en algunos apartados.

4.4.3 Modelo TFT

TFT es un ejemplo de modelo basado en redes neuronales. Como su nombre indica, está basado en la implementación de *transformers*, los cuales fueron introducidos por Vaswani *et al.* en 2017 en el artículo *Attention Is All You Need* [55]. Esta arquitectura se utiliza frecuentemente para la predicción de series temporales y proyectos de procesamiento de lenguaje, para lo cual toma como entrada valores pasados de la variable en una ventana de tiempo determinada, covariables estáticas que puedan proporcionar información contextual y variables exógenas. Una de las particularidades de este modelo es que puede componer un codificador a modo de modelo predictor y un decodificador que utilice el primero para obtener predicciones a varios instantes de distancia en el futuro como predicciones horarias a 12 horas vista, lo que se conoce como *multi-horizonte*. Para ello, TFT soporta la utilización de variables exógenas que solo son conocidas durante la etapa de codificación y otras que son conocidas en todos los instantes futuros.

TFT genera como salida un conjunto de cuantiles. Cada cuantil q para la predicción τ-pasos-adelante en un momento t toma la forma que se muestra en la Figura 4.7, siendo los *known inputs* aquellas variables conocidas a futuro por el decodificador y *unknown inputs*, las que solo se desconoce su valor a la hora de decodificar. Además, utiliza su propia función de pérdida basada en cuantiles, que será explicada en la sección 4.5.4

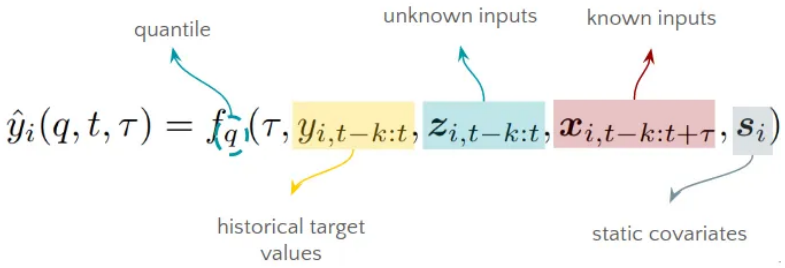


Figura 4.7: Cuantiles de predicción del modelo TFT [4].

En cuanto a la arquitectura de este tipo de modelos, que se muestra en la Figura 4.8 [4], el modelo hace uso de diferentes bloques con distintas arquitecturas internas y funciones específicas:

- **Selección de variables**: Se implementan bloques de este tipo específicos para cada tipo de entrada (covariables estáticas, entradas conocidas y entradas desconocidas a futuro), encargados de sopesar la importancia de cada variable de entrada, de forma que las siguientes capas o bloques puedan tomar estas entradas ponderadas según su importancia.

- **Bloques LSTM**: Este tipo de red neuronal recurrente modeliza las dependencias temporales a corto y largo plazo de la serie, permitiendo una mejor contextualización gracias a la recurrencia de este tipo de bloques.

- **Bloques GRN**: Este tipo de redes conocidas como residuales implementan conexiones *skip* entre capas, que permite la transmisión de información de una capa a otra a la cual no es directamente adyacente. Esta característica permite al modelo generalizar mejor en diferentes escenarios, además de reducir el número de parámetros entrenados innecesariamente.

- **Mecanismo de Atención**: Este tipo de algoritmos asignan pesos a la importancia de cada valor V en base a las relaciones entre las claves y las consultas, K, Q, es decir:

$$Attention(Q, K, V) = \alpha(Q, K)V$$

  Por otra parte, la autoatención o *self-attention* utiliza consultas, claves y valores de la misma entrada, aprendiendo la relevancia de cada paso de tiempo respecto al resto de la secuencia, capturando así las dependencias temporales. En TFT, la arquitectura de autoatención multipunto o *multihead attention* utiliza de forma paralela varios hilos de autoatención. Los resultados de estos hilos se concatenan en un solo tensor que luego será multiplicado por la matriz de pesos de los parámetros previamente seleccionados.

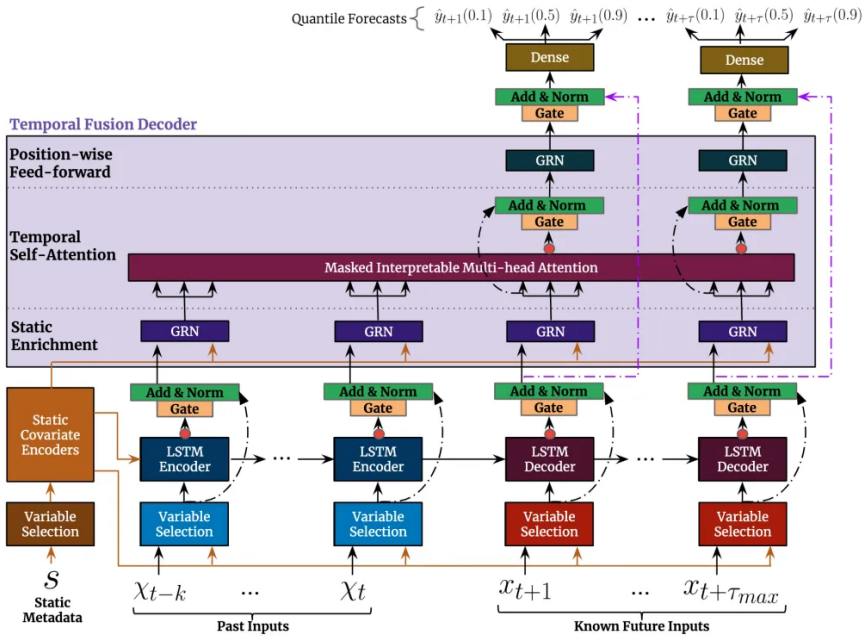


Figura 4.8: Arquitectura básica del modelo TFT [4].

4.4.4 Modelo NHITS

El modelo NHITS, o en español Interpolación Jerárquica para Series Temporales, es un modelo avanzado de predicción de series temporales basado también en redes neuronales, diseñado para mejorar la precisión y eficiencia en comparación con arquitecturas previas como NBEATS [5, 6]. NHITS utiliza una arquitectura jerárquica de interpolación que permite al modelo aprender representaciones de diferentes resoluciones temporales, capturando patrones a múltiples escalas temporales. Esta técnica reduce la complejidad computacional y mejora la capacidad del modelo para predecir a largo plazo y manejar múltiples niveles de estacionalidad y tendencia. Este modelo fue introducido por Challu *et al.*[56] como una ampliación del modelo NBEATS que incorpora un muestreo de datos en diferentes frecuencias de muestreo (*Multi-rate Sampling*) y una interpolación jerárquica de la salida (*Hierachical Interpolation*), como se explicará más adelante.

A grandes rasgos, la arquitectura del modelo consiste en bloques formados por perceptrones multicapa, que a su vez componen *stacks* residuales, que también se agrupan formando el modelo final, como puede verse en la Figura 4.9.

Cada nivel de la jerarquía realiza predicciones en su propia resolución temporal, lo que se conoce como *multi-rate sampling*, permitiendo capturar patrones en las diferentes escalas temporales y filtrar mejor el ruido de la serie temporal. Cada *stack* de bloques se especializa

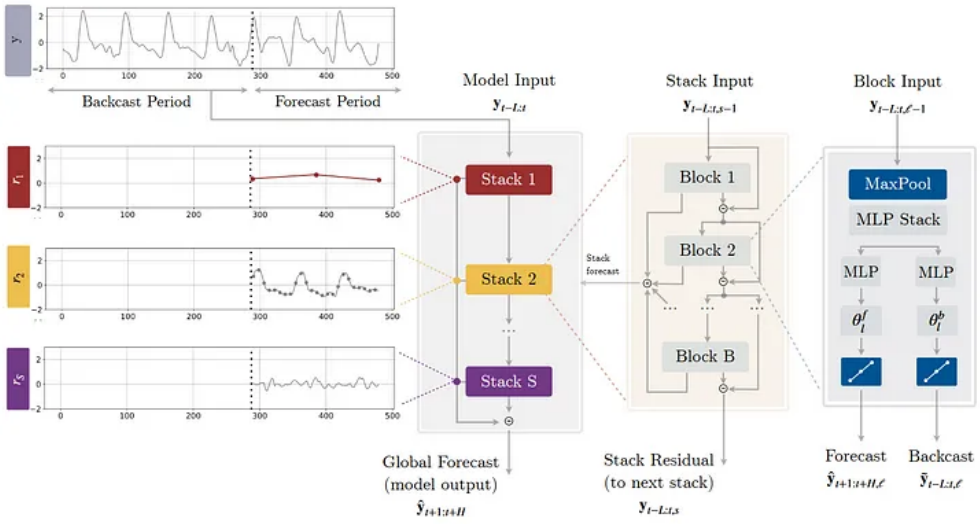


Figura 4.9: Arquitectura del modelo NHITS [5].

en una resolución temporal distinta. Para obtener este remuestreo, la arquitectura implementa en sus bloques capas de MaxPooling con diferentes tamaños de *kernel* en cada nivel que determinan la resolución temporal de salida de esta operación. Esta operación de remuestreo puede ejemplificarse con una simple imagen, en la Figura 4.10, donde se aprecia cómo los diferentes factores de muestreo afectan a la señal obtenida.

A continuación, el mecanismo de Interpolación Jerárquica se encarga de la combinación de las predicciones. A diferencia de NBEATS, donde cada *stack* tiene la misma cardinalidad y resolución temporal, NHITS introduce el concepto de ratio de expresividad, que simplemente indica el número de predicciones por unidad de tiempo que realiza cada *stack*. Teniendo este factor en cuenta, se combinan sus salidas para obtener predicciones finales, permitiendo obtener modelos mucho más ligeros en cuanto a parámetros entrenables y, por lo tanto, más rápidos.

4.4.5 Modelos ARIMA

La familia de modelos ARIMA es, a diferencia de los modelos de AA previamente explicados, un modelo estadístico basado en técnicas clásicas de este campo de las matemáticas. Como su nombre señala, los modelos basados en ARIMA utilizan componentes auto regresivos(AR) y medias móviles (MA) para la modelización de una serie temporal. Su alta interpretabilidad y simplicidad de implementación lo hacen un candidato esencial a la hora de seleccionar herramientas para la modelización de series temporales. Por otra parte, ARIMA requiere que la serie temporal sea estacionaria, concepto explicado en la Subsección 4.3.1, por lo que se

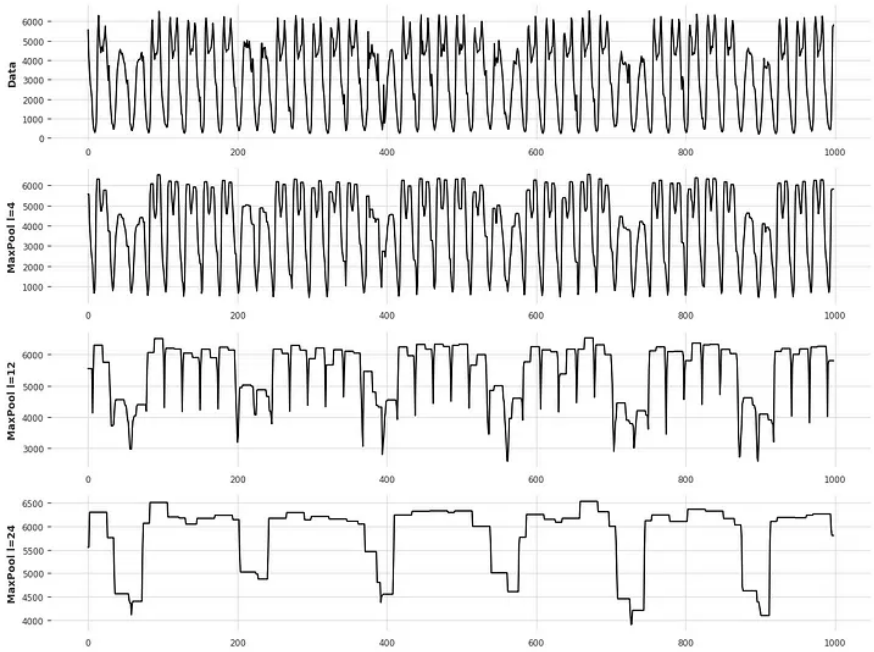


Figura 4.10: Ejemplo de operación de MaxPooling para remuestreo de series temporales [6].

emplearán las pruebas estadísticas vistas. Un modelo de este tipo en un instante t y para una serie de tiempo Y tiene la forma [57]:

$$Y_t = \phi_1 Y_{t-1} + \cdots + \phi_t Y_{t-p} + a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q},$$

donde p representa el orden de autoregresión (número de instantes anteriores que se toman en cuenta) y q, el orden de la componente de media móvil; a_t es el residuo en el momento t. Una vez obtenidos p, q, resultaría en un modelo $ARIMA(p, d, q)$, donde d es el número de diferenciaciones aplicadas para convertir la serie temporal en estacionaria. En la Figura 4.11 se muestra un ejemplo de una serie antes y después de una diferenciación, convirtiéndola en estacionaria ($d = 1$) [7].

Como se ha indicado previamente, uno de los requisitos del proyecto será emplear la variable exógena *lluvia* a la hora de entrenar modelos. Además, se busca modelar también la componente estacional de la serie temporal explicada en la Sección 4.3. Para abarcar estas características, se hará uso de SARIMAX, el cual supone una expansión de los modelos ARIMA capaz de incorporar la modelización de la estacionalidad y las variables exógenas. Para ello, en lugar de un modelo $ARIMA(p, d, q)$, donde d es el número de diferenciaciones aplicadas

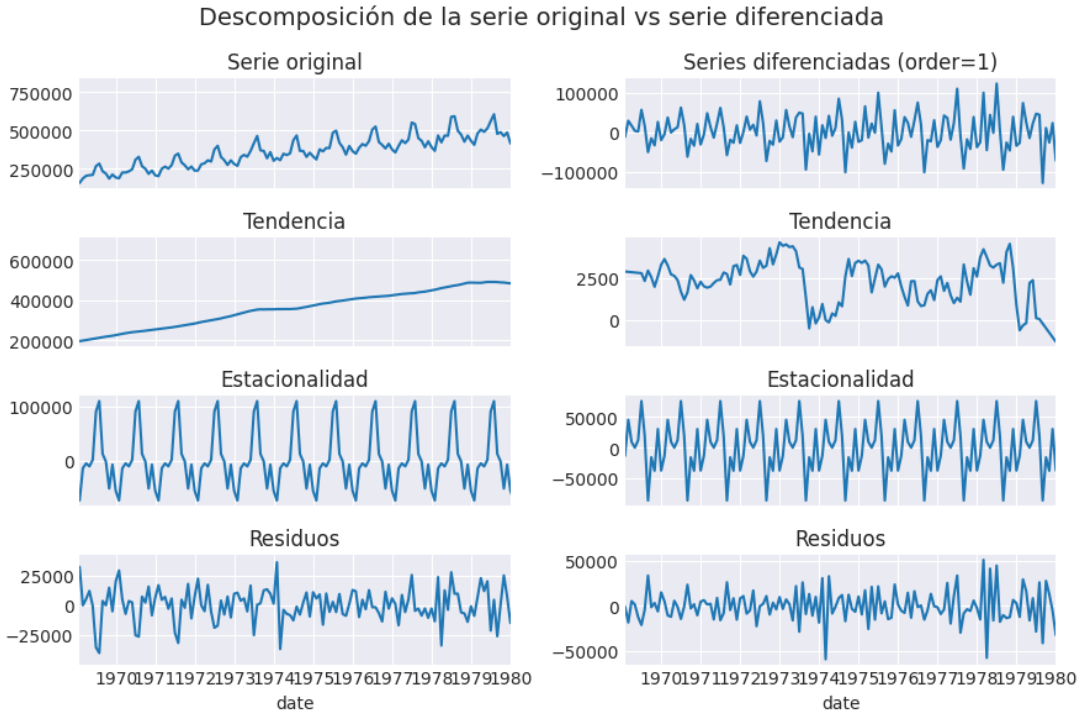


Figura 4.11: Descomposición en sus componentes de una serie temporal antes y después de ser diferenciada [7].

para convertir la serie temporal en estacionaria, será necesario el ajuste de los parámetros:

$$SARIMAX(p,d,q)\text{x}(P,D,Q)_s$$

En este caso, (P, D, Q) hacen referencia a los parámetros de auto regresión, media móvil y diferenciación de la componente estacional de la serie, respectivamente, siendo s el periodo de esta componente.

A la hora de ajustar los parámetros de este tipo de modelos, la librería de Python *pmarima* incluye la función *auto_arima*, la cual utiliza el AIC (Criterio de Información de AKAIKE) para obtener la mejor configuración de parámetros en cada caso de forma automatizada.

4.4.6 Modelos basados en árboles

Los arboles de decisión son un tipo de algoritmo muy utilizado en AA tanto en tareas de clasificación como de regresión. Como su nombre indica y como puede apreciarse en la Figura 4.12 [8], su estructura se asemeja a un árbol invertido, donde los nodos no terminales representan las tomas de decisiones que el algoritmo plantea y las hojas, los resultados a los que

conlleva cada decisión. En concreto, se plantea la utilización de modelos basados en árboles de regresión en este proyecto, los cuales producen salidas numéricas continuas. Durante su fase de entrenamiento se producen sucesivas divisiones del espacio de los predictores para generar regiones no solapantes [58], para luego generar predicciones de la variable respuesta dentro de cada región.

