



UNIVERSIDAD
DE MÁLAGA



UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento de Economía y Dirección de Empresas

Área de conocimiento: Organización de Empresas

TRABAJO FIN DE GRADO

**ESTUDIO Y ALGORITMOS PARA LA ACTIVACIÓN ANTICIPADA DE
ELEMENTOS DE CLIMATIZACIÓN**

Grado en Ingeniería de Organización Industrial

Mención en Sistemas de Producción

Alumna: Vázquez Salazar, Marina

Tutor académico: Calero Castro, Santiago

Cotutor externo: Requena García, Gabriel

Málaga, septiembre de 2023

Agradecimientos

A mi familia, por guiarme, apoyarme y creer en mí desde el primer momento. Gracias por vuestro apoyo incondicional durante estos años, habéis hecho que todo sea posible.

Resumen

Este proyecto tiene como objetivo la búsqueda, evaluación y adaptación de un modelo óptimo dentro de unos algoritmos predictivos existentes. Esto permitirá la predicción del tiempo que tarda una estancia en alcanzar una temperatura consigna determinada. Este tiempo dependerá de una serie de variables: humedad, temperatura objetivo, temperatura inicial, etc. Para ello, la empresa Airzone proporcionará las bases de datos necesarias de una instalación anónima, con el objetivo de encontrar el modelo que mejor funcione al predecir la diferencia de tiempo. De esta forma, el usuario obtendrá una temperatura deseada a una hora programada, logrando el menor gasto energético posible y un aumento de la eficiencia energética.

Palabras clave

Aprendizaje automático, eficiencia energética, zonificación, algoritmos, variables.

Abstract

This project aims to search, evaluate and adapt an optimal model within existing predictive algorithms. This will allow the prediction of the time it takes for a room to reach a certain set temperature. This time will depend on a series of variables: humidity, target temperature, initial temperature, etc. To do this, the Airzone company will provide the necessary databases from an anonymous facility, with the aim of finding the model that works best when predicting the time difference. In this way, the user will obtain a desired temperature at a programmed time, achieving the lowest possible energy expenditure and an increase in energy efficiency.

Keywords

Machine learning, energy efficiency, zoning, algorithms, variables.

ÍNDICE DE CONTENIDO

1. ANTECEDENTES.....	15
2. ALCANCE Y OBJETO DEL TRABAJO	19
3. LA EMPRESA	23
3.1. ¿Qué es Airzone?	23
3.2. Ubicación de las instalaciones de la empresa	24
3.3. Airzone y la eficiencia energética.....	25
4. BASES DE DATOS.....	29
4.1. Objetos de DeviceEvents.json	30
4.2. Objetos de Weather.json	32
4.3. Ejemplos	33
4.3.1. Ejemplo “DeviceEvents.json”.....	33
4.3.2. Ejemplo “Weather.json”.	36
5. LENGUAJE DE PROGRAMACIÓN.....	39
5.1. Librerías usadas	40
6. APRENDIZAJE AUTOMÁTICO O MACHINE LEARNING	43
7. INTRODUCCIÓN AL CONTENIDO DEL PROYECTO.....	47
8. PASO PREVIO - ARCHIVOS DE TEXTO	53
9. SECCIÓN A: PREPROCESAMIENTO Y SPLIT	55
9.1. Tipos y recuento de parámetros en DeviceEvents.json	55
9.2. Funciones de preprocesamiento.....	56
9.3. Procesamiento de los JSON y obtención de los datos	58
9.4. Creación y eliminación de variables por conocimiento del problema.....	59
9.5. Normalización y división de los datos en entrenamiento y test.....	62
9.6. Análisis de correlación entre variables X	63
10. SECCIÓN B Y C: SELECCIÓN DE VARIABLES.....	69
10.1. Análisis de correlación y sus variables	63

10.1.1. Estudio de invierno.....	71
10.1.2. Estudio de verano.	73
10.1.3. Estudio conjunto.	75
10.2. Todas las variables previamente filtradas	63
10.3. LASSO y sus variables.....	63
10.3.1. Validación cruzada	78
10.3.2. Variables obtenidas	80
10.4. Conclusiones de la selección de variables.....	85
11. SECCIÓN D: EJECUCIÓN DE MODELOS Y EVALUACIÓN	87
11.1. Explicación de modelos	88
11.1.1. Regresión multivariante	88
11.1.2. Árboles de decisión (Decision Trees).	90
11.1.3. Bosques aleatorios (Random Forest).	91
11.1.4. XGBoost.	92
11.2. Hiperparámetros	95
11.3. Métricas de evaluación.....	97
11.3.1. Coeficiente de Determinación (R2).....	97
11.3.2. Error Absoluto Medio (MAE).	98
11.3.3. Error Cuadrático Medio (MSE).	99
11.3.4. Raíz del Error Cuadrático Medio (RMSE).....	100
11.3.5. Otras consideraciones sobre las métricas de evaluación.	101
12. RESULTADOS OBTENIDOS	103
13. CONCLUSIONES	109
14. LÍNEAS FUTURAS	113
15. BIBLIOGRAFÍA DE CONTENIDO	115
16. BIBLIOGRAFÍA DE IMÁGENES.....	123
17. ANEXOS.....	125

ÍNDICE DE ILUSTRACIONES

Ilustración 1. Gráfico de temperatura respecto al tiempo.....	20
Ilustración 2. Logo de Airzone	23
Ilustración 3. Ubicación de las instalaciones.....	25
Ilustración 4. Certificación BREEAM	26
Ilustración 5. Certificación LEED	27
Ilustración 6. Certificación VERDE.....	27
Ilustración 7. Certificación WELL Building Standard.....	27
Ilustración 8. Base de datos DeviceEvents.json (1)	34
Ilustración 9. Base de datos DeviceEvents.json (2)	35
Ilustración 10. Base de datos Weather.json.....	36
Ilustración 11. Logo de Python	39
Ilustración 12. Ejemplo de aprendizaje supervisado por regresión	45
Ilustración 13. Ejemplo de aprendizaje supervisado por clasificación.....	46
Ilustración 14. División de contenido.....	47
Ilustración 15. Esquema A - Preprocesamiento y Split.....	49
Ilustración 16. Esquema B y C - Selección de variables	50
Ilustración 17. Esquema D - Ejecución y Evaluación	51
Ilustración 18. Listado-DeviceEvents.txt	53
Ilustración 19. Listado-Weather.txt	53
Ilustración 20. Dataset almacenado en TSV.....	58
Ilustración 21. Mapa de calor correlaciones X de invierno.....	65
Ilustración 22. Mapa de calor correlaciones X de verano	66
Ilustración 23. Mapa de calor correlaciones X del conjunto	67
Ilustración 24. Gráfico de barras invierno B1	71
Ilustración 25. Gráfico de barras invierno C1	72
Ilustración 26. Gráfico de barras verano B1.....	73

