Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN CIENCIA E ENXEÑARÍA DE DATOS

Procesamiento y análisis de datos meteorológicos de la isla de Tenerife

Estudante: Mateo Sabuz García

Dirección: Guillermo de Bernardo Roca

Marcos Félix Rodríguez Silva

A Coruña, febrero de 2024.

Se lo dedico a mis padres, a Mauro, a Celia, a mi abuela y a Roger

Agradecimientos

En primer lugar quería darle las gracias a mis tutores por ayudarme con este tramo final de mis estudios de grado. Gracias a Marcos por permitirme hacer este proyecto, dejándome compaginar de una mejor forma este trabajo con mis tareas dentro de la empresa. Gracias a Guillermo por la paciencia y por aceptar este segundo proyecto después de que el anterior no se pudiera realizar por motivos ajenos a nosotros.

Gracias mamá, papá y Mauro por todo el apoyo continúo que recibí durante toda esta etapa como alumno, haciendo muchos esfuerzos para que pudiera vivir esta etapa de la mejor forma. Gracias también a Celia por apoyarme siempre de la mejor forma desde la distancia, aunque nos separen muchos kilómetros es como si te tuviera a mi lado.

Por último pero no menos importante gracias a todos los amigos que conocí y me acompañaron durante toda esta etapa. Gracias a Prieto, Chou, Brais, Dei y muchos más, con vosotros compartí momentos que nunca olvidaré. Empecé esta carrera sin conocer apenas a nadie y la termino con gente que considero muy buenos amigos, gracias a vosotros esta etapa se me hizo mucho más llevadera.

Resumen

El objetivo de este proyecto de fin de grado es el desarrollo de una solución para el procesamiento y visualización de datos meteorológicos procedentes de bases de datos internas del Cabildo de Tenerife, los cuales son distintas métricas recogidas por una gran cantidad de sensores de distintas estaciones distribuidas por toda la isla.

Este trabajo se ha dividido en diferentes fases para conseguir este objetivo. La primera de ellas es una fase introductoria donde se trata tanto la motivación como los objetivos principales que se quieren lograr, seguida de una explicación de las distintas tecnologías empleadas en el desarrollo del proyecto. A continuación se analizaron los datos de origen y las necesidades de explotación de los mismos, permitiendo continuar con las siguientes fases de diseño e implementación de la solución. Por último se muestra esta solución acompañada de unas conclusiones e ideas de trabajo futuro.

Para la realización del proyecto se utilizó por un lado Apache Airflow para el procesamiento de los datos de origen, la cual es una herramienta totalmente programada con el lenguaje Python. Este mismo lenguaje también se utilizó tanto para el análisis exploratorio como para el desarrollo de los modelos de predicción de los datos una vez procesados. La parte de visualización de los datos se llevó a cabo con Power BI, implementando distintos *dashboards* interactivos.

En este proyecto se utilizó una metodología basada en sprints, con un seguimiento continuo por parte de los tutores elaborando distintas entregas semanales.

Abstract

The objective of this end-of-degree final project is to develop a solution for the processing and visualization of meteorological data from internal databases of Cabildo de Tenerife. These databases contain various metrics collected by a large number of sensors from different stations distributed throughout the island.

This work has been divided into different phases to achieve this objective. The first phase is an introductory stage where both motivation and the main objectives to be achieved are addressed, followed by an explanation of the various technologies used in the project's development. Next, the source data and their exploitation needs were analyzed, allowing for the continuation of the subsequent phases of solution design and implementation. Finally, the solution is presented along with conclusions and ideas for future work.

In the project, Apache Airflow was used for processing the source data, a tool entirely programmed in Python language. This same language was also employed for both exploratory

analysis and the development of prediction models for the processed data. The data visualization part was carried out using Power BI, implementing various interactive dashboards.

The project employed a sprint-based methodology, with continuous monitoring by directors and the submission of different weekly deliverables.

Palabras clave:

- ETL
- Visualización
- Análisis de datos
- Predicción
- Series temporales
- Redes neuronales

Keywords:

- ETL
- Visualization
- Data Analysis
- Prediction
- Time series
- Neural networks

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

En esta sección presenta tanto la motivación del trabajo, como los objetivos y estructura del mismo.

1.1 Motivación

Tenerife [1] es, a día de hoy, uno de los destinos favoritos de muchos turistas de todo el mundo. Desde sus playas paradisiacas hasta sus paisajes de montaña y vegetación, hacen que sea el destino idóneo para diferentes tipos de turistas, desde aquellos que quieren pasar unos días de tranquilidad hasta un tipo de público mucho más aventurero. Aunque el turismo representa la mayor parte del PIB de la Tenerife, no se puede despreciar la industria agrícola de la isla, ya que gracias a la diversidad climática del territorio existe una industria muy variada.

El estudio climático es muy importante para ambas industrias ya que dependen en gran parte de él. En la actualidad existen un gran número de páginas de previsión meteorológica, que comúnmente ofrecen la predicción de las métricas más comunes como pueden ser la temperatura, precipitación y velocidad del viento de la próxima semana o a veces de los 14 próximos días.

La mayoría de las herramientas de análisis climático más utilizadas actualmente están principalmente enfocadas a la predicción meteorológica. Por otra parte, existen otras herramientas o proyectos enfocados en la publicación de datos masivos de interés que se suelen conocer como "datos abiertos". Vista esta situación, es de interés poder contar con una herramienta que contemple tanto la representación de los datos históricos para ayudar a identificar tendencias o patrones en el clima, y a su vez contar con estimaciones futuras, consiguiendo así una herramienta completa que permita abarcar distintos tipos de análisis.

Tenerife cuenta con una gran de estaciones meteorológicas donde sus distintos sensores recogen métricas todos los días como la temperatura, precipitación, velocidad del viento, radiación solar y humedad relativa. El objetivo principal de este proyecto es integrar todas

estas medidas de los sensores en varios *dashboards* interactivos para poder tanto visualizar la evolución histórica de las mediciones como también contar con una previsión futura de las más importantes.

1.2 Objetivos

El desarrollo de este trabajo se puede dividir principalmente tres bloques de trabajo principales. El primer bloque se centra en un análisis en detalle de la fuente de datos, que en este caso se trata de una una base de datos, donde se almacenan todos los registros de los sensores. En este primer bloque también se platean las necesidades de explotación de los datos una vez analizados en la fuente de origen. El siguiente bloque se centra en el procesamiento de los mismos, donde mediante un proceso ETL se aplicarán distintas transformaciones en los datos para conseguir un formato mas amigable para el análisis posterior. El tercer y último bloque se enfoca tanto la creación de visualizaciones como en la exploración de técnicas de predicción de algunas de las métricas con las que nutrir la herramienta de análisis.

En lo que respecta a todas estas fases del trabajo mencionadas, se plantean los siguientes objetivos concretos:

- Analizar la base de datos de origen y definir las posibilidades de explotación de la misma. Para ello se diseñarán procesos para extraer la información relevante de la fuente de datos. Una vez se sabe que datos de valor se pueden extraer de la base de datos, se deben definir las necesidades de explotación de dichos datos.

- Implementar un proceso ETL para la transformación de los datos de origen. Ya que los a datos de origen pueden estar en un formato no adecuado para el análisis, es necesario transformarlos para posteriormente poder explotarlos. Para realizar este proceso ETL se utilizó la herramienta Apache Airflow.

- Desarrollar una herramienta constituida por distintos *dashboards* que permitan visualizar de manera intuitiva la evolución de las distintas métricas de interés. Estas visualizaciones permitirán tanto identificar tendencias o patrones a lo largo del tiempo como la comparación de distintas zonas geográficas.

- Explorar y comparar distintos métodos de predicción para la estimación de la temperatura y precipitaciones futuras, posteriormente integrando estas predicciones en el *dashboard* principal.

1.3 Estructura de la memoria

A lo largo de este documento se describe en detalle el desarrollo de este proyecto. Esta memoria explicativa se organiza en los siguientes capítulos:

- **Capítulo 1**: Introducción. Este primer capítulo introductorio describe como será este proyecto. En este capítulo encontramos la motivación, los objetivos del proyecto y la estructura de esta memoria.
- **Capítulo 2**: Fundamentos tecnológicos.En este capítulo se explican a nivel teórico las distintas tecnologías y herramientas empleadas en este proyecto. En este apartado además se muestran lineas de trabajo actuales en lo que respecta al análisis de datos climáticos.
- **Capítulo 3**: Metodología y planificación. En esta sección se explican las metodologías de seguimiento que se utilizaron para el desarrollo exitoso de este proyecto.
- **Capítulo 4**: Análisis. Aquí se presenta el análisis de los datos de origen y se detallan las posibilidades de explotación de los mismos.
- **Capítulo 5**: Diseño y implementación. En este apartado se explica el diseño y la implementación de los distintos componentes que forma parte de la solución final desarrollada.
- **Capítulo 6**: Conclusiones. En este punto se muestra la herramienta resultante y se presentan las conclusiones principales del proyecto y posibles lineas de continuación del mismo.

Capítulo 2

Fundamentos tecnológicos

En este capítulo se presenta el trabajo relacionado existente en el ámbito de este trabajo y se explican de forma teórica las distintas tecnologías utilizadas en este proyecto.

2.1 Trabajo relacionado

El principal objetivo de esta sección es la búsqueda de herramientas o proyectos desarrollados en el ámbito del análisis meteorológico, con el fin de analizar las distintas soluciones disponibles implementadas en trabajos previos que puedan ser de utilidad para el desarrollo de este.

En concreto, en este apartado se muestran distintas herramientas existentes actualmente, entre las cuales algunas están centradas en la parte predictiva, y otras ofrecen la posibilidad de descargar una importante cantidad de datos históricos, pero no permiten hacer ningún tipo de análisis de estos datos.

Cabe destacar que estas páginas de predicción meteorológica cuentan con datos en tiempo real recogidos por satélite. Esto es importante debido a que permite hacer predicciones mucho mas precisas, ya que además del registro histórico de mediciones, cuentan con la información de las borrascas y anticiclones que se forman en la tierra, permitiendo estimar sus direcciones y así afinar mucho más las predicciones.

Agencia Estatal de Meteorología (AEMET)

Una de las herramientas más utilizadas por el público general para consultar datos climáticos es la página web de AEMET. La herramienta se centra principalmente en la predicción futura, ofreciendo una previsión por horas del día en cuestión y una predicción de algunas medidas de los 7 días posteriores. Estas predicciones se realizan con un sistema desarrollado llamado HIRLAM, que utiliza los datos recogidos por satélite. El sistema HIRLAM [2] proporciona pronósticos meteorológicos de alta resolución en áreas geográficas limitadas. El modelo

utiliza ecuaciones matemáticas para simular la atmósfera y otros componentes del sistema climático, ayudando así a prever el tiempo en diferentes ubicaciones y momentos. En cuanto a análisis de datos históricos no cuenta con una gran cantidad de información. Al igual que en la predicción del tiempo, cuentan con la información histórica de los últimos 7 días.

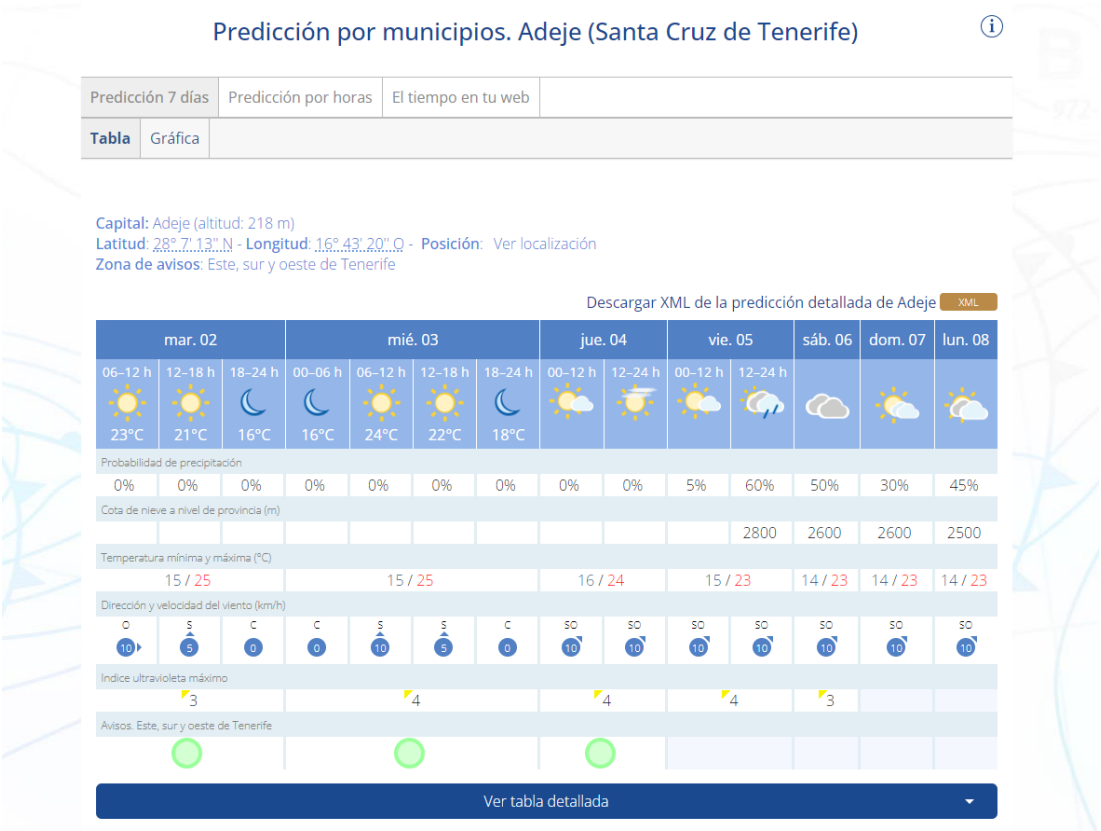


Figura 2.1: Predicción a 7 días del municipio de Adeje

eltiempo.es

Eltiempo.es [3] es el soporte digital de información meteorológica líder en España. Es una web española que principalmente se utiliza para consultar la previsión climática de España pero a su vez permite consultar prácticamente todos los lugares del mundo. A diferencia de AEMET, esta herramienta no publica información sobre el modelo climático desarrollado. Cuentan con un rango mayor de predicción, aumentando los 7 días de AEMET a 14. Esta herramienta no cuenta con un registro histórico de los datos, pero estos los utiliza para mostrar

la media de las medidas en cada mes del año.

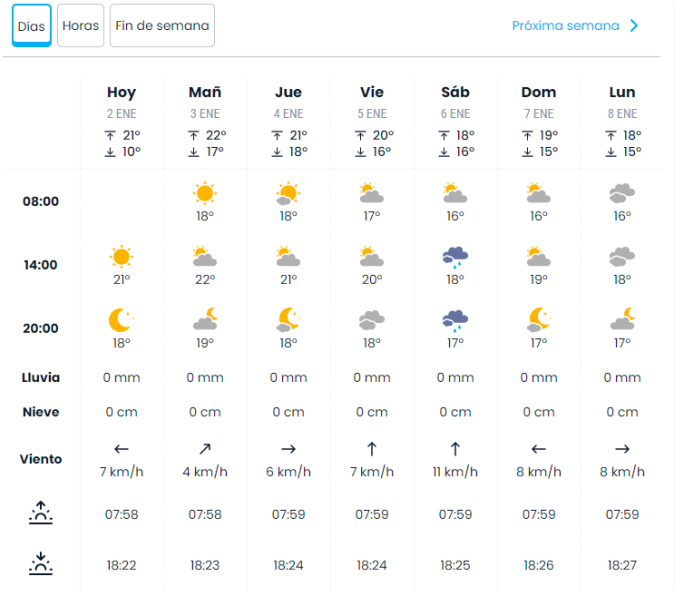
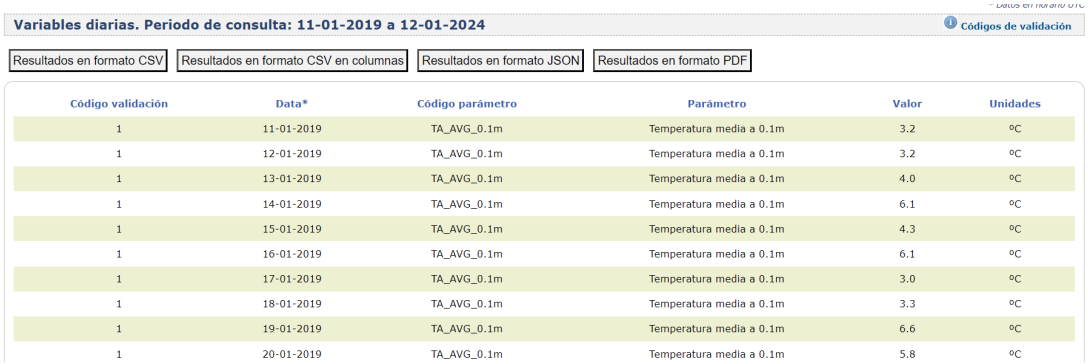


Figura 2.2: Predicción del municipio de Adeje

MeteoGalicia

MeteoGalicia [4] es la agencia meteorológica regional para Galicia. Herramientas como esta cuentan con una sección de datos abiertos donde se puede consultar hasta un rango de 5 años de históricos de datos de distintas medidas registradas. Estos datos se pueden descargar en distintos formatos pero no existe ninguna sección que contenta una herramienta de análisis de los mismos.



Variables diarias. Periodo de consulta: 11-01-2019 a 12-01-2024 — Códigos de validación

Resultados en formato CSV | Resultados en formato CSV en columnas | Resultados en formato JSON | Resultados en formato PDF

Código validación	Data*	Código parámetro	Parámetro	Valor	Unidades
1	11-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	3.2	ºC
1	12-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	3.2	ºC
1	13-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	4.0	ºC
1	14-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	6.1	ºC
1	15-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	4.3	ºC
1	16-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	6.1	ºC
1	17-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	3.0	ºC
1	18-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	3.3	ºC
1	19-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	6.6	ºC
1	20-01-2019	TA_AVG_0.1m	Temperatura media a 0.1m	5.8	ºC

Figura 2.3: Consulta de la temperatura media desde 2019

2.2 Tecnologías y herramientas empleadas

En esta sección se explican las tecnologías y herramientas utilizadas en este proyecto. En primer lugar se explican los componentes principales para el desarrollo del mismo, seguido de una explicación de ciertas herramientas auxiliares empleadas.

2.2.1 Tecnologías y herramientas principales

Apache Airflow

Apache Airflow [5] es una plataforma de código abierto de orquestación de flujos de trabajo desarrollada en Python. Permite la automatización y programación de tareas complejas, facilitando la gestión de procesos ETL (Extract, Transform, Load) y la programación de cronogramas de ejecución. Airflow utiliza un enfoque basado en código para definir flujos de trabajo, lo que lo hace flexible y escalable. Su interfaz web proporciona una visión clara y monitoreo en tiempo real de las tareas ejecutadas. Para gestionar la orquestación de flujos de trabajo, Airflow utiliza grafos acíclicos dirigidos.

Un DAG [6] (Directed Acyclic Graph o Grafo Acíclico Dirigido) es el elemento central en Apache Airflow. Representa visualmente un flujo de trabajo, donde los nodos del grafo son tareas y las aristas definen el orden de ejecución. Cada tarea en un DAG es una unidad de trabajo individual y puede ser una operación ETL, un script de Python, o cualquier tarea que se deba ejecutar.

Como se puede observar en la Figura 2.4, un proceso ETL se puede dividir en tres fases principales [7] :

- Extracción: Esta primera etapa consiste en la extracción de los datos necesarios de los orígenes de datos de interés. En lo que respecta a este proyecto, la fase de extracción

consiste en conectarse a la base de datos y extraer los datos de interés con consultas SQL.

- Transformación: Una vez se extraen los datos de interés puede ser necesario realizar una limpieza, correcciones o cambios estructurales en los mismos. En esta fase se aplican todas las transformaciones necesarias a los datos de origen para obtener una versión analizable de los mismos. En caso de tener que juntar en un mismo recurso los datos diversas fuentes se realizarían también en esta fase.

- Carga: Por último los datos procesados se cargan en un destino final para ser utilizados. En este proyecto se utilizan herramientas que aceptan todo tipo de fuentes de datos como pueden ser PowerBI o Python con sus librerías, por lo que se realizará un proceso que integre toda la información relevante sin necesidad de cargarlo en una fuente de datos.

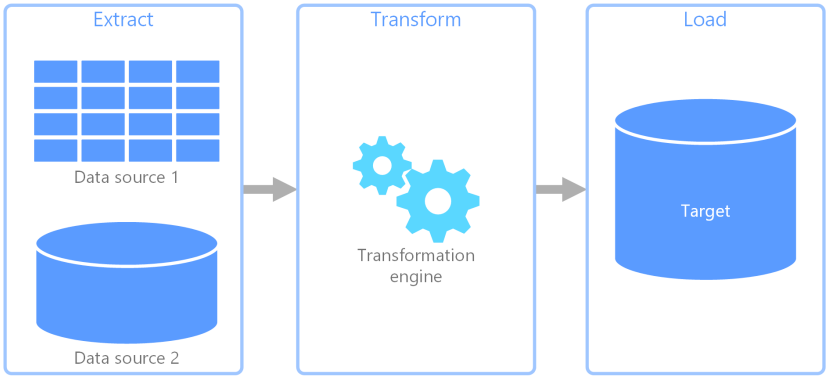


Figura 2.4: Proceso ETL

Power BI

Power BI [8] es un conjunto de servicios de software, aplicaciones y conectores que trabajan juntos para convertir sus fuentes de datos no relacionadas en perspectivas coherentes, visualmente inmersivas e interactivas. Esta herramienta permite al usuario unificar diferentes orígenes de datos para crear *dashboards* interactivos que facilitan el proceso de análisis de datos.

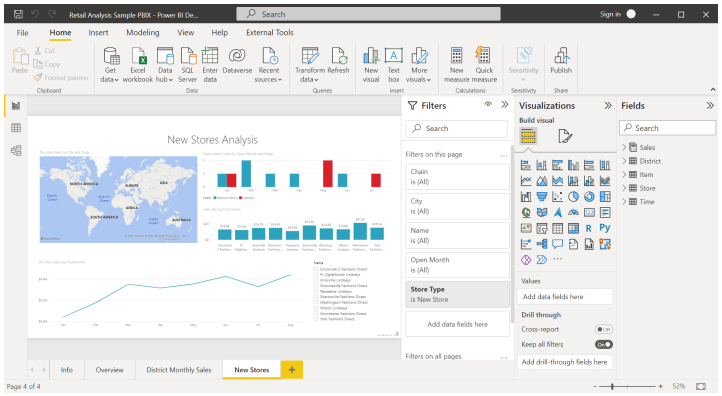


Figura 2.5: Ejemplo de *dashboard*

En concreto, para el desarrollo de este proyecto se utilizó su herramienta de escritorio Power BI Desktop.

2.2.2 ARIMA

ARIMA (Autoregressive Integrated Moving Average) [9] es un modelo estadístico utilizado en análisis de series temporales para prever valores futuros. Se compone de tres componentes: el modelo autoregresivo (AR) que considera la dependencia lineal de la serie en sus valores pasados, el componente de media móvil (MA) que modela la dependencia de los errores pasados, y la integración (I) que indica la cantidad de veces que se ha diferenciado la serie para hacerla estacionaria.

Red neuronal recurrente

Una red neuronal recurrente [10] (RNN) es un tipo de arquitectura de red neuronal diseñada para procesar secuencias de datos, donde la salida actual depende no solo de la entrada actual sino también de la información anterior en la secuencia. A diferencia de las redes neuronales feedforward convencionales, las RNN tienen conexiones que forman bucles, lo que les permite retener y utilizar información pasada. Esto las hace eficaces para modelar patrones temporales y secuenciales en datos, como texto, audio o series temporales. Cada unidad en una RNN mantiene una "memoria" interna que se actualiza continuamente a medida que procesa la entrada actual y la información previa. Sin embargo, las RNN también pueden enfrentar desafíos, como la dificultad para manejar dependencias a largo plazo debido a problemas de desvanecimiento o explosión del gradiente. En la figura 2.6 se puede observar el esquema de una red neuronal recurrente.