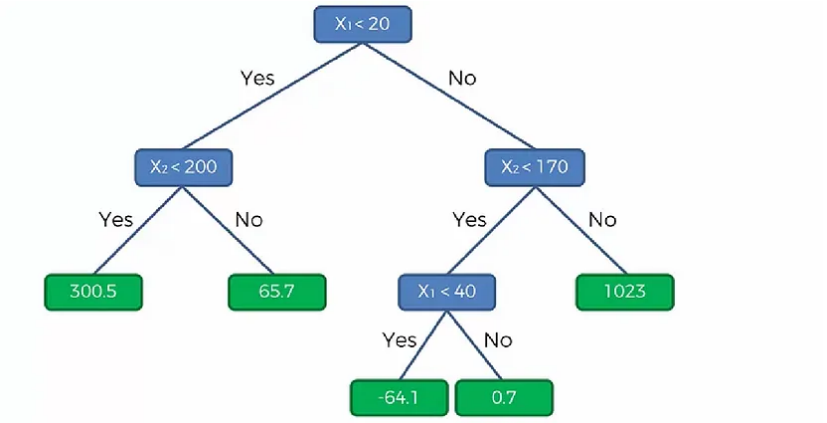


Figura 4.12: Diagrama de un árbol de decisión para regresión [8].

Estos modelos son sencillos y muy fácilmente interpretables, pero pueden ser propensos a sesgos y sobreajustes. Por lo general, los árboles pequeños tienen menor varianza y no consiguen representar relaciones complejas entre variables, mientras que los árboles demasiado ramificados tienden a estar sobreajustados a los datos de entrenamiento. Para afrontar esta problemática, fueron creados los métodos conocidos como *Ensemble*, que combinan varios modelos o árboles para reducir el sesgo y el sobreajuste. Las dos estrategias más utilizadas a la hora de combinar modelos son:

- ***Bagging***: Conocido también como *Bootstrap Aggregating*, consiste en la creación de múltiples modelos con diferentes subconjuntos de los datos de entrenamiento y combinar sus salidas mediante el promedio en caso de salidas continuas o el voto mayoritario, para salidas categóricas. El modelo *Random Forest* o Bosque Aleatorio se encuentra dentro de esta categoría.

- ***Boosting***: En este caso, la estrategia consistirá en el entrenamiento sucesivo de modelos sencillos, llamados *weak learners*, de forma que cada modelo pueda adquirir conocimiento y corregir errores producidos por los modelos previos. Se encuentran en este grupo los modelos de Refuerzo de Gradiente (GB), sobre los cuales se trabajará en este proyecto. En la Figura 4.13 se ejemplifica el proceso de entrenamiento de los sucesivos modelos y cómo el error desciende a medida que el número de *weak learners* aumenta.

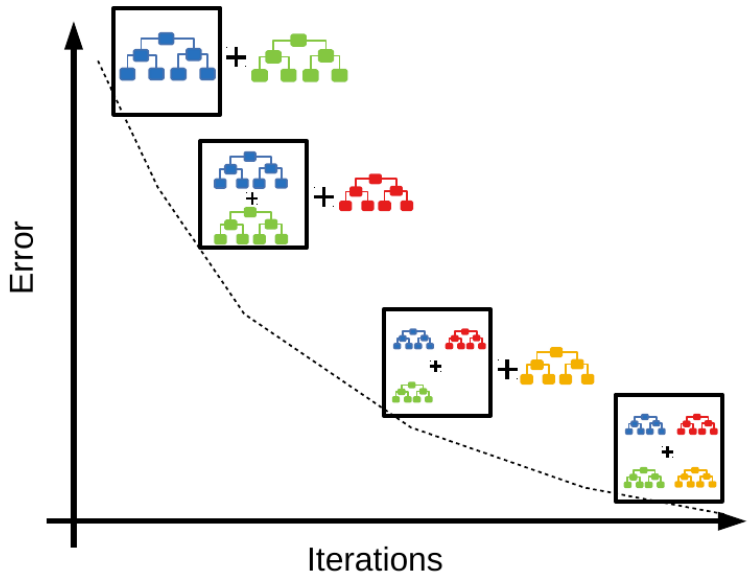


Figura 4.13: Simplificación del proceso de refuerzo (*Boosting*) en base al entrenamiento de sucesivos *weak learners* [9].

En esta ocasión, se emplearán dos modelos basados en el concepto de *Boosting*, concretamente implementaciones del algoritmo *Gradient Boosting*. Este algoritmo supone la posibilidad de utilizar cualquier función de coste, siempre que esta sea diferenciable, en base a lo cual calcula dicho gradiente. Los modelos de este tipo que se utilizarán son los siguientes:

- **Refuerzo Extremo de Gradiente (XGB)**: Busca la mayor velocidad computacional y rendimiento del modelo. Puede ser utilizado tanto para regresión como para clasificación, es altamente flexible y tiene un buen manejo de los datos faltantes. Además, ofrece regularización robusta, lo cual es efectivo para evitar el sobreajuste. Este modelo que, como ya se ha explicado está basado en árboles, realiza crecimiento nivel por nivel (*Level-wise tree growth*) de sus estructuras, lo que significa que se enfoca en el balanceo del árbol. Por lo general, resulta en árboles menos profundos (menos niveles) y puede ser más costoso en tiempo y memoria.

- **Máquina Ligera de Refuerzo de Gradiente LGBM**: Como su nombre señala, este tipo de modelo está diseñado para ser más ligero (ocupar menos memoria) que XGB. A diferencia de este, LGBM sigue una estrategia de expansión a nivel de nodo (*Leaf-wise tree growth*) [9], es decir, prioriza el crecimiento del árbol en base a sus mejores nodos, permitiéndole no tener que explorar el árbol por completo. Además, utiliza histogramas para agrupar los valores de las variables continuas en *bins* discretos para agilizar el entrenamiento y reducir el uso de memoria.

La Figura 4.14 ejemplifica las distintas estrategias que siguen los modelos XGB y LGBM a la hora de expandirse.

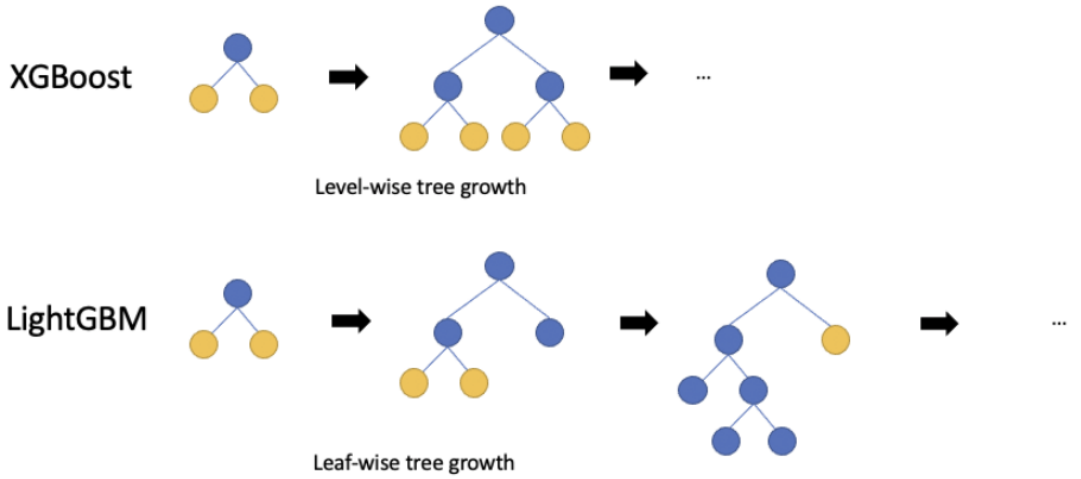


Figura 4.14: Diferentes técnicas de expansión de árboles

4.5 Métricas

A continuación se exponen las principales métricas o funciones de pérdida utilizadas para cuantificar la calidad de los distintos modelos que han sido ejecutados.

4.5.1 Coeficiente de Determinación

También conocido como R2 (o R cuadrado), indica cuánta variación tiene la variable a predecir respecto a la variable independiente, es decir, cómo de bien el modelo se está ajustando a las observaciones reales del conjunto de datos. Esta métrica toma en cuenta todas las variables independientes existentes en el conjunto de datos [59].

Su valor será 0 en el peor de los casos, o 1 si nuestro modelo tiene una precisión perfecta. La principal desventaja de esta métrica es su hipótesis de que cada variable está contribuyendo positivamente a la calidad del modelo, lo cual no siempre es correcto. Dicho de otra forma, R2 puede hacernos creer que el modelo está mejorando al añadir nuevas variables, cuando puede no ser cierto, aunque su valor haya aumentado al añadir más variables.

Se calcula de la forma:

$$R^2 = 1 - \frac{\sum(y_i - x_i)^2}{\sum(y_i - \mu_y)}.$$

En caso de querer estudiar más en detalle los cambios producidos por la introducción de nuevas variables al modelo, podría ser útil el uso de R2 ajustado, que penaliza la adición de variables independientes de la forma:

$$R^2 Ajustado = 1 - \frac{(1 - R^2)(N - 1)}{N - M - 1},$$

donde N es el número de filas del conjunto de datos y M, el número de variables.

4.5.2 Media del Error Absoluto (MAE)

Por su parte, MAE es una sencilla métrica que permite calcular la diferencia entre dos valores, en este caso entre el predicho y el real. Se utiliza el valor absoluto de estas diferencias para evitar que los errores de signo opuesto se cancelen. Además, se calcula la media de dichos errores, siguiendo la fórmula:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}.$$

Dado que cuantifica la diferencia entre el valor de salida del modelo y el valor real, un buen modelo tendrá un valor de MAE cercano a cero.

4.5.3 Media del Error Cuadrático (MSE)

Similar al cálculo del MAE, pero utilizando el cuadrado de las diferencia en lugar de su valor absoluto. Al igual que MAE, podemos comparar distintos modelos en base a esta métrica, siendo de mayor calidad el modelo que tenga un MSE más cercano a cero.

$$MSE = \frac{\sum_{i=1}^{n} (y_i - x_i)^2}{n}.$$

4.5.4 Función de pérdida de cuantil

Conocida por su nombre en inglés *Quantile Loss* [60], es la función de pérdida utilizada en modelos como TFT (4.4.3) y en problemas de regresión cuantílica. A diferencia de las funciones de pérdida absolutas como las anteriores MSE o MAE, la función de pérdida cuantílica está enfocada en minimizar los errores que se produzcan entorno a unos puntos específicos de la distribución de la variable objetivo (cuantiles).

Para un cuantil determinado q, donde $0 < q < 1$, se calcula la pérdida cuantílica o *Quantile Loss* como:

$$L_q(y, \hat{y}) = \begin{cases} q \cdot (y - \hat{y}) & \text{si } y \geq \hat{y}, \\ (q-1) \cdot (y - \hat{y}) & \text{si } y < \hat{y}, \end{cases}$$

donde y es el valor observado de la variable e $\hat{y}$, el valor predicho.

4.6 Tratamiento de anomalías y datos faltantes

Para concluir este capítulo, se incluye una explicación breve de los algoritmos utilizados en la fase de detección de anomalías y su posterior *imputación*, es decir, su sustitución por nuevos datos.

4.6.1 DBSCAN

Dentro del campo de los algoritmos de aprendizaje no supervisado, como se detallaba en la Sección 4.4, se encuentran algunos algoritmos destinados al *clustering* como es el caso del de Agrupación Espacial Basada en Densidad de Datos con Ruido (DBSCAN.) Este emplea la distancia entre un punto concreto y sus puntos más cercanos para determinar su pertenencia o no al grupo de dichos vecinos [10]. A diferencia de otros algoritmos de *clustering* como KMeans, cuya función suele estar limitada a establecer grupos solamente basados en la distancia de los puntos a un determinado punto central (por lo que 2 puntos muy cercanos entre sí pueden pertenecer a grupos cuyo centro está más alejado de los puntos que ellos mismos), DBSCAN puede determinar *clusters* de formas arbitrarias y no necesita que se le indique el número de agrupaciones (*clusters*) que debe formar.

Los algoritmos de*clustering* basados en densidad, como DBSCAN, identifican grupos distintivos en el conjunto de datos en base a la idea de que una agrupación debe ser una región contigua del conjunto de datos con una alta densidad de puntos de ese grupo, aislado de otros *clusters* por regiones de poca densidad no pertenecientes a dicho grupo. En cambio, KMeans asume que los *clusters* son circulares sin importar la distribución de los datos en el espacio. Esto puede visualizarse en la Figura 4.15 .

Otra de las ventajas de DBSCAN será la posibilidad de detectar puntos que no pertenecen a ningún grupo, permitiéndole clasificarlos como puntos anómalos. Este algoritmo necesita algunos parámetros de entrada:

- **MinPts**: Número mínimo necesario de puntos agrupados juntos para que una región sea considerada un*cluster*.

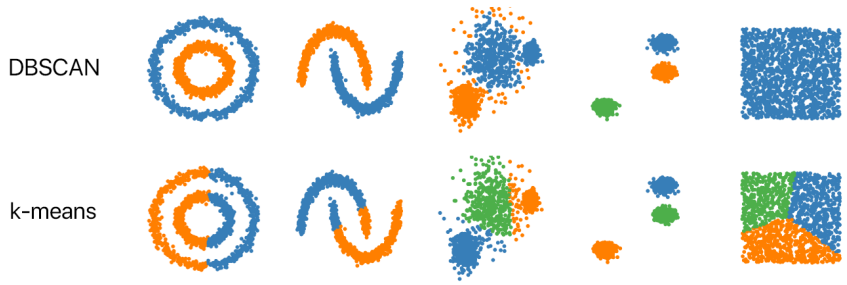


Figura 4.15: Diferencias entre los *clusters* formados por DBSCAN y KMeans para distintos escenarios [10].

- **Epsilon**: Umbral que se utilizará como límite a partir del cual la distancia entre puntos hace que no se consideren contiguos.

4.6.2 Imputación

Una vez se hayan identificado los posibles puntos anómalos, deberán ser retirados del conjunto de datos. En este proyecto, al estar trabajando con series temporales, los datos faltantes no son convenientes, ya que estas series perderían la continuidad de su información. Los valores inicialmente faltantes junto con los extraídos del proceso de detección de anomalías serán imputados, es decir, sustituidos por valores estimados con diferentes algoritmos. Como se mencionaba en la Sección 4.1, en la literatura podemos encontrar trabajos relacionados con este proyecto que proponen la técnica conocida como la Descomposición de Tendencia Estacional utilizando el método LOESS (STL), que tiene como objetivo preservar la estructura general de la serie temporal, en particular sus componentes tendenciales y estacionales, al completar los valores faltantes, siendo más probable que los valores imputados reflejen la dinámica real de la serie temporal, lo que da lugar a modelos más precisos.

Esta técnica es especialmente útil en escenarios en que se utilizan series temporales, ya que utiliza específicamente su estacionalidad y demás componentes de la serie temporal para regenerar la serie, siendo capaz de capturar patrones complejos.

Otras técnicas frecuentemente empleadas a la hora de trabajar con datos faltantes [61] son:

- **Imputación constante**: sustituye los datos faltantes por una constante.
- **Imputación de Media, Mediana o Moda**: Aunque menos perjudicial en el caso de las series temporales que el método anterior, este tipo de imputación podría perjudicar la variabilidad de la serie y modificar las componentes de estacionalidad.

- **Anterior o próximo**: También llamados Última Observación Llevada hacia Delante (LOCF) y Próxima Observación Retrotraída hacia Atrás (NOCB), sustituyen el valor faltante por el anterior o próximo a este que sean conocidos y válidos.
- **Imputación por media móvil**: Asigna al dato faltante el valor de la media móvil en ese punto de la serie. Más complejo de configurar al tener que seleccionar parámetros como el tamaño de la ventana, puede no ser eficaz en escenarios con muchos datos faltantes contiguos.
- **Interpolación Lineal**: Los valores faltantes se reemplazan según una ecuación lineal derivada de los valores disponibles, como la media entre el anterior y el próximo. Supone una relación lineal entre observaciones.
- **Interpolación polinómica o spline**: Utiliza polinomios de diferentes grados para estimar la serie, pero supone patrones concretos en la estacionalidad de la misma.
- **K vecinos más próximos**: Sustituye el valor faltante en función de sus vecinos más cercanos, generalmente con la media de los mismos.

Capítulo 5

Estudio de los conjuntos de datos

Como primera fase del desarrollo práctico del proyecto, se estudian los conjuntos de datos con el objetivo de analizarlos, extraer información relevante e integrar los subconjuntos de datos de distintas fuentes y formatos en un conjunto de datos único que pueda ser empleado en las siguientes fases del proyecto. Además, al tratarse de un proyecto en colaboración con empresas externas, esta labor puede ser realizada en múltiples ocasiones, en función de la llegada de nuevos datos o de correcciones en los mismos por parte de los colaboradores.