Ilustración 27. Gráfico de barras verano C1	74
Ilustración 28. Gráfico de barras conjunto B1	75
Ilustración 29. Gráfico de barras conjunto C1	76
Ilustración 30. Validación cruzada	79
Ilustración 31. Gráfico de barras invierno B3	81
Ilustración 32. Gráfico de barras invierno C3	81
Ilustración 33. Gráfico de barras verano B3	82
Ilustración 34. Gráfico de barras verano C3	82
Ilustración 35. Gráfico de barras conjunto B3	83
Ilustración 36. Gráfico de barras conjunto C3	83
Ilustración 37. Resumen selección de variables	86
Ilustración 38. Regresión multivariante	89
Ilustración 39. Árbol de decisión	90
Ilustración 40. Random Forest	92
Ilustración 41. Ejemplo adivinanzas XGBoost	94
Ilustración 42. Posibles modelos	103
Ilustración 43. R2 prueba - invierno	104
Ilustración 44. MAE prueba - invierno	104
Ilustración 45. MAE prueba de los 3 estudios	106
Ilustración 46. Balance coste y precisión	110

ÍNDICE DE ANEXOS

Anexo A. Listados listado-DeviceEvents.txt y listado-weather.txt.....	125
Anexo B. Código A0_Preprocesamiento.py – python_tfg.....	128
Anexo C. Código fn_preprocesamiento.py – python_tfg.....	130
Anexo D. Código A1_Preprocesamiento.py – python_tfg.....	135
Anexo E. Código A2_Inspeccion.py – python_tfg	138
Anexo F. Código A2i_distribucion_y_descriptiva.py – python_tfg	139
Anexo G. Excel A2i_Descripcion_estadistica.xlsx – python_tfg	141
Anexo H. Excel A2i_Descripcion_estadistica.xlsx – python_tfg_verano	142
Anexo I. Excel A2i_Descripcion_estadistica.xlsx – python_tfg_conjunto.....	143
Anexo J. Imágenes de la carpeta A2i_histogramas – python_tfg.....	144
Anexo K. Imágenes de la carpeta A2i_histogramas – python_tfg_verano	152
Anexo L. Imágenes de la carpeta A2i_histogramas – python_tfg_conjunto.....	160
Anexo M. Código fn_normalizacion.py – python_tfg	168
Anexo N. Código A3_Normalizacion.py – python_tfg.....	169
Anexo Ñ. Código A4_Correlacion.py – python_tfg.....	170
Anexo O. Excel A4_Correlacion_X.xlsx – python_tfg.....	172
Anexo P. Excel A4_Correlacion_X.xlsx – python_tfg_verano	175
Anexo Q. Excel A4_Correlacion_X.xlsx – python_tfg_conjunto.....	178
Anexo R. C0_Normalizacion.py – python_tfg.....	181
Anexo S. Código B1_Preparacion.py – python_tfg	183
Anexo T. B1_Variables.tsv– python_tfg.....	185
Anexo U. B1_Variables.tsv– python_tfg_verano	185
Anexo W. B1_Variables.tsv– python_tfg_conjunto	185
Anexo X. Código B2_Preparacion.py – python_tfg	186
Anexo Y. B2_Variables.tsv– python_tfg	187
Anexo Z. B2_Variables.tsv– python_tfg_verano	187

Anexo AA. B2_Variables.tsv– python_tfg_conjunto	188
Anexo BB. Código B3_Preparacion.py – python_tfg	189
Anexo CC. B3_Variables.tsv– python_tfg.....	191
Anexo DD. B3_Variables.tsv– python_tfg_verano.....	191
Anexo EE. B3_Variables.tsv– python_tfg_conjunto	191
Anexo FF. Código C1_Preparacion.py – python_tfg.....	192
Anexo GG. C1_Variables.tsv– python_tfg	194
Anexo HH. C1_Variables.tsv– python_tfg_verano.....	194
Anexo II. C1_Variables.tsv– python_tfg_conjunto.....	194
Anexo JJ. Código C2_Preparacion.py – python_tfg	195
Anexo KK. C2_Variables.tsv– python_tfg	196
Anexo LL. C2_Variables.tsv– python_tfg_verano	196
Anexo MM. C2_Variables.tsv– python_tfg_conjunto	197
Anexo NN. Código C3_Preparacion.py – python_tfg.....	198
Anexo ÑÑ. C3_Variables.tsv– python_tfg	200
Anexo OO. C3_Variables.tsv– python_tfg_verano.....	200
Anexo PP. C3_Variables.tsv– python_tfg_conjunto.....	200
Anexo QQ. Código fn_modelos.py– python_tfg	201
Anexo RR. Código D_Entrenamiento_y_evaluacion.py.....	206
Anexo SS. Excel D_Evaluacion.xlsx – python_tfg.....	219
Anexo TT. Excel D_Evaluacion.xlsx – python_tfg_verano.....	220
Anexo UU. Excel D_Evaluacion.xlsx – python_tfg_conjunto.....	221
Anexo VV. Excel D_Result_coef_imp de B y C – python_tfg.....	222
Anexo WW. Excel D_Result_coef_imp de B y C – python_tfg_verano	224
Anexo XX. Excel D_Result_coef_imp de B y C – python_tfg_conjunto.....	226

1. ANTECEDENTES

La climatización busca el bienestar térmico en un espacio cerrado. Esto se logra por medio de la calefacción, la refrigeración o la ventilación de una estancia, para adecuar así la humedad y la temperatura.

Según el Reglamento de Instalaciones Térmicas en Edificios (RITE) los valores de humedad relativa y de temperatura para gozar de un buen confort térmico según la época del año son los siguientes:

- Para verano, es conveniente una temperatura que ronde entre los 23 y los 25 grados centígrados y una humedad relativa que ronde entre el 45 y el 60 por ciento.
- Para invierno, es conveniente una temperatura que ronde entre los 21 y los 23 grados centígrados y una humedad relativa que ronde entre el 40 y el 50 por ciento.

Sin embargo, esto es solo una generalización. Estos valores varían según la persona, ya que cada una tiene una sensación térmica diferente, y el frío o calor que se siente va más allá de los grados que marque un termómetro.

Para obtener un buen confort térmico en las personas y una buena salubridad del aire, disponemos de sistemas de climatización, que son unos dispositivos mecánicos que varían la humedad, la temperatura, la pureza y circulación del aire.