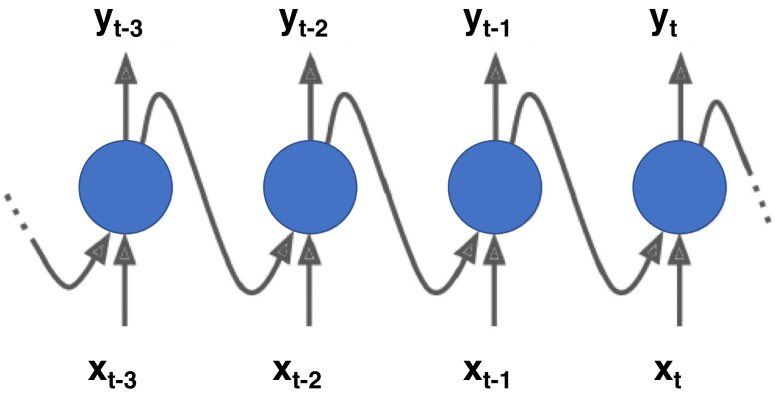


Figura 2.6: Red neuronal recurrente

Python

Python [11, 12] es un lenguaje de programación interpretado y orientado a objetos. La curva de aprendizaje de la sintaxis de Python es más sencilla que la de otros lenguajes de programación y con una legibilidad alta, lo que reduce los costes de mantenimiento de los programas.

Gracias a su sencillez y las posibilidades de uso que tiene, Python es a día de hoy uno de los lenguajes más populares. Python es el rey en lo que se refiere al análisis y manipulación de datos. Gracias a la gran cantidad de librerías que tiene, es el lenguaje de programación más utilizado por científicos e ingenieros de datos. Entre las librerías utilizadas en este proyecto cabe destacar:

- Pandas [13]: es un paquete de Python que proporciona estructuras de datos rápidas, flexibles y expresivas diseñadas para que trabajar con datos "relacionales" o "etiquetados" sea fácil e intuitivo. Este paquete nos permite aplicar una infinidad de transformaciones sobre los datos, convirtiéndose en la herramienta de análisis/manipulación de datos más potente de la actualidad. Esta librería se utilizó en el proyecto para las transformaciones en los datos y para manipularlos previamente a introducirlos a los modelos de predicción.

- Tensorflow [14]: es una biblioteca de código abierto desarrollada por Google, diseñada para facilitar la creación y entrenamiento de modelos de aprendizaje automático y redes neuronales. Utiliza un enfoque de flujo de datos mediante la representación de operaciones matemáticas como grafos computacionales, donde los nodos son tensores que representan datos multidimensionales. En el proyecto esta librería se utilizó para entrenar los modelos basados en redes de neuronas.

- Matplotlib [15]: es una completa biblioteca para crear visualizaciones estáticas, animadas e interactivas en Python. Con una sintaxis simple, los desarrolladores pueden generar gráficos de líneas, dispersión, barras y más. Además, Matplotlib permite personalizar estilos, colores y etiquetas para mejorar la presentación de los gráficos. Esta librería se utilizó para crear distintas gráficas

2.2.3 Tecnologías y herramientas auxiliares

Visual Studio Code

Visual Studio Code [16] es un editor de código fuente disponible para Windows,Linux y macOS. Se caracteriza por ser una interfaz ligera y minimalista que permite desarrollar proyectos en muchos lenguajes de programación destinos. Desde ella el usuario podrá tanto editar código como ejecutarlo y depurarlo. También cuenta con integración con Git y un gran número de extensiones creadas por otros desarrolladores que permiten añadir funcionalidades a la herramienta.

PostgreSQL

PostgreSQL [17] es un sistema de gestión de bases de datos relacionales de código abierto que amplía el lenguaje SQL combinado con muchas características que almacenan y escalan con seguridad las cargas de trabajo de datos más complicadas. Es el sistema de gestión de base de datos que utiliza la fuente de datos de origen.

Capítulo 3

Metodología y planificación

En este capítulo se explicará la metodología de trabajo utilizada y la planificación que se llevó a cabo para cumplir los objetivos del proyecto.

3.1 Metodología

Para el desarrollo de este proyecto se utilizó la famosa metodología basada en sprints llamada Scrum.

3.1.1 Scrum

Scrum [18, 19] es un marco de trabajo ágil que se utiliza comúnmente en el desarrollo de software, pero también puede aplicarse a otros contextos. Fue desarrollado para gestionar proyectos complejos, y se basa en principios de transparencia, inspección y adaptación. Scrum se centra en la entrega incremental de productos y en la colaboración efectiva entre los miembros del equipo.

Eventos clave

La metodología Scrum cuenta con los siguientes eventos principales para su correcto desarrollo:

- Sprint: Un ciclo de desarrollo con una duración fija.
- Reunión de Planificación del Sprint: Define el trabajo a realizar durante el *sprint*.
- Reunión Diaria de Scrum: Actualización diaria sobre el progreso y los impedimentos.
- Revisión del Sprint: Demostración del trabajo completado al final del *sprint*.
- Retrospectiva del Sprint: Reflexión sobre el *sprint* y mejoras para el próximo.

Beneficios

La metodología Scrum ofrece una serie de beneficios significativos en el desarrollo de proyectos, destacando la flexibilidad y adaptabilidad como pilares fundamentales. Al adoptar Scrum, los equipos pueden responder de manera ágil a los cambios en los requisitos del proyecto, permitiendo una mayor capacidad de adaptación a medida que evoluciona la comprensión del cliente. Además, al dividir el trabajo en iteraciones llamadas *sprints*, Scrum fomenta la entrega continua de incrementos de producto funcional, lo que facilita la retroalimentación temprana del cliente y la mejora constante. La transparencia en el proceso, facilitada por eventos como la reunión diaria de Scrum, promueve una mayor visibilidad y colaboración entre los miembros del equipo.

3.1.2 Diseño de los *sprints*

Como se mencionó al principio del capítulo, Scrum fue la metodología adoptada para el desarrollo de este proyecto. La realización del mismo se define en los siguientes *sprints*:

- **Sprint 1: Análisis de la base datos.** En esta fase se analiza el contenido de las distintas tablas de la base de datos y se seleccionan los datos de interés. En esta fase también se definen las necesidades de transformación de los datos en proceso ETL posterior.
- **Sprint 2: Desarrollo de la ETL.** En esta iteración se desarrolla el proceso ETL en Apache Airflow. En primer lugar se implementa el código necesario para la extracción de los datos de la base de datos utilizando las consultas SQL que se definen en el sprint anterior. Posteriormente debe desarrollarse el código necesario para aplicar las transformaciones necesarias en los datos.
- **Sprint 3: Diseño de las herramientas de explotación** En este sprint se diseñaran los distintos *dashboards* utilizando los datos previamente procesados.
- **Sprint 4: Desarrollo del dashboard.** Este *sprint* representa la parte principal del proyecto. En esta fase se implementan los distintos cuadros de mandos que permitirán hacer analítica descriptiva con PowerBI.
- **Sprint 5: Exploración de métodos de predicción.** En esta fase se exploran distintos métodos de predicción de alguna de las métricas extraídas de la base de datos. Para la implementación de los modelos predictivos se debe realizar un análisis exploratorio previo de las series temporales.
- **Sprint 6: Integración de los modelos en las visualizaciones.** En esta fase de completan los *dashboards* desarrollados en el cuarto *sprint* con las predicciones de los modelos.

- **Sprint 7: Redacción de la memoria.** Una vez desarrollada la parte técnica del proyecto, este debe documentarse en una memoria explicativa. Esta memoria integra todas las documentaciones que se fueron realizando durante todo el transcurso del proyecto. En ella se encuentran la motivación del proyecto, la explicación del desarrollo del mismo y las conclusiones ente otros.

3.2 Planificación

Desde el inicio del proyecto hasta el final del mismo, se trató de seguir una planificación inicial estimada. Una buena planificación inicial es muy importante para poder abordar proyectos de este tipo.

Esta planificación inicial se representó en un diagrama de Gantt. En este cada tarea representa a cada uno de los *sprints* mencionados en el apartado 3.1.2. Cabe destacar que al final de cada *sprint* como se explicó hay reuniones de retrospectiva y planificación de sprint, por lo que en algunos sprints se tuvieron que mejorar o corregir desarrollos de sprints anteriores por lo que la planificación inicial no es una representación exacta del trabajo realizado pero sí una estimación muy cercana a la realidad.

El diagrama de Gantt que representa esta planificación inicial puede verse en la Figura 3.1.

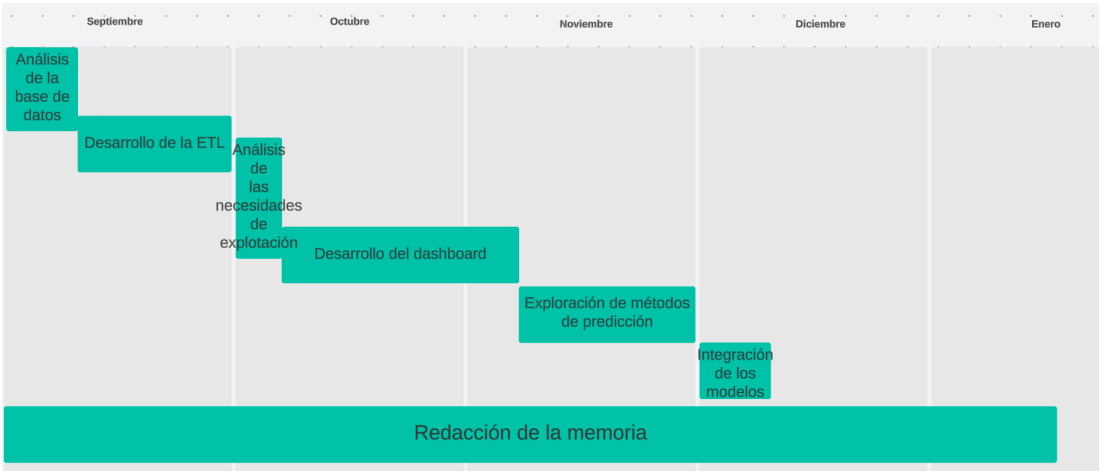


Figura 3.1: Planificación inicial

En la tabla 3.1 se muestra la comparativa entre las horas estimadas y las horas reales *sprint*. Hubo ligeros retrasos en alguno de los *sprints* pero en el cómputo general las horas reales se acercan bastante a las estimadas inicialmente. Los sprints cortos como el análisis de la base de datos, el diseño de las herramientas de explotación y la integración de los modelos en el cuadro de mandos se podrían denominar *microsprints*, ya que se consideraron como fases a parte de los *sprints* más importantes. Por un lado se experimentaron desviaciones significativas en el desarrollo ETL, implementación de los *dashboards* y y la redacción de la memoria. Estas desviaciones se compensaron en parte con la exploración de los modelos de predicción, ya que parte del conocimiento y código empleado para la predicción de la temperatura se pudo reutilizar para la predicción de las precipitaciones.

Sprint	**Horas estimadas**	**Horas reales**	**Desviación**
Análisis de la base de datos	10	15	+5
Desarrollo del proceso ETL	70	85	+15
Diseño de las herramientas de explotación	10	10	0
Desarrollo del dashboard	60	70	+10
Exploración de métodos de predicción	60	40	-20
Integración de los modelos en el dashboard	10	5	-5
Redacción de la memoria	60	70	+10
Total	280	295	+15

Tabla 3.1: Horas estimadas vs horas reales

3.3 Análisis de costes

En cuanto a la estructura de costes del proyecto, en primer lugar se deben separar los distintos recursos utilizados para su realización.

Con respecto a los recursos hardware utilizados, se cuenta con el ordenador propio que supone un coste 0 para el proyecto.

En cuanto a las herramientas y tecnologías utilizadas tampoco suponen un coste al proyecto ya que se trata siempre de software libre o de versiones gratuitas de herramientas, como es el caso de PowerBI. Como en este proyecto se utilizan gráficos integrados con Python, si en un futuro se quisiera utilizar el PowerBI Report Server o embeber el *dashboard* en una página web, sería necesario mejorar a la versión Pro de PowerBI, con un coste de 9,4 euros/mes. Este coste no se considera como coste del proyecto.

Por último falta mencionar los recursos humanos del proyecto, es decir, el científico de datos que lo desarrolla y los directores que supervisan el proyecto. En cuanto al científico de datos se estima un coste de 25€/hora. Como para la realización del proyecto se hizo uso de 280

horas de trabajo, se estima un coste 7000€ por parte del desarrollador. El coste por hora de un director de proyecto se estima en 35€. Aproximadamente se utilizaron 25 horas de reuniones por parte de los dos directores, por lo que se estima un coste de 1750€. Por tanto el coste global del proyecto sería aproximadamente de unos 8750€.

En la tabla 3.2 se pueden ver los costes mencionados desglosados.

Recurso	Coste/h	Coste
Científico de datos	25€	7000€
Director 1	35€	875€
Director 2	35€	875€
Power BI Desktop	0€	0€
Ordenador propio	0€	0€
Total		**8750€**

Tabla 3.2: Costes del proyecto

Capítulo 4
Análisis

En este capítulo se explicará el contenido de la fuente de datos y las posibilidades de análisis que esta ofrece.

4.1 Contexto previo

Antes de pasar con el análisis de la fuentes de datos es necesario explicar el contexto de este trabajo de fin de grado. Este trabajo se enmarca dentro de un proyecto empresarial. El proceso ETL recoge distintas utilidades a mayores de las necesarias para el desarrollo de este proyecto. Esto se traduce en que se mostrarán campos que no fueron eliminados que tienen otra finalidad pero que no se utilizaran en este trabajo. De ahora en adelante solo se explicarán aquellos campos necesarios para la explotación analítica realizada.

Asimismo, al ser un proyecto empresarial tampoco se podrá mostrar el nombre de la base de datos ni el nombre exacto de sus tablas ni de las columnas de las tablas. Prescindir de estos nombres no es algo relevante ya que lo importante es saber el contenido de dichas tablas.

4.2 Análisis de la fuente de datos

Este proyecto utiliza una única fuente de datos, la cual es una base de datos relacional PostgreSQL proporcionada por el Cabildo de Tenerife. En esta sección se analizará el contenido de las distintas tablas de la base de datos con el fin de seleccionar las tablas con la información más relevante.

4.2.1 Contenido

La base de datos cuenta con una gran cantidad de datos de interés para el diseño de una herramienta de explotación. Se puede encontrar la información detallada de cada una de las estaciones meteorológicas de la isla. También se cuenta con información geográfica de cada una

de las estaciones, tanto el municipio y localidad a la que pertenecen, como sus coordenadas geográficas exactas. Las estaciones meteorológicas cuentan con distintos sensores asociados a ellas. Estos sensores pueden ser de distintos tipos, es decir registran distintos tipos de medidas y pueden estar instalados de distintas formas. Por cada sensor existe un histórico de 9 años de registros diarios de sus mediciones. A partir de este contenido se buscará implementar una herramienta de visualización que permita explotar esta gran cantidad de información almacenada de las mediciones. La base de datos cuenta con un total de 180 tablas por lo que el análisis de centrará en las tablas consideradas mas relevantes que se pueden ver en la Figura 4.1. En el siguiente apartado se explicarán las mismas en detalle.

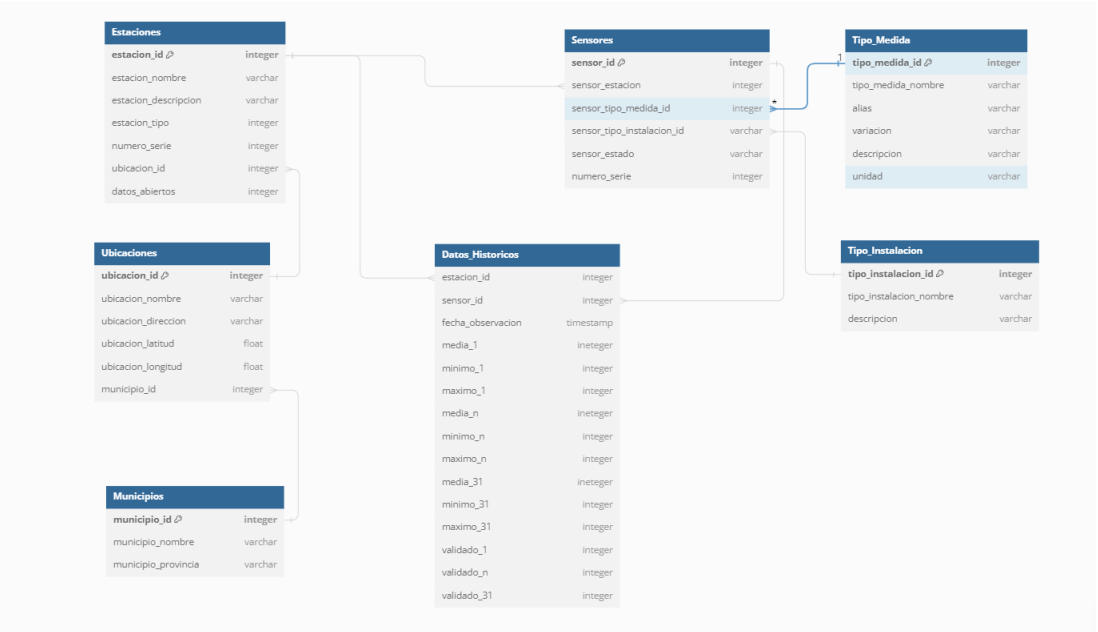


Figura 4.1: Diagrama de la base de datos

4.2.2 Tablas relevantes

- **Estaciones**. Esta tabla contiene toda la información acerca de la estaciones de la isla. Esta tabla podría ser relevante ya que contiene los nombre de la estación, el tipo de estación que es y su ubicación, estas dos últimas en forma de ubicación. Esta tabla contiene un campo que es necesario a la hora de diseñar las consultas SQL de extracción, ya que nos indica si son datos que pueden ser públicos, por lo que habrá que hacer un filtrado

en función de este campo. La tabla cuenta con un total de 139 estaciones diferentes, de las cuales únicamente se extraerán 62 debido a la condición del cumplimiento de este campo.

- **Ubicaciones**. Ya que en la tabla las ubicaciones están representadas con un identificador, esta tabla contiene las coordenadas geográficas exactas asociadas a ese identificador y asimismo al municipio al que pertenece el punto representado con un identificador. Combinando esta tabla con la anterior que contiene la información de las estaciones meteorológicas, ya se podrían representar en el mapa, que era uno de los objetivos de análisis que se buscaba.

- **Municipios** Como en la tabla anterior se vio que cada ubicación contenía el municipio al que pertenece representado por un identificador, esta tabla permite saber que municipio es cada uno de los identificadores. Gracias a contar con los nombres de los municipios, en un futuro se podrán agrupar los datos en función del municipio al que pertenecen, por lo que quedaría otro requisito de explotación resuelto.

- **Sensores** Esta tabla contiene la información de todos los sensores que reportan las medidas. Lo más relevante de esta tabla es que indica en que estación están instalados, que tipo de métrica mide y cual es su tipo de instalación. Como en las tablas anteriores, todos estos campos están representados por identificadores.

- **Tipos de medida** Como la tabla anterior contiene el tipo de métrica representado por un id, esta tabla proporciona la información del tipo de medida más detallada, indicando con un nombre que tipo de metida recoge (por ejemplo “Velocidad del viento”) y en que unidad se está midiendo.

- **Tipo de instalacion** En esta tabla se recogen los distintos tipos de instalación de los sensores. Esta tabla es importante explicarla ya que fue petición del cliente que únicamente se extrajeran datos de sensores que estuvieran instalado al aire libre, por lo que va a ser una tabla que se utilizará para hacer un filtrado en la consulta.

- **Datos históricos** Esta es la tabla de hechos la cual recoge todos los datos históricos. En cada uno de los registros está relacionado con una estación y un sensor en concreto. Cada fila de la tabla representa un mes de datos. En la tabla 4.1 se puede observar cómo es la estructura en la que se disponen los datos para el ejemplo del mes de enero de 2023. Cada fila cuenta con un total de 93 columnas, ya que existe una columna para cada día y para cada tipo de métrica (media, máximo y mínimo diario). A su vez existe una columna columna *validada* para cada día, donde se indica si los datos de ese día están validados o no.

estacion	sensor	fecha	media_1	maximo_1	minimo_1	...	media_31	maximo_31	minimo_31	validado_1	...	validado_31
1	2	2023-01-01	3.5	7.5	1.5	...	6.6	10.0	4.2	True	...	False

Tabla 4.1: Tabla de datos históricos

4.3 Requisitos de explotación

Una vez analizada la fuente de datos de origen y sus tablas más relevantes hay que dar paso a definir las necesidades de análisis que se buscará satisfacer con las distintas visualizaciones de la herramienta.

4.3.1 Necesidades de análisis

Para poder contar con una herramienta de análisis lo suficientemente completa es de interés poder contar con ciertas características. Entre estas características con las que debe contar la herramienta, se encuentra disponer de un histórico amplio, de varias métricas registradas en las distintas estaciones ubicadas en los municipios de la isla, y a su vez contar con predicciones de las más importantes. A continuación se detallan las distintas necesidades de análisis encontradas:

- **Diversidad de métricas.** Ya que la mayor parte de las herramientas de análisis climático cuentan con variedad de métricas, sería interesante poder contar con la evolución de distintas métricas y así poder ver la relación entre unas y otras. Tras analizar la fuente de datos se seleccionaron las siguientes medidas a representar:
  - Temperatura (media, máxima, mínima)
  - Precipitación (total)
  - Radiación solar (máxima)
  - Humedad relativa (media, máxima, mínima)
  - Velocidad del viento (media, máxima, mínima)

  Esta medidas fueron las que se consideraron más importantes basándose en las medidas que manejan las herramientas de análisis meteorológico más populares. Entre otras medidas que se descartaron se encuentran:
  - Presión atmosférica
  - Evaporación
  - Dirección del viento

- **Evolución histórica de los datos.** Al tratarse de una herramienta que busca poder ofrecer un análisis de datos históricos es importante poder contar con un registro histórico importante. Estos datos históricos lo más común es representarlos como series temporales, por lo que para poder encontrar patrones significativos que se repitan a lo largo del tiempo es importante contar con un histórico de al menos 5 años. Hay que tener en cuenta las limitaciones de memoria RAM que tiene el equipo donde se desarrolla este proyecto ya que un número muy elevado de registros puede provocar problemas en el procesamiento de los datos o en la implementación de las visualizaciones.

- **Datos por municipio.** A pesar de que Tenerife tiene una superficie reducida, puede ser interesante visualizar la evolución de las medidas en los distintos municipios. Esto permitiría hacer un análisis más detallado de las distintas zonas de la isla, y compararlas entre sí.

- **Datos por estación.** El histórico de datos los registran las distintas estaciones meteorológicas repartidas por toda la isla. Algunos municipios cuentan con más de una estación meteorológica, por lo que es interesante donde se ubica cada una de estas estaciones geográficamente en el mapa de la isla, y poder contar con la evolución de las métricas a nivel de cada estación, pudiendo hacer comparación también entre las distintas estaciones.

- **Predicciones futuras.** Por último, para complementar el análisis histórico, es interesante poder contar con la predicción futura de alguna de las métricas, consiguiendo así una herramienta de análisis lo más completa posible. El objetivo sería conseguir una estimación de cómo se van a comportar algunas de las medias en los días posteriores o predecir algún evento. Por ejemplo sería interesante poder tener delante la evolución histórica de una medida y poder ver al mismo tiempo la estimación para la semana siguiente.

4.4 Tablas y campos seleccionados

En base a las necesidades de explotación planteadas, se escogieron los siguientes campos de la fuente de datos analizada previamente:

- **Tabla Estaciones**
  - *estacion_id*: identificador de la estación
  - *estacion_nombre*: nombre de la estación
  - *ubicacion_id*: identificador de la estación

  - *datos_abiertos*: indica si los datos de la estación pueden ser públicos

- **Tabla Ubicaciones**
  - *ubicacion_id*: identificador de la estación
  - *ubicacion_latitud*: latitud de la ubicación
  - *ubicacion_longitud*: identificador de la estación
  - *municipio_id*: identificador del municipio

- **Tabla Municipios**
  - *municipio_id*: identificador del municipio
  - *municipio_nombre*: nombre del municipio

- **Tabla Sensores**
  - *sensor_id*: identificador del sensor
  - *sensor_estacion*: estaciones en la que está instalado el sensor
  - *sensor_tipo_medida_id*: identificador del tipo de medida que registra el sensor
  - *sensor_tipo_instalacion_id*: identificador del tipo de instalación del sensor

- **Tabla Tipos de medida**
  - *tipo_medida_id*: identificador del tipo de medida
  - *tipo_medida_nombre*: nombre del tipo de medida (temperatura, humedad Relativa, etc)

- **Tabla Tipo instalacion**
  - *instalacion_id*: identificador de la estación
  - *instalacion_nombre*: nombre del tipo de instalación (aire libre, interior, etc)

- **Tabla Datos Historicos**
  - *estacion_id*: identificador de la estación
  - *sensor_id*: identificador del sensor
  - *fecha_observacion*: mes y año de los registros
  - *media_1*: media diaria de la medida (media diaria registrada de la temperatura, humedad Relativa, etc)
  - *minimo_1*: mínimo diario de la medida (máximo diario registrado de la temperatura, humedad Relativa, etc)

- *maximo_1*: máximo diario de la medida (mínimo diario registrado de la temperatura, humedad Relativa, etc)
- ... (media, máximo y mínimo de cada uno de los días del mes)
- *media_31*: media diaria de la medida
- *minimo_31*: mínimo diario de la medida
- *maximo_31*: máximo diario de la medida
- *validado_1*: indica si los valores del día están validados (True/False)
- ... (validación de cada uno de los días del mes)
- *validado_31*: indica si los valores del día están validados (True/False)

Hay columnas que están replicadas para cada uno de los días del año (media,máximo,mínimo y validado), por tanto por cuestión de espacio el listado anterior se muestra de manera resumida.