5.1 Fuentes de datos y operaciones genéricas

Los conjuntos de datos que se exponen a continuación pertenecen a envíos periódicos de las empresas participantes en el proyecto. Estos datos son recibidos en archivos de tipo *csv* (valores separados por comas), *tsv* (valores separados por tabuladores) o excel. Las mediciones de cada una de las distintas estaciones son almacenadas en archivos específicos, es decir, si una EDAR tiene 10 puntos de medida, se estarían recibiendo 10 archivos diferentes.

La lectura de todos estos archivos se realizará a través del lenguaje de programación Python, concretamente haciendo uso de su librería Pandas, generando estructuras de datos del tipo Dataframe. Al contar con diferentes archivos, se necesitarán las funciones de esta librería para realizar la unión entre ellos y conformar un único *Dataframe*, como la función *merge* o *join* [62], que permiten la unión de las tablas como si se tratase de una base de datos relacional, haciendo uso de una clave común, que en este caso será la fecha y hora de las mediciones.

Previo a la unión de las distintas fuentes de datos, deben ser comprobadas las frecuencias de muestreo de los conjuntos de datos. Dado que se busca trabajar en formato horario, tanto para las predicciones como para el procesamiento de los datos, será necesario identificar y corregir aquellos conjuntos de datos que no cuenten con esa resolución temporal, como se muestra en la Figura 5.1

62	2020-01-01 07:24:30.000
63	2020-01-01 07:24:40.000
64	2020-01-01 07:24:50.000
65	2020-01-01 07:25:00.000
66	2020-01-01 07:49:00.000
67	2020-01-01 08:13:00.000
68	2020-01-01 08:37:00.000
69	2020-01-01 09:01:00.000
70	2020-01-01 09:25:00.000
71	2020-01-01 09:47:30.000
72	2020-01-01 09:47:40.000
73	2020-01-01 09:47:50.000
74	2020-01-01 09:48:00.000

Figura 5.1: Ejemplo de los datos muestreados irregularmente.

A indicación de las empresas colaboradoras, la ausencia de mediciones en un determinado momento debe interpretarse como que el valor medido no ha variado desde la última medición. Es decir, si la columna de tiempo realiza un salto temporal de 1 hora, todos los valores de medida de ese espacio de tiempo se corresponden con el valor del último instante previo al salto temporal. El procedimiento seguido en estos casos, por lo tanto, constará de tres pasos: remuestrear los datos con una resolución temporal de alta frecuencia (en el caso de la Figura 5.1, cada 10 segundos) para obtener una serie temporal espaciada regularmente en el tiempo, imputar los datos faltantes en estos nuevos instantes por el último valor anterior conocido, y por último, volver a muestrear los datos utilizando en esta ocasión la resolución temporal de una hora con el valor medio de la intensidad de caudal.

El proceso de remuestreo y rellenado mencionados antes pueden entenderse fácilmente con la figura 5.2. En ella, a partir de un conjunto de datos con muestreo irregular, es *resampleado* o remuestreado a la frecuencia deseada y posteriormente se rellenan esos datos faltantes haciendo uso de la función *ffill()*, que se corresponde con el rellenado hacia delante (LOCF). En el caso de este proyecto, tras el *ffill()* se utiliza un *resample('1h')* para alcanzar la resolución temporal deseada.

5.2 Datos de Sonderso

Como se mostraba en la Sección 4.2.2 y en la Figura 4.3, esta localización cuenta con una red de alcantarillado que puede dividirse en dos secciones, una por el oeste y otra por el este. A consecuencia de esto, las EBAR nombradas como Vefps009 y Soenps005 serán de

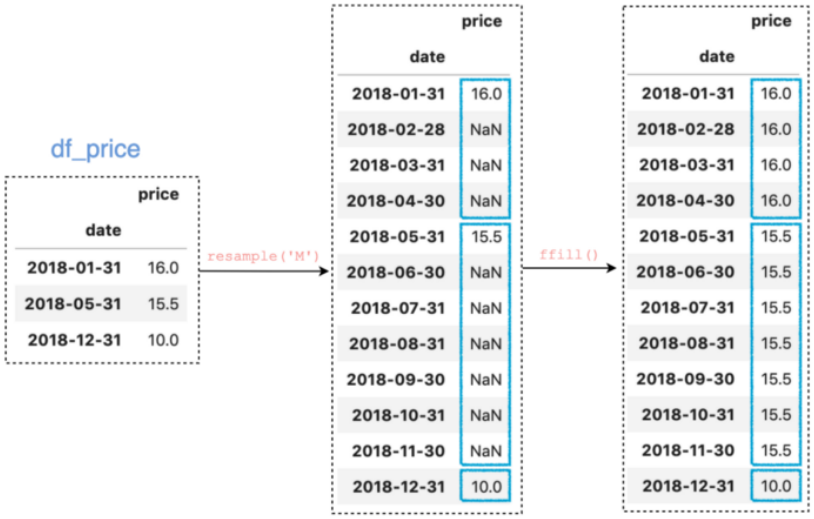


Figura 5.2: Ejemplo del proceso de remuestreo y rellenado con Pandas [11].

suma importancia, pues a ellas llegará todo el volumen de agua agregado desde las otras bombas de su sección. Estos datos se procesarán en todo momento con frecuencia horaria y en unidades de volumen (m^3/h), a menos que se indique lo contrario. Además, por simplicidad, las variables correspondientes al caudal serán denominadas como el nombre de la estación a la que representan, siendo la variable de nombre *Sonderso* realmente *Caudal_EDAR_Sonderso*.

En la Figura 5.3 se muestra el conjunto de datos que se utilizará como caudal de la EDAR de Sonderso, que se corresponde con el envío más reciente de datos por parte de esta empresa. En el eje horizontal se muestra el rango de fechas que abarca, de Noviembre de 2021 a Septiembre de 2023. En el eje vertical, en m^3/h, se muestra el volumen de agua que llega a la estación en cada instante de tiempo. Se observa que la serie de tiempo contiene una franja de datos de volumen medio, rodeados de múltiples picos de crecida del caudal.

Por otra parte, la Figura 5.4 muestra el volumen que asume la EBAR principal de la sección oeste, conocida como Vefps009. Se observa a simple vista que tienen una forma muy similar.

Por último, la EBAR principal de la sección este, identificada como Soenps005, se ilustra en la gráfica de la Figura 5.5. En comparación con las otras, esta será la de menor caudal (esto se observa mejor en la Figura 5.6).

Como se mencionaba, todas estas series temporales presentan grandes subidas y bajadas en momentos concretos y teniendo formas muy similares. Estas oscilaciones se deben a la llegada de un mayor volumen de agua causado por eventos de precipitaciones intensas, los cuales son

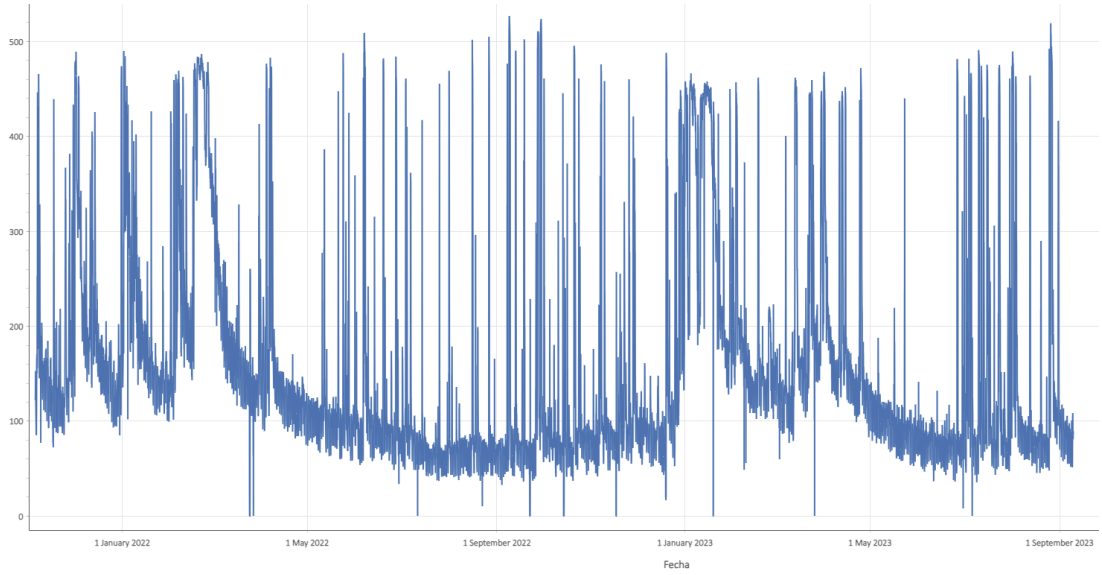


Figura 5.3: Caudal de la EDAR de Sonderso

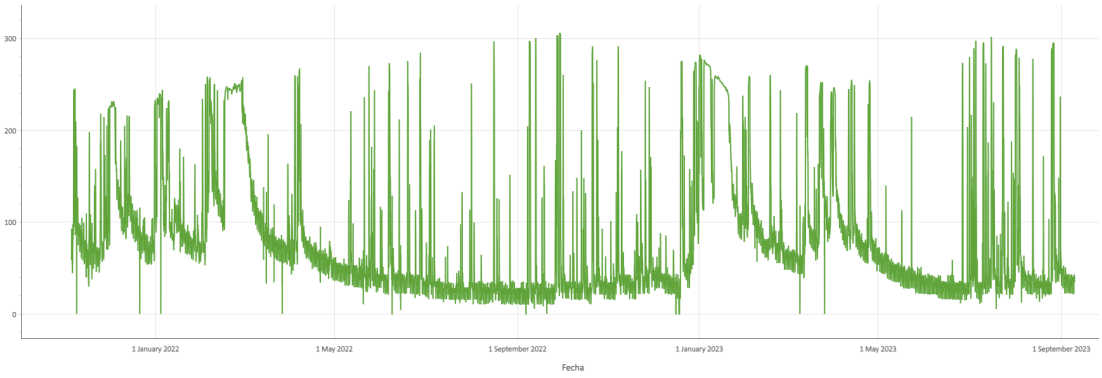


Figura 5.4: Caudal inicial de Vefps009

críticos para las instalaciones. Además, se debe tener en cuenta que los caudales de Vefps009 y Soenps005 combinados llegan a la central de Sonderso, por lo que serán siempre menores que esta y los aumentos de caudal que sufren estarán afectando a la estación principal en donde desembocan. Puede apreciarse la influencia de la lluvia sobre las estaciones en la Figura 5.7, donde se puede ver en rojo la intensidad de lluvia en cada momento. El dato original de lluvia medido en segundos se ha convertido a formato horario mediante la media, por lo que la variable *lluvia* representará la intensidad media de lluvia medida en cada hora.

Una vez presentado el conjunto de datos del que se parte, puede comenzarse con el análisis exploratorio de los datos. En primer lugar, comprobando la existencia de datos faltantes y de instantes en los que los caudales valen cero, cosa que no debería suceder ya que la empresa

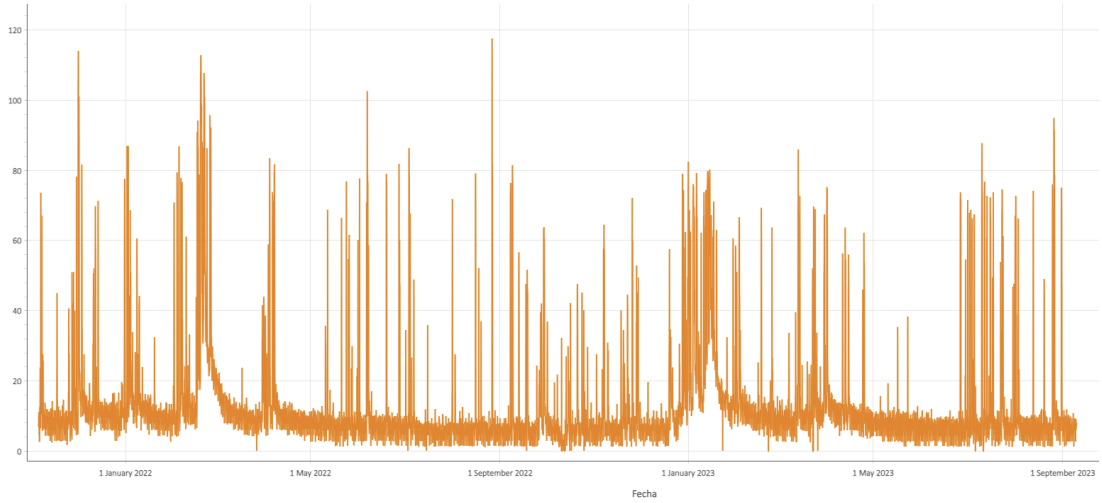


Figura 5.5: Caudal inicial de Soenps005.

indica que en ningún momento la estación está completamente vacía. También comprobaremos la existencia de datos nulos en la variable *lluvia*, ya que en esta sí pueden existir ceros en momentos en los que no haya llovido. Se muestran los resultados de estas comprobaciones, hechas en Python, en la Tabla 5.1.

Serie	Datos faltantes	Nº ceros	Mediciones válidas
Sonderso	2	3	16154
Vefps009	2	22	16135
Soenps005	2	9	16148
Lluvia	2	14938	16157

Tabla 5.1: Comprobación inicial de ceros y nulos en Sonderso.

Tras comprobar la situación de estos datos faltantes, se decide simplemene eliminar esas observaciones del conjunto de datos, ya que se corresponden con los momentos en que se han producido cambios de hora (finales de marzo de los años 2022 y 2023), donde no se han hecho mediciones. En el caso de los ceros de cada variable, más adelante se sustituirán por valores nulos que serán imputados.

Como se ha mencionado en la Sección 3, Python cuenta con una librería que permite utilizar la API del DMI, mediante la cual pueden obtenerse los datos meteorológicos de la ciudad de Sonderso. En este caso, se han extraído los valores de temperatura, humedad y presión en el

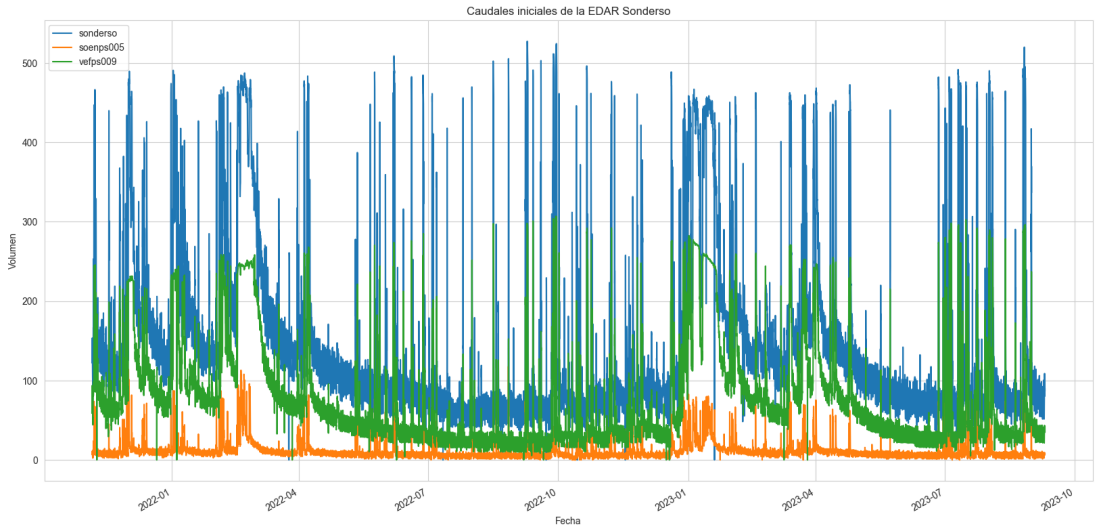


Figura 5.6: Caudales iniciales de las principales estaciones de Sonderso.

periodo de tiempo que cubre el conjunto de datos. Se muestran en la Figura 5.8. La presión está medida en hPa, la humedad, en puntos porcentuales y la temperatura, en grados Celsius.

También será importante conocer los rangos de valores en los que se presentan las variables, así como su media y su desviación típica. Esto se refleja en la Tabla 5.2. En este caso se observa la alta desviación típica de todas las series de tiempo de caudales, así como unos máximos mucho mayores a sus medias, que se corresponden con las mencionadas temporadas de lluvias intensas.

Serie	**Media**	**Desviación**	**Máximo**	**Mínimo**
Sonderso	154.02	109.81	527.31	0.00
Vefps009	81.42	68.11	306.87	0.00
Soenps005	12.09	13.45	117.69	0.00
Lluvia	0.02	0.13	5.65	0.00
Temperatura	9.41	6.72	35.10	-12.40
humedad	81.25	15.90	100	28.10
Presión	1015.02	11.17	1050.30	969.70

Tabla 5.2: Estadísticas de las variables del conjunto de datos de Sonderso.