Actualmente, los sistemas climatización no generan el confort deseado, gastan más energía de la necesaria, no pueden apagarse a distancia e incluso en muchos casos estéticamente dejan mucho que desear.

La empresa Airzone resuelve todos estos inconvenientes adaptándose a todo tipo de edificios y máquinas mejorando el rendimiento de la instalación.

Los hogares que tienen un sistema de climatización con rejillas, normalmente constan de unas rejillas en cada habitación que permiten el flujo de aire más o menos caliente. Generalmente sólo se puede elegir la temperatura en el salón sin saber la temperatura que se obtendrá en el resto de habitaciones. Esto supone tanto un derroche de energía en aquellas habitaciones donde no hay nadie como una falta de confort para los usuarios que se encuentran en las otras habitaciones.

Con Airzone se puede elegir la temperatura en cada habitación y se podrán apagar las zonas que no se estén utilizando, esto se hace por medio de unas rejillas inteligentes, un termostato inteligente en cada habitación y una central de sistema que analiza la información y regula los equipos. Con ello, se logra integrar todas las partes de la residencia.

Además de hacer uso de este sistema en zonas residenciales, también se puede adaptar para negocios o entornos dedicados al uso terciario. Por lo que además de adaptarse a la climatización por rejillas, se adapta a otros tipos de climatización como son splits de aire acondicionado, cassettes o radiadores.

Por otra parte, el usuario dispone de “Airzone Cloud”, que es una aplicación móvil donde se controla el sistema donde y cuando se desee, permitiendo establecer un control programable de la climatización de la estancia.

Sin embargo, esto supone un problema ya que, si se programa la climatización a una hora determinada, esta se activa a esa hora, pero necesita un tiempo hasta alcanzar la temperatura objetivo que el usuario demande. Este tiempo depende de varios factores como son la diferencia de temperatura entre la temperatura objetivo y la inicial, las condiciones exteriores, el tamaño de la estancia, la potencia de climatización, etc.

Esto provoca que los usuarios no estén en confort durante un tiempo indeterminado al no disponer de la temperatura deseada.

Actualmente, los usuarios encienden el equipo de camino a casa para que su hogar los espere con la temperatura prevista, ya que la climatización necesita un tiempo para conseguir la temperatura objetivo establecida por la persona. Esto presenta una serie de problemas:

- Si el usuario arranca la climatización demasiado tarde, no estará en confort ya que no conseguirá alcanzar la temperatura objetivo en el tiempo que desea.
- Si el usuario arranca la climatización demasiado pronto, es probable que consiga la temperatura deseada, pero esto implicará un consumo de energía innecesario.

Este problema dio lugar a la idea de implantar la posibilidad de obtener la temperatura que el usuario desea a una hora determinada.

Para ello, el sistema de climatización arrancará un tiempo previo al predeterminado por el usuario, la diferencia de tiempo entre el momento en el que se enciende la máquina y el momento en el que alcanza la temperatura deseada por el usuario será el tiempo óptimo necesario para que el sistema de climatización alcance la temperatura predeterminada en el instante predeterminado.

Hay que destacar, que ya hay empresas referentes en el sector como Google Nest o Ecobee cuyos termostatos ya utilizan algoritmos de aprendizaje automático y varios sensores avanzados para a predecir el tiempo en el que la habitación alcanza la temperatura objetivo determinada por el usuario. Esta innovación es útil e interesante ya que contribuye a una mayor comodidad en el hogar y al incremento de la eficiencia energética del mismo.

2. ALCANCE Y OBJETO DEL TRABAJO

El objetivo principal de este Trabajo de Fin de Grado será la búsqueda, evaluación y adaptación de un modelo óptimo dentro de unos algoritmos predictivos existentes. Esto permitirá la determinación del tiempo de antelación al que se debe activar la climatización en una estancia para alcanzar una temperatura deseada por el usuario, logrando un menor gasto energético.

Para ello, se hará uso de unas bases de datos anónimas, proporcionadas por la empresa Airzone, de una estancia que será de ámbito residencial. La estancia será de dicho ámbito y no de uso terciario porque generalmente en estas últimas, como pueden ser oficinas, se produce más movimiento de personas que entran y salen del habitáculo, además de que existe una mayor posibilidad de que tanto las ventanas como las puertas permanezcan abiertas. Todo este ruido provocado por esos factores externos podría verse reflejado desfavorablemente en la evolución de la curva de temperatura ambiente. Por lo que se presupone que las condiciones de entorno de las estancias de tipo residencial son más constantes, y por ende también lo será su curva de temperatura al existir menos factores que afecten al modelo. En consecuencia, se presupone que el modelo obtendrá mejores resultados.

La configuración de la instalación de la estancia a estudiar será de suelo radiante, de aire o de una combinación de ambos.

Se realizará el estudio en una sola habitación de la estancia de tipo residencial, ya que estudiando y hallando el algoritmo correspondiente para una de las habitaciones de la estancia, se podrá aplicar el mismo estudio al resto de las habitaciones de la misma. La base de datos proporcionará históricos sobre la evolución de las temperaturas en esta habitación, y de los cambios o eventos que surjan, como puede ser una parada parcial del sistema de climatización (donde deje de calentar o enfriar la zona ya que se encuentra en la temperatura demandada), o por una parada total del equipo por parte del usuario (donde apague el sistema completamente).

Agregando a lo anterior, las muestras de ejemplo que se proporcionarán de las bases de datos seguirán prácticamente el mismo patrón. Se darán situaciones donde el sistema, que se encuentra en un estado de parada total, arranque y se vaya aproximando a la temperatura que el usuario demanda, hasta que llegue a 0.5 grados Celsius por encima de la misma. Cuando esto ocurra, tendrá lugar una parada parcial del sistema de

climatización, hasta que la temperatura de la habitación se encuentre a menos de 0.5 grados Celsius por debajo de la temperatura demandada. En este momento el sistema volverá a ponerse en marcha. El proceso se repetirá de forma indefinida a menos que tenga programado un apagado de la instalación a cierta hora o el usuario apague completamente la instalación.

A continuación, se muestra gráficamente lo explicado mediante una representación de la temperatura ambiente de la estancia respecto al tiempo.