Capítulo 5

Diseño e implementación

Este capítulo se centra en el diseño e implementación de los distintos componentes del proyecto. En primer lugar se explicará el proceso ETL, seguido de la construcción de la herramienta de análisis y, por último, se explorarán distintos métodos de predicción con el que completar la herramienta.

5.1 Proceso ETL

En esta sección se detallan los distintos procesos que sufren los datos desde el inicio hasta el final y cómo se implementan estos procesos.

5.1.1 Resultado esperado

En este apartado se mostrarán los archivos CSV que se buscará generar a partir del proceso ETL. El proceso ETL se diseñó para poder utilizarse sobre una base de datos y cualquier formato de almacenamiento. En este proyecto se decidió quedarse con el resultado almacenado en archivos CSV para posteriormente cargarlos en la herramienta de análisis.

- **Un CSV con la información detallada de las estaciones meteorológicas.** Este recurso consta de 62 dos filas, donde cada una representa la información de cada estación meteorológica. Los campos de este recurso son los siguientes:
  - estacion_id
  - estacion_nombre
  - fecha_instalacion
  - fecha_baja
  - municipio_id
  - municipio_nombre

  - localidad_nombre
  - paraje
  - latitud
  - longitud
  - utm_x
  - utm_y
  - altitud
  - datalogger_tipo
  - sensores_cantidad

- **5 CSVs de medidas diarias.** Se generará un archivo CSV para cada una de las medidas, ya que se buscará posteriormente cargarlos en la herramienta por separado ya que cada medida será una página independiente. Los campos de cada uno de los recursos son los siguientes:

  - estacion_id
  - estacion_nombre
  - sensor_id
  - sensor_alias
  - sensor_nombre
  - fecha_observacion
  - metrica
  - valor

  Debido a la sencillez del modelo final que se cargará a PowerBI, no se decidió crear una dimensión para los sensores y para la fecha de observación. De la misma manera, por temas prácticos, se incluyó el nombre de la estación en los datos históricos para no tener que hacer el join con la tabla de las estaciones.

5.1.2 Transformaciones necesarias

Para poder generar el resultado mencionado anteriormente, se identificaron una serie de transformaciones a aplicar en los datos de origen. Para conseguir dicho resultado es necesario desarrollar dos DAGs. El primer DAG generará el archivo CSV que contiene la información de cada una de las estaciones. El otro DAG se encargará de generar los distintos recursos que

contienen la información histórica de las medidas registradas por cada estación meteorológica en cada año.

A continuación se explicarán las trasformaciones necesarias que se deben implementar en cada uno de los DAGs a desarrollar.

- **Detalle de las estaciones**. Debido a que se diseñó una consulta SQL que contiene los filtros fundamentales necesarios para obtener los datos que se buscan, este es relativamente simple, ya que no es necesario realizar apenas transformaciones.
- **Datos históricos diarios**. Como podíamos observar en la tabla 4.1 los datos de origen están pivotados por lo que es necesario despivotarlos con el fin de obtener los datos de la forma que se puede ver en la tabla 5.1. De esta forma los datos ya podrán cargarse en la herramienta de análisis ya que tendrían la forma buscada. En la parte de construcción de los DAGs se explicará paso por paso cómo se utilizaron distintos operadores de transformación para conseguir el resultado deseado.

estacion_id	estacion_nombre	sensor_id	sensor_alias	sensor_nombre	fecha_observacion	metrica	valor
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	media	3.33
2	ABONACOP	2	WSP	Velocidad del viento	2023-01-01	minimo	2
3	ABONACOP	3	WSP	Velocidad del viento	2023-01-01	maximo	5.5

Tabla 5.1: Tabla de datos históricos objetivo

5.1.3 Extracción

Conexión a la base de datos

Para poder extraer los datos es necesario tener un método de conexión a la base de datos. En Apache Airflow, un *Hook* es una interfaz que facilita la interacción con sistemas externos al proporcionar una abstracción sobre las conexiones a estos sistemas. En lugar de lidiar directamente con la complejidad de la conexión y las operaciones específicas de cada sistema externo, los *Hooks* permiten encapsular esa lógica, promoviendo la reutilización de código.

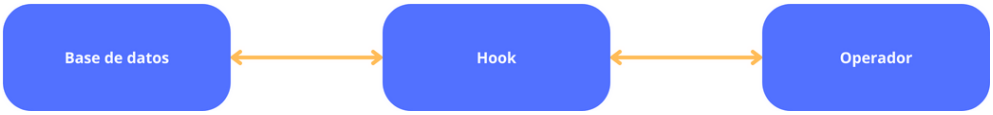


Figura 5.1: Funcionamiento del *Hook*

En Airflow, en el pipeline del DAG, cada tarea esta programada dentro de un operador. El

operador de extracción se encarga de interactuar con la base de datos PostgreSQL, utilizando el *Hook* de PostgreSQL, para obtener los datos necesarios. La ventaja principal de utilizar el *Hook* de Airflow en vez de programar la lógica de la conexión radica en la reutilización de código, ya que este operador de extracción desarrollado para bases de datos PostgreSQL podría ser reutilizado en futuras extracciones que utilicen el mismo sistema de gestión de bases de datos.

En la Figura 5.2 se puede observar cómo se declara una conexión en la interfaz gráfica de Airflow. Para utilizarla posteriormente en el operador de extracción, se declaró el *Hook* de PostgreSQL y se indica el *Connection Id* de la conexión que se quiere utilizar.

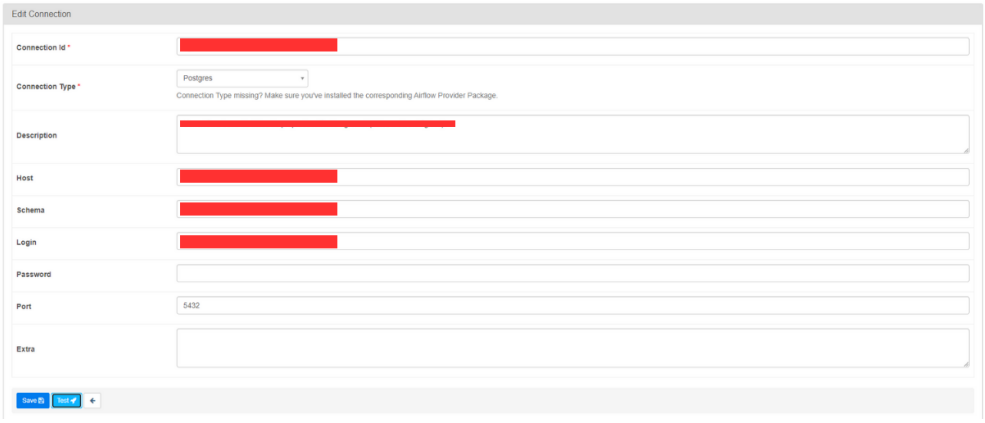


Figura 5.2: Configuración de una conexión

Consultas SQL de extracción empleadas

Una vez que se ha diseñado un método para la conexión con la base de datos PostgreSQL, es necesario abordar la implementación en el operador de extracción de cómo extraer los datos de interés, buscando siempre la reutilización de código para futuras extracciones. Airflow permite definir variables en su interfaz gráfica, que después pueden ser utilizadas en el código del DAG.

Las variables en Airflow son clave para la reutilización de código, ya que un mismo operador puede utilizar distintas variables en distintos DAGs. Para el operador de extracción desarrollado, las consultas SQL de extracción se declararon en variables de Airflow. El operador lee la consulta y utilizando el *Hook* extrae los datos de la base de datos utilizando dicha consulta y genera un archivo CSV con los mismos.

Para el desarrollo ETL se implementaron dos DAGs. El primero es para generar un recurso con toda la información de cada una de las estaciones (nombre, ubicación,etc.) y el segundo genera los recursos de las mediciones por estación y por año. Cada uno de estos DAGs utiliza una consulta SQL de extracción que son las siguientes:

- **Datos históricos diarios**

```
SELECT
  estacion_id,
  estacion_nombre,
  s.sensor_id,
  tipo_medida_id,
  sensor_alias,
  sensor_nombre,
  fecha_observacion,
  media_1,
  minimo_1,
  maximo_1,
  ...
  media_31,
  minimo_31,
  maximo_31,
  validado_1,
  ...
  validado_31
FROM
  tabla_historico
INNER JOIN estaciones ws ON wh.estacion_id = ws.estacion_id
  INNER JOIN sensores s ON wh.sensor_id = s.sensor_id
INNER JOIN tipo_medidas dt ON tipo_medida_id = dt.tipo_medida_id
WHERE
  opendata = true
  AND s.instalacion_tipo = 1
  AND dt.tipo_medida_id IN (10, 11, 12, 13, 15)
ORDER BY
  wh.sensor_id,
  tabla_historico.fecha_observacion
```

  En esta consulta se extraen las métricas diarias de los tipos de medidas seleccionados en el capítulo de Análisis 4. Estas métricas son la temperatura, radiación solar, humedad relativa, velocidad del viento y precipitación, filtradas por *tipo_medida_id*. Asimismo, se aplica el filtro de *opendata* que garantiza que los datos pueden ser públicos y se seleccionan los sensores que fueron instalados en el exterior (*instalacion_tipo=1*).

- **Detalle de las estaciones**

```
select
  estacion_id,
  estacion_nombre,
  fecha_instalacion,
  fecha_baja,
  municipio_id,
  municipio_nombre,
  localidad_nombre,
  paraje,
  latitud,
  longitud,
  utm_x,
  utm_y,
  altitud,
from
  estaciones ws
  left join ubicaciones l on ws.ubicacion_id = l.ubicacion_id
  left join localidades l2 on l.localidad_id = l2.localidad_id
  left join municipios m on l2.municipio_id = m.municipio_id
where
  opendata is true
order by
  ws.estacion_id
```

En esta consulta se extrae toda la información significativa de las estaciones meteorológicas. Entre esta información relevante se encuentra su nombre y información geográfica de cada estación. Se aplica el filtro al campo *opendata* para extraer únicamente aquellas estaciones que pueden ser públicas.

5.1.4 Diseño de los DAGs

- **Detalle de las estaciones**. Debido a que no se necesita aplicar apenas transformaciones en estos datos se trata de un DAG simple. En primer lugar se utilizó el operador de extracción mencionado en la sección anterior. Este operador utiliza una consulta definida por variable de Airflow y la ejecuta a través del Hook. Una vez se obtienen los datos, se vuelcan en un CSV que pasará al siguiente bloque de tareas. En la Figura 5.3 se puede observar que hay un bloque de tareas previo a la extracción que se llama *operaciones_previas*, donde simplemente se eliminan de la carpeta los ficheros generados en la ejecución anterior y se muestra la variable de configuración utilizada en la ejecución.

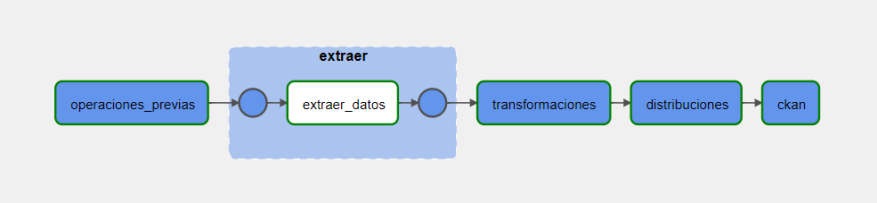


Figura 5.3: Bloque de tareas de extracción

El siguiente bloque de tareas es el bloque de tareas de transformaciones. En primer lugar se aplicó un operador que elimina CSVs vacíos. Esto se implementó ya que en algunas ocasiones se observó con otras bases de datos que por problemas ajenos, estas bases de datos fallan y no devuelven datos. Por cuestiones de rendimiento, si no se devuelven datos y el operador de extracción genera un CSV vacío, lo eliminará y pasará a los siguientes bloques de tareas. Una vez el CSV generado en la extracción pasa por este operador, se ejecuta el operador que modifica la fecha, la cual en los datos de origen esta en formato *datetime* y se transforma a formato ISO.

En la Figura 5.4 se puede observar el bloque de tareas explicado previamente, con cada una de las tareas que lo componen.

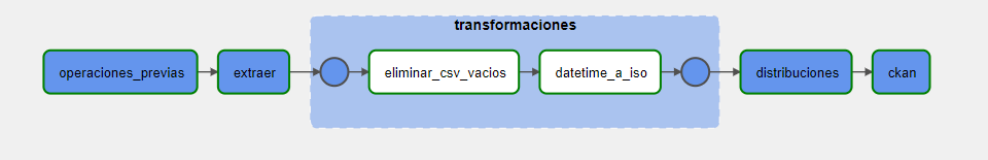


Figura 5.4: Bloque de tareas de transformación

- **Datos históricos diarios**. En primer lugar cuenta con un bloque de extracción que funciona de la misma forma que el del DAG anteriormente explicado pero utilizando su respectiva consulta SQL. A continuación del bloque de extracción, se encuentra el bloque de transformaciones que se refleja en la Figura 5.5.

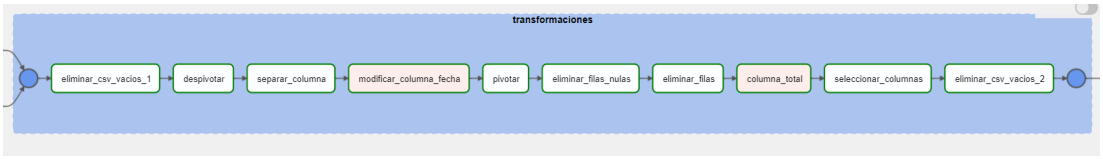


Figura 5.5: Bloque de tareas de transformación

A continuación se explicará mas en que se hace en detalle en cada una de las fases:

1. Eliminar CSVs vacíos. Este operador, al igual que el del DAG anterior, elimina todos los CSVs vacíos si se generaron durante la extracción.
2. Despivotar columnas. El despivoteo de columnas, también conocido como "unpivot", se refiere a la transformación de datos en una tabla para cambiar las columnas que contienen valores en varias columnas especializadas. En lugar de tener datos distribuidos en columnas específicas, se reorganizan en filas, manteniendo una columna de identificación única y otra columna que contiene los valores previamente dispersos.

   En este caso, las columnas que hay que despivotar son las columnas que contienen las métricas de cada día y las columnas que indican si cada día del mes está validado (media_1, minimo_1, maximo_1,..., media_31, minimo_31, maximo_31 y validado_1,..., validado_31). A continuación, en la Figura 5.6, se puede observar como es el resultado de esta transformación.

estacion_id	estacion_nombre	sensor_id	sensor_alias	sensor_nombre	fecha_observacion	type	value
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	media_1	5
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	minimo_1	2
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	maximo_1	7
...	...	...	...	...	...	...	
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	validado_1	True
...	...	...	...	...	...	...	...

Figura 5.6: Primer unpivot

3. Separar columna. El siguiente operador separa la columna *type* en dos. Debido a que el mes y el año están almacenados en la columna fecha_observación, es necesario concatenar en la fecha la información del día que contiene la columna *type*. En la Figura 5.7 se observa el resultado de separarla.

estacion_id	estacion_nombre	sensor_id	sensor_alias	sensor_nombre	fecha_observacion	type	day	value
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	media	1	5
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	minimo	1	2
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	maximo	1	7
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	validado	1	True

Figura 5.7: Separación de la columna type

4. Modificar la columna fecha_observacion. Como se comentó en el anterior paso, la columna type se divide con el fin de poder tener la fecha del día en la columna fecha observación. En este paso se realiza la concatenación del día a la fecha. En la Figura se puede observar como quedaría para el día 2 de mes.

estacion_id	estacion_nombre	sensor_id	sensor_alias	sensor_nombre	fecha_observacion	type	day	value
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-02	media	2	5
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-02	minimo	2	2
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-02	maximo	2	7
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-02	validado	2	True

Figura 5.8: Concatenar day a fecha_observacion

5. Pivotar columnas. Al contrario de despivotar columnas, pivotar las columnas implica transformar datos en filas a una estructura tabular, donde las columnas representan categorías previamente distribuidas. Es necesario pivotar la columnas de type y value para poder hacer un filtrado posteriormente de los datos que están validados, ya que con la disposición de columnas actual no se puede hacer dicho filtrado. A continuación, en la Figura 5.9, se muestra como quedarían los datos tras la ejecución de esta tarea.

estacion_id	estacion_nombre	sensor_id	sensor_alias	sensor_nombre	fecha_observacion	day	media	minimo	maximo	validado
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-02	2	5	2	7	True

Figura 5.9: Pivot

6. Filtrar días que no existen. Recordando la estructura inicial, tanto los distintos tipos de métrica (media, minimo y total) como la columna de *validado* iban desde el día 1 hasta el 31. Teniendo en cuenta que hay meses que no tienen 31 días, se debe hacer un filtrado de estos.
7. Filtrar días no validados. Como se mencionó anteriormente, únicamente se puede contar con los datos validados, por lo que todas aquellas filas que contienen el valor *False* en la columna *validado* se eliminaron.
8. Construir columna *total*. Se decidió dentro del proyecto crear la columna total para las precipitaciones para que la persona que utilice los datos posteriormente no tuviera una idea equivocada del significado de los registros.
9. Eliminar columna *day* y *Validado*. Estas dos columnas ya no son útiles en el resultado final, por lo que se eliminaron.
10. Despivotar las columnas de nuevo. Las columnas que contienen los valores de las métricas se despivotaron de nuevo, obteniendo de esta forma la estructura que se buscaba en un inicio, teniendo en cuenta que existe la métrica total, que será nula para todos los tipos de medida a excepción de la precipitación diaria. En la Figura 5.10 se puede observar la estructura final de los datos después de aplicar todas las transformaciones. Esta estructura de datos será la que posteriormente se cargue en la herramienta de explotación

estacion_id	estacion_nombre	sensor_id	sensor_alias	sensor_nombre	fecha_observacion	metrica	valor
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	media	5
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	minimo	2
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	maximo	7
1	ABONACOP	1	WSP	Velocidad del viento	2023-01-01	Total	NULL

Figura 5.10: Resultado tras el unpivot

11. Crear un CSV por cada medida. Por último, se creó un fichero para cada una de las medidas, obteniendo así un CSV para cada una ellas (Temperatura, Humedad Relativa, etc.).

Ejemplo de implementación de un DAG: datos históricos

En este apartado se explicará como se implementó el DAG que procesa los datos históricos de las medidas registradas.

Un DAG esta compuesto por uno a varios grupos de tareas (Figura 5.11), donde estos a su vez están compuestos de las distintas tareas que se realizan. Cada tarea es una operación que realiza un operador de Airflow sobre los datos. En la Figura 5.12 se muestra el código del bloque de tareas más importante, el de transformaciones. Este bloque esta compuesto por cada una de las operaciones que se explicaron anteriormente. Cada uno de los bloques de código definidos dentro del grupo son una tarea.

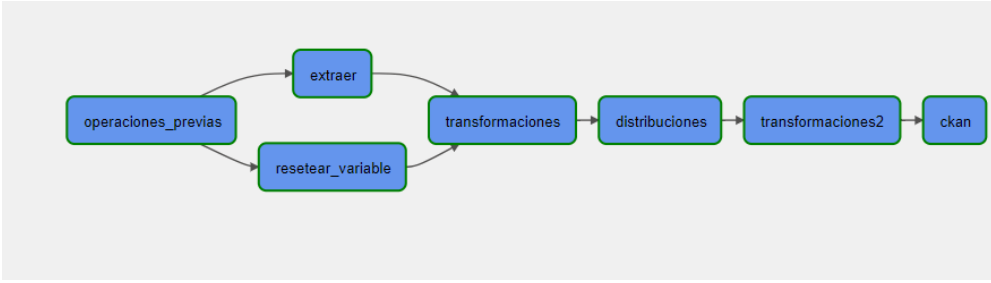


Figura 5.11: ETL de los datos históricos

```
with TaskGroup(group_id="transformaciones", tooltip="En este grupo se muestran todas las transformaciones") as transformations:

    empty_csv_delete1 = empty_csv_delete(
        task_id="eliminar_csv_vacios_1",
        path=datos_diarios_config["empty_csv_delete"]["path"]
    )

    csv_columns_unpivot_1 = csv_columns_unpivot(
        task_id="despivotar",
        path=datos_diarios_config["csv_columns_unpivot_1"]["path"],
        pc_vars=datos_diarios_config["csv_columns_unpivot_1"]["pc_vars"],
        pc_var_name=datos_diarios_config["csv_columns_unpivot_1"]["pc_var_name"],
        pc_value_name=datos_diarios_config["csv_columns_unpivot_1"]["pc_value_name"]
    )

    separate_column = csv_column_separate(
        task_id="separar_columna",
        path=datos_diarios_config["separate_column"]["path"],
        scc_column=datos_diarios_config["separate_column"]["scc_column"],
        scc_column_1=datos_diarios_config["separate_column"]["scc_column_1"],
        scc_column_2=datos_diarios_config["separate_column"]["scc_column_2"],
        scc_column_separator=datos_diarios_config["separate_column"]["scc_column_separator"]
    )

    modify_date_column = PythonOperator(
        task_id="modificar_columna_fecha",
        python_callable=modify_date_column,
        op_kwargs={
            "path":"/dags/ODT/datos_meteorologicos/datos_diarios/tmp/"
        }
    )

    total_column = PythonOperator(
        task_id="columna_total",
        python_callable=create_total_column,
        op_kwargs={
            "path":"/dags/ODT/datos_meteorologicos/datos_diarios/tmp/"
        }
    )
```

Figura 5.12: TaskGroup de transformaciones

Una vez se definen las operaciones que se deben ejecutar sobre los datos, es necesario

definir el orden de estas. Para indicar el orden en el que deben ejecutarse, se escribe al final del grupo de tareas el nombre de cada una de las tareas seguida de un ».

```
tarea1 >> tarea2 >> tarea3 >> tarea4
```

Ejemplo de implementación de un operador: operador de unpivot

Como se acaba de explicar previamente, cada una de las tareas que componen el DAG las realiza un operador previamente programado. A continuación se mostrará como está implementado un operador en concreto. La tarea escogida para explicar es una operación de unpivot. Explicada la implementación de un operador, la implementación del resto es similar.

```
csv_columns_unpivot_1 = csv_columns_unpivot(
    task_id="despivotar",
    path=datos_diarios_config["csv_columns_unpivot_1"]["path"],
    pc_vars=datos_diarios_config["csv_columns_unpivot_1"]["pc_vars"],
    pc_var_name=datos_diarios_config["csv_columns_unpivot_1"]["pc_var_name"],
    pc_value_name=datos_diarios_config["csv_columns_unpivot_1"]["pc_value_name"]
)
```

Figura 5.13: Tarea de unpivot

A continuación se muestra el código del operador en cuestión. La propia operación de unpivot se realiza con funciones de la librería Pandas. Como se puede observar en la Figura 5.14, la operación se realiza en función de los parámetros de entrada de la función pc_vars, pc_var_name y pc_value_name. Estos parámetros se observa que se introducen a través del código de la tarea del DAG en la Figura 5.13. Se introducen de esa forma ya que estos parámetros se indican dentro de la variable de la interfaz gráfica de Airflow.

```python
class csv_columns_unpivot(BaseOperator):
    def __init__(
        self,
        path: str,
        pc_vars: str,
        pc_var_name: str,
        pc_value_name: str,
        **kwargs
    ):
        super().__init__(**kwargs)
        self.path = path
        self.pc_vars = pc_vars
        self.pc_var_name = pc_var_name
        self.pc_value_name = pc_value_name


    def execute(self, context):
        path_files = os.getcwd()+self.path

        files=[]
        for file in os.listdir(path_files):
            if not file.startswith('.'):
                files.append(file)

        files=[x for x in files if '.csv' in x]

        for file in files:
            df=pd.read_csv(path_files+file,encoding ='utf-8-sig')
            if df.shape[0]>1:
                try:
                    df2=df.melt(id_vars=self.pc_vars.split(','),var_name=self.pc_var_name,value_name=self.pc_value_name)
                    df2.to_csv(path_files+file, index=False,encoding='utf-8-sig')
                    logging.info("%s %s","Fichero despivotado:",file)
                except:
                    error="Fichero no pivotado: "+file
                    raise ValueError(error)
            else:
                error="Fichero vacio - SE DETIENE EJECUCION: "+file
                raise ValueError(error)
```

Figura 5.14: Código del operador de unpivot

Desarrollando los operadores de esta forma se permite su reutilización en ETL futuras, ya que es configurable en función de los nombres de las columnas de los datos de entrada.