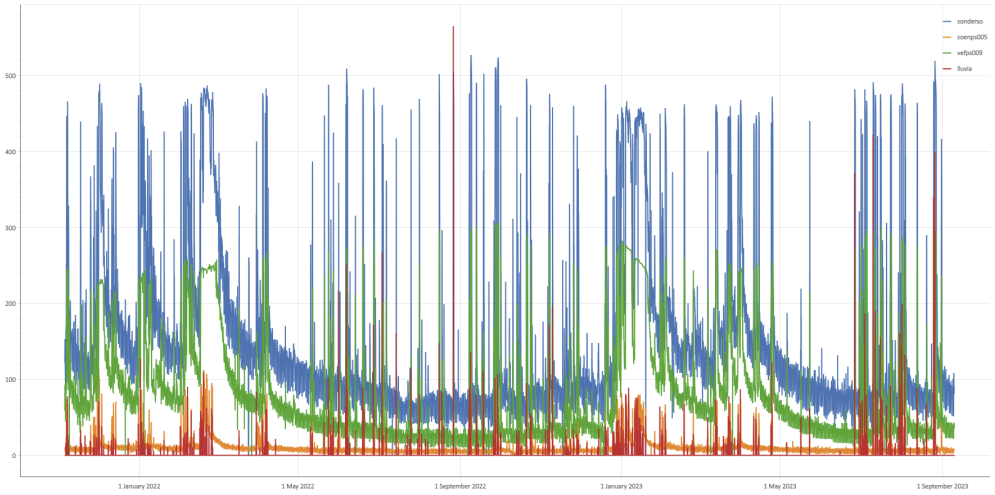


Figura 5.7: Influencia de la lluvia en los caudales de Sonderso

Además de estas analíticas univariantes, se realiza un análisis multivariante para conocer las posibles interacciones entre las variables del conjunto de datos. En este caso, en la Figura 5.9 se muestran gráficamente las correlaciones entre las distintas variables del conjunto de datos. Como es de esperar, Vefps009 y Soenps005 tienen valores muy altos de correlación con la serie principal Sonderso, mientras que la lluvia parece ser la de menor correlación. En general, las variables climáticas no parecen estar altamente correlacionadas con los caudales.

Dado que se trata de series temporales, tal como se explica en la Sección 4.3, será necesario el análisis de sus componentes, así como estudiar si se trata de un proceso estacionario. Para analizar sus componentes, la librería *stasmodels* incorpora la función *seasonal_descompose*, la cual descompone la serie introducida en sus componentes de serie temporal, como se muestra en la Figura 5.11. Por motivos de una mejor visibilidad de la imagen, se muestra solamente la descomposición de un fragmento de los datos, dado que de incluirse entero la imagen en este documento sería ilegible. Además, se muestra en la Figura 5.10 la tendencia extraída de la serie completa.

A continuación, se realiza una comprobación de posible proceso estacionario sobre la serie, utilizando los tests ADF y KPSS. Estos tests devuelven p-valores de 0.00002 y 0.001, respectivamente, por lo que deberían rechazarse sus hipótesis nulas al ser mucho menores que cualquier nivel de significancia lógico. Teniendo en cuenta que esto significaría asumir que la serie es estacionaria según ADF y a su vez no estacionaria según KPSS, no habría evidencia clara de que la serie sea estacionaria o no. En esta situación, puede ser útil aplicar una diferenciación sobre los datos y repetir los tests.

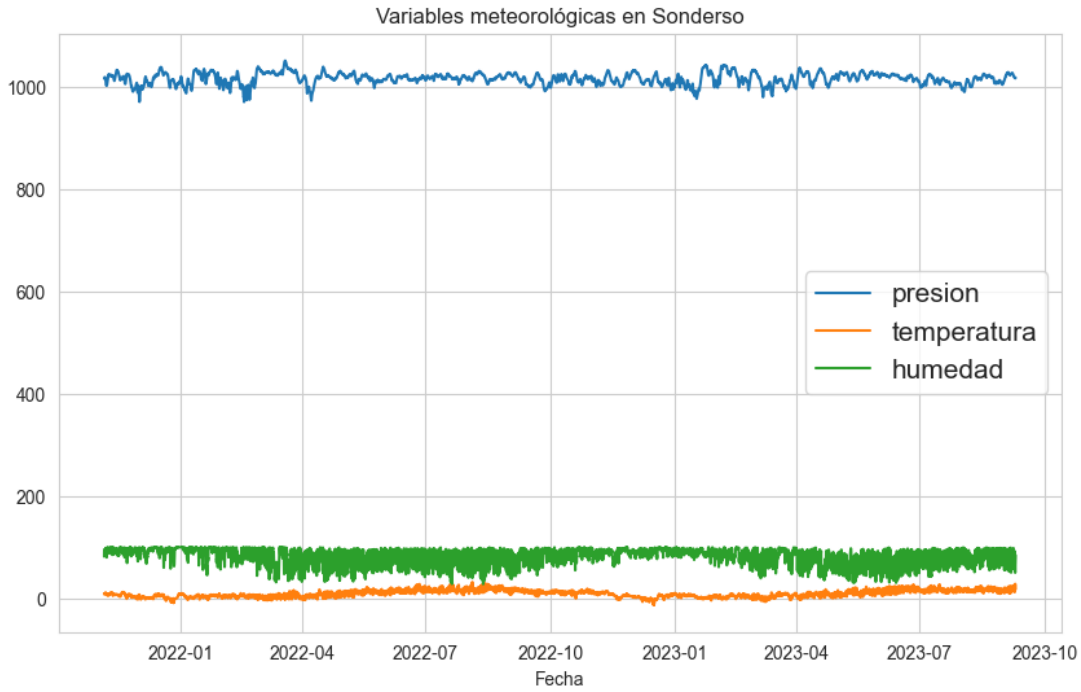


Figura 5.8: Mediciones meteorológicas extraídas mediante la API del DMI.

Habiendo diferenciado la serie, los nuevos p-valores son de aproximadamente 0 para ADF y 0.04 para KPSS. Podría decirse que el test de KPSS no es significativo si se establece un nivel de significancia menor que 0.04, siendo todavía el test de ADF significativo, concluyendo en una ligera evidencia de que la serie una vez diferenciada sí podría ser estacionaria.

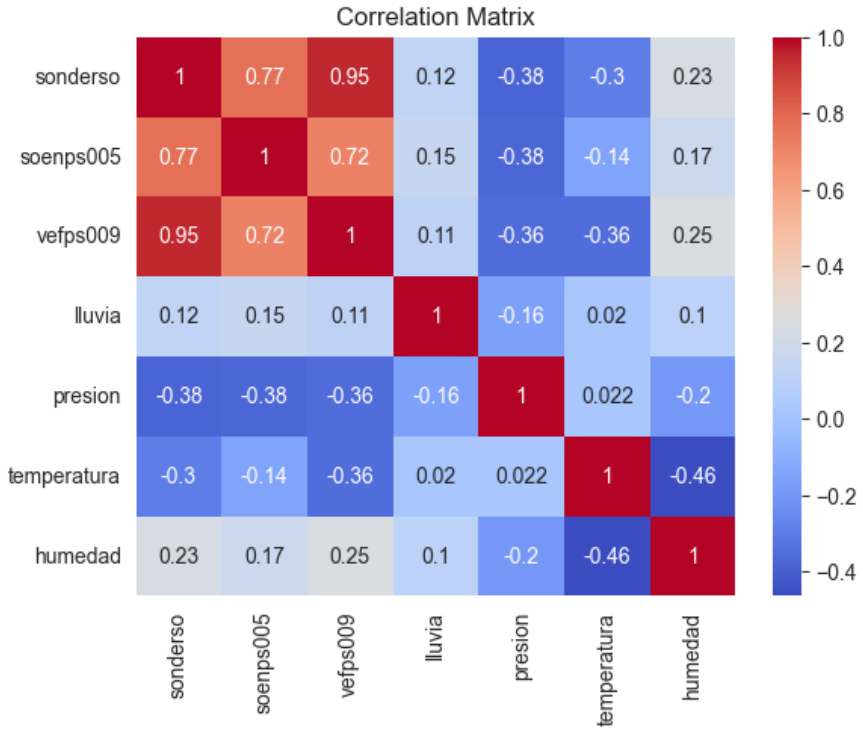


Figura 5.9: Correlaciones de las variables iniciales del conjunto de datos de Sonderso.

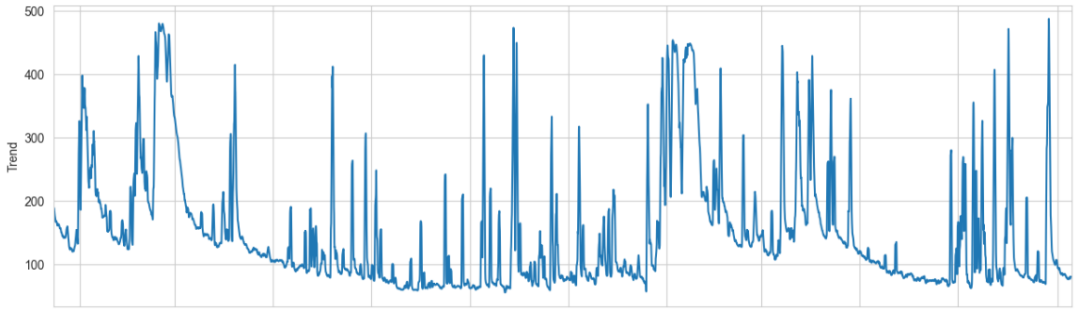


Figura 5.10: Componente de tendencia de la serie completa de Sonderso.

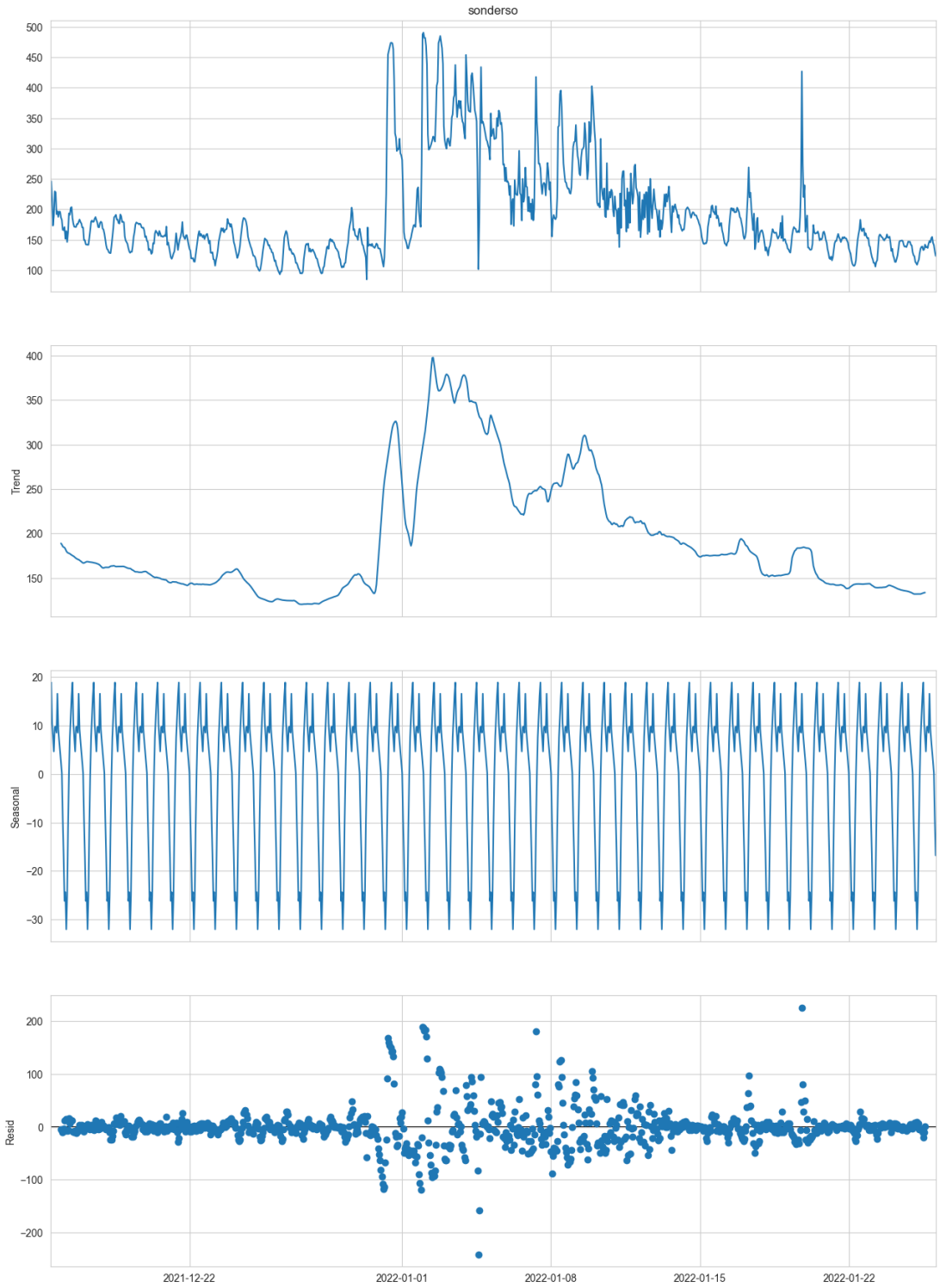


Figura 5.11: Descomposición de la serie temporal Sonderso en sus componentes.

5.3 Datos de Moaña

Como se comenta en secciones previas en esta memoria, existe una segunda EDAR, situada en la comarca de Moaña, en Pontevedra, España. Aunque finalmente este Trabajo de Fin de Grado (TFG) está centrado específicamente en los procedimientos llevados a cabo en la estación danesa de Sonderso y no se entrará en detalle en los procedimientos de esta segunda estación, el proyecto Life Reseau buscará también aplicar estas técnicas a la planta de Moaña. Para comenzar, los datos enviados contienen información sobre los caudales de las distintas EBAR, el funcionamiento de sus bombas y los niveles (alturas del agua en los pozos) de cada estación. Sin entrar en los mismos detalles que se explicaron para la estación de Sonderso en la Sección 5.2, se han integrado los sucesivos envíos de datos mensuales en un único conjunto de datos y eliminado aquellas variables que se han indicado como no utilizables o innecesarias. Además, se ha hecho uso de la web de Puertos del Estado para la extracción de la información meteorológica necesaria, a la cual se añade en este caso el nivel de la marea (*marea*) del puerto de Vigo, la cual puede resultar influyente en esta estación por su ubicación, mostrada en la Figura 4.4. Las variables utilizadas se muestran en la Figura 5.12, siendo *entrada_sanbarto* el caudal que entra en la estación y *nivel_metros* la profundidad en metros de su pozo.

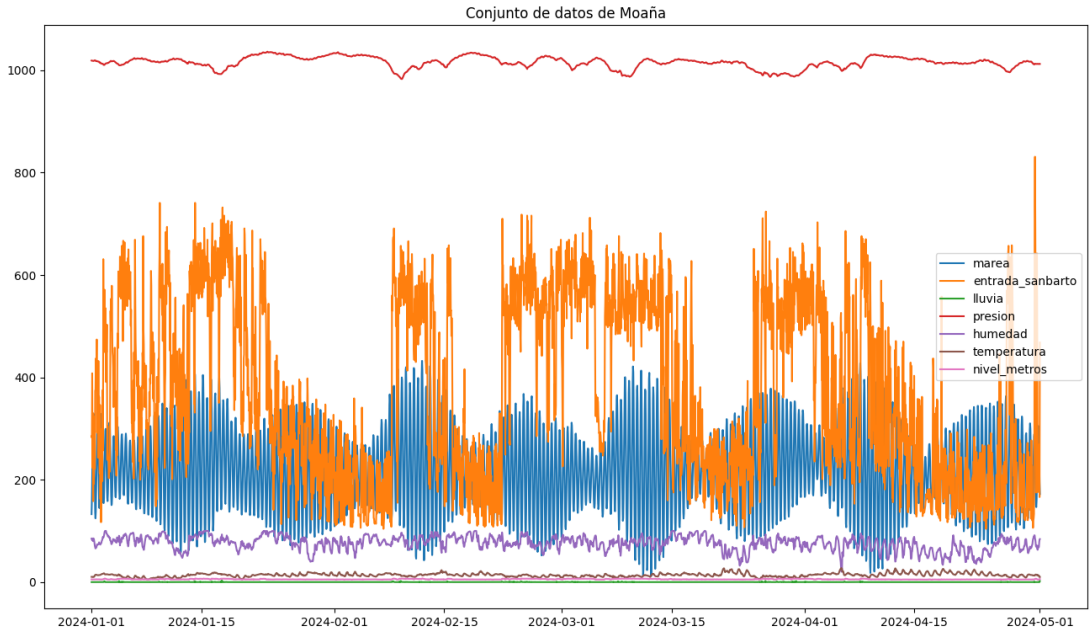


Figura 5.12: Conjunto de datos de la EDAR de Moaña, ya procesados.

Este conjunto de datos presenta una serie de problemas de compleja solución, los cuales requerirán trabajo futuro y colaboración con la empresa que los genera para su correcta medición, por lo que finalmente se ha decidido centrar este TFG en la estación danesa. Además

de los problemas de inconsistencia de variables que ha provocado el descarte de múltiples variables debido a la falta de gran cantidad de mediciones en ellas, estos datos presentan una variabilidad demasiado alta, como se ve en la Figura 5.13, presente a lo largo de toda la serie temporal y que no se corresponde con el comportamiento de ningún otro punto de la red ni con ninguna variable conocida.