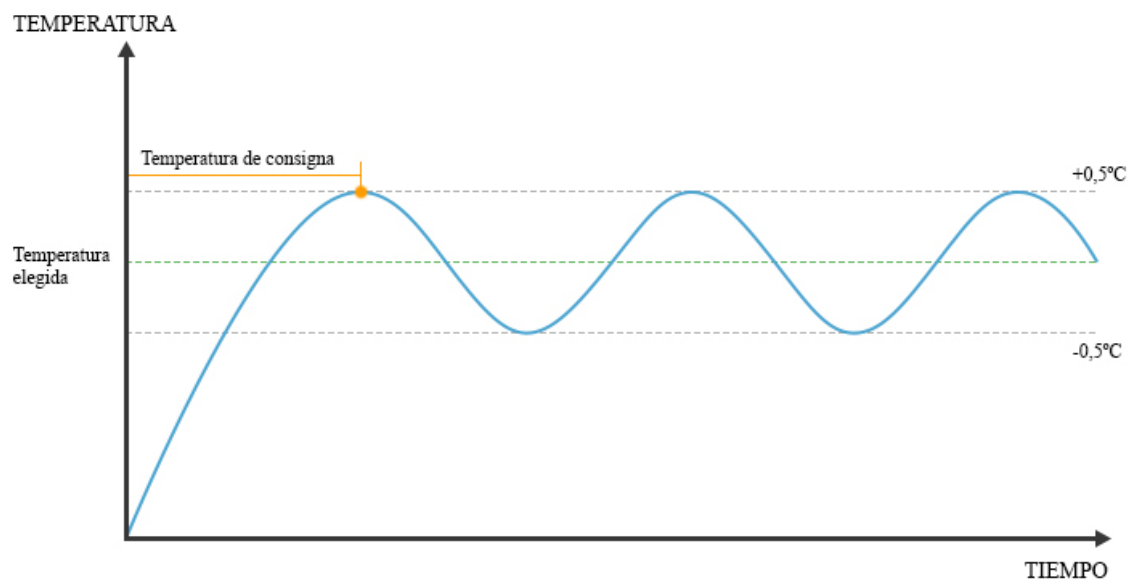


Ilustración 1. Gráfico de temperatura respecto al tiempo

Fuente: Elaboración propia

Para realizar el proyecto, interesa conocer la diferencia de temperatura que existe desde el encendido de la máquina hasta la temperatura consigna. Por esta misma razón, se ignorarán aquellas bases de datos donde se realice un apagado anticipado por parte del usuario, ya que no permite llegar a la temperatura consigna.

Además, cabe destacar, que se realizarán tres estudios de forma paralela, uno con bases de datos de invierno, otro con bases de datos de verano, y otro mixto con el conjunto de ambas. De esta forma se podrá estudiar si el mismo modelo sirve para todas las estaciones del año, o, por el contrario, si hacen falta dos modelos que dependan de si lo que se quiere es enfriar la habitación o calentarla.

Antes de usar cualquier modelo predictivo, es importante entender que no todas las características de las bases de datos serán relevantes para la predicción, por lo que se va a llevar a cabo un análisis para identificar cuáles de ellas influyen en mayor medida a los resultados.

Estas características se seleccionarán como datos de entrada para los modelos predictivos. Estos datos se usan para entrenar los modelos y para realizar predicciones.

Entre la gran diversidad de modelos predictivos que existen, en este proyecto se van a estudiar cuatro de ellos. Estos modelos son:

- El Modelo de Regresión multivariante,
- El Modelo de Árboles de decisión (Decision Trees),
- El Modelo de Bosques aleatorios (Random Forest),
- Y el Modelo de XGBoost.

Para obtener el mejor modelo, se usará una serie de métricas para evaluar los rendimientos de los modelos. Estas métricas son

- El Coeficiente de Determinación (R^2),
- El Error Absoluto Medio (MAE),
- El Error Cuadrático Medio (MSE),
- Y la Raíz del Error Cuadrático Medio (RMSE).

3. LA EMPRESA

La empresa con la que se va a trabajar a lo largo del proyecto se llama Airzone. Esta empresa va a proporcionar los datos de unas instalaciones anónimas con las que se trabajará para encontrar el mejor algoritmo correspondiente. Tras ello, la empresa incorporará el algoritmo hallado en su sistema, permitiendo al usuario programar una temperatura en la estancia a una hora determinada.



Ilustración 2. Logo de Airzone

Fuente: Airzone Comunicación, 2022b

3.1. ¿Qué es Airzone?

Airzone forma parte de la Corporación Empresarial Altra, grupo empresarial con un gran sentido de la ética. Airzone se dedica a la investigación, el desarrollo, la fabricación y la comercialización de soluciones inteligentes para el control de todo tipo de instalaciones de climatización (*Airzone Comunicación, 2022*).

La empresa pone a disposición del usuario la posibilidad de climatizar zonas sin renunciar al confort y logrando una mayor eficiencia energética, lo cual supone un gran ahorro económico debido al elevado precio de la energía actualmente. Estas zonas a climatizar podrán ser entornos residenciales o dedicados al uso terciario.

Airzone trabaja en tres líneas de negocio que son fundamentales para adaptar el funcionamiento de los equipos a las necesidades de los usuarios: un sistema de climatización por zonas, la calidad de aire, y su sistema Aidoo, que permite controlar la climatización de zonas a través de una aplicación informática para dispositivos móviles o tabletas, consiguiendo de esta forma mayores facilidades para el cambio de temperatura deseado por los usuarios, y aumentando la comodidad y los recursos energéticos en hogares y negocios.

La empresa, tiene ya más de 20 años de experiencia, y se ha convertido en un referente en el sector debido a su apuesta por la sostenibilidad y el futuro tecnológico. Se encuentra constantemente en busca de mayor innovación y desarrollo de nuevas soluciones de climatización, ya que de esta forma consigue un mayor confort para los usuarios (*Airzone Comunicación, 2022c*)

Se apuesta por proporcionar un servicio completo para cualquier proyecto, desde su asesoramiento preliminar hasta su desarrollo posterior, proporcionando a los clientes tanto las herramientas como el software que les sea necesario. Estos sistemas son compatibles con las principales marcas de fabricantes y se integran con otros sistemas de control domóticos y de control de edificios actualmente tan de moda. Además, permite la integración de varios sistemas de control a los dispositivos de control.

Es conveniente este uso del sistema de zonificación para el aire acondicionado si se cumple algunas de estas condiciones:

- Que se disponga de un equipo de conductos que estén instalados en casa y se quiera mejorar el confort de la misma.
- Que sea imposible instalar una unidad interior para cada habitación, ya sea por límite de espacio, por ejemplo.
- Que se cuente con un presupuesto para la colocación de un equipo de conductos, pero no para una instalación múltiple de los mismos.

3.2. Ubicación de las instalaciones de la empresa

La empresa Airzone se sitúa en la localidad malagueña, en el Parque Tecnológico de Andalucía (PTA) o también llamado actualmente como Málaga TechPark. Este está especializado en el sector de las tecnologías de la información y la comunicación (TIC) y se encuentra situado en el distrito de Campanillas.