Validación de la ETL

Hasta ahora se fueron explicando paso todas las transformaciones que realiza el DAG implementado. Una de las ventajas que ofrece Apache Airflow para el desarrollo de procesos ETL, es la capacidad de monitorización que tiene. Una vez se implementados los DAG, si estos fueron implementados correctamente aparecerán en la pantalla de inicio de la interfaz web de Airflow, de no ser así, en la parte superior de la interfaz, saldrá un mensaje de error indicando en cuestión del código desarrollado.

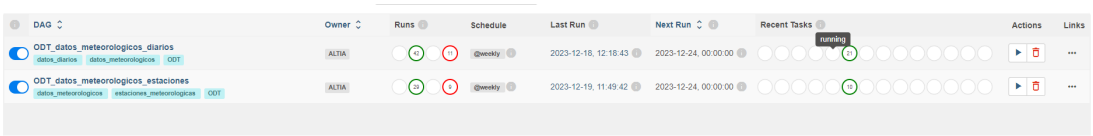


Figura 5.15: DAGs desarrollados

Si se entra en uno de los DAG (por ejemplo en *odt_datos_meteorologicos_diarios*) se podrán ver muchos más detalles sobre sus ejecuciones. En primer lugar, se puede consultar el

historial de ejecuciones pasada, su tiempo de ejecución y si fue una ejecución exitosa o fallida.

Si la ETL falló en alguna de las tareas, Airflow proporciona el log de la ejecución en cuestión para poder saber que falló. Por ejemplo en la Figura 5.16 se puede ver un error que sucedió en una de las ejecuciones debido a un fallo de programación con las columnas del dataframe.

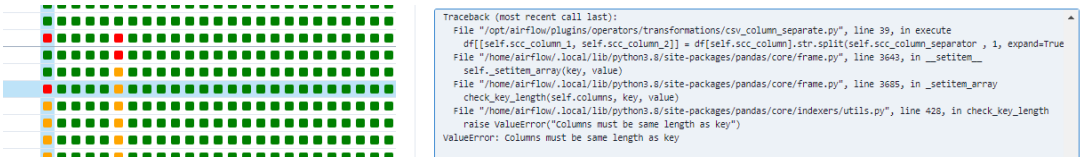


Figura 5.16: Error en el operador de separar columnas

Después de ejecutar ambos DAGs se hizo un resumen del proceso. Para el DAG que genera los datos históricos se procesaron 9 años, desde el 1 de enero de 2015 hasta el 9 de noviembre de 2023, que son los años disponibles en la base de datos utilizada para este proyecto, y se obtuvieron un total de 3 millones y medio de filas de registros para poder cargar en la herramienta de análisis. El DAG que genera la información de las estaciones generó un total de 62 filas correspondientes a la estaciones que son de carácter público (la base de datos contaba con 139 estaciones). La tabla 5.2 recoge esta información que se acaba de mencionar.

DAG	**Nº de filas de salida**
Detalle de las estaciones	62
Histórico	3568276

Tabla 5.2: Resumen de la ejecución

5.2 Diseño de los dashboards de explotación

La herramienta de visualización se diseñó con el objetivo de cubrir las principales necesidades de análisis descritas en anteriores capítulos.

En los apartados siguientes se explica en primer lugar que se busca implementar en la herramienta de explotación para satisfacer todas las necesidades de análisis. Posteriormente se detalla la carga de los datos y algunos ajustes necesarios con Power Query. Por último se

muestra la organización de la herramienta y se explican en detalle los distintos *dashboards* implementados en PowerBI.

5.2.1 Finalidades de la herramienta de explotación

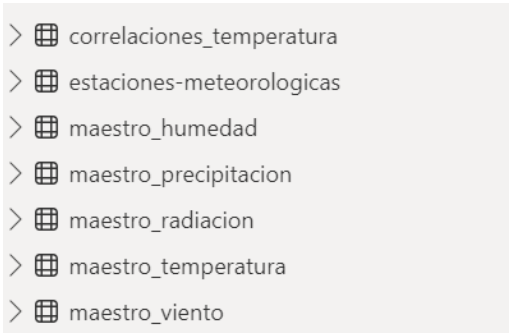
En este apartado se muestra la estructura que se busca dentro de la herramienta de explotación. A continuación se detallan cada una de las ideas de análisis que debe de tener el *dashboard* que se busca desarrollar:

- **Una página para cada medida**. El cuadro de mandos debe de tener una página para cada una de las medidas. Para navegar entre ellas se debe implementar un menú de navegación intuitivo que permita cambiar de una medida a otra.
- **Un comparador de estaciones para cada una de las medidas**. Para cada una de las medidas existentes, debe existir un comparador que permita comparar dos estaciones o una misma estación en tramos temporales distintos. Este comparador debe ir por separado a la página de la información general de cada una de las medidas.
- **Página de temperatura**. Esta página debe contener la información geográfica de cada una de las estaciones visualizada en un mapa. Sería de interés poder contar con la matriz de correlación entre las temperaturas de las estaciones e un mismo municipio, ya que la página entera deberá filtrarse por municipio. Al tratarse de datos representados como series temporales, es necesaria una visualización de la evolución diaria en cada una de las estaciones. Por último, esta página debe incluir la predicción que se desarrollará en la siguiente sección.
- **Página de precipitación**. Al igual que en el caso de la temperatura, se debe contar con un mapa de las estaciones. Esta página debe contener un KPI que muestre la predicción desarrollada en la siguiente sección.
- **Páginas de radiación solar, humedad relativa y velocidad del viento**. Estás páginas deberán contener su debido mapa con las estaciones representadas y la evolución histórica de la medida en cuestión como las páginas explicadas anteriormente. Sería de interés contar con alguna visualización de evolución temporal agrupada (mensual, anual, etc) ya que estas páginas no contarán con predicciones.

5.2.2 Carga de los datos

En el proceso ETL se generaron 5 archivos CSV que contienen el histórico de cada una de las medidas. Estos archivos contienen un histórico de 9 años (2015-2023). Los archivos de temperatura, humedad relativa y velocidad del viento cuentan con los registros medios, máximos

y mínimos diarios de cada una de las estaciones. El CSV de radiación solar cuenta únicamente con la radiación solar máxima. Por último, el CSV de las precipitaciones diarias cuanta con la precipitación total recogida por cada estación. A parte de los datos históricos también se generó un CSV con la información de cada una de las estaciones. Como se busca generar una visualización de matriz de correlación para la página de temperatura, se guardaron los datos de correlación en un archivo CSV generado con un script a parte del proceso ETL en Airflow. En la Figura 5.2.2 se pueden ver las tablas cargadas en la herramienta.



Una vez cargados los datos, estos no se pueden visualizar debido a cómo se representan los decimales en los archivos CSV. PowerBI utiliza como separador decimal la coma, mientras que el separador decimal en los archivos de entrada es el punto. Esto provoca que los datos se cargan como si fuera texto y no se puede cambiar a tipo numérico. En primer lugar se hizo una sustitución de todos los puntos por comas en los campos que contengan los valores de las medidas con el editor Power Query de PowerBI. Una vez cambiado el separador decimal, se cambió el tipo del campo a número decimal. Una vez hechas estas dos transformaciones, los datos ya se pueden representar.

Posteriormente se definieron las relaciones entre las fuentes de datos cargadas. Power BI tiene una limitación en cuanto a las relaciones. En el caso de la tabla de correlaciones entre estaciones, es necesario hacer una relación entre `estacion1` y `estacion2` con el campo de id_estacion de la tabla con la información de las estaciones, para poder hacer un filtrado dentro de la visualización. PowerBI no permite hacer una relación de dos campos de una tabla con el mismo campo de la tabla que se quiere relacionar. Para solucionar esto se creó una tabla auxiliar con la información de los nombres de las estaciones y los municipios a los que pertenecen.

estacion1	estacion2	valor
ARICO_01	ARICO_01	1
GALLETAS	ARICO_01	0,897064027201
GUIAIS01	ARICO_01	0,929961810263237
OROTAV01	ARICO_01	0,905438891200556
HELECHO1	ARICO_01	0,828716903330063
RAVELO01	ARICO_01	0,886161304704119
TEJINA01	ARICO_01	0,903731930708411
GUAN1HA1	ARICO_01	0,814787978169177
VILAFLOR	ARICO_01	0,827867767833689
TOPONEGR	ARICO_01	0,944390745861069
ABONACOP	ARICO_01	0,968253962235195
TEGUESTE	ARICO_01	0,927722864347818
LLANITOP	ARICO_01	0,930933571614894
RATINO 01	ARICO_01	0,92874730429314
ANAVISTH	ARICO_01	0,869972861824485
TRIGOTH	ARICO_01	0,937525748574698
URSULATH	ARICO_01	0,937463372503807
HOYOSTH	ARICO_01	0,831240229959236
PALMATH	ARICO_01	0,927695814627012
TACORTH	ARICO_01	0,929428092858747
MATANTH	ARICO_01	0,926857246230356
VICTOTH	ARICO_01	0,926912880420274
SUERTETH	ARICO_01	0,931683444702297

Figura 5.17: Tabla de las correlaciones entre las estaciones

Una vez creada esta tabla auxiliar ya se pueden establecer todas las estaciones entre las tablas. En la Figura 5.18 se puede observar como quedaría el modelo de datos final que se utilizará para crear las visualizaciones.

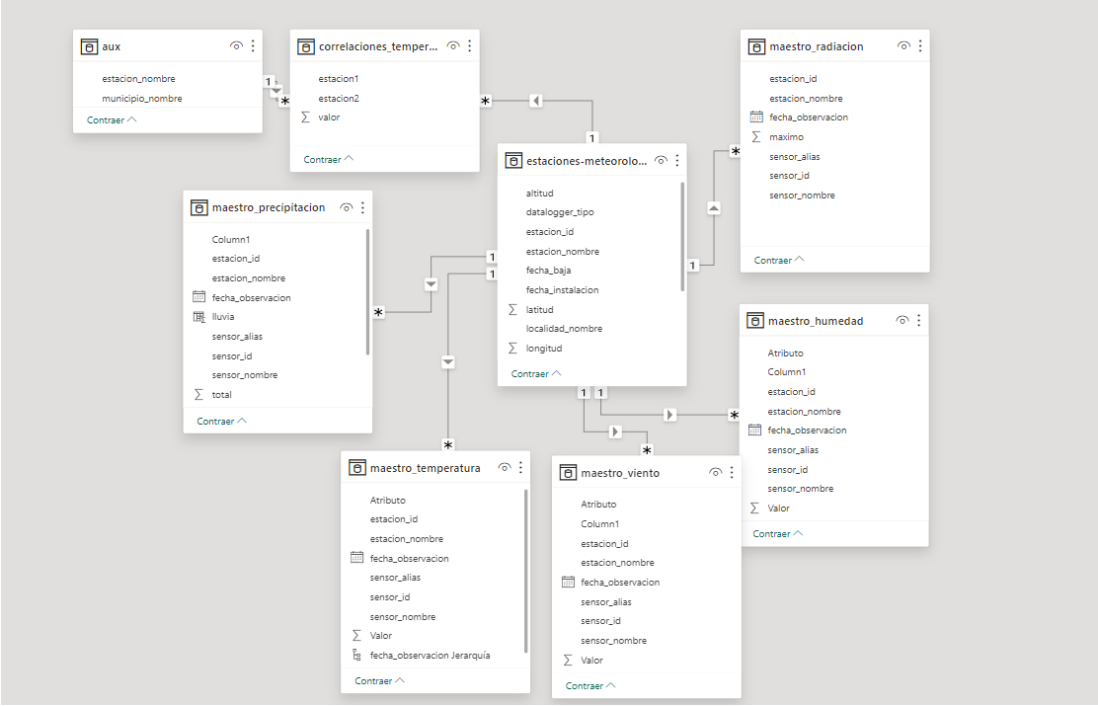


Figura 5.18: Modelo de datos de PowerBI

5.2.3 Estructura de la herramienta de análisis

La herramienta se organiza en dos menús de navegación. En primer lugar, consta de un menú de inicio. En este menú aparecen las distintas medidas disponibles (temperatura, velocidad del viento, etc) y se podrá navegar a cada una de ellas para consultar los datos acerca de esa medida en particular. Este menú descrito se puede visualizar en la Figura 5.19.

Figura 5.19: Menú principal de la herramienta

Una vez se navega a la medida que se quiere consultar, se puede o bien navegar a la página de **vista general** de la medida o bien al **comparador de estaciones**. En la Figura 5.20 se observa el menú de la temperatura. La página de vista general contiene la información general de los distintos municipios de la isla representada mediante distintas visualizaciones, mientras que el comparador de estaciones sirve para hacer un análisis más detallado.

Figura 5.20: Menú de los datos de temperatura

5.2.4 Páginas de vista general

Para el desarrollo de los *dashboards* se decidió dividirlo por medidas por eso se cargaron en archivos distintos. Esto quiere decir que cada tipo de medida tendrá sus respectivas visualizaciones. Dentro de cada medida se diseñaron dos tipos de *dashboard*. Uno que muestra la información general del municipio a lo largo del histórico disponible, que es el que se explicará en este apartado, y un comparador de estaciones que se explicará más adelante su fin.

Temperatura

En primer lugar se explicará el diseño del *dashboard* de información general de la temperatura que se puede ver en la Figura 5.21. Los filtros disponibles son el municipio y el tipo de métrica, ya que en los datos de temperatura están disponibles la media, el máximo y el mínimo diario.

Aprovechando que la tabla que contiene la información de las estaciones tiene las columnas de coordenadas geográficas, se implementó una visualización de mapa. Esto puede ser útil para poder ubicar las distintas estaciones meteorológicas en el mapa de un municipio en concreto. También es interesante saber su posición geográfica ya que en algunos municipios las estaciones están relativamente separadas por lo que es interesante saber la ubicación de cada una para poder entender las diferencias en los registros entre unas y otras. A mayores de ubicar las estaciones en el mapa, se muestra su temperatura media anual.

También se implementó una matriz de correlación entre las distintas estaciones del municipio. Esta visualización permite saber como de relacionadas están las curvas de temperaturas de las estaciones que se ven en el mapa. Debido a la limitación que se comentó con anterioridad de PowerBI, se añadió un filtro donde debe indicarse el mismo municipio, ya que el filtro de municipio que filtra el *dashboard* en general, únicamente filtra una de las columnas de la matriz de correlación cargada en la herramienta.

Uno de los principales objetivos de representación era mostrar el histórico de registros tomados por los sensores. Para representar esto se creó una visualización en forma de serie temporal de los registros diarios de cada uno de las estaciones de los municipios. Si se desea ver la curva de una estación en concreto, se puede visualizar haciendo *click* en la estación que se desea ver en la leyenda de la propia gráfica o en el mapa.

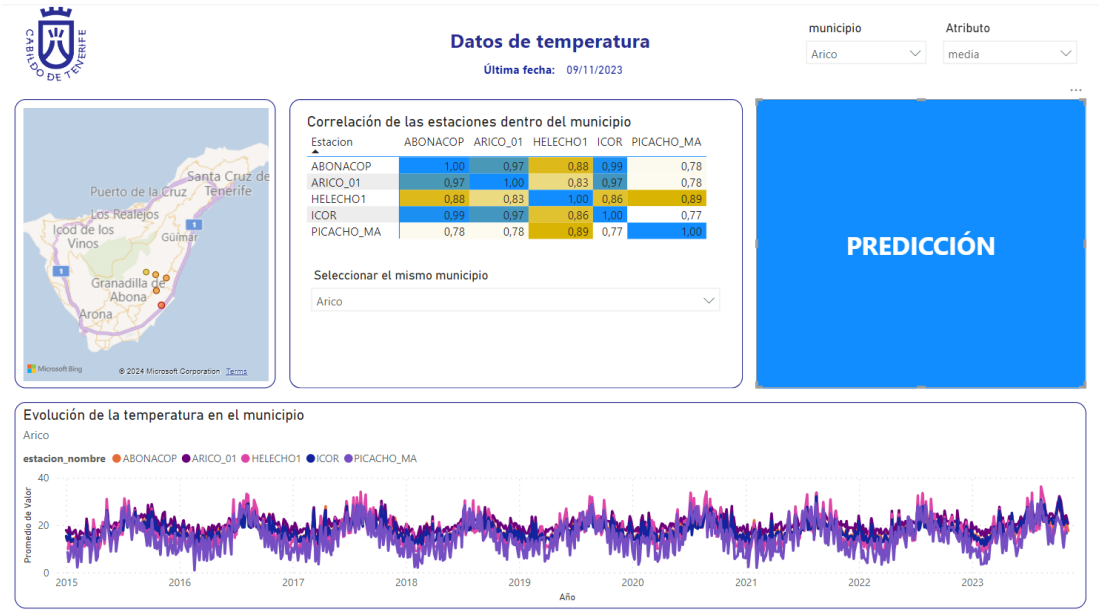


Figura 5.21: Página de vista general de la temperatura

La visualización de predicción no se explicará en este apartado, sino que se detallará en la sección de integración de los modelos en PowerBI 5.4. En esta sección se mostrará la implementación de la visualización y la vista final de esta página con el gráfico integrado.

Precipitación

En el caso de la página de las precipitaciones se siguió una estructura similar (Figura 5.22). En primer lugar se muestran las estaciones y su media de precipitación anual. Al igual que la temperatura también se muestra la evolución de la precipitación diaria del municipio. La página también cuenta con la precipitación anual media, en la que llama la atención la diferencia de 2023 con respecto al resto de años. En la siguiente sección se detalla más en profundidad los efectos de esta diferencia en las precipitaciones. En esta página no se incluyó la correlación. La correlación suele ser más relevante cuando se busca entender relaciones lineales entre variables. En el caso de la temperatura, es más probable que existan patrones lineales, ya que las estaciones climáticas tienden a experimentar cambios en conjunto.

Al igual que con la temperatura, se implementará un KPI que indicará la probabilidad de lluvia predicha por los modelos implementados posteriormente. Está visualización se mostrará en la sección mencionada anteriormente, donde se explica la creación y la integración de esta visualización.

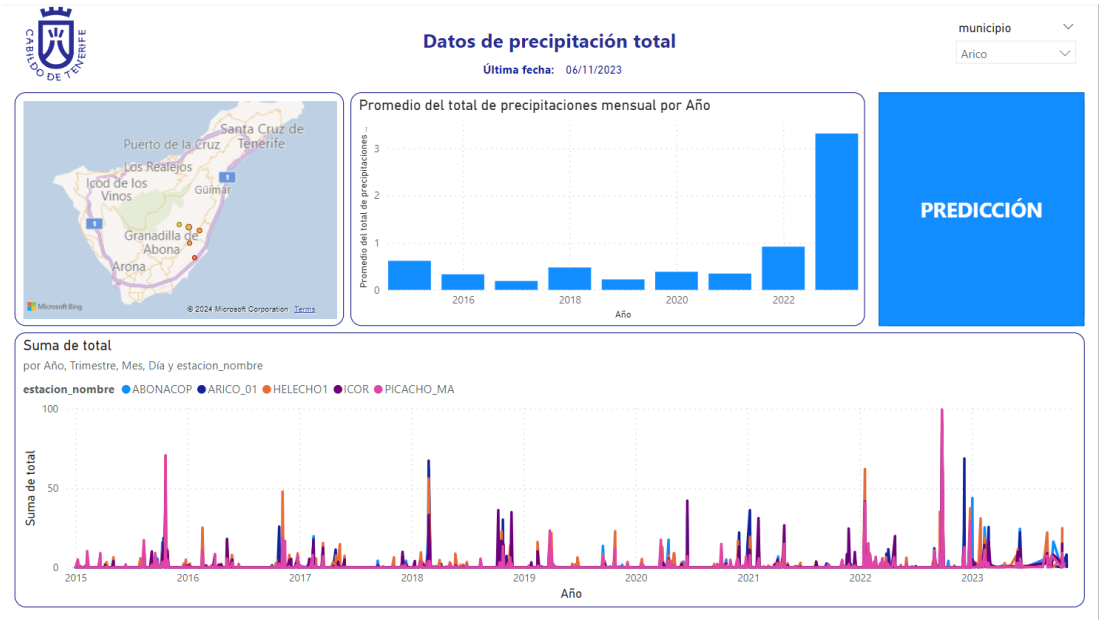


Figura 5.22: Página de vista general de las precipitaciones

Radiación solar, Humedad relativa y Velocidad del viento

Los tres *dashboards* para estas medidas son muy similares por lo que se explican en conjunto en este apartado. En la Figura 5.23 se puede observar la página de los datos de la radiación solar máxima. En el la sección A.1 del Apéndice A se incluyen las páginas que muestran los datos de humedad relativa y la velocidad del viento. En primer lugar no contendrán ningún tipo de predicción. Esta predicción que tendrán la temperatura y la precipitación se remplazó por una evolución mensual del municipio en cuestión. Al igual que en las páginas de temperatura y precipitación, todas estas páginas cuentan con un mapa con las estaciones ubicadas con su valor medio histórico y la evolución diaria de cada una de las estaciones del municipio a consultar.

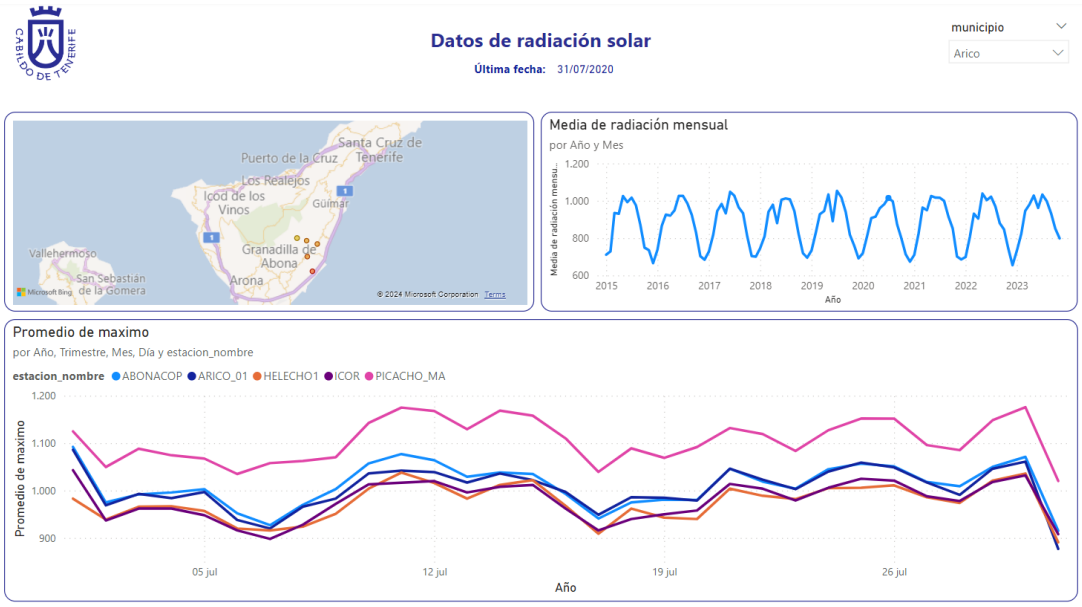


Figura 5.23: Página de vista general de la radiación solar

5.2.5 Comparador de estaciones

La páginas explicadas anteriormente son de gran utilidad para conocer la información general de cada uno de los municipios. También puede existir la necesidad de hacer un análisis más fino. Por ejemplo puede ser de interés analizar un mismo tramo temporal entre dos estaciones en concreto. En la Figura 5.24 se puede ver como se compara la estación ARICO_01 con ICOR en el mes de enero de 2015. También puede ser interesante comparar una misma estación en tramos temporales distintos para ver qué cambios se dieron de un periodo a otro. En la Figura 5.25 se puede ver la comparación de la estación ARICO_01 en distintas fechas. Para cubrir esta necesidad de análisis más concreto se diseñó un comparador de estaciones para cada una de las medidas. Este comparador se podrá filtrar por tramo temporal dentro del histórico de datos y por los municipios y estaciones que se quieren comparar.