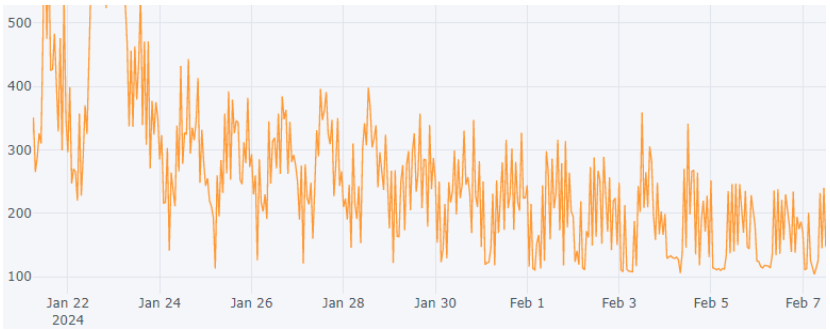


Figura 5.13: Detalle de la extrema variabilidad del caudal de la estación de Moaña

En cuanto a la correlación con otras variables del conjunto, únicamente el *nivel_metros* resulta relevantemente correlacionada con el caudal de *entrada_sanbarto*, con un 0.81. Su correlación con la lluvia apenas supera el 0.2.

A pesar de estas problemáticas, se presentan en la Sección 7.7 algunas posibles predicciones sobre este conjunto de datos utilizando los modelos de AA que se habían implementado .

Capítulo 6

Procesado de datos anómalos

Tras explorar los datos de los que se dispone y realizar un breve preprocesado, se afrontará la problemática de los datos anómalos, que serán debidamente detectados y procesados y posteriormente imputados. Los conceptos teóricos relacionados con esta sección han sido explicados anteriormente en la Sección 4.6.

Para el desarrollo de esta sección se ha utilizado el algoritmo DBSCAN expuesto en el Subapartado 4.6.1. Este algoritmo es capaz de trabajar en espacios multidimensionales, es decir, con múltiples variables, por lo que se han hecho pruebas con múltiples configuraciones de ellas, también generando nuevas variables a partir de las ya existentes. Se recogen en la Tabla 6.1 algunas de las variables generadas.

Variable	Significado	ID
cum_lluvia_nh	Acumulación de lluvia en las N horas previas.	A
horas_sin_lluvia	Horas sin llover.	B
diff_hora_previa	Diferencia con observación anterior.	C
diff_rolling_mean	Diferencia con media móvil.	D
diff_mean_hora	Diferencia con la media de esa hora.	E
hour_sin, hour_cos	Transformación mediante codificación cíclica.	F

Tabla 6.1: Variables generadas a partir de las existentes.

El objetivo principal de esta sección será la comprobación mecánica de la existencia de posibles datos anómalos que pudiesen no ser detectados de forma manual y visual, buscando en todo momento una selección de anomalías comprensible y justificable en base a las variables conocidas del conjunto de datos.

Además de las variables, DBSCAN deber ser ajustado en cuanto a sus parámetros epsilon y minimo de puntos, ya explicados en el Apartado 4.6.1. El proceso de detección de datos anómalos en una serie temporal se realiza generalmente a base de diferentes pruebas utilizando distintas combinaciones de parámetros. Para el ajuste de todos estos campos, se irán realizando sucesivas pruebas guiadas por los resultados visuales obtenidos en cada una y por el coeficiente de *Silhouette*, que determina la cohesión de los *clusters* obtenidos.

En la Tabla 6.2 se muestran algunas de las mejores configuraciones obtenidas, utilizando como columna IDs los identificadores de la Tabla 6.1. Todas las configuraciones mostradas incluyen la variable de caudal de Sonderso (V = Vefps009, S = Soenps005); *Nº* se corresponde con el número de puntos marcados como anómalos, de un total de 16159.

IDs	eps	minPts	Nº	Clus.	silho.
B, C, D	0.901	25	339	2	0.68
B, C	0.999	25	145	2	0.65
V, S	0.546	25	145	2	0.61

Tabla 6.2: Resultados de algunas de las configuraciones probadas para DBSCAN.

Finalmente, en base a los resultados visuales y a la mayor puntuación de Silhouette, se utilizó la configuración que incluye las variables adicionales *horas_sin_lluvia*, *diff_hora_previa*, *diff_rolling_mean*. En la Figura 6.1 se visualiza la salida del modelo DBSCAN que habría producido tal configuración. Esta proyección en forma de gráficos de dispersión es difícil de valorar a simple vista, así que se han elaborado diferentes gráficas para analizar los resultados obtenidos en puntos previa y manualmente seleccionados como críticos.

Para comenzar con los resultados de la detección de anomalías, se destaca una sección de la serie de tiempo de Sonderso en la cual las covariables Vefps005 y Soenps005, afluentes directos del caudal de Sonderso, no presentan una variabilidad muy alta y tampoco existe presencia de lluvia, pero el sensor de Sonderso habría estado generado variaciones nada habituales y sin sentido en el contexto de una red interconectada de canalización. Se muestra en la Figura 6.2, siendo las líneas naranja, rojo y azul las variables Sonderso, Vefps009 y Soenps005, respectivamente. En la Figura 6.3, se muestra el mismo espacio temporal tras eliminar los puntos seleccionados como anómalos por el DBSCAN, y en la Figura 6.4, el mismo espacio de tiempo una vez se ha aplicado al algoritmo de imputación STL.

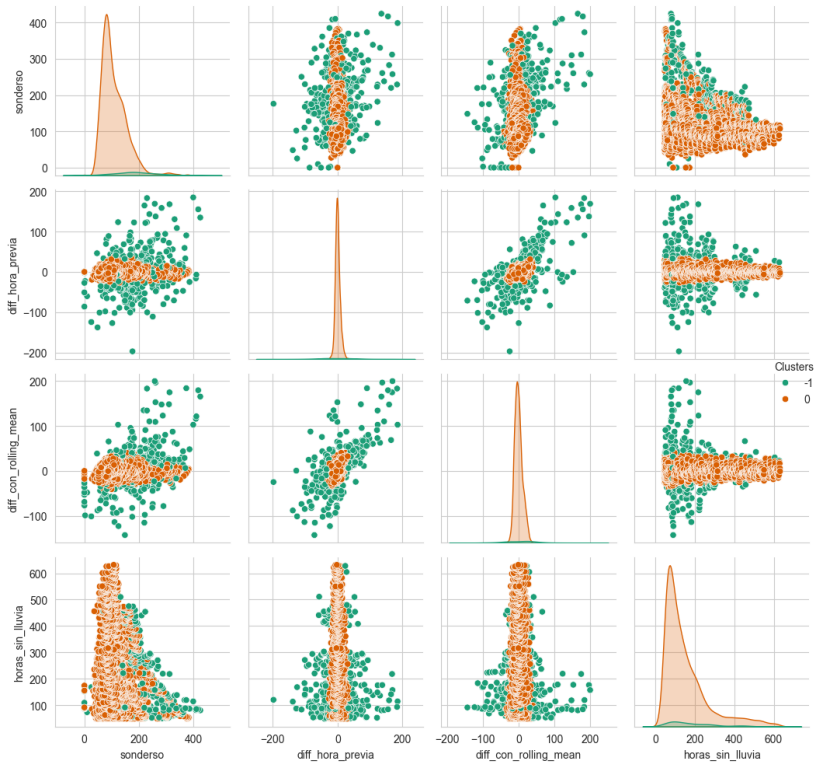


Figura 6.1: Salida del modelo DBSCAN para una de las configuraciones elegidas.

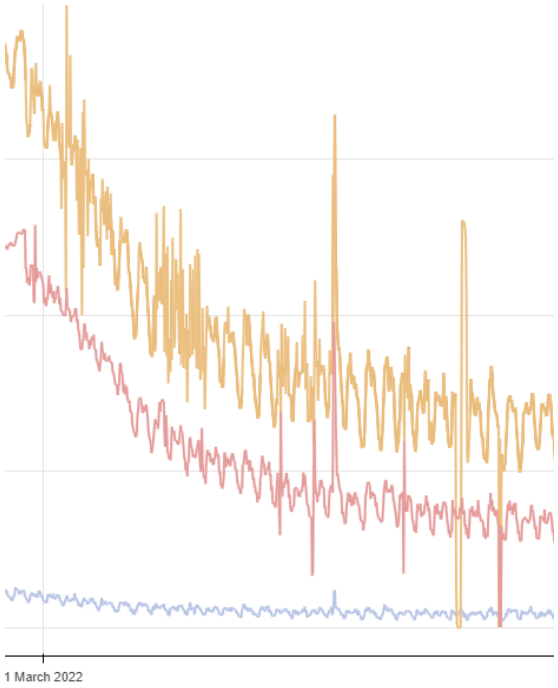


Figura 6.2: Zona 1 seleccionada por posibles datos anómalos.

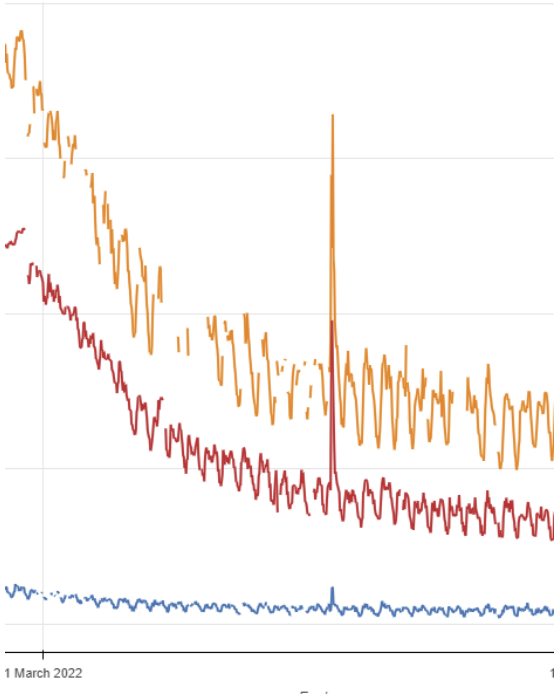


Figura 6.3: Zona 1 de posibles datos anómalos eliminados con DBSCAN.

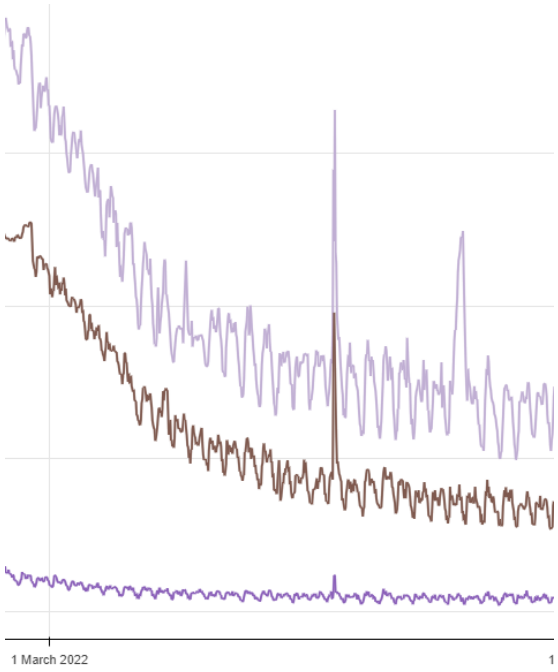


Figura 6.4: Zona 1 de datos anómalos imputada con STL.

A modo de visualización general, se presenta en la Figura 6.5 el resultado final del proceso de detección y eliminación de datos anómalos y la posterior imputación utilizando STL de los faltantes generados. En rasgos generales, este proceso combinado de detección de anomalías con DBSCAN y posterior imputación con STL ha obtenido resultados satisfactorios. Se adjunta el conjunto de datos resultante para las variables principales de caudal una vez limpiadas e imputadas, en la Figura 6.6

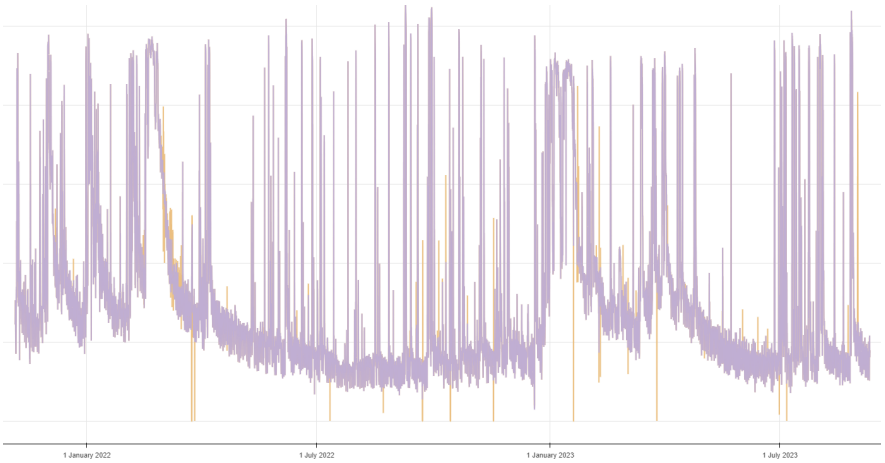


Figura 6.5: Serie temporal de Sonderso completa, tras eliminación de atípicos e imputación STL.

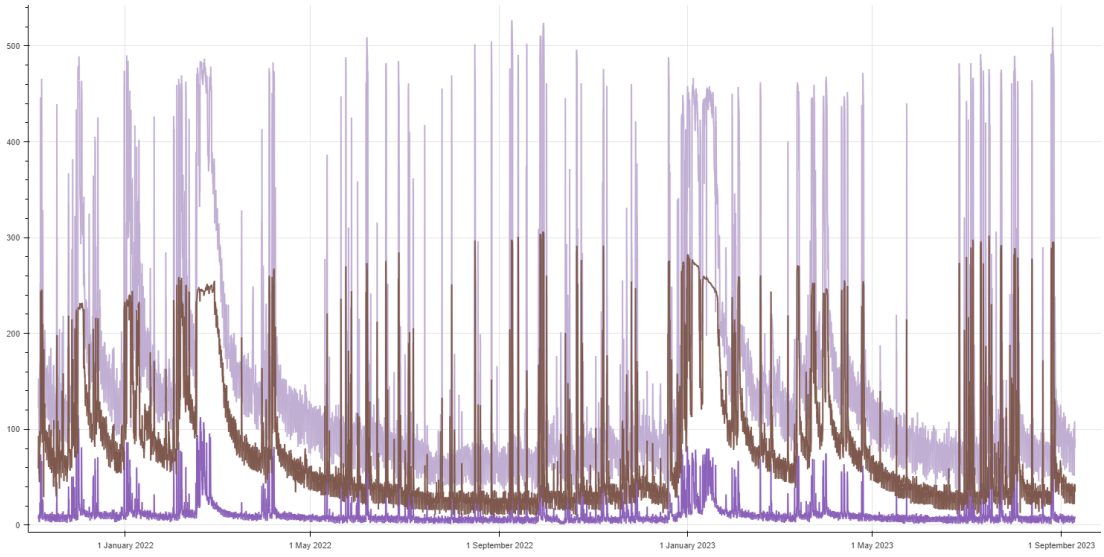


Figura 6.6: Variables de caudal del conjunto de datos tras su limpieza.

Capítulo 7

Modelado

Finalmente, la etapa de modelado tendrá como objetivo el entrenamiento de diferentes modelos de AA para la obtención de predicciones sobre el conjunto de datos. Se utilizarán los modelos expuestos en la Sección 4.4. En esta sección se tratará de exponer brevemente los mejores resultados obtenidos para cada modelo, tratando de realizar comparaciones entre ellos con las diferentes métricas vistas en el Apartado 4.5.

En este apartado, las gráficas de resultados que se muestran se corresponden con la generación de predicciones sobre todo el conjunto de *test*.

7.1 TFT

Como se ha expuesto en las secciones anteriores, concretamente en la Subsección 4.4.3, este modelo ha sido utilizado en base al *framework* AiSystemFramework y con la optimización de parámetros ofrecida por NNI. En base a las variables previamente expuestas, se ha tratado de obtener la mejor configuración capaz de predecir el caudal de la variable *Sonderso*. De entre los mejores modelos obtenidos, destacarán dos, que explicaremos a continuación. El primero de ellos, tomará como entrada únicamente las variables *Sonderso* y *lluvia*, formando ambas partes del codificador y decodificador, es decir, el modelo conocerá en todo momento el valor de la variable de lluvia a la hora de predecir la variable de caudal. Para simplificar la comparativa, la denominaremos configuración **simple**.

Tras su entrenamiento, se visualizan las diferentes configuraciones de parámetros probadas en la Figura 7.1, siendo la métrica la función de pérdida de cuantil (véase la Subsección 4.5.4). En dicha imagen se puede ver que el parámetro de *learning rate* podría ser el más determinante a la hora de conseguir buenos modelos, ya que por lo general los modelos con menor valor en este parámetro han obtenido una mejor métrica final. En otros hiperparámetros no se aprecia una zona en la que todos sus modelos sean buenos como sucede con el *learning rate*.