La empresa se encuentra en una de las calles principales del Parque Tecnológico, Calle de Marie Curie, número 21.



Ilustración 3. Ubicación de las instalaciones

Fuente: Google maps, (s.f.)

Málaga cuenta con un envidiable clima mediterráneo y ha provocado que el sector tecnológico crezca considerablemente en la ciudad causando que ya se empiece a ver como una apuesta para el futuro digital, (Recio Á.,2023). Por lo que podríamos decir que, al encontrarse la compañía en dicha ciudad, y exactamente en dicho parque, se favorece al ambiente innovador y al entorno tecnológico de la misma.

Sin embargo, esta fábrica se le ha quedado pequeña a Airzone, que se encuentra en un crecimiento formidable. La empresa ha adquirido otro terreno de 8500 metros cuadrados en el parque tecnológico con el que se pretende hacer otra fábrica de 5500 metros cuadrados, pretenden que se ejecute su apertura en abril o mayo en el siguiente año 2024 (Triguero N, 2022). Además, se espera que esta inversión ayude a crecer a la compañía tanto por Europa como por Estados Unidos.

3.3. Airzone y la eficiencia energética

A lo largo de estos últimos años, la eficiencia energética ha dejado de ser una opción para convertirse en un requisito, debido a la fuerte subida del precio de la luz.

Estudios demuestran que aproximadamente el 50% de la energía consumida se debe a los sistemas de climatización, siendo la principal causa de consumo energético. De esta forma, AIRZONE crea soluciones más eficientes desde el punto de vista energético,

ya que, al zonificar y controlar todos los parámetros de funcionamiento, como la humedad y la temperatura del aire, los consumidores pueden disfrutar al máximo del confort térmico y de la eficiencia del sistema de climatización, además esto último implica grandes ahorros de energía. Con ello, se logra un ahorro energético de entre un 30 y un 55 por ciento al usar de un sistema de regulación, este ahorro dependerá de la temperatura exterior y la carga parcial del equipo de producción térmica.

Con Airzone se permite maximizar el uso de los elementos que emiten la energía en forma de calor, logrando el máximo control y eficiencia, gracias a sus algoritmos.

Al conseguir una mayor sostenibilidad mediante la optimización del control térmico y la mejora de la eficiencia energética, se logra proteger el medio ambiente y luchar contra el cambio climático, ya que se usa la cantidad precisa de energía en cada habitación provocando una reducción del consumo del sistema.

Su sistema permite conseguir varias certificaciones energéticas que se le otorgan a construcciones sostenibles. Estas certificaciones se muestran a continuación:

- BREEAM (Building Research Establishment Environmental Assessment Methodology). Este sistema evalúa y certifica la sostenibilidad en la edificación de forma técnicamente más avanzada. En esta certificación se evalúan 10 categorías y Airzone puede mejorar 5 de ellas, que son: Gestión, Salud y bienestar, Energía, Contaminación, e Innovación.



Ilustración 4. Certificación BREEAM

Fuente: BREEAM® (2023, 20 enero).

- LEED (Leadership in Energy & Environmental Design). Es un sistema de certificación que tiene un reconocimiento internacional, y que se usa para el rendimiento medioambiental en la construcción o renovación de edificios, busca manejo razonable de la energía de materiales y del agua. En esta certificación se contemplan 8 criterios y Airzone puede mejorar alguno de ellos al igual que en

BREEAM, que son: Energía y Atmósfera, Calidad del Ambiente Interior, Innovación en diseño.



Ilustración 5. Certificación LEED

Fuente: Méndez y Méndez (2009)

- VERDE. A diferencia de las dos certificaciones mencionadas anteriormente, esta es de carácter nacional, de España. Es la certificación que desarrolla la GBCe (Green Building Council España). Estudia 7 áreas diferentes de las que Airzone puede influir en: Energía y Atmósfera, y Concepto de Calidad.



Ilustración 6. Certificación VERDE

Fuente: GBCe | Certificación VERDE, (s. f.)

- WELL Building Standard. Es una certificación que asegura que la estancia donde se ha certificado se preocupa y trabaja por el bienestar y la salud de los ocupantes de la misma. Al igual que la certificación VERDE, WELL consta de 7 áreas de actuación. El impacto de Airzone es causado en la característica de confort.



Ilustración 7. Certificación WELL Building Standard

Fuente: Moon (2022)

La obtención de estos certificados energéticos presenta una serie de ventajas, entre ellas se pueden destacar: la reducción del consumo de energía y el ahorro económico que esto supone; el aumento del valor del inmueble haciéndolo más atractivo a la hora de vender o alquilar; y sobre todo la ayuda a la sostenibilidad y mantenibilidad del planeta, ya que se colabora en la creación de un mundo más limpio y sostenible (*Energetico, C.*, 2021).

La instalación de Airzone siempre va a ser una buena opción a la hora de la obtención de certificaciones energéticas, que hacen las estancias más atractivas y rentables en cuanto al consumo de energía.

La empresa se encuentra siempre en busca de la mejora continua de sus equipos y de una mayor eficiencia energética para el beneficio de un ahorro energético considerable. Por ello, se realiza el desarrollo de este proyecto ya que, si se consigue obtener el tiempo que tarda la máquina desde su encendido hasta que alcance la temperatura objetivo, se conseguirá reducir y optimizar los tiempos de funcionamiento de la máquina eliminando los que no son requeridos.

4. BASES DE DATOS

Tal y como ya se ha mencionado, se necesitan unas bases de datos que sigan un modelo específico para obtener soluciones de calidad. Para lograrlo, se deben usar las bases de datos de una misma habitación perteneciente a una estancia residencial que tenga programado su encendido siempre a la misma hora, y que llegue a la temperatura consigna sin que el usuario precipite el apagado del equipo antes de que alcance dicha temperatura. En las bases de datos se tiene registrada la temperatura de la habitación cada cierto tiempo.

La empresa proporcionará las bases de datos en formato JSON, cuyas siglas vienen del inglés y significan JavaScript Object Notation, en español traducido como notación de objetos de JavaScript. JSON es un formato de intercambio de datos que resulta sencillo de interpretar y crear para las máquinas y fácil de leer y escribir para los programadores. Además, consta de una velocidad de procesamiento bastante alta.