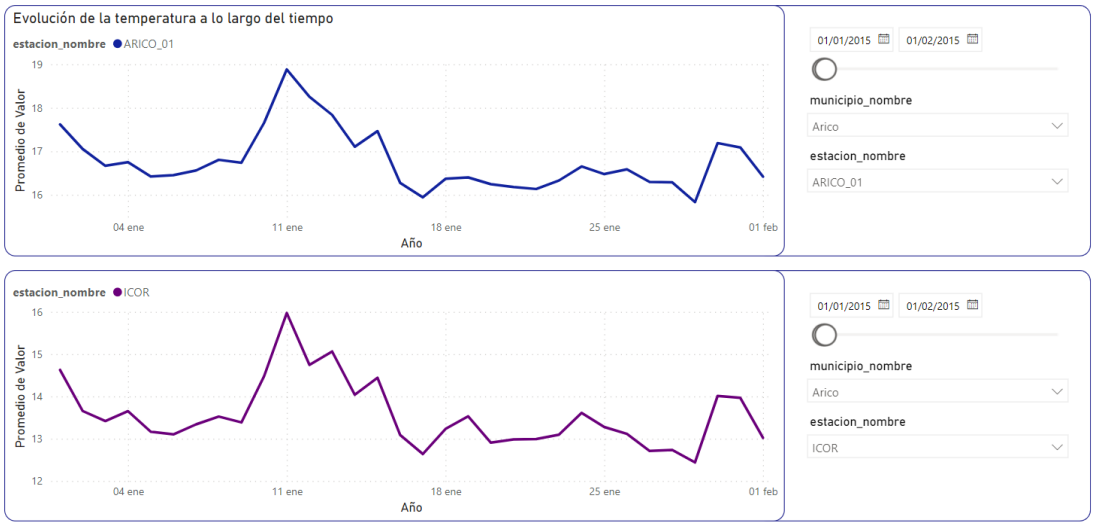


Figura 5.24: Distintas estaciones, mismo tramo temporal

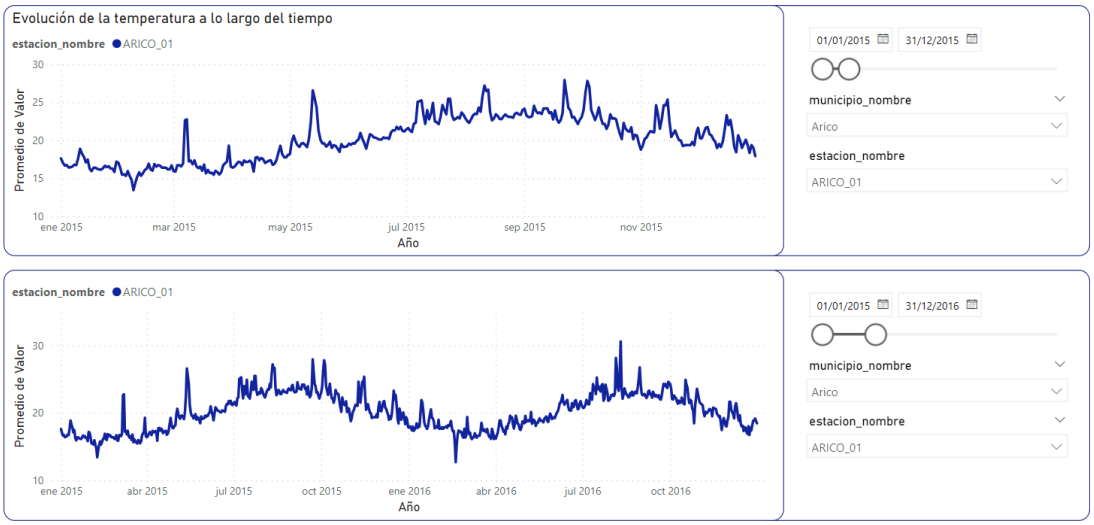


Figura 5.25: Misma estación, distintos tramos temporales

5.3 Diseño de modelos de predicción

En esta sección se describirá todo el proceso de diseño de estos modelos con los que complementar el *dashboard*. En primer lugar se describirán las medidas escogidas para predecir, seguido de un análisis exploratorio de los datos, finalizando con el diseño de los modelos en cuestión.

5.3.1 Variables de interés a predecir

Después de haber extraído y tratado los datos de origen se cuenta con 5 tipos distintos de medidas reportadas por los sensores: temperatura, radiación solar, velocidad del viento, humedad relativa y precipitación. Tanto la temperatura, como la humedad relativa y la velocidad del viento, cuentan con las métricas medias, máximas y mínimas diarias. En el caso de la radiación solar únicamente se cuenta con la radiación máxima diaria y, en el caso de la precipitación, únicamente se cuenta con la precipitación total del día, almacenada en la columna *total* creada para esta medida.

La temperatura media diaria y la precipitación total diaria, junto a la velocidad del viento son las tres medidas con más interés general (Figura 5.26). La temperatura media diaria y la precipitación total diaria se consideraron las dos mas relevantes, por lo que serán las variables de interés a predecir. La velocidad del viento se decidió excluirla dentro del desarrollo de modelos predictivos debido a tiempos de desarrollo del proyecto. La estimación de la velocidad del viento y del resto de medidas que no se predicen en este trabajo y se podría abordar dentro de un trabajo futuro como continuación a este proyecto.

Figura 5.26: Medidas que muestra Google cuando se busca el tiempo de un lugar

Cabe destacar ya de entrada, que la capacidad predictiva de los modelos está limitada

por la cantidad de características de las que se dispone (6 medidas tomadas desde sensores terrestres). Este proyecto, no cuenta con información recogida por radares satelitales, así como cuentan con ella distintas herramientas de predicción meteorológica muy utilizadas en la actualidad, como pueden ser AEMET o Meteogalicia. Por tanto, no se busca competir contra estas herramientas, sino que se busca ofrecer funcionalidades de predicción que puedan ser útiles y que se integren dentro de un análisis global.

5.3.2 Análisis exploratorio

Antes de pasar al diseño de los modelos de predicción es necesario hacer un primer análisis exploratorio de los datos. Realizar un análisis exploratorio antes de diseñar modelos de predicción es importante ya que permite comprender la naturaleza de los datos, identificar patrones y outliers, seleccionar variables relevantes, y entender las relaciones entre estas variables. Esto ayuda a mejorar la calidad del modelo al reducir el riesgo de *overfitting* o *underfitting*, optimizando así la capacidad predictiva y la interpretación de los resultados. La parte de análisis más básica ya se hizo en pasos anteriores, como es la limpieza y transformación de los datos. Los objetivos de este análisis exploratorio son objetivos más concretos destinados al cálculo de las predicciones, y estos no se cubren en los *dashboards* diseñados anteriormente. Se amplía la información contenida en los cuadros de mandos con gráficas y resultados numéricos más específicos.

En este proyecto, los datos de estudio son series temporales, lo que quiere decir que representan la evolución de una variable o conjunto de variables a lo largo del tiempo. Las series temporales se caracterizan por estar organizadas secuencialmente, con puntos de datos registrados en intervalos temporales regulares. Este enfoque permite analizar patrones, tendencias y ciclos que pueden surgir en el comportamiento de la variable en cuestión.

En la Figura 5.27 se puede ver la representación de las curvas de temperatura a lo largo del 2022, donde cada curva representa la evolución de una estación meteorológica. En esta figura se pueden ver algunos patrones comunes en las curvas. Por ejemplo, se pueden observar picos en días concretos que afectan a todas las estaciones, pero la evolución de estas tienen mucha variabilidad, por lo que, además de tener un gran número de estaciones, esta visualización no permite hacer un análisis muy fino.

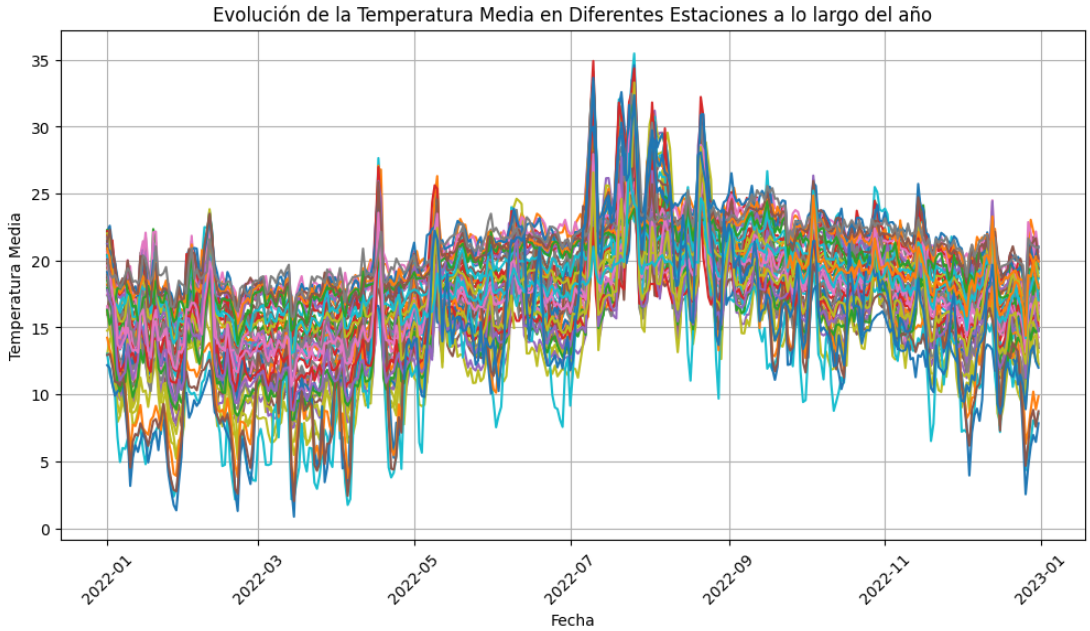


Figura 5.27: Evolución de la temperatura media a lo largo de 2022

Análisis de correlación entre variables

En primer lugar es de interés saber cómo de relacionadas están las distintas medidas existentes. Para ello, se calculó la matriz de correlación entre las distintas variables. Medir las correlaciones ayuda a identificar la influencia relativa de cada variable en la predicción, evitando multicolinealidad. Además, permite seleccionar características relevantes para el modelo, mejorando la eficacia predictiva. Con una comprensión sólida de las correlaciones, se pueden ajustar modelos más robustos y confiables, optimizando la capacidad de prever resultados futuros.

	Temperatura	Humedad	Precipitacion	Radiacion	Viento
Temperatura	1.000000	-0.323237	-0.159369	0.270004	-0.204167
Humedad	-0.323237	1.000000	0.313040	-0.042686	-0.178242
Precipitacion	-0.159369	0.313040	1.000000	-0.370376	0.088676
Radiacion	0.270004	-0.042686	-0.370376	1.000000	0.054181
Viento	-0.204167	-0.178242	0.088676	0.054181	1.000000

Figura 5.28: Correlación entre las variables

Analizando la matriz de correlación de la Figura 5.28, para la variable de temperatura, existe una correlación negativa no muy alta con la humedad, esto es normal ya que cuanto más se calienta el aire, disminuye la humedad relativa. También existe correlación negativa con el viento, que este generalmente hace que disminuya la temperatura ambiente. Por último existe una correlación positiva con la radiación, lo que es lógico ya que cuando los días están despejados el sol calienta más el ambiente. A pesar de que existan correlaciones, estas no son correlaciones muy fuertes, por lo que se considera que esta significancia es baja y se optó por no utilizar estas variables extra en el modelo final. A pesar de eso, posteriormente se mostrará dentro de la predicción con redes neuronales recurrentes, una aproximación de predicción utilizando todas las variables para confirmar esta conclusión.

En cuanto a la variable de precipitación nos encontramos con una situación similar. Una correlación positiva con la humedad ya que las precipitaciones aumentan la humedad relativa, y una correlación negativa con la radiación, ya que cuando llueve el cielo esta cubierto por las nubes, lo que hace que la radiación disminuya.

Análisis de autocorrelaciones

Algunos modelos, como las redes neuronales recurrentes, permiten escoger un número determinado de valores anteriores para calcular la predicción, por lo que nos interesa que estos valores anteriores tengan cierta correlación con el valor a predecir. Este tipo de correlación se denomina la autocorrelación. Las correlaciones entre variables pueden ser útiles a la hora de incluir o no ciertas variables en el modelo en función de sus correlaciones con la variable de interés. Como se trata de series temporales, es interesante conocer las autocorrelaciones de las variables de interés. La autocorrelación es una medida estadística que evalúa la relación lineal entre valores de una serie temporal y sus propios valores pasados. En otras palabras, indica la similitud entre observaciones en momentos sucesivos. Un coeficiente de autocorrelación cercano a 1 sugiere una fuerte correlación positiva, mientras que cercano a -1 indica una fuerte correlación negativa. Un valor cercano a 0 señala una baja autocorrelación.

En el caso de la temperatura (Figura 5.29), encontramos una correlación muy alta entre el valor actual y sus valores pasados. En la gráfica se puede visualizar hasta el retardo 35, y la autocorrelación hasta ese retardo no baja de 0,5. Esto puede ser de gran ayuda a la hora de escoger cuantas secuencias pasadas deben entrar al modelo, siempre y cuando este lo permita, ya que como se comentó antes, en el caso de las redes neuronales recurrentes se debe escoger el tamaño de las entradas que serán la entrada del modelo.

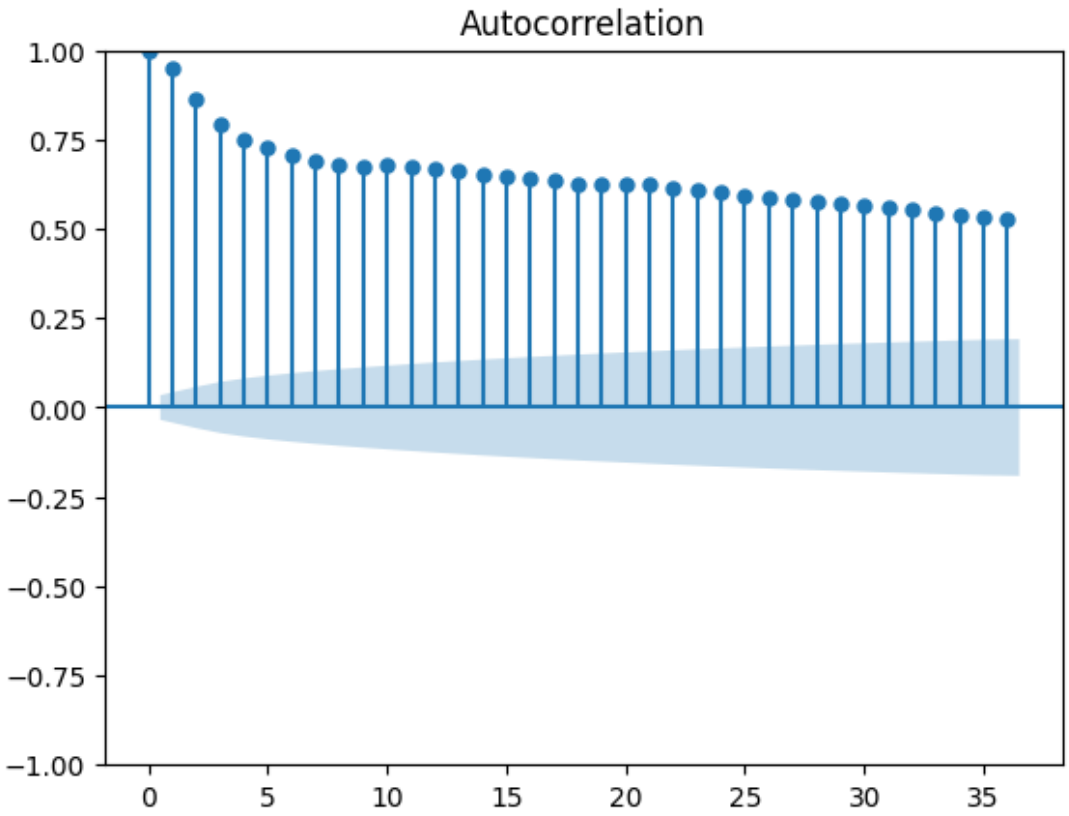


Figura 5.29: Autocorrelación de la temperatura

Como se puede ver en la Figura 5.30, la autocorrelación de la precipitación es un tanto diferente . No se aprecia una autocorrelación tan fuerte como la que presenta la temperatura. Se puede apreciar que a partir del retardo 5 la autocorrelación está en torno a 0. De igual manera que en el caso de la temperatura, sirve de guía para saber cuántas secuencias pasadas introducirle al modelo si este lo permite, por lo que si se utiliza únicamente la propia serie temporal de precipitación para calcular valores futuros, no se debería pasar del retardo 5 o 6, ya que a partir de esos retardos la autocorrelación es prácticamente nula. Tendría sentido no utilizar un retardo superior a 2, ya que a partir de este valor la autocorrelación baja a 0,10 aproximadamente. En todo caso, a partir del retardo 5, la autocorrelación es tan baja que no tendría sentido considerarla.

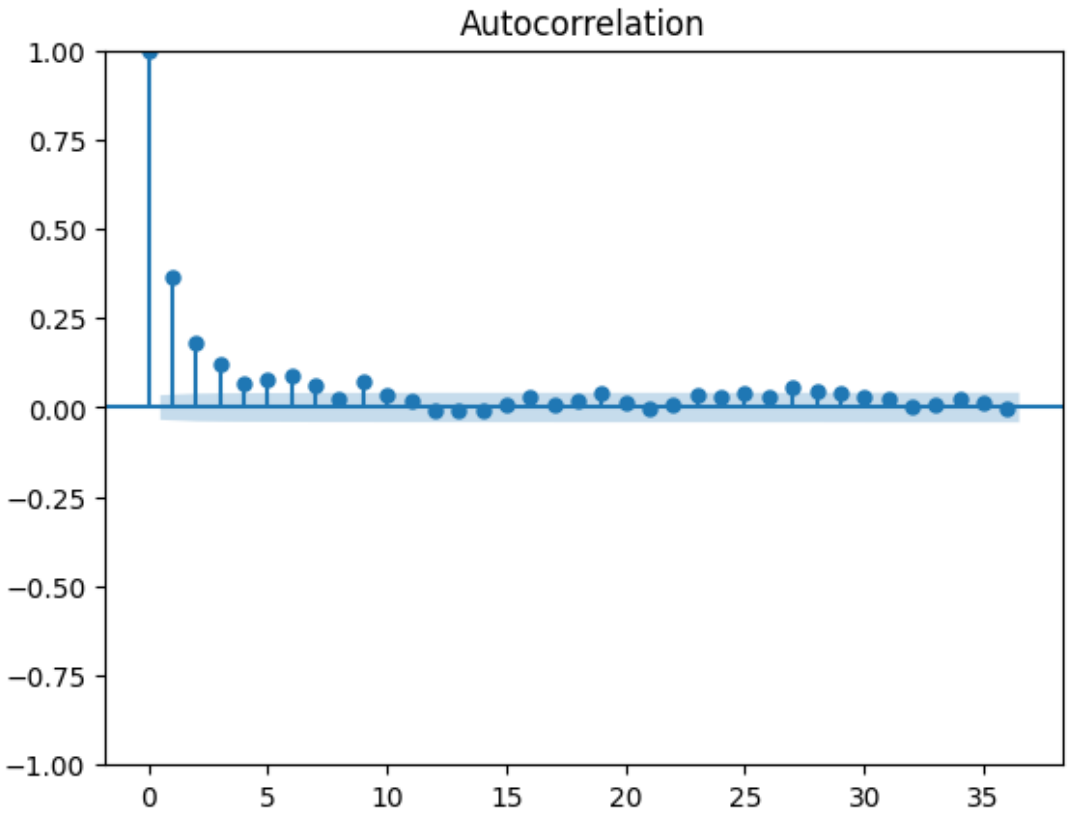
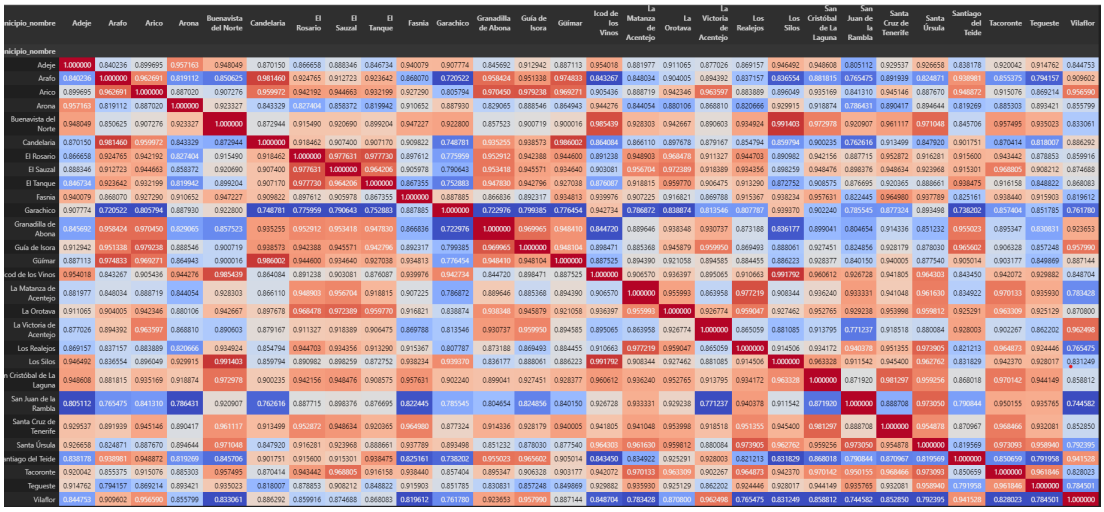


Figura 5.30: Autocorrelación de la precipitación

Correlación entre municipios

A continuación se analizó la correlación entre los municipios, la cual se puede ver en la matriz de la Figura 5.31. Los datos están distribuidos por estaciones, que estas se pueden agrupar en municipios. De entrada sería interesante agrupar las estaciones por municipios para no tener que ajustar 62 modelos distintos. Aun así hay un total de 28 municipios distintos, y habría que ajustar de igual forma una gran cantidad de modelos distintos. Dado que se trata de un territorio con un tamaño reducido, puede ser interesante ajustar un modelo global promediando todas las estaciones, que sirva para predecir la temperatura del municipio en cuestión. Esto es de interés debido al coste computacional y a las horas de implementación que se deben dedicar a construir 62 modelos diferentes, sin esperar una gran mejora con respecto a utilizar uno global. Viendo las correlaciones entre los distintos municipios se puede saber si esta aproximación podría ser válida.