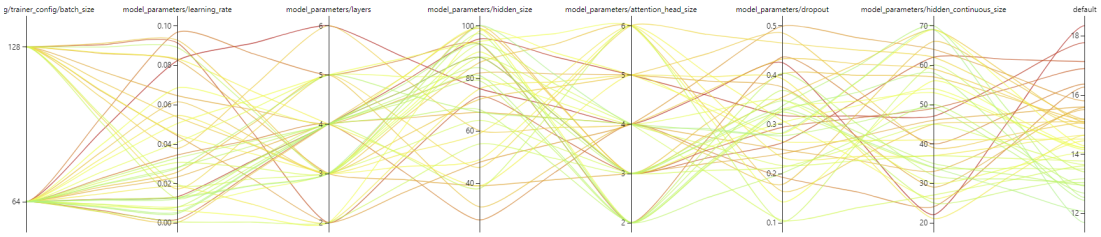


Figura 7.1: Optimización de hiperparámetros llevada a cabo por NNI.

Por otra parte, destacará también el resultado obtenido por el modelo que toma como variables de codificador *Sonderso*, *Vefps009*, *Soenps005* y *lluvia*, a la que denominaremos **compleja**. En la Tabla 7.1 se proporciona la comparativa entre estas dos configuraciones del modelo, mostrando sus parámetros óptimos en base al entrenamiento, así como sus métricas principales.

Parámetro	**Modelo simple**	**Modelo complejo**
tamaño batch	64	128
learning rate	0.013	0.016
capas	3	3
tamaño capas	100	99
tamaño cabezal atención	4	5
dropout	0.282	0.266
Quantile Loss	11.75	10.22
R2	0.912	0.909
MAE	15.73	15.338

Tabla 7.1: Comparativa de modelos TFT.

Por último, se adjunta una comparativa visual de los resultados de estos dos modelos. En la Figura 7.2 se muestra en azul el caudal original de Sonderso, en amarillo, la configuración **compleja** y en verde, la **simple**. Puede apreciarse que ambas configuraciones obtienen resultados generalmente buenos, como indican sus métricas, aunque ambos podrían estar generando errores en algunos de los puntos más altos de las gráficas. Por otra parte, ambos parecen estar captando casi a la perfección el inicio de las tendencias de subida del caudal, lo cual es el momento más crítico y, por lo tanto, donde más relevancia tienen los errores de los modelos.

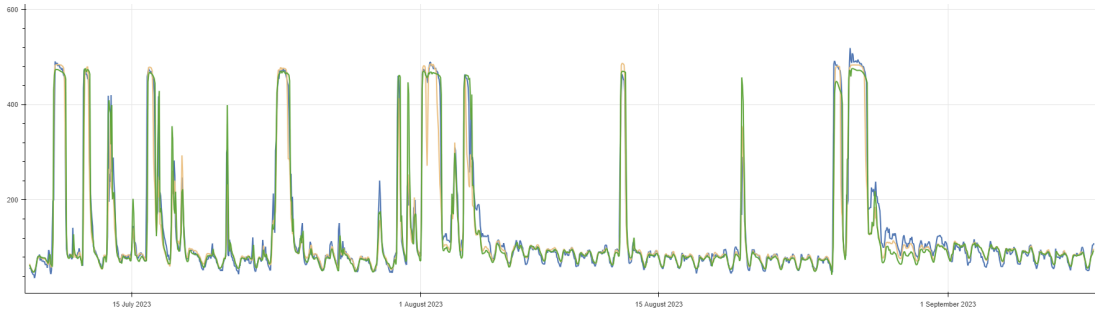


Figura 7.2: Comparativa entre configuraciones seleccionadas de TFT.

7.2 SARIMAX

En cuanto a los modelos de tipo SARIMAX, se automatizará el proceso de selección de parámetros p, d, q, P, D, Q utilizando la función *auto_arima*, la cual está implementada en múltiples librerías de Python. Para conservar la similitud con los demás modelos, en este caso también denominaremos como **simple** a la configuración que toma como entrada las variables de caudal de *Sonderso* y la *lluvia*, siendo el modelo **complejo** el compuesto por las variables *Sonderso*, *Vefps009*, *Soenps005* y *lluvia*. En la Tabla 7.2 se puede apreciar que ambos modelos han resultado en la misma configuración de parámetros SARIMAX, pero el modelo denominado **complejo** habría obtenido un mejor resultado en ambas métricas, por lo que parecería lógico afirmar que SARIMAX es más preciso con la configuración que cuenta con mayor número de variables, incluso superando los resultados del TFT.

Parámetro	Modelo simple	Modelo complejo
(p, d, q)	(2, 0, 0)	(2, 0, 0)
(P, D, Q)	(0, 0, 0)	(0, 0, 0)
s	24	24
R2	0.911	0.934
MAE	14.068	12.930

Tabla 7.2: Comparativa de modelos SARIMAX.

Por el contrario, si se analizan los resultados de forma gráfica como propone la Figura 7.3, donde la línea verde representa las predicciones del modelo **simple** y la roja, del modelo **complejo**, se ve claramente que estas predicciones son peores que las obtenidas por el modelo TFT en la Figura 7.2. Analizando más en detalle los resultados, se llega a la conclusión de que las métricas benefician al modelo SARIMAX por el motivo de que gran parte del conjunto

de datos presenta una determinada estructura estacional que se corresponde con el ciclo de caudal diario en momentos en los que no hay eventos lluviosos, siendo SARIMAX un predictor casi perfecto en estos momentos, pero sobreestimando gravemente los mayores aumentos del caudal, que son precisamente puntos críticos.

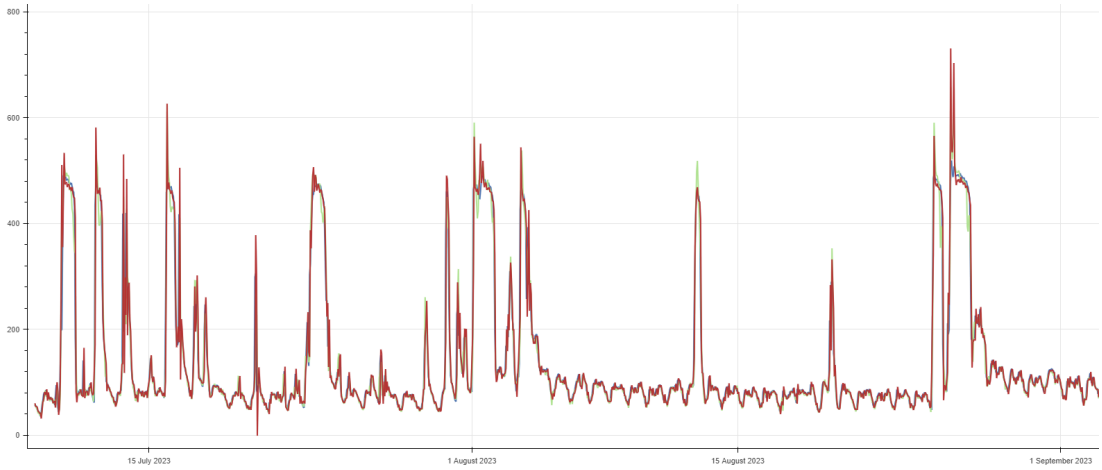


Figura 7.3: Comparativa entre configuraciones seleccionadas de SARIMAX.

7.3 LGBM

Continuando con los modelos ejecutados, es el turno de los modelos LGBM. En este caso, se utilizan implementaciones de este modelo de la librería *lightgbm*. Esta librería facilita también funciones para la búsqueda de hiperparámetros óptimos. Se muestra en la Tabla 7.3 la mejor configuración para los modelos seleccionados. De nuevo, se denomina modelo **simple** a la configuración que toma como entrada las variables de caudal de *Sonderso* y la *lluvia*, siendo el modelo **complejo** el compuesto por las variables *Sonderso*, *Vefps009*, *Soenps005* y *lluvia*.

En esta ocasión, ambos modelos parecen obtener mejores métricas que las seleccionadas para TFT, pero en este caso la comparativa gráfica sí es favorable o, cuanto menos, similar a simple vista. En la Figura 7.4 se ilustran en azul, rosa y morado, respectivamente, el valor real de *Sonderso*, las predicciones con el mejor modelo **simple** de LGBM, y las predicciones con la mejor configuración de la versión **compleja** del mismo. En el caso del **simple**, de color rosa, se aprecian algunas desviaciones menores en las predicciones, que probablemente hayan supuesto las diferencias respecto al **complejo**. También se puede apreciar en la Tabla 7.3 que la mejor configuración del modelo simple ha requerido de mayor número de estimadores y más profundos, aún habiendo obtenido peores métricas, lo que podría señalar que las relaciones

Parámetro	Modelo simple	Modelo complejo
Estimadores (árboles)	200	100
Profundidad máxima árboles	5	4
Learning rate	0.014	0.013
R2	0.916	0.949
MAE	13.802	10.806

Tabla 7.3: Comparativa de modelos LGBM.

entre variables resultan más difíciles de modelar al haber menos variables de entrada. En cambio, el *learning rate* es casi idéntico entre ellos.

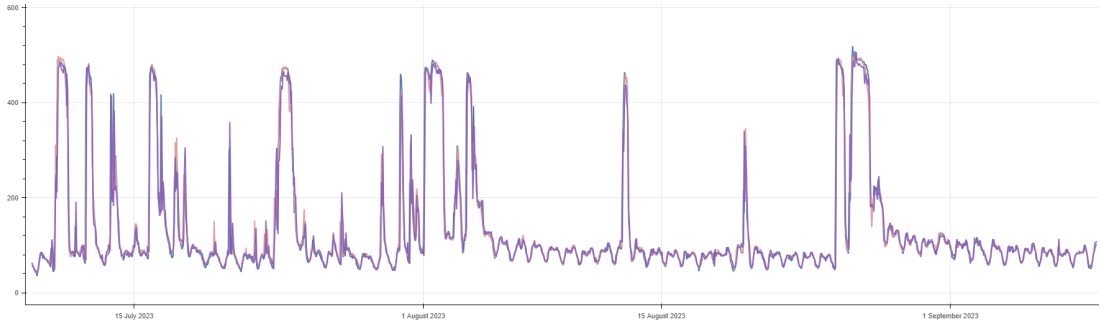


Figura 7.4: Comparativa entre configuraciones seleccionadas de LGBM.

7.4 XGB

El modelo XGB también cuenta con su propia implementación en librerías de Python, en este caso la llamada *xgboost*. Al igual que LGBM en el apartado anterior, sus parámetros óptimos han sido calculados en base a las funciones incluidas en dicha librería, y se recogen en la Tabla 7.4.

Las métricas de las mejores configuraciones del modelo XGB apuntan a una ligera mejoría respecto a su similar LGBM en cuanto a R2, pero no en cuanto a MAE. Además, en esta ocasión la diferencia entre modelos viene dada por el parámetro *learning rate*, ya que son idénticos en cuanto a número de estimadores y profundidad. Esta situación es inversa a lo que sucedía en LGBM, donde es el *learning rate* el punto de acuerdo entre ambos. Finalmente, se muestra en la Figura 7.5 la comparativa entre ambas configuraciones, siendo de nuevo la línea azul el caudal real de Sonderso, en lila el **complejo** y en marrón, el modelo **simple**. En este caso

Parámetro	Modelo simple	Modelo complejo
Estimadores (árboles)	200	200
Profundidad máxima árboles	3	3
Learning rate	0.068	0.079
R2	0.918	0.946
MAE	13.842	11.124

Tabla 7.4: Comparativa de modelos XGB.

se aprecia en múltiples instantes de tiempo cómo el modelo **simple** tiene mayor precisión en los segmentos estacionales de inactividad pluvial, pero el **complejo** (lila), que se equivoca en varios puntos de esta franja estacional, predice con mayor exactitud los puntos altos de los picos de crecimiento del caudal. Dada esta situación, parece más sensato seleccionar el modelo **complejo** entre los propuestos para XGB, pues obtiene mejores métricas que su rival y mayor precisión en los momentos críticos de subida del caudal.

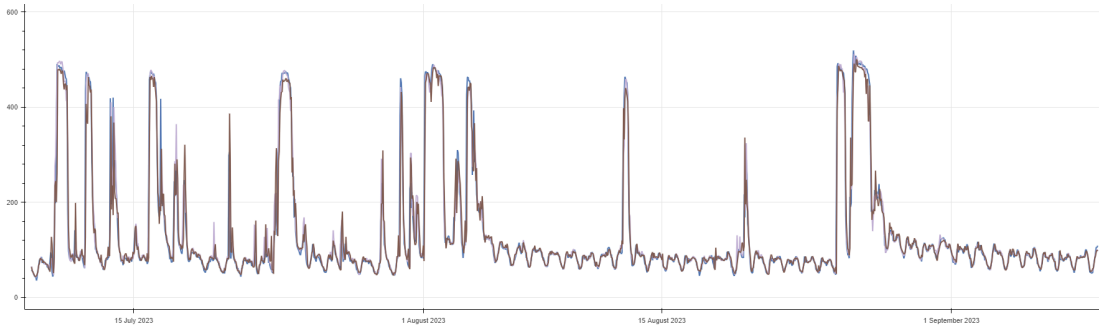


Figura 7.5: Comparativa entre configuraciones seleccionadas de XGB.

7.5 NHITS

Para finalizar el capítulo de modelización, analizaremos los resultados obtenidos con el modelo NHITS. En este caso, se utilizará la implementación de la librería PyTorch, la cual permite el ajuste de parámetros como el *learning rate* de forma automatizada. Este parámetro resulta ser prácticamente idéntico entre ambas configuraciones, las cuales han sido con diferencia las que peores métricas han obtenido de entre todos los modelos, como vemos en la Tabla 7.5.

Parámetro	Modelo simple	Modelo complejo
Tamaño de batch	64	64
Bloques por stack	2	2
Learning rate	0.002	0.001
R2	0.835	0.902
MAE	19.963	15.413

Tabla 7.5: Comparativa de modelos NHITS.

Aunque analizando la métrica no se aprecie una diferencia tan grande con el resto de modelos, es a la hora de visualizar sus predicciones en la Figura 7.6 donde se aprecia que este modelo está cometiendo grandes errores en algunos de los picos de máximo caudal. Parece que NHITS no se esté adecuando lo suficientemente bien a los datos en las zonas de caudal alto, a diferencia de las zonas sin lluvia. De entre las opciones presentadas para NHITS, su versión **compleja** (con las variables *Sonderso*, *Vefps009*, *Soenps005* y *lluvia*) obtiene resultados notablemente mejores que los de su versión **simple**. Es probable que este modelo necesite ser reentrenado con diferentes configuraciones en cuánto al número de bloques y stacks que utiliza.

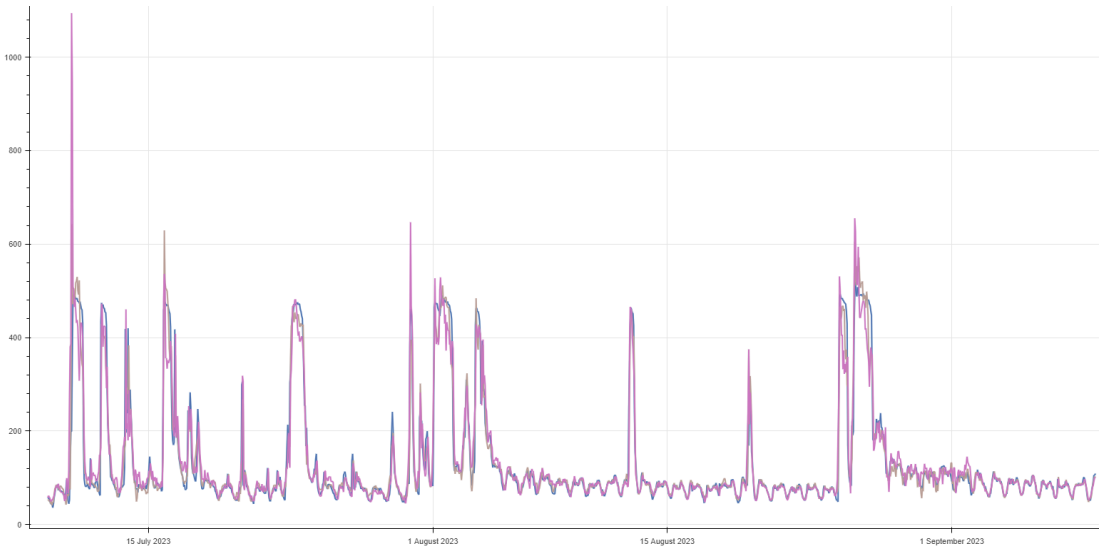


Figura 7.6: Comparativa entre configuraciones seleccionadas de NHITS.

7.6 Comparativa final de modelos

En esta última sección, se recogen de manera breve las diferentes conclusiones y métricas de los modelos presentados a lo largo de todo este capítulo. Por simplicidad, se mantiene la convención de nombrar modelos **complejos** a aquellos entrenados sobre las variables *Sonderso*, *Vefps009*, *Soenps005* y *lluvia*, mientras que denominamos **simples** a los que solo cuentan con *Sonderso* y *lluvia*.