Para la realización de este proyecto se usarán dos tipos de bases de datos, aquellas que pertenezcan a los meses de verano, y aquellas que pertenezcan a los meses de invierno. Dichos archivos proporcionarán los datos de cada una de las zonas. Cada zona lleva asignada un identificador único (ID). Para cada zona se tienen datos según el marco temporal, desde que arranca la climatización hasta que se para por haber alcanzado la temperatura consigna. Para cada zona y marco temporal existen dos tipos de base de datos. Ambos documentos proceden de colecciones distintas dentro de la base de datos de MongoDB, que se usan en el backend de la empresa, es decir, en el sistema corporativo usado en la dirección de la empresa. MongoDB es un sistema de base de datos de código abierto orientado a documentos, que guarda estructuras de datos BSON (una especificación similar a JSON) con un esquema dinámico que ofrece una integración de datos de forma fácil y rápida (*MongoDB*, s.f.).

Los dos tipos de bases de datos que se van a proporcionar en este proyecto son los siguientes:

- “*DeviceEvents.json*”: esta base de datos facilita información de todos los eventos que ocurren en el interior de la estancia en el periodo de tiempo en el que se alcanza la temperatura consigna.

- “*Weather.json*”: esta base de datos facilita información sobre el clima exterior de la ubicación donde se encuentra situada la zona en el periodo de tiempo correspondiente. Estos datos se obtienen a través de “Open Weather API”, que es un servicio en línea que se encarga de proporcionar datos meteorológicos globales a través de interfaces de programación de aplicaciones, esto incluye datos meteorológicos actuales, pronósticos, predicciones inmediatas y datos históricos para cualquier ubicación geográfica.

Ambos archivos son un array de objetos JSON.

A continuación, se explica el formato de los dos tipos de bases de datos.

4.1. Objetos de DeviceEvents.json

La base de datos **DeviceEvents.json** está compuesta por los siguientes objetos:

- ***ts***: es un objeto que contiene información de la fecha de dicha muestra. El objeto se compone de una propiedad “*\$date*”, que a su vez es un objeto con una única propiedad “*\$numberLong*”, cuyo valor es la fecha de la muestra en formato UNIX. UNIX es un sistema que se usa para la descripción de instantes de tiempo, se define como la cantidad de segundos transcurridos desde la medianoche UTC (Tiempo Universal Coordinado), que serían las 12:00 am, del día 1 de enero de 1970, sin contar los segundos intercalares, es decir, los ajustes de segundos que se producen para mantener los estándares de emisión de tiempo cercanos al tiempo solar medio.

- ***meta***: es un objeto que contiene información de la zona para su referencia, así como el nombre del parámetro que ha cambiado:

- *d_id*: identificador de la zona
- *ws_id*: identificador del sistema al cual pertenece la zona.
- *param*: nombre del parámetro de la zona que ha cambiado. Los parámetros que se encuentran en la base de datos son:
 - *power*: para el encendido o apagado de la zona
 - *active*: para el arranque o la parada de la máquina

- *humidity*: para la humedad
 - *local_temp*: para la temperatura ambiente medida por el termostato
 - *setpoint_air_heat*: nos proporciona la temperatura proporcionada por el usuario, es la temperatura que el usuario desea, esa a la que tiene que llegar el sistema de climatización.
- **value**: es el nuevo valor del parámetro. Los valores cambian en cuanto a tipo dependiendo de “*meta.param*”. Para cada tipo de parámetro, los valores que se pueden tomar son:
- *power*: es un parámetro booleano, es decir, solo admite dos respuestas posibles. Estas respuestas son: “*true*” para el encendido y “*false*” para el apagado.
 - *active*: es un parámetro booleano también, con respuestas “*true*” para el arrancado y “*false*” para la parada.
 - *humidity*: es un número que indica el porcentaje, de 0 a 100, de humedad.
 - *local_temp*: es un objeto con dos propiedades:
 - *Celsius*: proporciona la temperatura en unidades centígrados con una precisión de 0.1 grados.
 - *Fah*: proporciona la temperatura en unidades Fahrenheit, con precisión de 1 grado.
 - *setpoint_air_heat*: al igual que *local_temp* es un objeto con dos propiedades:
 - *Celsius*: proporciona la temperatura en unidades centígrados con una precisión de 0.1 grados.
 - *Fah*: proporciona la temperatura en unidades Fahrenheit, con precisión de 1 grado.
 - **old**: es el valor anterior al cambio. Tiene el mismo formato que el campo *value*.
 - **id**: es un campo interno de MongoDB para referenciar cada entrada.

4.2. Objetos de Weather.json

La base de datos **Weather.json** está compuesta por los siguientes objetos:

- **ts**: es igual que para “*DeviceEvents.json*”. La fecha indica el clima previsto para esa hora. Se toma esta fecha como inicio de la previsión para la siguiente hora. Por ejemplo, si la hora que indica en este campo es las 08:00, entonces el documento se refiere a la previsión climática que ocurre entre las 08:00 y 09:00.

- **location_id**: es el objeto que identifica la ubicación, solo es usado para uso interno como referencia cruzada.

- **weather**: es un objeto que contiene la información climática para dicha zona. En este encontramos:

- *temp*: es un objeto que indica la temperatura media exterior en grados centígrados y Fahrenheit.
- *feels_like*: es un objeto que indica la temperatura aparente media exterior también en grados centígrados y Fahrenheit.
- *pressure*: es un número que indica la presión atmosférica a nivel del mar en hectopascales, hPa.
- *humidity*: es un número que indica la humedad de la zona en porcentaje.
- *dew_point*: es un objeto que indica la temperatura atmosférica correspondiente al punto de rocío.
- *uvi*: es un número que indica la intensidad de radiación ultravioleta proveniente del sol en la superficie terrestre, el índice UV.
- *clouds*: es un número que indica el porcentaje de nubosidad de 0 a 100.
- *visibility*: es un número que indica la visibilidad media en metros.
- *wind_speed*: es un número que indica la velocidad media del viento en metros por segundo, m/s.
- *wind_deg*: es un número que indica la dirección del viento en grados.
- *wind_gust*: es un número que indica la velocidad de las rachas de viento en metros por segundo, m/s.

- *pop*: es un número que indica la probabilidad de lluvia, su rango de valores va de 0 a 1. Es 0 cuando existe un 0% de probabilidad de lluvia y es 1 cuando existe un 100% de probabilidad de lluvia.
- *weather*: es un objeto que contiene descripción textual del tiempo, para el uso en la aplicación.

Todos estos objetos constituyen las distintas bases de datos proporcionadas por la empresa Airzone para realizar este proyecto.

4.3. Ejemplos

A continuación, se muestran como ejemplo dos pequeñas partes de cada una de las bases de datos ya mencionadas con una breve explicación de lo ocurrido en cada parte de la misma.

4.3.1. Ejemplo “DeviceEvents.json”

Se muestran dos imágenes para la explicación de la base de datos que proporciona información del interior de la vivienda.