nicipio_nombre	Adeje	Arafo	Arico	Arona	Buenavista del Norte	Candelaria	El Rosario	El Sauzal	El Tanque	Fasnia	Garachico	Granadilla de Abona	Guía de Isora	Güímar	Icod de los Vinos	La Matanza de Acentejo	La Orotava	La Victoria de Acentejo	Los Realejos	Los Silos	San Cristóbal de La Laguna	San Juan de la Rambla	Santa Cruz de Tenerife	Santa Úrsula	Santiago del Teide	Tacoronte	Tegueste	Vilaflor
nicipio_nombre																												
Adeje	1.000000	0.840236	0.899695	0.957163	0.948049	0.870150	0.866658	0.888346	0.846734	0.940079	0.907774	0.845692	0.912942	0.887113	0.954018	0.881977	0.911065	0.877026	0.869157	0.946492	0.946808	0.805112	0.929537	0.926658	0.838178	0.920042	0.914762	0.844753
Arafo	0.840236	1.000000	0.962691	0.819112	0.850625	0.981460	0.924765	0.912723	0.923642	0.868070	0.720522	0.958424	0.951338	0.974833	0.843267	0.848034	0.904005	0.894392	0.837157	0.836554	0.881815	0.765475	0.891939	0.824871	0.938981	0.855375	0.794157	0.909602
Arico	0.899695	0.962691	1.000000	0.887020	0.907276	0.959972	0.942192	0.944663	0.932199	0.927290	0.805794	0.970450	0.979238	0.969271	0.905436	0.888719	0.942346	0.963597	0.883889	0.896049	0.935169	0.841310	0.945146	0.887670	0.948872	0.915076	0.869214	0.956590
Arona	0.957163	0.819112	0.887020	1.000000	0.923327	0.843329	0.827497	0.858372	0.819942	0.910652	0.887930	0.829065	0.888546	0.864943	0.944276	0.844054	0.880106	0.866810	0.820666	0.929915	0.918874	0.786431	0.890417	0.894644	0.819269	0.885303	0.893421	0.855799
Buenavista del Norte	0.948049	0.850625	0.907276	0.923327	1.000000	0.872944	0.915490	0.920690	0.899204	0.947227	0.922800	0.857523	0.900719	0.900016	0.985439	0.928303	0.942667	0.890603	0.934924	0.991403	0.972978	0.920907	0.961117	0.971048	0.845706	0.957495	0.935023	0.833061
Candelaria	0.870150	0.981460	0.959972	0.843329	0.872944	1.000000	0.918462	0.907400	0.907170	0.909822	0.748781	0.935255	0.938573	0.986002	0.864084	0.866110	0.897678	0.879167	0.854794	0.959794	0.900235	0.762616	0.913499	0.847920	0.901751	0.870414	0.818007	0.886292
El Rosario	0.866658	0.924765	0.942192	0.827404	0.915490	0.918462	1.000000	0.977631	0.977730	0.897612	0.775959	0.952912	0.942388	0.944600	0.891238	0.948903	0.968478	0.911327	0.944703	0.890982	0.942156	0.887715	0.952872	0.916281	0.915600	0.943442	0.878853	0.859916
El Sauzal	0.888346	0.912723	0.944663	0.858372	0.920690	0.907400	0.977631	1.000000	0.964206	0.905978	0.790643	0.953418	0.945571	0.934640	0.903081	0.956704	0.972389	0.918389	0.934356	0.898259	0.948476	0.898376	0.948634	0.923968	0.915301	0.968805	0.908212	0.874688
El Tanque	0.846734	0.923642	0.932199	0.819942	0.899204	0.907170	0.977730	0.964206	1.000000	0.867355	0.752883	0.947830	0.942796	0.927038	0.876087	0.918815	0.959770	0.906475	0.913290	0.872752	0.908575	0.876695	0.920365	0.889661	0.938475	0.916158	0.848822	0.868083
Fasnia	0.940079	0.868070	0.927290	0.910652	0.947227	0.909822	0.897612	0.905978	0.867355	1.000000	0.887885	0.866836	0.892317	0.934813	0.939976	0.907225	0.916821	0.869788	0.915367	0.938234	0.957631	0.822445	0.964980	0.937789	0.825161	0.938440	0.915903	0.819612
Garachico	0.907774	0.720522	0.805794	0.887930	0.922800	0.748781	0.775959	0.790643	0.752883	0.887885	1.000000	0.722976	0.799385	0.776454	0.942734	0.786872	0.838874	0.813546	0.807787	0.939370	0.902240	0.785545	0.877324	0.893498	0.738202	0.857404	0.851785	0.761780
Granadilla de Abona	0.845692	0.958424	0.970450	0.829065	0.857523	0.935255	0.952912	0.953418	0.947830	0.866836	0.722976	1.000000	0.969965	0.948410	0.844720	0.889646	0.938348	0.930737	0.873188	0.836177	0.899041	0.804654	0.914336	0.851232	0.955023	0.895347	0.830831	0.923653
Guía de Isora	0.912942	0.951338	0.979238	0.888546	0.900719	0.938573	0.942388	0.945571	0.942796	0.892317	0.799385	0.969965	1.000000	0.948104	0.898471	0.885360	0.945879	0.959950	0.869493	0.888061	0.927451	0.824856	0.928179	0.878030	0.965602	0.906328	0.857248	0.957990
Güímar	0.887113	0.974833	0.969271	0.864943	0.900016	0.986002	0.944600	0.934640	0.927038	0.934813	0.776454	0.948410	0.948104	1.000000	0.887525	0.894390	0.921058	0.894585	0.884455	0.886223	0.928377	0.840150	0.940005	0.877540	0.905014	0.903177	0.849869	0.887144
icod de los Vinos	0.954018	0.843267	0.905436	0.944276	0.985439	0.864084	0.891238	0.903081	0.876087	0.939976	0.942734	0.844720	0.898471	0.887525	1.000000	0.906570	0.936397	0.895065	0.910663	0.991792	0.960612	0.926728	0.941805	0.964303	0.843450	0.942072	0.929882	0.848704
La Matanza de Acentejo	0.881977	0.848034	0.888719	0.844054	0.928303	0.866110	0.948903	0.956704	0.918815	0.907225	0.786872	0.889646	0.885368	0.894390	0.906570	1.000000	0.955993	0.863958	0.977219	0.908344	0.936240	0.933331	0.941048	0.961630	0.834922	0.970133	0.935930	0.783428
La Orotava	0.911065	0.904005	0.942346	0.880106	0.942667	0.897678	0.968478	0.972389	0.959770	0.916821	0.838874	0.938348	0.945879	0.921058	0.936397	0.955993	1.000000	0.926774	0.959047	0.927462	0.952765	0.929238	0.953998	0.959812	0.925291	0.963309	0.925129	0.870800
La Victoria de Acentejo	0.877026	0.894392	0.963597	0.868810	0.890603	0.879167	0.911327	0.918389	0.906475	0.869788	0.813546	0.930737	0.959950	0.894585	0.895065	0.863958	0.926774	1.000000	0.865059	0.881085	0.913795	0.771237	0.918518	0.880084	0.928003	0.902267	0.862202	0.962498
Los Realejos	0.869157	0.837157	0.883889	0.820666	0.934924	0.854794	0.944703	0.934356	0.913290	0.915367	0.807787	0.873188	0.869493	0.884455	0.910663	0.977219	0.959047	0.865059	1.000000	0.914506	0.934172	0.940378	0.951355	0.973905	0.821213	0.964073	0.924446	0.765475
Los Silos	0.946492	0.836554	0.896049	0.929915	0.991403	0.859794	0.890982	0.898259	0.872752	0.938234	0.939370	0.836177	0.888061	0.886223	0.991792	0.908344	0.927462	0.881085	0.914506	1.000000	0.963328	0.911542	0.945400	0.962762	0.831829	0.942370	0.928017	0.831249
n Cristóbal de La Laguna	0.946808	0.881815	0.935169	0.918874	0.972978	0.900235	0.942156	0.948476	0.908575	0.957631	0.902240	0.899041	0.927451	0.926377	0.960612	0.936240	0.952765	0.913795	0.934172	0.963328	1.000000	0.871920	0.981297	0.959256	0.868018	0.970142	0.944149	0.858812
San Juan de la Rambla	0.805112	0.765475	0.841310	0.786431	0.920907	0.762616	0.887715	0.898376	0.876695	0.822445	0.785545	0.804654	0.824856	0.840150	0.926728	0.933331	0.929238	0.771237	0.940378	0.911542	0.871920	1.000000	0.888708	0.973050	0.790844	0.950155	0.935765	0.744582
Santa Cruz de Tenerife	0.929537	0.891939	0.945146	0.890417	0.961117	0.913499	0.952872	0.948634	0.920365	0.964980	0.877324	0.914336	0.928179	0.940005	0.941805	0.941048	0.953998	0.918518	0.951355	0.945400	0.981297	0.888708	1.000000	0.954878	0.870967	0.968466	0.932081	0.852850
Santa Úrsula	0.926658	0.824871	0.887670	0.894644	0.971048	0.847920	0.916281	0.923968	0.889661	0.937789	0.893498	0.851232	0.878030	0.877540	0.963303	0.961630	0.959812	0.880084	0.973905	0.962762	0.959256	0.973050	0.954878	1.000000	0.819569	0.973093	0.958940	0.792393
ntiago del Teide	0.838178	0.938981	0.948872	0.819269	0.845706	0.901751	0.915600	0.915301	0.938475	0.825161	0.738202	0.955023	0.965602	0.905014	0.843450	0.834923	0.925291	0.928003	0.821213	0.831829	0.868018	0.790844	0.870967	0.819569	1.000000	0.850659	0.791958	0.941528
Tacoronte	0.920042	0.855375	0.915076	0.885303	0.957495	0.870414	0.943442	0.968805	0.916158	0.938440	0.857404	0.895347	0.906328	0.903177	0.942072	0.970133	0.963309	0.902267	0.964873	0.942370	0.970142	0.950155	0.968466	0.973093	0.850659	1.000000	0.961846	0.828023
Tegueste	0.914762	0.794157	0.869214	0.893421	0.935023	0.818007	0.878853	0.908212	0.848822	0.915903	0.851785	0.830831	0.857248	0.849869	0.929882	0.935930	0.925129	0.862202	0.924446	0.928017	0.944149	0.935765	0.932081	0.958940	0.791958	0.961846	1.000000	0.784501
Vilaflor	0.844753	0.909602	0.956590	0.855799	0.833061	0.886292	0.859916	0.874688	0.868083	0.819612	0.761780	0.923653	0.957990	0.887144	0.848704	0.783428	0.870800	0.962498	0.765475	0.831249	0.858812	0.744582	0.852850	0.792393	0.941528	0.828023	0.784501	1.000000

Figura 5.31: Correlación entre los municipios

Basándose en la matriz de correlaciones de la Figura 5.31, se puede observar una correlación alta entre prácticamente todos los municipios. La correlación mínima es de 0,72 y la correlación media es de 0,9. Vista esta alta correlación, se optó por entrenar un modelo global con los datos de todas las estaciones promediados. Si esta correlación no fuera significativa, se debería de ajustar un modelo al menos para cada municipio. Visto esto experimentar con modelo global para calcular predicciones de cada uno de los municipios puede ser una buena idea en términos de eficiencia.

Al existir una alta correlación entre los valores diarios de las distintas estaciones meteorológicas, estos se pueden promediar. Esta versión promediada se utilizará para testear la capacidad de los modelos, y si este modelo global funciona correctamente, se probará con las distintas estaciones meteorológicas.

Análisis de valores nulos

Se hizo un análisis de valores nulos de la curva histórica. Se encontraron 85 valores nulos en la variable del total de precipitación. Estos valores nulos se pueden interpretar como que esos días no llovió, pero como hay muchos valores que son 0, se puede interpretar también que hubo un error de medición y no se registró el valor. En este caso se utilizó un *KNN Imputing* para imputar esos valores faltantes. El *KNN Imputing* es un método de imputación de datos faltantes que utiliza el algoritmo K-Nearest Neighbors para estimar los valores ausentes basándose en los valores de vecinos cercanos en un conjunto de datos.

5.3.3 Predicción de la temperatura

En este apartado se explicará el desarrollo de las distintas aproximaciones hasta llegar a una aproximación final, la cual se implementará dentro de los *dashboards* presentados anteriormente. Se buscará hacer una predicción de la temperatura media diaria de los próximos 7 días, consiguiendo así complementar el *dashboard* con la evolución futura de una semana entera. Por cuestiones de tiempo se escogió la temperatura media pero sería de interés abordar en un trabajo futuro la predicción de la temperatura máxima y la mínima en ese rango temporal de una semana.

Se presentarán distintas aproximaciones, entre las cuales se encuentran un ARIMA, el Prophet Model de Meta y una Red neuronal recurrente. A mayores se hicieron otras pruebas como la predicción de la curva mensual anual con los modelos ARIMA o probar los propios métodos de predicción de PowerBI, pero se decidió no incluirlo en esta memoria. Los modelos de PowerBI están diseñados para integrarse directamente en los *dashboards*, por lo que no se pueden validar como los otros. Debido a esta dificultad de validación, este no se incluyó dentro de este trabajo. En el Apéndice A, se puede ver un ejemplo de predicción con el método nativo de PowerBI.

División de los datos

Antes de pasar con los modelos en cuestión, cabe destacar la división de los datos empleada como conjunto de entrenamiento y como conjunto de test. Se cuenta con un total de 9 años de histórico, con el 2023 hasta el 9 de noviembre que fue la fecha de la última ejecución del DAG utilizada para el desarrollo de este proyecto. Se decidió utilizar como conjunto de entrenamiento el rango de años de 2015 a 2022, utilizando los datos de 2023 como conjunto de test.

En la Figura 5.32 se puede ver la evolución de la curva, representando en azul los datos de la curva que se utilizarán para el entrenamiento, y en naranja para el test de los modelos. Se puede observar que 2023 es un año un tanto especial, ya que se aprecia que no sigue exactamente los mismos patrones que los años anteriores, por lo que es interesante para saber qué capacidad de adaptación tienen los modelos.

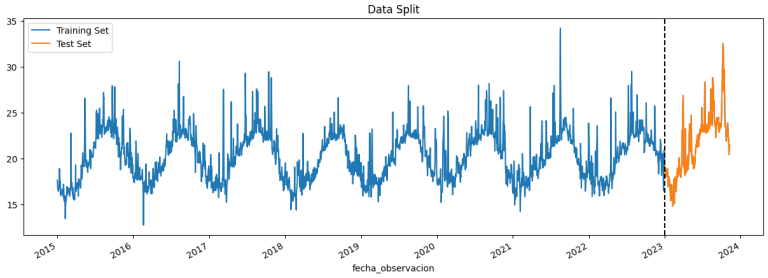


Figura 5.32: Análisis de valores nulos

ARIMA

En primer lugar antes de ajustar el modelo ARIMA se hizo una búsqueda de los parámetros p,d y q óptimos. El parámetro p es el número de términos autorregresivos (AR), d es orden de diferenciación y q Número de términos de media móvil (MA). Se hizo una validación cruzada probando probando todas las combinaciones posibles de p y q de 1 hasta 5 basándose en el criterio de información bayesiano (BIC). Como la curva no fue diferenciada el orden de diferenciación es 0. Una vez ajustados todos los modelos, el que obtuvo un BIC más bajo fue un ARIMA(1,0,1).

Una vez obtenidos los parámetros p,d y q del mejor modelo, este se ajustó de nuevo con los parámetros ganadores. En la Figura 5.33 se puede ver el ajuste del modelo en los datos de entrenamiento. El modelo obtuvo un RMSE de 1,076. Es un resultado bastante bueno dado que viendo la Figura 5.33, la curva azul representa los datos reales y la naranja las predicciones. Como se puede observar la curva estimada se ajusta casi a la perfección a los datos reales. Esto no es significante ya que la evaluación importante es la que se hace con el conjunto de test.

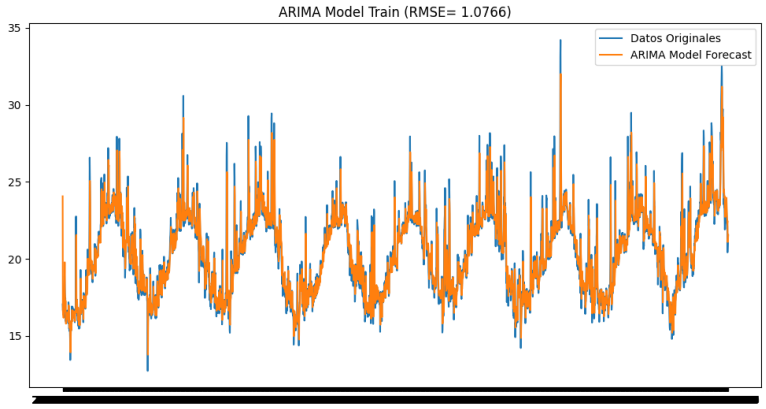


Figura 5.33: ARIMA(1,0,1) con los datos de entrenamiento

Una vez visto el ajuste en los datos de entrenamiento se probó el mismo con los datos de test. El modelo se probará de dos formas distintas. Una aproximación es intentar predecir toda la curva de test, teniendo que generar así un único archivo de predicciones. La otra aproximación de este modelo es calculando solo la predicción de los 7 días posteriores, teniendo que generar así, un modelo cada día que pase. En primer lugar se trató de predecir la curva entera de los datos de 2023 disponibles para testear. Esta prueba no fue muy satisfactoria. En la Figura 5.34 se puede observar que la linea naranja (los datos estimados), es constante, por lo que la capacidad de predicción de la curva entera es muy limitada.

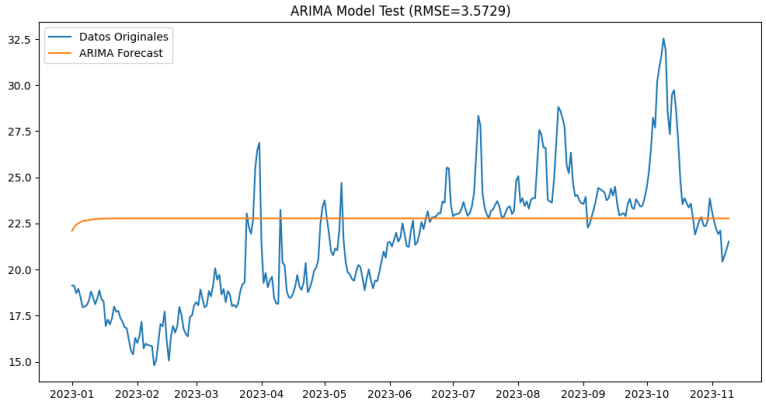


Figura 5.34: ARIMA(1,0,1) con los datos de test

A continuación, se hizo la segunda aproximación comentada anteriormente. Dado que el objetivo es predecir únicamente los 7 días, en vez de obtener los 7 días siguientes a partir de la curva de todo el año se decidió evaluar un ARIMA cada día para predecir únicamente los 7 días posteriores. Se implementó un bucle que ajusta un modelo nuevo por cada día de los datos de test. Es decir, cada día que pasa, se ajusta un modelo nuevo con los parámetros mencionados anteriormente, y se testea la predicción de los 7 días siguientes con los datos originales. Haciendo esto, se obtuvo un RMSE de 1,93.

En aproximaciones siguientes, se utilizará el RMSE de 1,93 que obtuvo el ARIMA en la segunda aproximación de test, como error a batir por los modelos.

Prophet Model

Prophet [20] es un modelo de pronóstico de series temporales desarrollado por Meta y abierto al público general. Diseñado para prever tendencias temporales, es especialmente efectivo para conjuntos de datos con patrones estacionales. Utiliza un enfoque aditivo que descompone la serie temporal en componentes como tendencia, estacionalidad y días festivos. Prophet es fácil de usar y requiere mínima intervención del usuario, adaptándose bien a datos de diversas industrias. Su capacidad para manejar *outliers* y su escalabilidad lo hacen

ampliamente utilizado en diversos sectores. Prophet permite indicar días festivos, pero en este caso no es muy útil ya que los días festivos no tienen ningún impacto en la temperatura, así como lo pueden tener en las ventas de un negocio.

A diferencia del modelo ARIMA, Prophet es un modelo más sencillo de implementar ya que no necesita ninguna validación cruzada de hiperparámetros.

El modelo se ajustó con los mismos datos de entrenamiento que en el caso anterior. En la Figura 5.35 los puntos son los valores reales y la franja azul son los intervalos de confianza de las predicciones. Se obtuvo un un RMSE de 1.42, superior al modelo ARIMA con los datos de entrenamiento. Esto se debe a que el modelo Prophet tiene en cuenta los outliers a la hora de ajustar el modelo, ya que viendo la Figura 5.35, el ajuste del modelo no esta apenas condicionado por los outliers, ya que estos no entran nunca dentro de las bandas de confianza.

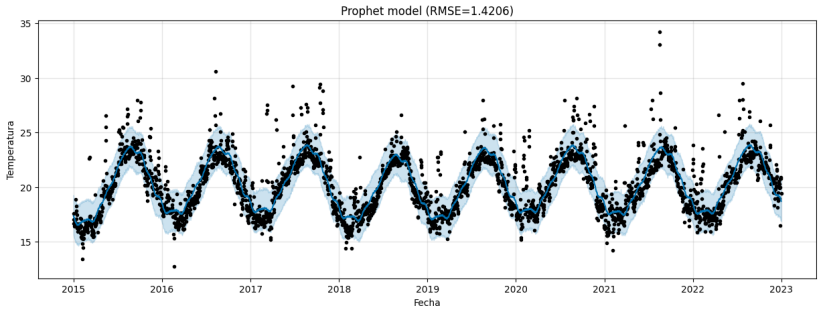


Figura 5.35: Prophet Model con los datos de entrenamiento

Al igual que con el ARIMA, ya que Prophet también permite hacer una predicción futura de toda la curva de 2023, de igual forma se hicieron dos aproximaciones de test.

La primera aproximación de predicción de toda la curva funcionó de una forma claramente superior a la predicción del ARIMA como se puede ver en la Figura 5.36. De igual forma el RMSE sigue siendo bastante alto, esto se debe a los datos extremos que el modelo no ajusta, por lo que estos aumentan el error medio de una manera considerable. Como se comentó anteriormente, 2023 se trató de un año atípico, teniendo más valores extremos que otros años.

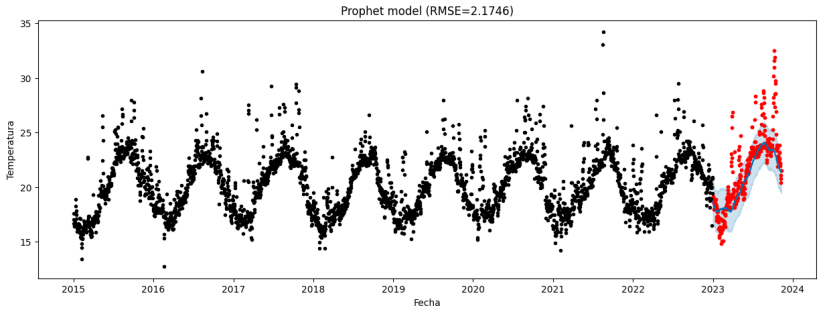


Figura 5.36: Prophet Model con los datos de test

Posteriormente se hizo la aproximación de test prediciendo únicamente los 7 días próximos. Ajustando un modelo Prophet en cada punto de los datos de 2023 con todo el histórico de datos, y prediciendo los 7 días posteriores, se obtuvo un RMSE de 1.91, ligeramente inferior al error del ARIMA que era 1.93.

Después de esta segunda aproximación, se puede considerar que el modelo Prophet funciona de mejor que el modelo ARIMA. Obtuvo una mejor ligera mejor segunda aproximación de test e incluso podría utilizarse la predicción de todo el año para no tener que calcular un nuevo modelo cada día, ya que si el valor a predecir no se trata de un valor muy extremo, el modelo generalmente hará una estimación más o menos acertada.

En este punto tenemos dos posibles modelos a utilizar con unas características similares, ya que su para uso en el *dashboard* en producción, habría que ajustar un nuevo modelo en cada municipio que se quiera consultar, funcionando el segundo modelo ligeramente mejor que el primero.

Red LSTM

La última aproximación realizada fue una red LSTM. Hay dos principales motivaciones para hacer uso de una red entrenada. En primer lugar, mejorar los errores vistos hasta ahora, el error a batir es 1.91 del modelo Prophet de Meta. Otra motivación es solucionar algunas limitaciones que tienen los dos modelos anteriores. En primer lugar tanto el modelo ARIMA como Prophet no se pueden almacenar, ya que sería necesario ajustar un modelo nuevo para cada día que pase. Otra limitación es que no se puede experimentar con variables extras para ver si estas aportan una mejora significativa en la predicción. Utilizando una red de neuronas, en primer lugar se podría entrenar utilizando los datos globales promediados y guardar los pesos del modelo, y posteriormente invocar a este modelo ya entrenado para hacer la predicción del municipio que se quiere consultar. En segundo lugar, se podrían evaluar las otras variables accesorias por si ofrecen alguna mejora en las predicciones.

Dentro de esta última aproximación, se hicieron dos pequeñas aproximaciones distintas:

- **Utilizando únicamente la variable de temperatura**. Esto mismo se hizo con el modelo ARIMA y Prophet.
- **Utilizando todas las variables disponibles**. Esto no se puede hacer con los modelos ARIMA y Prophet debido a sus limitaciones.

Antes de entrenar el modelo en cuestión fue necesario hacer un pequeño preprocesado en los datos, orientado específicamente al entrenamiento del modelo. En primer lugar se hizo un escalado en los datos. Los datos escalados [21] pueden ser más resistentes al ruido y a los valores atípicos. Al limitar los datos al rango 0-1, se reduce la amplificación de efectos

extremos que podrían ocurrir. El método de escalado utilizado fue una normalización Min-Max. Véase a continuación la fórmula del escalado:

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

Una vez se obtuvieron los datos normalizados, se procedió a construir las secuencias de entrada de la red. Como en el análisis exploratorio se observó una alta autocorrelación, se decidió construir secuencias de tamaño 15. Como lo que se quiere predecir los 7 próximos días, las secuencias se construyen siguiendo una estructura "Muchos a Muchos", es decir, a partir de los 15 días anteriores se predicen los siguientes 7 días.