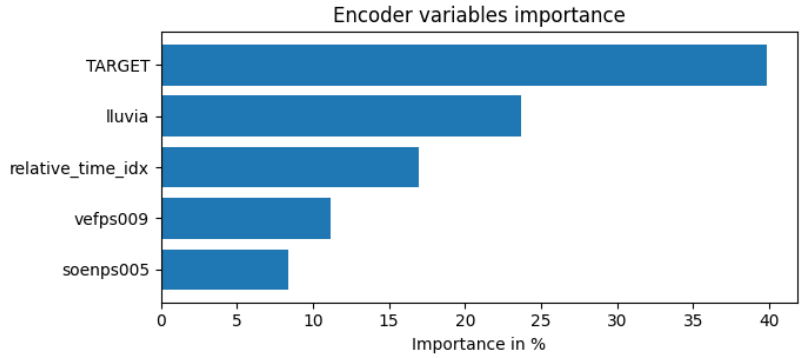
En aspectos generales, como se ve en la Tabla 7.6 y como se ha mostrado en sus respectivas gráficas en las secciones anteriores, los modelos que mayor precisión han logrado han sido los basados en GB, obteniendo los mayores R2 y los menores MAE de entre todos los modelos. Concretamente lo han logrado en su versión **compleja**, es decir, utilizando las cuatro variables principales del conjunto de datos. En contraposición, el modelo NHITS ha obtenido los peores resultados, siendo especialmente inexacto en la predicción de los momentos iniciales de los aumentos de caudal, momento que representa el punto más crítico ante situaciones de crecidas del volumen de agua, cuya sobreestimación podría suponer el vertido excesivo de aguas no tratadas al medio de forma innecesaria.

Modelo	Tipo	R2	MAE
TFT	complejo	0.909	15.338
TFT	simple	0.912	15.732
SARIMAX	complejo	0.934	12.930
SARIMAX	simple	0.911	14.068
LGBM	complejo	0.949	10.806
LGBM	simple	0.916	13.802
XGB	complejo	0.946	11.124
XGB	simple	0.918	13.842
NHITS	complejo	0.902	15.413
NHITS	simple	0.835	19.963

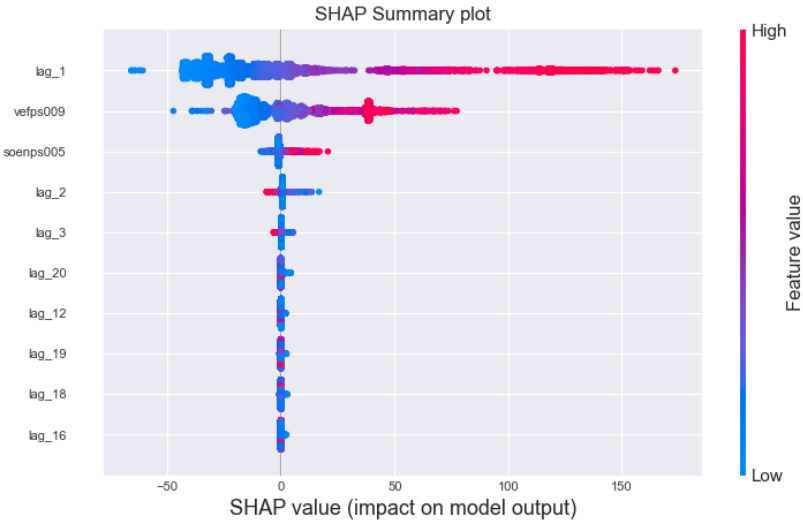
Tabla 7.6: Comparativa de modelos para Sonderso.

En general, la versión **compleja** de cada modelo ha obtenido mejores resultados que su versión **simple**, por lo que cabría pensar que las variables añadidas a la versión compleja (*Vefps009* y *Soenps005*) estarían influyendo la toma de decisiones del modelo de forma posi-

tiva y son relevantes. Puede estudiarse la aportación de cada variable a la salida del modelo utilizando los valores SHAP. En la Figura 7.7 se comparan estos valores para los casos concretos de LGBM (el modelo con mejores métricas) y TFT (único modelo que empeora en su versión compleja frente a la simple).



(a) TFT complejo



(b) LGBM complejo

Figura 7.7: Puntuaciones SHAP para TFT y LGBM.

En el caso de TFT, se observa cómo se asigna una mayor importancia a la variable de lluvia frente a los caudales de *Vefps009* y *Soenps005* (siendo TARGET la variable *Sondesso*), mientras que el modelo LGBM asigna mayores puntuaciones a las variables de caudal *Vefps009* y *Soenps005*, desestimando la lluvia. Es posible que los modelos hayan aprendido que *Vefps009* y *Soenps005* están directamente relacionadas con la actividad meteorológica, por lo que optan por usar las variables de caudal o la lluvia, pero no ambas.

7.7 Algunas predicciones en Moaña

Como se menciona en la Sección 5.3 y otras secciones previas, aunque finalmente este trabajo haya estado centrado en la estación de Sonderso, se analizarán algunos de los resultados obtenidos al aplicar los modelos estudiados a lo largo del proyecto sobre este segundo conjunto de datos.

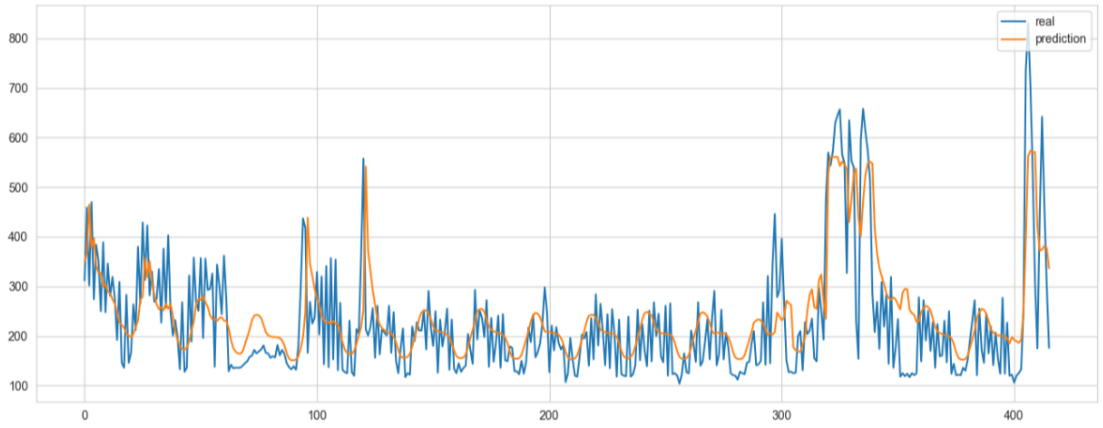
En la Figura 7.8 se muestran las predicciones obtenidas por los principales modelos de AA sobre la serie de tiempo de Moaña para su conjunto de test, realizando predicciones hora a hora y uniéndolas para componer las figuras mostradas. En todos los casos, las variables utilizadas para realizar esta comparativa serán *entrada_sanbarto* a modo de objetivo y *lluvia, nivel_metros* como exógenas. Las métricas correspondientes se recogen en la Tabla 7.7, donde se puede apreciar la gran diferencia entre ellas y las obtenidas en la Tabla 7.6 para Sonderso. Destaca cómo TFT parece no conseguir modelar la variabilidad de la serie y tender a una especie de media móvil de la misma.

Modelo	**R2**	**MAE**
TFT	0.490	63.042
SARIMAX	0.457	66.516
LGBM	0.632	51.271
XGB	0.608	53.455
NHITS	0.526	62.067

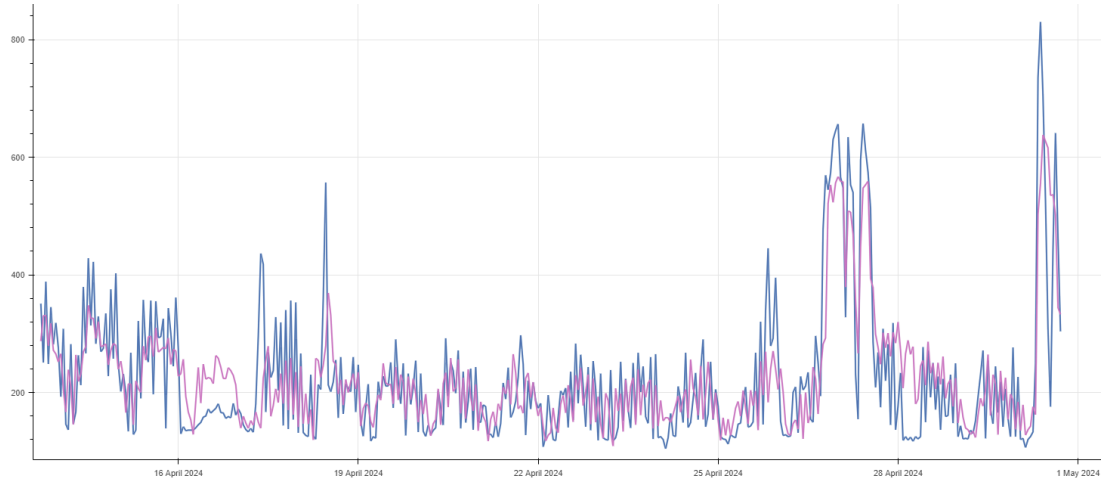
Tabla 7.7: Comparativa de modelos para Moaña.

Además de la distancia entre valores reales y predichos en estos modelos, si se observa en detalle la Figura 7.8 se puede destacar cómo, en muchos de los instantes de tiempo, los modelos parecen sufrir un pequeño retardo, que podría ser consecuencia de la alta variabilidad, la cual no siempre tiene la misma frecuencia de oscilación, y que los modelos podrían estar sobreajustando.

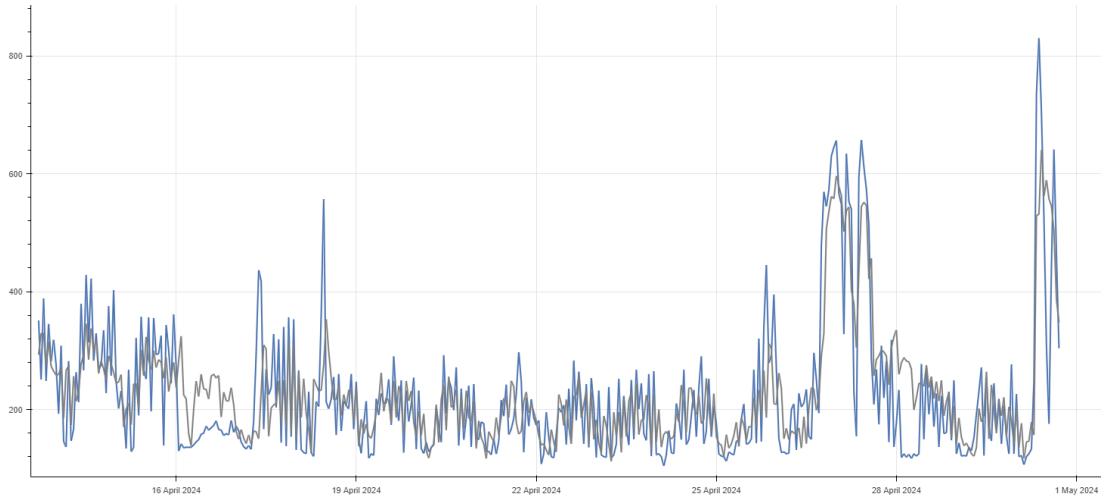
Estas bajas precisiones y los problemas derivados de la extraña disposición de los datos utilizados en la estación de Moaña son un claro ejemplo del Garbage in, Garbage out (GIGO), ya explicado en la Sección 4.4.2, el cual resume las consecuencias que un conjunto de datos de mala calidad supone a la hora de elaborar modelos de Aprendizaje Automático (AA).



(a) Predicciones del modelo TFT (naranja) sobre los datos de Moaña (azul)



(b) Predicciones del modelo LGBM (morado) sobre los datos de Moaña (azul)



(c) Predicciones del modelo XGB (gris) sobre los datos de Moaña (azul)

Figura 7.8: Mejores modelos obtenidos para predicciones en Moaña.

Capítulo 8

Conclusiones

En este último capítulo de la memoria se exponen las conclusiones finales del TFG, tratando de recoger en un breve resumen las decisiones tomadas en base a los distintos resultados obtenidos en las fases anteriores. Además, se relacionarán los contenidos de este trabajo con los campos de estudio de la Ciencia e Ingeniería de Datos, y se detallarán los próximos pasos en el desarrollo de este proyecto y sus posibles ampliaciones a futuro.

8.1 Conclusión final del trabajo

A lo largo de este proyecto se han tratado diferentes aspectos relacionados con el proceso de análisis de datos, detección de anomalías y datos faltantes y, sobre todo, modelización para la obtención de predicciones. En cada uno de estos procesos se ha requerido una amplia investigación y documentación acerca de los métodos, contextualización del trabajo e implementación de las herramientas necesarias.

Este trabajo, dentro del contexto de Life Reseau, proyecto en el que está integrado, significa una pequeña parte del esfuerzo de desarrollo e implementación de las soluciones que se buscan conseguir a largo plazo en el marco de la sostenibilidad y la gestión eficiente de los recursos hídricos. A medida que el tiempo pase y el proyecto avance, considero que pueden alcanzarse muchos de los objetivos marcados, ampliando esta solución a múltiples EDAR en situaciones similares.

En cuanto a los resultados obtenidos, son satisfactorios, pues el objetivo principal de este TFG era el estudio, la implementación, análisis y comparativa de diferentes técnicas basadas en AA capaces de generar predicciones sobre el conjunto de datos aportado. También creo que no son resultados finales, sino un punto de partida hacia un sistema capaz de integrar diversos modelos de AA y de predecir con mayor exactitud los posibles aumentos del caudal que se sucedan.

Finalmente, considero que los resultados que se están obteniendo para la EDAR de Moaña son, de momento, el inicio del proceso que se siguió con la de Sonderso. En este caso, el piloto de Moaña todavía requiere trabajo y colaboración por parte de las empresas que integran el proyecto, teniendo como objetivo a corto plazo la mejoría en los sistemas de medición del caudal, que precedan a un mejor estudio de sus datos y, finalmente, se obtengan resultados suficientemente satisfactorios.

8.2 Aprendizaje realizado y relación con las competencias del grado

A lo largo de la titulación se han adquirido conocimientos en diferentes ámbitos, como recoge la guía oficial del grado [63]. Son varias las temáticas y campos de estudio que se abordan en este proyecto, pero podemos destacar los que se explican a continuación. Para comenzar, se han adquirido amplios conocimientos sobre la programación en Python, sus diferentes herramientas y librerías y sus aplicaciones, en un contexto de desarrollo profesional y de trabajo en equipo. Como cabe esperar de un grado como el Grado en Ciencia e Ingeniería de Datos (GCID) u otras ingenierías, la programación es esencial para el desarrollo de todo tipo de proyectos.

En segunda instancia, se ha conocido, implementado y experimentado distintos algoritmos y modelos de AA, que es otra de las bases fundamentales de este grado. El Aprendizaje Automático (AA) es un campo de estudio en constante evolución y muy latente hoy en día, con gran demanda en el mercado laboral y aplicaciones casi ilimitadas en multitud de proyectos.

Durante las fases de análisis de datos y extracción de información de ellos, se han puesto en práctica conocimientos del área de estadística, la cual explora multitud de técnicas imprescindibles para el correcto desarrollo de este tipo de proyectos, así como algoritmos basados en fundamentos algebraicos y métodos numéricos para comprender el funcionamiento de los modelos y algoritmos utilizados.

Por último, la existencia de múltiples fuentes de datos, su integración y utilización de herramientas de almacenamiento de conjuntos de datos en la nube ponen en valor las competencias adquiridas en la titulación en el campo de las bases de datos y la computación.

En resumen, ese proyecto es multidisciplinar, y abarca la mayoría de campos de estudio de la titulación y bases de la actualidad de la Ciencia y la Ingeniería de Datos.

8.3 Trabajo futuro

En general, el proyecto presenta múltiples oportunidades de mejora, sobre todo en cuanto a la continuación del estudio de los métodos de AA para la obtención de predicciones. Los siguientes pasos en este aspecto serán el estudio y aplicación de nuevos modelos de predicción que puedan conseguir mejores resultados. Algunos de los modelos que formarán parte de los siguientes pasos del proyecto podrían ser LagLlama [64] o DeepAR [65].

Pese a que los resultados obtenidos en el caso de la estación danesa de Sonderso han sido satisfactorios y avanzan acorde a lo planificado, a lo largo de este trabajo hemos visto la problemática que presenta el estudio de la planta de tratamiento de Moaña, en la cual se estará trabajando en colaboración con la empresa que la gestiona para obtener más datos y de mejor calidad, aplicando por lo tanto las técnicas expuestas en esta memoria y obteniendo resultados satisfactorios.

Por otra parte, el proyecto buscará ser integrado en un entorno de visualización que permita al usuario, en este caso la empresas colaboradoras, obtener predicciones en tiempo real de los caudales de sus estaciones, así como consultar todos los conjuntos de datos disponibles desde una plataforma integrada y unificada. En este sentido, podría buscarse escalar el proyecto y ampliar la colaboración con un mayor número de empresas, alcanzando así un marco de trabajo común en el que los distintos países puedan obtener retroalimentación acerca de la gestión de sus EDAR y trabajar por la mejoría en la gestión de los recursos hídricos, su tratamiento eficiente y la reducción de la contaminación de los medios acuáticos y naturales.