1	[{				
2	"ts": {				
3	"\$date": {				
4	"\$numberLong": "1671093727538"				
5	}				
6	},				
7	"meta": {				
8	"d_id": "3a1b0c80058f5a04e07fc0a4",				
9	"param": "power",				
10	"ws_id": "E8:4F:25:E4:87:FD"				
11	},				
12	"old": false,				
13	"value": true,				
14	"_id": {				
15	"\$oid": "639adddfa7fd28f9d1fbdd83"				
16	}				
17	}, {				
18	"ts": {				
19	"\$date": {				
20	"\$numberLong": "1671093727701"				
21	}				
22	},				
23	"meta": {				
24	"d_id": "3a1b0c80058f5a04e07fc0a4",				
25	"param": "active",				
26	"ws_id": "E8:4F:25:E4:87:FD"				
27	},				
28	"_id": {				
29	"\$oid": "639adddfa7fd28f9d1fbddcc"				
30	},				
31	"value": true,				
32	"old": false				
33	}, {				

En esta parte se muestra la fecha y hora de la muestra en formato UNIX.

Objeto que muestra el identificador de la zona "d_id"; el parámetro cambiado "power" que indica el encendido o apagado de la zona; y el identificador del sistema al cual pertenece la zona "ws_id".

Antes ("old"), "power" estaba en false, es decir, la máquina estaba apagada, y en este momento ("value") cambia a true, lo que indica el encendido de la máquina.

Se vuelve a mostrar la nueva fecha y hora en formato UNIX.

El identificador de la zona y el sistema son los mismos ("d_id" y "ws_id"). El parámetro cambiado va a ser "active", que indica el arranque o la parada de la máquina.

Muestra la referencia de la entrada. Se ve que antes la maquina estaba en un estado de parada, false, y que ahora ha arrancado, true.

Ilustración 8. Base de datos DeviceEvents.json (1)

Fuente: Elaboración propia

33 v	}, {	
34 v	"ts": {	
35 v	"\$date": {	
36	"\$numberLong": "1671093788036"	En esta parte se muestra la nueva fecha y hora de la muestra en formato UNIX.
37	}	
38	},	
39 v	"meta": {	
40	"d_id": "3a1b0c80058f5a04e07fc0a4",	El identificador de la zona y el sistema son los mismos ("d_id" y "ws_id"). El parámetro cambiado va a ser "local_temp", que nos indica la temperatura en Celsius y en Fahrenheit.
41	"param": "local_temp",	
42	"ws_id": "E8:4F:25:E4:87:FD"	
43	},	
44 v	"value": {	
45	"celsius": 17.7,	Se muestra que antes ("old"), la temperatura marcaba 17.8°C y que ahora marca 17.7°C. También se proporcionan estos datos en grados Fahrenheit. Además se muestra la referencia de la entrada ("_id").
46	"fah": 64	
47	},	
48 v	"old": {	
49	"celsius": 17.8,	
50	"fah": 64	
51	},	
52 v	"_id": {	
53	"\$oid": "639ade1ca7fd28f9d1fc0c13"	
54	}	
55 v	}, {	
56 v	"ts": {	
57 v	"\$date": {	
58	"\$numberLong": "1671093911709"	Se vuelve a mostrar la nueva fecha y hora en formato UNIX.
59	}	
60	},	
61 v	"meta": {	
62	"d_id": "3a1b0c80058f5a04e07fc0a4",	El identificador de la zona y el sistema son los mismos ("d_id" y "ws_id"). El parámetro cambiado va a ser "humidity", el cual nos indica el porcentaje de humedad que hay en la habitación.
63	"param": "humidity",	
64	"ws_id": "E8:4F:25:E4:87:FD"	
65	},	
66	"value": 51,	Antes el porcentaje de humedad se encontraba en 52, y ahora en un 51%. Se vuelve a mostrar la referencia de la nueva entrada.
67	"old": 52,	
68 v	"_id": {	
69	"\$oid": "639ade97a7fd28f9d1fc66ab"	
70	}	
71 v	}, {	

Ilustración 9. Base de datos DeviceEvents.json (2)

Fuente: Elaboración propia

La base de datos sigue proporcionando valores de temperatura y humedad continuamente hasta que un dato proporcione el valor de la temperatura consigna que sea indicada. En este caso, la base de datos mostrará cómo se desactiva el parámetro “active” dándole un valor de “false”, que indica la parada de la máquina.

La máquina se para (active → false) cuando la estancia se encuentra 0.5 grados por encima de la temperatura que el usuario desea. Gracias a esta parada, logramos que la temperatura no supere en 0.5 grados Celsius a la deseada en la estancia. Una vez la temperatura empiece a bajar de nuevo, la máquina volvería a arrancar, dándole a “active” un valor de “true”. Sin embargo, este estudio y las bases de datos

proporcionadas se van a enfocar en la primera parada de la máquina, ya que esta información sirve para el resto.

4.3.2. Ejemplo “Weather.json”

The diagram shows a JSON object for weather data. It is divided into three main sections by brackets on the right:

- Time and Location:** Lines 2-6. Includes `"ts": { "date": { "numberLong": "1671091200000" } }` and `"location_id": { "id": "60f54872fcb21406bc672c78" }`. Annotation: "Nos indica la fecha y la hora en la que se va a mostrar la información del clima." and "Objeto que muestra la ubicación donde se va a dar lugar el clima del exterior de la estancia."
- Weather Data:** Lines 10-39. Includes temperature, feels like, pressure, humidity, dew point, uvi, clouds, visibility, wind speed, wind degree, wind gust, and a list of weather conditions. Annotation: "Toda esta parte contiene información climática que hemos explicado anteriormente en este mismo punto."
- Probability of Precipitation:** Line 39. `"pop": 0.78`.

```
1 v [{
2 v   "ts": {
3 v     "date": {
4 v       "numberLong": "1671091200000"
5 v     }
6 v   },
7 v   "location_id": {
8 v     "id": "60f54872fcb21406bc672c78"
9 v   },
10 v  "weather": {
11 v    "temp": {
12 v      "celsius": 12.2,
13 v      "fah": 54
14 v    },
15 v    "feels_like": {
16 v      "celsius": 11.7,
17 v      "fah": 53
18 v    },
19 v    "pressure": 997,
20 v    "humidity": 86,
21 v    "dew_point": {
22 v      "celsius": 9.9,
23 v      "fah": 50
24 v    },
25 v    "uvi": 0.03,
26 v    "clouds": 100,
27 v    "visibility": 10000,
28 v    "wind_speed": 8.9,
29 v    "wind_deg": 89,
30 v    "wind_gust": 13.98,
31 v    "weather": [
32 v      {
33 v        "id": 804,
34 v        "main": "clouds",
35 v        "description": "overcast clouds",
36 v        "icon": "04d"
37 v      }
38 v    ],
39 v    "pop": 0.78
40 v  },
41 v }, {
```

Ilustración 10. Base de datos Weather.json

Fuente: Elaboración propia

La información climática de esta base de datos se repite continuamente cada vez que pasa una hora, tal y como se ha explicado anteriormente en el apartado 4.2. Objetos de Weather.json.