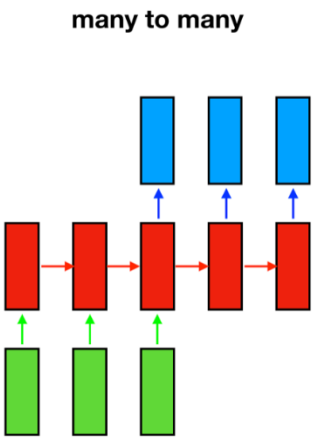


Figura 5.37: Secuencias "Muchos a Muchos"

Una vez se obtuvieron las secuencias construidas y normalizadas, se procedió a definir el modelo. El modelo se definió en las siguientes capas que se pueden ver en la Figura 5.38. En este modelo se incluyó una capa de entrada de 15 neuronas, en base a la decisión de utilizar los 15 días anteriores, y una salida de 7 neuronas en base a la predicción deseada . Para entrenar el modelo se escogió como optimizador un ADAM [22] y como medida de error el MSE (Error Cuadrático Medio). A mayores, para intentar mitigar el sobreentrenamiento se definió un *Early Stopping* utilizando el error del conjunto de validación y una paciencia de 10 iteraciones si el modelo no mejora su error. Como se puede observar en la Figura 5.39, el modelo paró de entrenar por parada temprana a las 74 iteraciones .

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 15, 128)	66560
dropout (Dropout)	(None, 15, 128)	0
lstm_1 (LSTM)	(None, 15, 128)	131584
dropout_1 (Dropout)	(None, 15, 128)	0
lstm_2 (LSTM)	(None, 15, 64)	49408
dropout_2 (Dropout)	(None, 15, 64)	0
lstm_3 (LSTM)	(None, 15, 64)	33024
dropout_3 (Dropout)	(None, 15, 64)	0
lstm_4 (LSTM)	(None, 32)	12416
dropout_4 (Dropout)	(None, 32)	0
dense (Dense)	(None, 7)	231

...
Total params: 293223 (1.12 MB)
Trainable params: 293223 (1.12 MB)
Non-trainable params: 0 (0.00 Byte)

Figura 5.38: Modelo utilizado

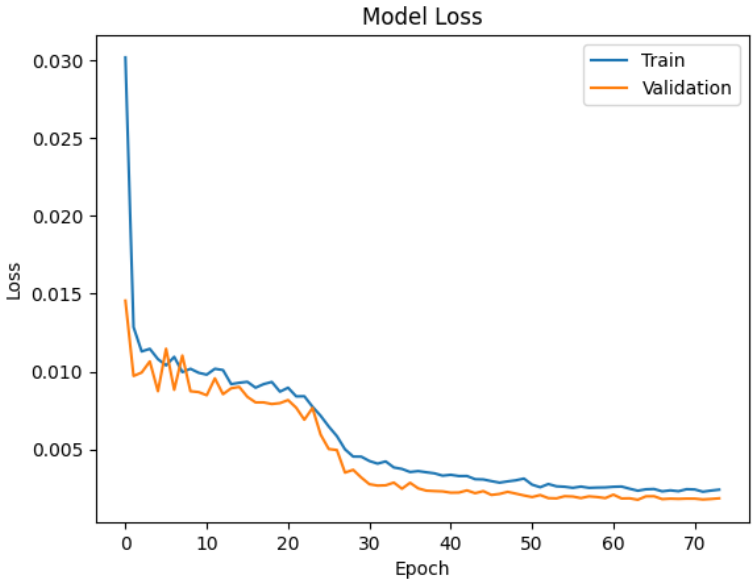


Figura 5.39: Evolución del entrenamiento

Una vez el modelo esta entrenado, se procedió a evaluarlo con los datos de test. Para

poder medir el RMSE entre los valores predichos y los valores reales fue necesario hacer una transformación inversa al escalado que se hizo sobre las secuencias, ya que las predicciones calculadas por la red están en el rango 0-1 tal como los datos de entrada. Una vez hecho el escalado inverso, se calculó el RMSE de los datos predichos. Esta aproximación obtuvo un error de 1.82 mejorando el obtenido por el modelo Prophet.

Prueba con todas las variables

Como las redes neuronales, a diferencia de los anteriores dos modelos de series temporales, permiten incluir más características de entrada. Aprovechando que el resto de variables tenían cierta correlación con la variable de interés, se decidió hacer una última aproximación utilizando el resto de variables a mayores de la temperatura en cuestión para calcular las predicciones. Se utilizo el mismo modelo y se entrenó de la misma forma. Esta aproximación obtuvo un RMSE de 1.86, por lo que no consiguió mejorar la primera aproximación dentro de esta iteración con redes neuronales. En la Tabla 5.3 se pueden ver los resultados de ambas aproximaciones con redes neuronales.

Número de características	RMSE
1 (temperatura media)	1.82
5 (temperatura media + resto)	1.86

Tabla 5.3: Comparación entre las dos aproximaciones con redes neuronales recurrentes

Modelo escogido

Una vez probadas distintas aproximaciones para tratar de predecir 7 días a futuro de la temperatura media diaria, el modelo que ofreció un mejor resultado fue la red neuronal recurrente con celdas LSTM, utilizando como entrada secuencias de 15 días de temperatura media.

Además de la superioridad del modelo en cuanto a resultados con los datos de test, como se puede observar en la tabla 5.4, cabe destacar que el uso de un modelo global es muy beneficioso en cuanto al rendimiento del *dashboard*. Los modelos que se utilicen van a ejecutarse por código Python programado en PowerBI. En el caso del modelo ARIMA y el modelo Prophet, cada vez que se quiera consultar un municipio distinto, se tendrá que ajustar un nuevo ARIMA (o Prophet) y calcular sus predicciones. El código Python en PowerBI tarda más en ejecutarse que cuando se ejecuta desde la IDE de programación, por lo que optimizar el código se vuelve importante. Si se utiliza un modelo global como en el caso de la red neuronal

recurrente, basta con almacenar los pesos del modelo y cargarlos cada vez que se quieran hacer nuevas predicciones, obteniendo así un código más eficiente y unos tiempos de ejecución más rápidos.

Modelo	RMSE
ARIMA	1.93
Prophet	1.91
RNN	1.86

Tabla 5.4: Comparación de resultados de las distintas aproximaciones

5.3.4 Predicción de las precipitaciones

Aprovechando que anteriormente ya se definió una arquitectura de un modelo neuronal recurrente, esta se reutilizó para la predicción de las precipitaciones. Al tratarse de un problema de clasificación no se probaron los modelos ARIMA y Prophet debido a que estos no permiten abarcar este tipo de problemas. Aunque la mayor parte de la arquitectura del modelo se puede reutilizar, se tuvieron que hacer ciertos ajustes para adaptarlo al problema en cuestión. En la Figura 5.40 se puede observar el modelo utilizado.

En primer lugar se cambió el número de secuencias de entrada, para la predicción de la temperatura se utilizaron los 15 días anteriores mientras que para este problema, basándose en los resultados obtenidos en la matriz de autocorrelación, se utilizaron los 5 últimos días ya que eran los que mayor correlación tenían.

Se hizo un ajuste en la última capa del modelo, ya que se puso una capa de activación sigmoide, para que el resultado de esta sea la probabilidad de pertenecer a la clase 1. El optimizador utilizado fue el mismo pero se cambió la función de pérdida, ya que en este caso se utiliza la entropía cruzada binaria, mostrando también en el entrenamiento la precisión del modelo.

```
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_5 (LSTM)               (None, 5, 128)            71680

 dropout_5 (Dropout)         (None, 5, 128)            0

 lstm_6 (LSTM)               (None, 5, 128)            131584

 dropout_6 (Dropout)         (None, 5, 128)            0

 lstm_7 (LSTM)               (None, 5, 64)             49408

 dropout_7 (Dropout)         (None, 5, 64)             0

 lstm_8 (LSTM)               (None, 5, 64)             33024

 dropout_8 (Dropout)         (None, 5, 64)             0

 lstm_9 (LSTM)               (None, 32)                12416

 dropout_9 (Dropout)         (None, 32)                0

 dense_1 (Dense)             (None, 1)                 33
...
Total params: 298145 (1.14 MB)
Trainable params: 298145 (1.14 MB)
Non-trainable params: 0 (0.00 Byte)
```

Figura 5.40: Modelo utilizado para la predicción de las precipitaciones

Preprocesado y división de los datos

Para poder resolver el problema tuvo que hacerse un pequeño preprocesado previo. La variable de total de precipitación diario contiene la cantidad de lluvia en mm^3, por lo que se creó una nueva columna que indica 1 si llovió o 0 si no llovió. Para crear esta columna se definió un umbral de 0.1 para descartar posibles errores de medición o mediciones muy bajas debido a la humedad que no podría considerarse como lluvia.

Como en la predicción de la temperatura, se utilizó el rango de 2015-2022 como conjunto de entrenamiento. Como se puede observar en la Figura 5.41 hay muchos más días que no llovió que que sí, como era de esperar de una isla como Tenerife. Esto puede llevar a problemas de sesgo en el modelo por un desbalanceo en las clases. Cabe destacar que se utilizó un umbral muy conservador donde es posible que se estén infraestimando los días sin lluvia. Posteriormente se verá el impacto que puede tener este desbalanceo en el modelo.

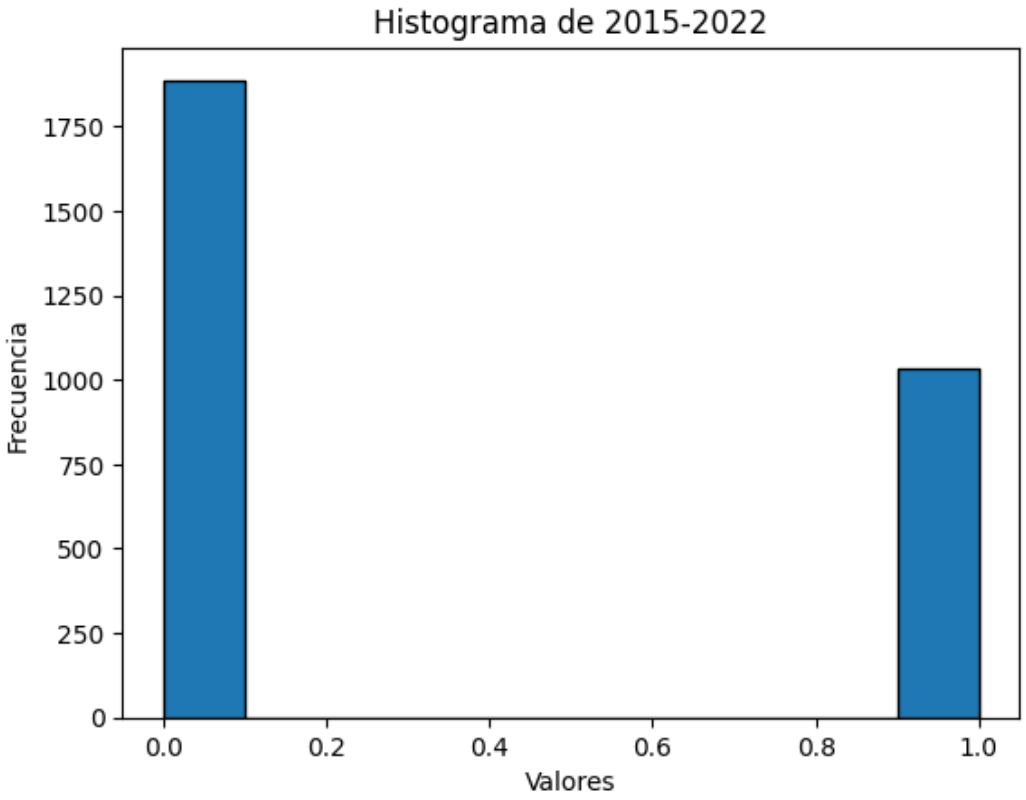


Figura 5.41: Histograma entrenamiento

De igual forma se utilizó 2023 como conjunto de test. Es interesante ver lo diferente que es este histograma comparado con el histograma de entrenamiento. En 2023 hubo más días de lluvia que de no lluvia (siempre teniendo en cuenta que se utilizó un umbral de 0.1 mm^3). Esto es muy interesante para ver como se comporta el modelo, ya que esta distribución distinta entre los conjuntos de entrenamiento y test afectan a los modelos de predicción. Anteriormente se mencionó la existencia de un desbalanceo de clases en favor a la clase de no lluvia. En 2023 existe un desbalanceo contrario, por lo que es interesante para ver el sesgo del modelo entrenado con los datos de entrenamiento y evaluado en el conjunto de test del año 2023.

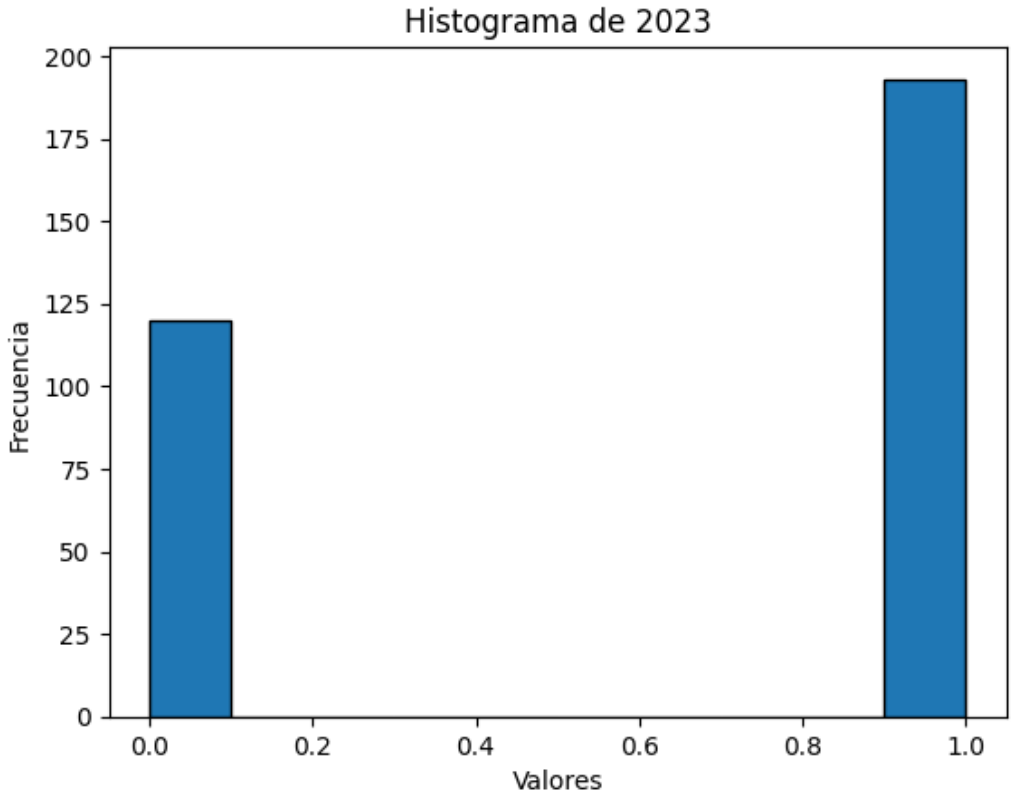


Figura 5.42: Histograma test

En los siguientes apartados se verán dos aproximaciones que se hicieron para resolver este problema de clasificación binaria. Una primera aproximación entrenando el modelo que se muestra al principio de la sección, y una segunda aproximación entrenando este nuevo

modelo pero intentando mitigar el desbalanceo de carga.

Red LSTM sin pesos

En esta primera aproximación no se tuvo en cuenta el desbalanceo entre clases. Al igual que en la predicción de temperatura, se utilizó un *Early Stopping* utilizando la pérdida como métrica de parada temprana. En la Figura 5.43 se puede observar la evolución del error en el entrenamiento.

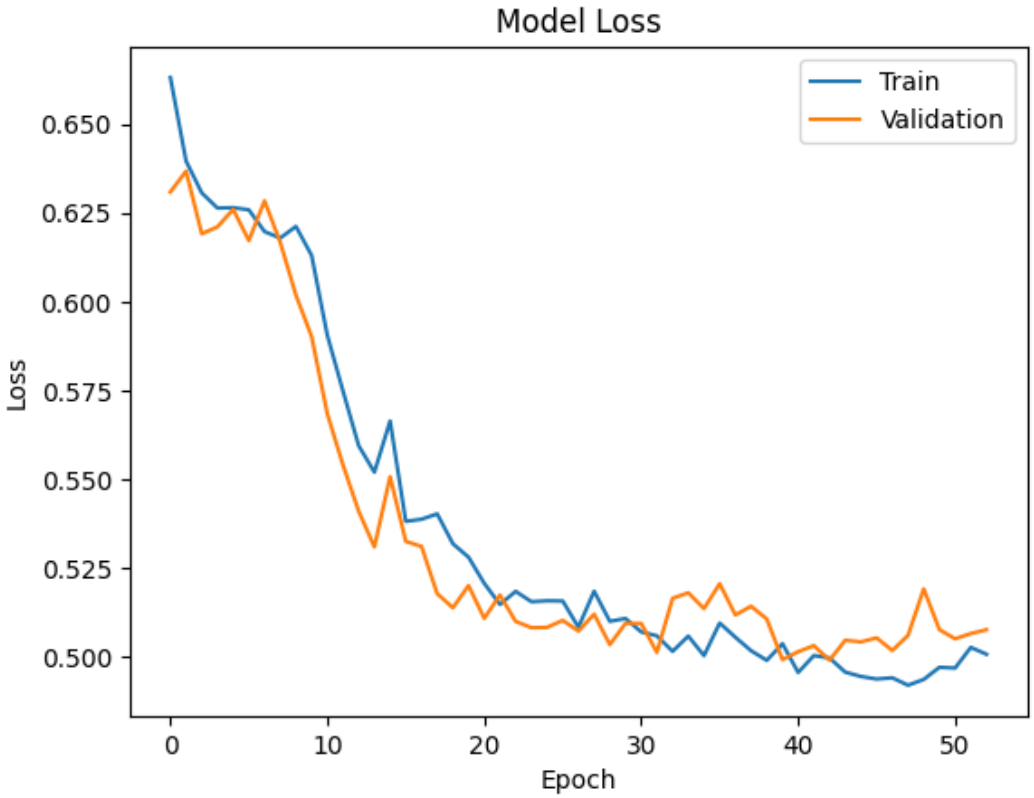


Figura 5.43: Evolución del error durante el entrenamiento

A continuación se testeó el modelo con los datos de 2023. Para ello se utilizó un umbral de 0.5 para determinar si llovió o no, ya que la capa de salida devuelve la probabilidad de pertenecer a la clase 1, es decir la probabilidad de que llueva.

A partir del test se generó la matriz de confusión de la Figura 5.44. Como era de esperar, hubo un 36% de falsos negativos, es decir, muchos días de lluvia que se predijeron como días

de no lluvia. Esto puede deberse en gran parte al desbalanceo que hay entre las clases, sumado a que en los datos de test hay menos datos de la clase dominante que en el conjunto de entrenamiento.

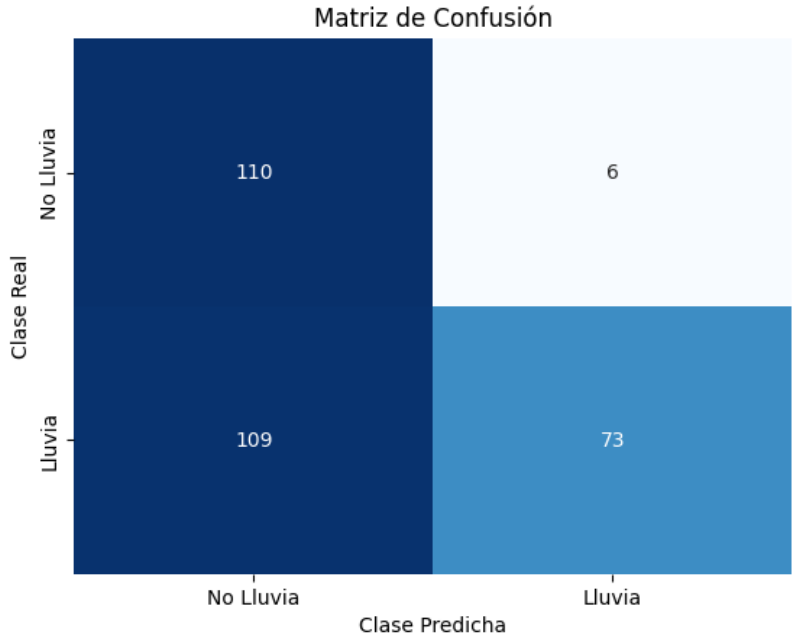


Figura 5.44: Matriz de confusión de la aproximación sin pesos

Las métricas de *accuracy*, *precision*, *recall* y *F1-score* calculadas a partir de la matriz de confusión se observan en la tabla 5.5.

Accuracy	Precision	Recall	F1-Score
0.61	0.92	0.40	0.55

Tabla 5.5: Aproximación sin pesos

El mejor valor lo ofrece la métrica *precision*, lo cual es normal ya que al predecir casi siempre los valores como no lluvia, el valor de esta métrica va a ser alto. El resultado del resto de medidas es muy bajo. Estos malos resultados, en primer lugar tienen que ver con la propia limitación de los modelos dado a que no se cuenta con una gran variedad de métricas para entrenarlos. En segundo lugar, bajo la suposición de que el desabalanceo de clases afecta mucho negativamente al rendimiento del modelo, se intentará abordar en la siguiente aproximación

con el objetivo de mejorar estas métricas de partida.

Red LSTM con pesos

Una vez vistos los resultados de la primera aproximación, se decidió hacer una segunda aproximación introduciéndole al modelo unos pesos calculados. Los pesos se calcularon en base a la proporción de elementos de cada clase. En primer lugar se calculó la proporción de elementos de cada clase. Una vez calculadas las proporciones, los pesos se calcularon como la inversa de la proporción de la clase en cuestión. De esta forma los elementos que pertenezcan a la clase dominante en el conjunto de entrenamiento tendrán un menor peso y de esta forma los ejemplos de los días de lluvia cobran más importancia a la hora de ajustar el modelo. Haciendo el mismo entrenamiento pero añadiendo los pesos, la matriz de confusión mostró mejores resultados (Figura 5.45). Se puede ver que al añadir los pesos se mitiga en gran parte el efecto del desbalanceo entre clases.

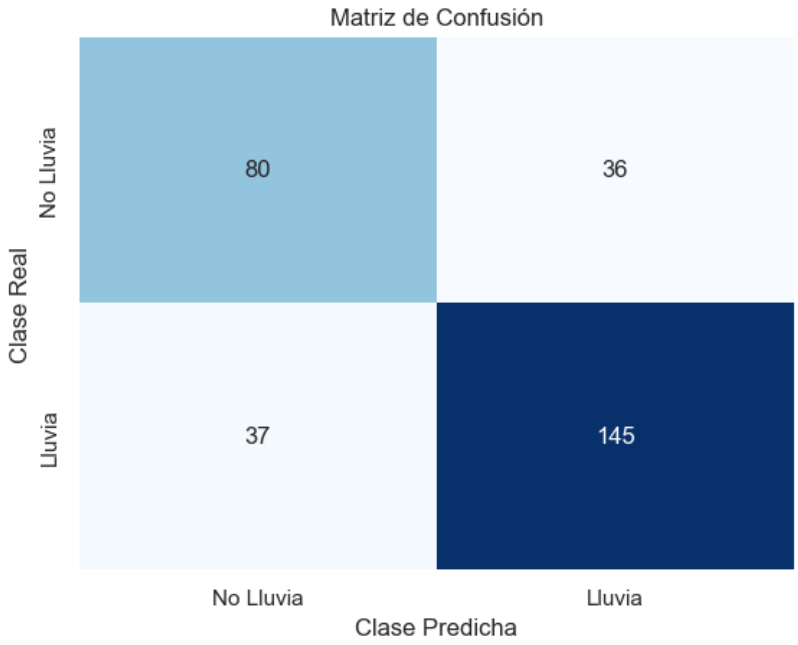


Figura 5.45: Matriz de confusión de la aproximación con pesos

En cuanto a las métricas calculadas a partir de la matriz de confusión anterior, se observa una mejora en todas menos en la precisión, lo cual era de esperar ya que la *precision* de la primera aproximación era muy alta debido al sesgo de predicción de la misma clase.

Accuracy	Precision	Recall	F1-Score
0.76	0.81	0.80	0.80

Modelo escogido

En la tabla 5.6 se muestran las métricas calculadas a partir de la matriz de confusión de cada una de las aproximaciones anteriores. En primer lugar se encuentra una mejora sustancial en la aproximación con pesos en las métricas de *recall* y *F1-score*. El *F1-score* es muy importante ya que brinda una evaluación más completa del rendimiento del modelo en la clasificación cuando existe un desequilibrio entre las clases positiva y negativa. La *precision* mide la estabilidad de las predicciones, por lo que es lógico que en el modelo con pesos sea inferior ya que no predice todo el rato lo mismo por el sesgo de clases. En base a la *accuracy*, el modelo con pesos acierta más que el modelo sin ellos.