Lista de acrónimos

AA Aprendizaje Automático. i, 2, 3, 13–17, 23–26, 29, 31, 50, 56, 65, 67–69, 73

ADF Augmented Dickey-Fuller. 23, 45, 46

API Application Programming Interface. iv, 43, 46

ARIMA Autoregressive Integrated Moving Averager. ii, 16, 17, 29, 30

DBSCAN Density-Based Spatial Clustering of Applications with Noise. iv, 16, 36, 51–53, 55

DMI Danish Meteorological Institute. iv, 12, 43, 46

DNN Dense Neural Network. 16

EAPI Extracción, análisis, procesado e integración de datos. 7, 8, 10

EBAR Estación de Bombeo de Aguas Residuales. 18, 20, 21, 40, 41, 49

EBARs Estaciones de Bombeo de Aguas Residuales. 18

EDAR Estación Depuradora de Aguas Residuales. i, 2, 3, 6, 7, 12, 17–20, 22, 39, 41, 49, 67–69

EDARs Estaciones Depuradoras de Aguas Residuales. 1

GB Gradient Boosting. 17, 32, 63

GCID Grado en Ciencia e Ingeniería de Datos. 68

GIGO Garbage in, Garbage out. 26, 65

GRN Gated Residual Network. 27

h.e. Habitante Equivalente. 19, 72

IA Inteligencia Artificial. 23

ITG Instituto Tecnológico de Galicia. 1, 5, 14

KPSS Kwiatkowski Phillips Schmidt and Shin. 23, 45, 46

LGBM Light Gradient Boosting Machine. ii, iv, v, 12, 33, 34, 59, 60, 64

LOCF Last Observation Carried Forward. 38, 40

LSTM Long Short-Term Memory. 27

MAE Mean Absolute Error. ii, 35, 57, 58, 60–63

MSE Mean Squared Error. ii, 35

NBEATS Neural Basis Expansion Analysis for Time Series Forecasting. 28, 29

NHITS Neural Hierarchical Interpolation for Time Series. ii–v, 17, 28, 29, 61–63

NNI Neural Network Intelligence. i, iii, 13, 14, 25, 56

NOCB Next Observation Carried Backward. 38

SARIMAX Seasonal Autoregressive Integrated Moving Averager using eXogenous regressors. ii, iv, v, 16, 30, 58, 59

shap SHapley Additive exPlanations. 73

STL Seasonal-Trend decomposition using LOESS. iv, 16, 37, 52, 54, 55

TFG Trabajo de Fin de Grado. i, 1–4, 7, 8, 15, 49, 67

TFT Temporal Fusion Transformer. ii–v, 14, 16, 17, 26–28, 35, 56–59, 64, 65

UE Unión Europea. 1, 15

XGB Extreme Gradient Boosting. ii, iv, v, 12, 33, 34, 60, 61

Glosario

AIC (Criterio de Información de AKAIKE) El criterio de información de Akaike (AIC) es un estimador de la calidad relativa del modelo que tiene en cuenta su complejidad. Este criterio de información penaliza los modelos complejos en favor del los sencillos para evitar el sobreajuste. 31

cuantiles Los cuantiles son puntos tomados a intervalos regulares de la función de distribución de una variable aleatoria. 26, 35

Dataframe Pandas DataFrame es una estructura de datos tabulares bidimensional, potencialmente heterogénea y de tamaño variable, con ejes etiquetados (filas y columnas). 39

Diagrama de Gantt Herramienta de gestión de proyectos que ilustra el trabajo realizado durante un período de tiempo en relación con el tiempo previsto para el trabajo. 8

Ensemble Un ensemble de árboles de regresión es un modelo predictivo compuesto por una combinación ponderada de varios árboles de regresión. En general, la combinación de varios árboles de regresión aumenta la capacidad predictiva [66]. 32

framework Conjunto de herramientas, guías y estructuras predefinidas que se utilizan para desarrollar y organizar software de manera eficiente. Ofrece un conjunto de prácticas estandarizadas y componentes reutilizables que aceleran el proceso de desarrollo [67]. 12, 14

Habitante equivalente El habitante equivalente es una unidad de población equivalente que corresponde a la carga contaminante media de las aguas residuales, establecida en 60 g de materia orgánica por habitante y día [68]. Se conoce como h.e.. 20

MaxPooling El Max Pooling o reducción de resolución por máximos es una operación de reducción de dimensionalidad que calcula el valor máximo en una ventana determinada de tiempo cuyo tamaño viene determinado por el tamaño del *kernel* elegido. 29

SHAP Los valores SHapley Additive exPlanations (shap) son una forma de explicar la salida de cualquier modelo de AA. Utiliza un enfoque de teoría de juegos que mide la contribución de cada jugador al resultado final. A cada característica se le asigna un valor de importancia que representa su contribución al resultado del modelo, por lo que shap muestra cómo afecta cada rasgo a cada predicción final, la importancia de cada rasgo en comparación con los demás y la dependencia del modelo de la interacción entre rasgos. 64

Silhouette El coeficiente de Silueta o simplemente *Silhouette* es una métrica para evaluar la calidad del agrupamiento obtenido con algoritmos de *clustering*. El valor de la silueta es una medida de la similitud de un objeto a su propio cúmulo (cohesión) en comparación con otros cúmulos (separación). Este coeficiente toma valores entre -1 y +1, siendo el valor positivo la máxima cohesión. 52

Bibliografía

[1] "¿Qué es SCRUM? Conoce el framework que agiliza el trabajo en equipo," 2022. [En línea]. Disponible en: https://www.escueladenegociosydireccion.com/revista/business/scrum-framework-agiliza-trabajo-equipo/

[2] A. E. de Abastecimientos de Agua y Saneamiento (AEAS), "Informe sobre aguas residuales en España," 2017. [En línea]. Disponible en: https://www.aeas.es/images/publicaciones/informacion-sector/2017_-_Informe_depuracin_AEAS_Da_mundial_del_agua_2017.pdf

[3] V. Morales-Oñate, "Series de tiempo," 2023. [En línea]. Disponible en: https://bookdown.org/victor_morales/SeriesdeTiempo/an%C3%A1lisis-descriptivo-de-una-serie-temporal.html

[4] M. Labiadh, "Understanding Temporal Fusion Transformer," 2023. [En línea]. Disponible en: https://medium.com/dataness-ai/understanding-temporal-fusion-transformer-9a7a4fcde74b

[5] J. Dancker, "N-HiTS — Making Deep Learning for Time Series Forecasting More Efficient," 2024. [En línea]. Disponible en: https://towardsdatascience.com/n-hits-making-deep-learning-for-time-series-forecasting-more-efficient-d00956fc3e93

[6] M. Peixeiro, "All About N-HiTS: The Latest Breakthrough in Time Series Forecasting," 2022. [En línea]. Disponible en: https://towardsdatascience.com/all-about-n-hits-the-latest-breakthrough-in-time-series-forecasting-a8ddcb27b0d5

[7] J. A. Rodrigo and J. E. Ortiz, "Modelos ARIMA y SARIMAX con python," 2024. [En línea]. Disponible en: https://cienciadedatos.net/documentos/py51-arima-sarimax-models-python

[8] M. Soni, "Árboles de decisión, random forest, gradient boosting y C5.0," 2020. [En línea]. Disponible en: https://maniksonituts.medium.com/what-is-decision-tree-regression-dcd0ea40a323

[9] "XGBOOST vs LightGBM," 2024. [En línea]. Disponible en: https://neptune.ai/blog/xgboost-vs-lightgbm

[10] N. S. Chauhan, "DBSCAN clustering algorithm in machine learning," 2023. [En línea]. Disponible en: https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

[11] "Pandas resample tricks you should know for manipulating time-series data," 2020. [En línea]. Disponible en: https://towardsdatascience.com/pandas-resample-tricks-you-should-know-for-manipulating-time-series-data-7e9643a7e7f3

[12] "Instituto Tecnológico de Galicia," 2024. [En línea]. Disponible en: https://itg.es/centro-tecnologico/

[13] "El proyecto Life Reseau," 2024. [En línea]. Disponible en: https://life-reseau.eu/es/el-proyecto/

[14] "Programa LIFE," 2024. [En línea]. Disponible en: https://www.miteco.gob.es/es/ministerio/servicios/ayudas-subvenciones/programa-life.html

[15] "Scrum: qué es y por qué es una de las metodologías ágiles," 2024. [En línea]. Disponible en: https://ausum.cloud/scrum-metodologia-agil-mas-popular-en-empresas/

[16] E. Meardon, "¿Qué son los diagramas de Gantt?" 2024. [En línea]. Disponible en: https://www.atlassian.com/es/agile/project-management/gantt-chart

[17] G. Sobregau, "¿Cuál es el salario de data scientist, ingeniero de datos y big data?" 2020. [En línea]. Disponible en: https://nuclio.school/blog/salario-de-un-data-scientist-un-ingeniero-de-datos-y-un-analista-de-big-data/#Salario-de-un-cientifico-de-datos

[18] A. M. Loreto, "¿Cuánto gana un Data Scientist en España en 2024?" 2024. [En línea]. Disponible en: https://www.hackaboss.com/blog/cuanto-gana-data-scientist-salario-espana

[19] M. Buenning, "¿Cuánto tiempo duran los portátiles?" 2024. [En línea]. Disponible en: https://www.ninjaone.com/es/blog/cuanto-tiempo-duran-los-ordenadores-portatiles-para-las-organizaciones/

[20] "Python." [En línea]. Disponible en: https://es.wikipedia.org/wiki/Python

[21] "Guido Van Rossum." [En línea]. Disponible en: https://es.wikipedia.org/wiki/Guido_van_Rossum

[22] "TIOBE Index for June 2024." [En línea]. Disponible en: https://www.tiobe.com/tiobe-index/

[23] "Pandas user guide." [En línea]. Disponible en: https://pandas.pydata.org/docs/user_guide/index.html

[24] "Numpy user guide." [En línea]. Disponible en: https://numpy.org/doc/stable/user/index.html#user

[25] "DMI Open Data API." [En línea]. Disponible en: https://github.com/LasseRegin/dmi-open-data

[26] "math - Mathematical functions." [En línea]. Disponible en: https://docs.python.org/es/3/library/math.html

[27] "PyTorch Documentation." [En línea]. Disponible en: https://pytorch.org/docs/stable/index.html

[28] "XGboost Documentation." [En línea]. Disponible en: https://xgboost.readthedocs.io/en/stable/

[29] "Welcome to LightGBM's documentation!" [En línea]. Disponible en: https://lightgbm.readthedocs.io/en/stable/

[30] "Welcome to skforecast - Skforecast Docs." [En línea]. Disponible en: https://skforecast.org/0.12.1/index.html

[31] "Documentation scikit-learn: machine learning in Python." [En línea]. Disponible en: https://scikit-learn.org/0.21/documentation.html

[32] "Matplotlib documentation." [En línea]. Disponible en: https://matplotlib.org/stable/index.html

[33] "Pandas-bokeh." [En línea]. Disponible en: https://patrikhlobil.github.io/Pandas-Bokeh/

[34] "Seaborn." [En línea]. Disponible en: https://seaborn.pydata.org/

[35] "MinIO: S3 and Kubernetes Native Object Storage for AI." [En línea]. Disponible en: https://min.io/

[36] "NNI Documentation." [En línea]. Disponible en: https://nni.readthedocs.io/en/stable/

[37] "Neural Network Intelligence." [En línea]. Disponible en: https://en.wikipedia.org/wiki/Neural_Network_Intelligence

[38] G. Cloud, "¿Qué es una máquina virtual?" [En línea]. Disponible en: https://cloud.google.com/learn/what-is-a-virtual-machine?hl=es

[39] IBM, "¿Qué son las Máquinas Virtuales (MV)." [En línea]. Disponible en: https://www.ibm.com/es-es/topics/virtual-machines#:~:text=Una%20m%C3%A1quina%20virtual%20es%20una%20representaci%C3%B3n%20virtual%20o%20emulaci%C3%B3n%20de,como%20%22host%22%20o%20anfitri%C3%B3n.

[40] "Introducción a las series temporales." [En línea]. Disponible en: https://www.ibm.com/docs/es/spss-statistics/saas?topic=forecasting-introduction-time-series

[41] "Tipos de análisis exploratorio de datos." [En línea]. Disponible en: https://www.ibm.com/es-es/topics/exploratory-data-analysis#Herramientas+de+an%C3%A1lisis+de+datos+exploratorios

[42] "Implement DBSCAN Clustering in Python with a Real-World Data," 2023. [En línea]. Disponible en: https://blog.deepsim.xyz/dbscan-clustering-python-real-world-data/

[43] "Data Imputation Demystified | Time Series Data." [En línea]. Disponible en: https://medium.com/@aaabulkhair/data-imputation-demystified-time-series-data-69bc9c798cb7

[44] B. Lim, S. O. Arık, N. Loeff, and T. Pfister, "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/S0169207021000637

[45] S. K. Pantelis Linardatos, Vasilis Papastefanopoulos. Theodor Panagiotakopoulos, "CO2 concentration forecasting in smart cities using a hybrid ARIMA–TFT model on multivariate time series IoT data," 2023. [En línea]. Disponible en: https://cienciadedatos.net/documentos/py09_gradient_boosting_python

[46] J. A. Rodrigo, "Gradient Boosting con Python." [En línea]. Disponible en: https://cienciadedatos.net/documentos/py09_gradient_boosting_python

[47] iAgua, "EDAR en españa." [En línea]. Disponible en: https://www.iagua.es/data/infraestructuras/estaciones-depuradoras-aguas-residuales-espana

[48] "Serie Temporal." [En línea]. Disponible en: https://es.wikipedia.org/wiki/Serie_temporal

[49] "Introducción a las series temporales." [En línea]. Disponible en: https://www.ibm.com/docs/es/spss-statistics/saas?topic=forecasting-introduction-time-series

[50] A. Sable, "Introduction to Time Series Analysis." [En línea]. Disponible en: https://blog.paperspace.com/introduction-time-series-analysis/

[51] "Componentes de una serie temporal." [En línea]. Disponible en: https://www5.uva.es/estadmed/datos/series/series1.htm#:~:text=Se%20denomina%20tendencia%20de%20una,representar%20la%20tendencia%20(creciente)

[52] IBM, "¿Qué es el Machine Learning (ML)?" [En línea]. Disponible en: https://www.ibm.com/es-es/topics/machine-learning

[53] "¿En qué consiste el ajuste de hiperparámetros?" [En línea]. Disponible en: https://aws.amazon.com/es/what-is/hyperparameter-tuning/

[54] S. Navarro, "Optimización de hiperparámetros en Deep Learning," *KeepCoding.io*, 2024. [En línea]. Disponible en: https://keepcoding.io/blog/optimizacion-hiperparametros-deep-learning/

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[56] C. Challu, K. G. Olivares, B. N. Oreshkin, F. Garza, M. Mergenthaler-Canseco, and A. Dubrawski, "N-HiTS: Neural Hierarchical Interpolation for Time Series Forecasting," 2022.

[57] U. N. A. de México, "Modelación ARIMA," pp. 2–3. [En línea]. Disponible en: http://www.ptolomeo.unam.mx:8080/jspui/bitstream/132.248.52.100/363/7/A7.pdf

[58] J. A. Rodrigo, "Árboles de decisión, random forest, gradient boosting y C5.0." [En línea]. Disponible en: https://www.cienciadedatos.net/documentos/33_arboles_decision_random_forest_gradient_boosting_C50.html

[59] N. A. L. Cosio, "Métricas en regresión," 2021. [En línea]. Disponible en: https://medium.com/@nicolasarrioja/m%C3%A9tricas-en-regresi%C3%B3n-5e5d4259430b

[60] V. Efimov, "Quantile Loss and Quantile Regression," 2023. [En línea]. Disponible en: https://towardsdatascience.com/quantile-loss-and-quantile-regression-b0689c13f54d

[61] A. Abulkhair, "Data Imputation Demystified | Time Series Data," 2023. [En línea]. Disponible en: https://medium.com/@aaabulkhair/data-imputation-demystified-time-series-data-69bc9c798cb7

[62] "Merge, join, concatenate and compare in pandas." [En línea]. Disponible en: https://pandas.pydata.org/docs/user_guide/merging.html

[63] "Grado en ciencia e ingeniería de datos." [En línea]. Disponible en: https://estudos.udc.es/es/study/detail/614g02v01

[64] A. Ashok, "Lag-Llama: Towards Foundation Models for Probabilistic Time Series Forecasting." [En línea]. Disponible en: https://time-series-foundation-models.github.io/lag-llama.pdf

[65] D. Salinas, V. Flunkert, and J. Gasthaus, "DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks," 2019.

[66] "Ensembles de árboles de regresión." [En línea]. Disponible en: https://es.mathworks.com/help/stats/regression-tree-ensembles.html

[67] Susana Meijomil, "Qué es el framework: Definición, para qué sirve y ejemplos," *inboundcycle*, 2024. [En línea]. Disponible en: https://www.inboundcycle.com/diccionario-marketing-online/framework

[68] R. V.-S. et al., "Estaciones Depuradoras de Aguas Residuales (EDAR)," p. 6, 2021. [En línea]. Disponible en: https://informemarbalear.org/wp-content/uploads/2021/03/imb-depuradores-esp.pdf