Una vez visto el rendimiento de ambos modelos, uno entrenado sin pesos y otro entrenado con pesos calculados en base a la proporción de cada clase, el modelo entrenado con pesos obtuvo un claro mejor rendimiento. Este modelo se almacenó y se utilizará para crear visualizaciones en el cuadro de mandos. Cabe destacar que para evaluar el rendimiento se aplicó un umbral de 0.5 para clasificar en una clase u otra, pero para el dashboard se utilizará la salida de la función sigmoide, ya que lo que se buscaba era calcular la probabilidad de lluvia del día siguiente.

	Accuracy	Precision	Recall	F1-Score
Sin pesos	0.61	0.92	0.40	0.55
Con pesos	0.76	0.81	0.80	0.80

Tabla 5.6: Comparación entre las dos aproximaciones

5.4 Integración de los modelos en el *dashboard*

En esta sección se explica como se integraron los modelos desarrollados, en el cuadro de mandos diseñado anteriormente. Esto se puede hacer debido a que PowerBI permite definir objetos visuales con Python y permite integrarlos en los *dashboards*, con código completamente del usuario. Esto permite una gran flexibilidad cuando las visualizaciones que te permite la herramienta no cumplen tus requisitos. En este caso es muy útil para la automatización

del proceso. Para poder utilizar las visualizaciones de la herramienta es necesario cargar los datos de una fuente externa, por lo que habría que generar un recurso de predicciones diario con las predicciones de todas las estaciones. De esta forma se hace todo dentro de la propia herramienta, ya que carga el modelo que se guardó con los pesos ajustados y se evalúan los datos actuales con el.

PowerBI tiene funcionalidades de predicción integradas en la herramienta, pero estas tienen algunas limitaciones. Una de las limitaciones es que estas no se podían realizar en páginas que tuvieras filtros de estación o municipio. La principal limitación de estos métodos es que no tienen un mecanismo que permita evaluar los resultados de estos modelos para poder compararlos con los desarrollados en este proyecto. Principalmente por este segundo motivo se descartó incluir este método dentro de este trabajo.

5.4.1 Página de datos de temperatura

La gráfica que se visualiza en el *dashboard* está generada con la herramienta Matplotlib. En primer lugar se carga el modelo y se evalúa con los datos que se cargan en PowerBI. Una vez se tienen las predicciones, estas se visualizan con un gráfico de líneas generado con Matplotlib. Si no hay ningún error de código, esta gráfica se puede mover y colocar en la posición que se desee como se ve en la Figura 5.46.

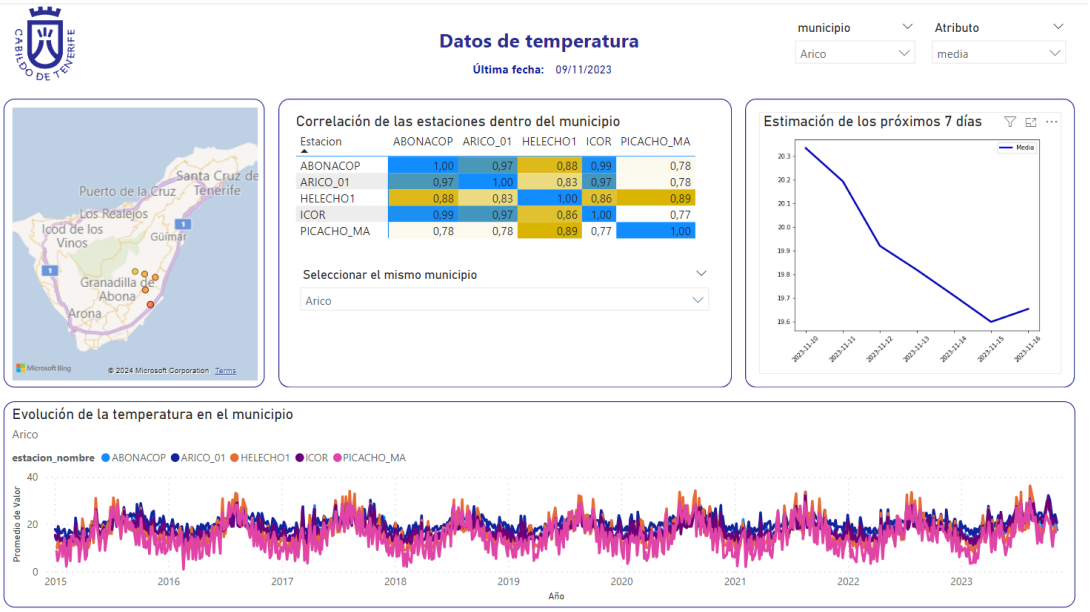


Figura 5.46: Página de datos de temperatura

5.4.2 Página de datos de precipitación

De igual forma que en el caso anterior, se evalúa el modelo almacenado con los datos de PowerBI y se calcula la predicción. Esta visualización fue implementada como una matriz 1x1 creada con la librería Seaborn y muestra la probabilidad de que llueva el día siguiente en el municipio. En la Figura 5.47 se puede ver donde se integró la visualización dentro del *dashboard*.

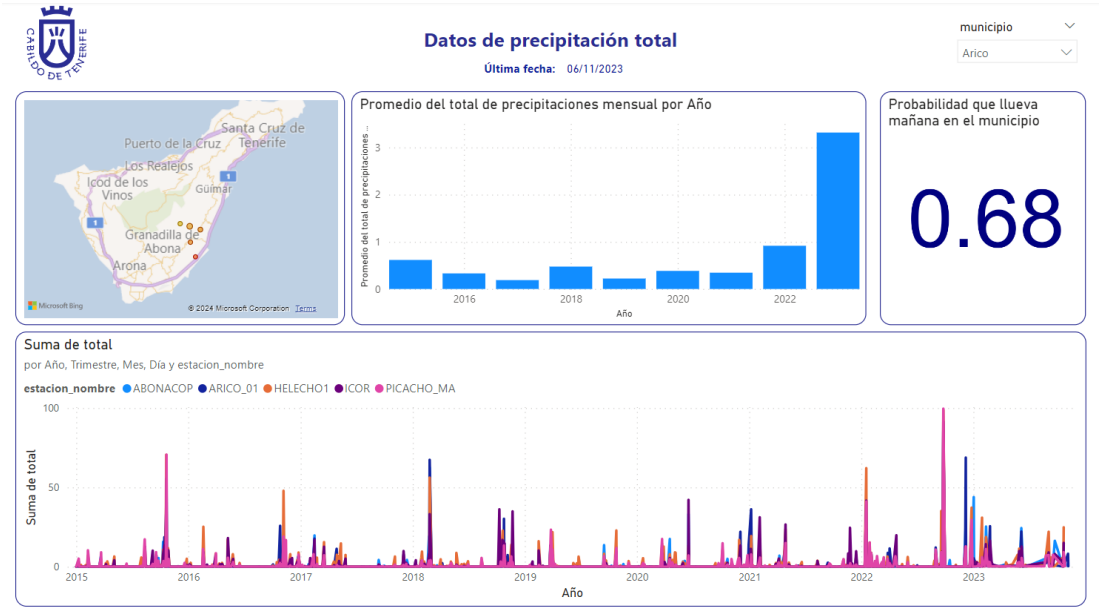


Figura 5.47: Página de datos de precipitación

5.4.3 Código Python en PowerBI

Las gráficas generadas previamente se implementaron dentro de la propia IDE de programación que tiene PowerBI. El editor se abre automáticamente cuando se presiona en una visualización Python. Para poder utilizar los datos cargados dentro de la herramienta, en primer lugar es necesario arrastrarlos a la caja de Valores de la misma forma que se hace en otras visualizaciones. Una vez arrastrados las columnas que se quieren utilizar, PowerBI autogenera código Python que carga los datos en una variable que llama *dataset*:

```
dataset = pandas.read_csv(
'input_df_61951505-6725-4b4c-80e2-85985a353ea8.csv')
```

En la Figura 5.48 se puede ver como se cargan los datos de PowerBI en un *dataframe*, y como se cargan el modelo y el escalador. Una vez todo está cargado, ya se pueden calcular las predicciones para los datos en cuestión.

```
dataset = pandas.read_csv('input_df_4ec62082-7582-4880-9b4f-dc78ffe341b2.csv')
modelo = tf.keras.models.load_model('path_model_7')
escalador = joblib.load('path_scaler')
array = dataset[-15:]["Valor"]

datos_transformados = escalador.transform(array.to_numpy().reshape(-1, 1))
prediccion = modelo.predict(datos_transformados.reshape((1, 15, 1)))
temperatura_predicha = escalador.inverse_transform(prediccion)
```

Figura 5.48: Código en PowerBI

El código completo empleado para la generación de las visualizaciones de la predicción de la temperatura y de las precipitaciones se puede ver en el Apéndice B.

Capítulo 6

Conclusiones

En este último capítulo se repasará el trabajo realizado a lo largo de este proyecto. A continuación se proponen algunas líneas de trabajo futuro del proyecto y una valoración personal del mismo.

6.1 Conclusiones del proyecto

A lo largo de este proyecto se realizaron diversas implementaciones para conseguir una solución final. Se desarrolló un proceso ETL utilizando DAGs de Apache Airflow, extrayendo los datos de la base de datos proporcionada por el Cabildo de Tenerife y procesándolos con éxito para poder utilizarlos en procesos posteriores. .

Con todo esto, se construyó una herramienta de visualización de datos de la isla de Tenerife que contiene una gran cantidad de datos históricos para visualizar de distintas formas, cubriendo así todas las necesidades de análisis que se planteaban en los primeros pasos del proyecto.

Por último, se exploraron distintos métodos de predicción para predecir tanto la temperatura como la probabilidad de lluvia, para posteriormente poder complementar toda la información del *dashboard* implementado con estimaciones futuras. El modelo que ofreció un mejor rendimiento fue una red neuronal recurrente haciendo uso de celdas LSTM. En el caso de la predicción de la precipitación del día siguiente, se vio el efecto que sucede cuando existe el desbalanceo de clases y como utilizando pesos calculados en base a las proporciones se puede mitigar este efecto en gran medida.

6.2 Trabajo futuro

Una vez cumplidos los objetivos de este proyecto de fin de grado, es de interés destacar algunos puntos en los que se podría continuar:

- **Predicción**. Incluir predicciones con un mayor rango o añadir nuevas medidas a predecir podría hacer que el cuadro de mandos sea aun más completo. También se podrían explorar nuevos métodos de predicción y, de utilizar nuevos datos recogidos por satélites, explorar modelos climáticos que se están utilizando en las herramientas más populares.
- **Aumentar el histórico**. Otro punto de interés sería incluir en el desarrollo un histórico mucho mayor. Sería de gran interés contar con una herramienta que ofrezca por ejemplo un histórico desde el año 2000. A pesar de que la herramienta está diseñada para el manejo de una cantidad de datos mayor, en este proyecto se utilizó un rango de años ligeramente corto debido a que eran los datos disponibles en la base de datos de inicio, pero de tener más habría que recortar de todas formas debido a las capacidades del equipo utilizado.
- **Utilizar datos satelitales**. Sería interesante poder contra con una fuente de datos satelital para poder integrar otras visualizaciones, ya que las visualizaciones de este proyecto se centran en la evolución de series temporales.
- **Utilizar otros formatos de almacenamiento de los datos**. En este proyecto se utilizaron archivos CSV para el almacenamiento. Se podrían utilizar otros formatos para almacenar los datos de una forma más eficiente y más flexible.

6.3 Valoración personal

Para finalizar con esta memoria, me gustaría dar una valoración personal sobre el desarrollo de este proyecto. En primer lugar, este trabajo me ha permitido trabajar con muchos conceptos trabajados en el grado en distintos ámbitos: bases de datos, *machine learning*, etc. A mayores me ha permitido conocer como es un proceso de datos dentro de una organización privada, y así poder ver su importancia y evaluar mis habilidades en el manejo de datos reales.

Apéndices

Apéndice A

Dashboards de PowerBI

A.1 Dashboards de la humedad relativa y velocidad del viento

En la Figura A.1 se muestra como se ve la página principal para la medida de humedad relativa. En el Figura A.2, se muestra la página para la velocidad del viento.

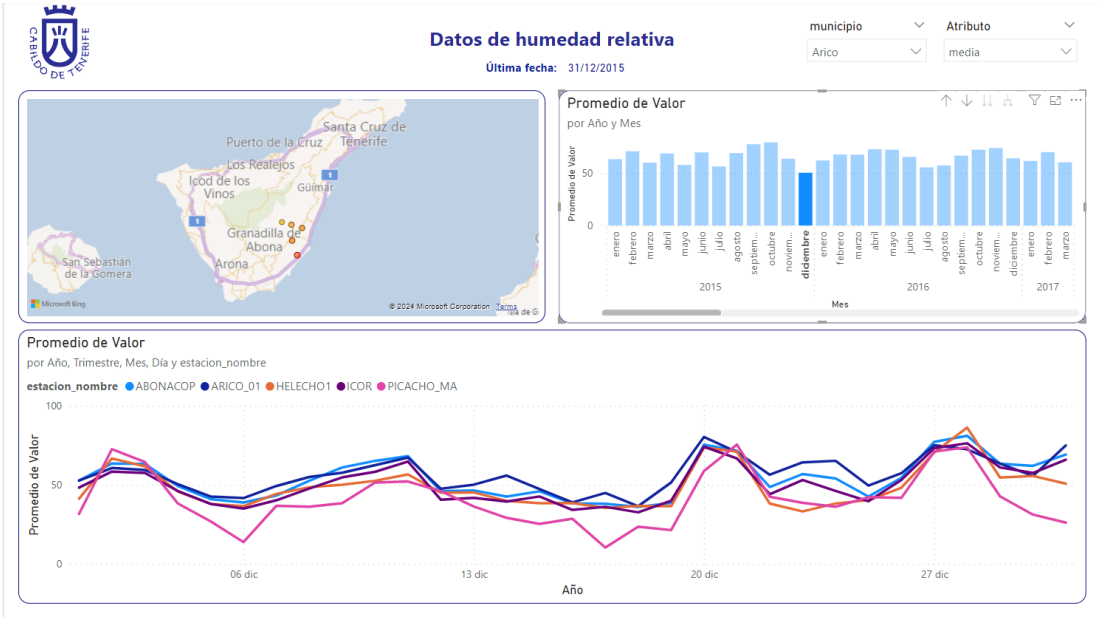


Figura A.1: Página de vista general de la humedad relativa

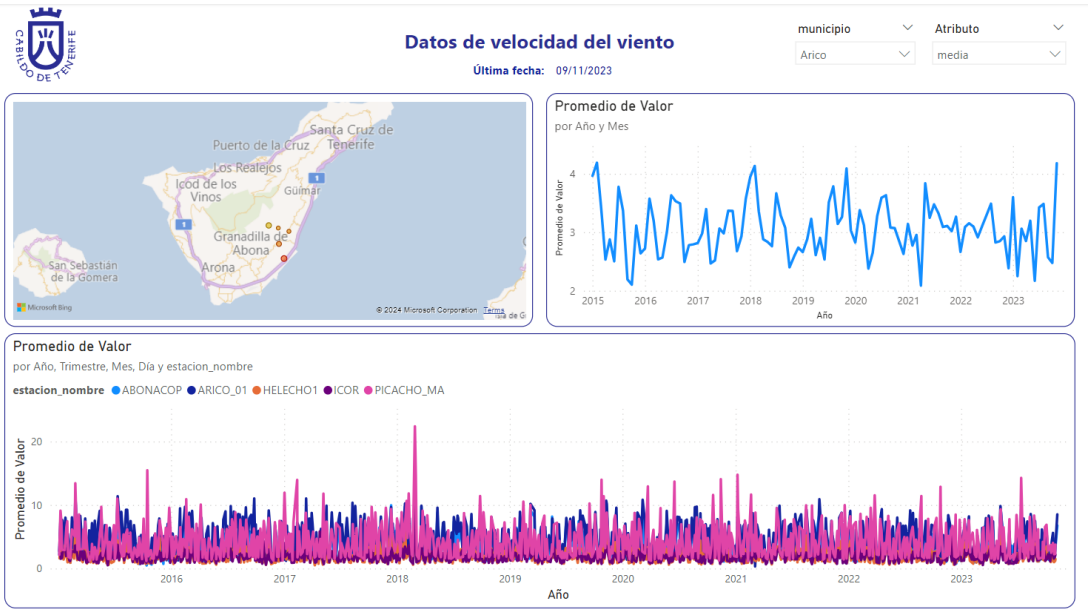


Figura A.2: Página de vista general de la velocidad del viento

A.2 Predicción con el método nativo de PowerBI

En la Figura A.3 se muestra un ejemplo de predicción hecha con la característica de predicción integrada dentro de PowerBI. Según un artículo del equipo de PowerBI en el foro oficial de Microsoft[23], los métodos que utilizan para hacer las predicciones de series temporales se basan en un *exponential smoothing*.

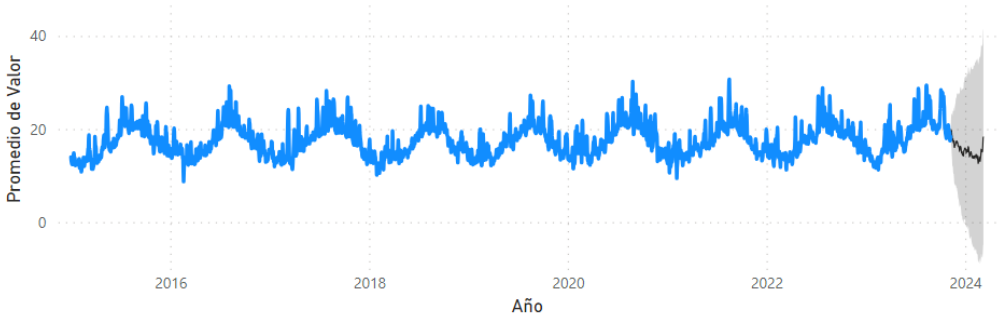


Figura A.3: Predicción del método de PowerBI

Apéndice B

Integración de python en PowerBI

B.1 Código pyhton utilizado en PowerBI

En la Figuras B.1 y B.2 se muestra el código empleado para integrar las predicciones de precipitación y temperatura respectivamente en el *dashboard*.

```
modelo = tf.keras.models.load_model('path_model_rain')
escalador = joblib.load('path_scaler_rain')

# Supongamos que ya has cargado tu dataset antes de este código

# Convertir la columna 'fecha_observacion' a formato de fecha
dataset['fecha_observacion'] = pd.to_datetime(dataset['fecha_observacion']).dt.date

# Agrupar por fecha y calcular la media de la columna 'media'
dataset = dataset.groupby('fecha_observacion')[['total']].mean().reset_index()
dataset['lluvia'] = dataset['total'].apply(lambda x: 1 if x > 0.1 else 0)
dataset = dataset.rename(columns={"total":"total_precipitacion"})

#####################################################################
prueba = dataset[-5:][["total_precipitacion","lluvia"]]
datos_transformados = escalador.transform(prueba)
prediccion = modelo.predict(datos_transformados.reshape((1, 5, 2)))

#####################################################################

plt.figure(figsize=(6.555555555555556,5.555555555555556), dpi=72)


# Crear un mapa de calor de la matriz de correlación
sns.set(style="white")
cmap = sns.color_palette(["#FFFFFF"], as_cmap=True)
sns.heatmap(prediccion, annot=True, fmt=".2f", cmap=cmap, cbar=False, square=True,xticklabels=False,yticklabels=False,annot_kws={"size": 200,"color":"navy"})

# Añadir etiquetas y mostrar el mapa de calor
plt.show()
```

Figura B.1: Código de generación de la predicción de la precipitación

```
modelo = tf.keras.models.load_model('path_model_7')
escalador = joblib.load('path_scaler')

# Supongamos que ya has cargado tu dataset antes de este código

# Convertir la columna 'fecha_observacion' a formato de fecha
dataset['fecha_observacion'] = pd.to_datetime(dataset['fecha_observacion']).dt.date
fecha_inicial = dataset["fecha_observacion"].to_numpy()[-1]
fechas_futuras =  [fecha_inicial + timedelta(days=i) for i in range(8)][1:]

# Agrupar por fecha y calcular la media de la columna 'media'
dataset = dataset.groupby('fecha_observacion')[['Valor']].mean().reset_index()

########################################################################
array = dataset[-15:]["Valor"]
datos_transformados = escalador.transform(array.to_numpy().reshape(-1, 1))
prediccion = modelo.predict(datos_transformados.reshape((1, 15, 1)))
temperatura_predicha = escalador.inverse_transform(prediccion)
########################################################################

plt.figure(figsize=(6.55555555555556,5.55555555555556), dpi=72)

# Graficar la línea con un estilo de curvas neón
plt.plot(fechas_futuras,temperatura_predicha[0], linewidth=3, color='blue', label='Media', markersize=8)

# Agregar efecto de sombra
plt.plot(fechas_futuras,temperatura_predicha[0], linewidth=3, color='black', alpha=0.3)

# Rotar las etiquetas del eje x para una mejor legibilidad
plt.xticks(rotation=45)

# Mostrar leyenda
plt.legend()
```

Figura B.2: Código de generación de la predicción del temperatura

Lista de acrónimos

AEMET Agencia Estatal de Meteorología. 4, 48

ARIMA Autoregressive Integrated Moving Average. 9, 54–58, 61

BIC Criterio de Información Bayesiano. 55

DAG Directed Acyclic Graph. 25–27, 29–31, 34–37, 54

Bibliografía

[1] W. oficial del Gobierno de Canarias, "Isla de tenerife," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www3.gobiernodecanarias.org/medusa/wiki/index.php?title=Isla_de_Tenerife#Sectores_econ.C3.B3micos

[2] W. oficial de AEMET, "Predicción numérica del tiempo," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.aemet.es/es/idi/prediccion/prediccion_numerica

[3] W. oficial de ElTiempo.es, "El tiempo," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.eltiempo.es/noticias/autor/eltiempo-es#:~:text=Eltiempo.es%20es%20el%20soporte,hora%20y%20en%20tiempo%20real.

[4] U. de Vigo, "Meteogalicia," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://lifetec.uvigo.es/service/meteogalicia/

[5] W. oficial de Apache Airflow, "What is airflow™?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://airflow.apache.org/docs/apache-airflow/stable/index.html

[6] ——, "Dags," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://airflow.apache.org/docs/apache-airflow/stable/core-concepts/dags.html

[7] W. oficial de IBM, "What is etl?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.ibm.com/topics/etl

[8] W. oficial de Microsoft, "What is power bi?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://learn.microsoft.com/en-us/power-bi/fundamentals/power-bi-overview

[9] M. P. G. Casimiro, *Análisis de series temporales: Modelos ARIMA*, 1st ed. UPV, 2009.

[10] W. oficial de IBM, "What are recurrent neural networks?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.ibm.com/topics/recurrent-neural-networks

[11] W. oficial de Python, "What is python? executive summary," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.python.org/doc/essays/blurb/

[12] W. oficial de AmazonAWS, "¿qué es python?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/python/

[13] W. oficial de NVIDIA, "Pandas," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.nvidia.com/en-us/glossary/data-science/pandas-python/

[14] W. oficial de Tensorflow, "Por qué tensorflow," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.tensorflow.org/about?hl=es-419

[15] W. oficial de Matplotlib, "Matplotlib: Visualization with python," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://matplotlib.org/

[16] W. oficial de Microsoft, "Visual studio code," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://azure.microsoft.com/es-es/products/visual-studio-code

[17] W. oficial de PostgreSQL, "What is postgresql?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.postgresql.org/about/

[18] W. oficial de Atlassian, "What is scrum?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.atlassian.com/agile/scrum

[19] W. oficial de Scrum.org, "What is scrum?" consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://www.scrum.org/resources/what-scrum-module

[20] W. oficial de Meta, "Forecasting at scale," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://facebook.github.io/prophet/

[21] W. oficial de Google, "Normalización," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://developers.google.com/machine-learning/data-prep/transform/normalization?hl=es-419

[22] C. University, "Adam," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://optimization.cbe.cornell.edu/index.php?title=Adam

[23] W. oficial de Microsoft, "Describing the forecasting models in power view," consultado el 9 de febrero de 2024. [En línea]. Disponible en: https://powerbi.microsoft.com/es-es/blog/describing-the-forecasting-models-in-power-view/