Con ambas bases de datos se entrenarán los modelos que determinarán el tiempo en el que la habitación logre alcanzar la temperatura consigna demandada por el usuario.

Sin embargo, no todos estos datos que proporcionan estos archivos son relevantes para el estudio. Las bases de datos proporcionan mayor cantidad de datos de los que realmente se necesitan en el proyecto. En consecuencia, muchas variables son irrelevantes en el modelo, por lo que previamente al estudio y análisis de algoritmos predictivos se tendrán que evaluar los pesos de cada una de las variables en el modelo, descartando aquellas que resulten despreciables para el estudio.

5. LENGUAJE DE PROGRAMACIÓN

El lenguaje de programación que se va a usar para la obtención del algoritmo correspondiente es Python.

El creador del mismo es Guido Van Rossum y lo lanzó en 1991 aunque lo comenzó a desarrollar a finales de los ochenta. Se trata de un lenguaje de alto nivel, ya que expresa los algoritmos del lenguaje de programación de acuerdo con nuestras capacidades cognitivas haciendo que el ser humano sea capaz de entenderlos, a diferencia de los lenguajes de bajo nivel que expresan los algoritmos de acuerdo a la capacidad de las máquinas de ejecutar órdenes. Se trata de un lenguaje con gran facilidad de aprender, escribir y leer. Python, a diferencia de otros lenguajes de programación, está compuesto por una programación versátil y flexible que es capaz de adaptarse a todo tipo de proyectos.

Es uno de los lenguajes de programación más importantes en nuestros días. Su código ha sido utilizado para el desarrollo de aplicaciones de todo tipo, como son Instagram, Spotify, Netflix, Google, YouTube o Pinterest (*Bahit, E., 2020*).



Ilustración 11. Logo de Python

Fuente: Welcome to Python.org (2023)

Una de las razones por las que se va a hacer uso de este lenguaje de programación es porque se han proporcionado las bases de datos en formato JSON, y Python consta de una librería JSON que viene incorporada por defecto en el lenguaje. Además, con dicho lenguaje se podrán instalar una serie de librerías ya creadas que harán la ejecución de los programas de forma más fácil y llevadera (*KeepCoding, R., 20 enero 2022*).

5.1. Librerías usadas

Las librerías en programación son un conjunto de archivos de código que se usan para el desarrollo de un software. De esta forma se facilita la programación y proporciona funcionalidades comunes que ya han sido resueltas por programadores anteriormente (*datos.gob.es*, 2022).

En este proyecto se han empleado diferentes tipos de librerías ya creadas por otros programadores. Entre ellas se encuentran las siguientes:

- Librería “*os*”: esta librería proporciona una forma versátil a la hora de manipular y acceder a un sistema de archivos en un sistema operativo. Es útil a la hora de hacer uso de las rutas correspondientes a distintos archivos del sistema operativo. (*OS — interfaces misceláneas del sistema operativo — documentación de Python - 3.10.12*, s. f.)
- Librería “*Pandas*”: esta librería está diseñada para la manipulación y el análisis de datos en el lenguaje Python. Es una librería flexible que permite cargar, limpiar, manipular y transformar los datos (*Pandas - Python Data Analysis Library*, s. f.).
- Librería “*NumPy*”: esta librería trabaja con arrays que permiten representar colecciones de datos de un mismo tipo en varias direcciones, además de realizar operaciones matemáticas. Es útil a la hora de realizar cálculos con los datos o para manipular matrices (*NumPy*, s. f.).
- Librerías “*matplotlib*” y “*seaborn*”: ambas librerías resultan útiles a la hora de crear visualizaciones estáticas, animadas e interactivas en Python y para realizar análisis exploratorios (EDA) y evaluar la importancia de las características (*Matplotlib — visualization with Python*, s. f.) (*Seaborn: Statistical Data Visualization — Seaborn 0.12.2 documentation*, s. f.).
- Librería “*scikit-learn*”: esta librería sirve para el aprendizaje automático en Python. Incluye gran variedad de modelos y técnicas de evaluación y selección de características. Ejemplos podrían ser regresión lineal, árboles de decisión, bosques aleatorios, validación cruzada, métricas de evaluación y otros métodos de selección de características (*SciKit-Learn: Machine Learning in Python — SciKit-Learn 0.16.1 documentation*, s. f.).

- Librería “*XGBoost*”: esta librería es una implementación eficiente de máquinas de aumento de gradiente. Consta de un alto rendimiento y alta velocidad a la hora de tratar grandes y complejos conjuntos de datos grandes (*XGBoost Documentation — XGBoost 1.7.6 documentation*, s. f.).
- Librería “*JSON*”: esta librería proporciona un formato ligero de intercambio de datos que está inspirado en la sintaxis literal del objeto JavaScript (*json — Codificador y decodificador JSON*, s. f.).
- Librería “*Openpyxl*”: esta librería es muy útil a la hora de leer y escribir archivos en formato Microsoft Excel (*Ltd*, 2023).

6. APRENDIZAJE AUTOMÁTICO O MACHINE LEARNING

El Aprendizaje Automático o Machine Learning es una disciplina del campo de la Inteligencia Artificial (IA) que proporciona a los ordenadores la capacidad de identificar patrones en datos masivos y elaborar predicciones usando el análisis predictivo a través de algoritmos. Con esto, los ordenadores pueden realizar tareas específicas de forma autónoma. (*Corporativa, Iberdrola*, s. f.).

El análisis predictivo es uno de los usos más frecuentes en el Machine Learning al ser una herramienta muy útil para las empresas. En este proyecto, se pretende aplicar una serie de algoritmos predictivos ya existentes que sean capaces de predecir el tiempo que tarda una habitación en alcanzar una temperatura. Para ello, existen modelos predictivos en los que se usan un conjunto de datos de entrenamiento y otro conjunto más reducido para realizar pruebas y validar la precisión del modelo con el fin de realizar los mejores pronósticos. (*Las 11 técnicas más utilizadas en el modelado de análisis predictivos - Insight | Keyrus*, s. f.).

Los algoritmos de aprendizaje automático construyen un modelo que representa el conocimiento extraído de una serie de datos facilitados a la entrada del mismo (*Charte*, s. f.). Dependiendo de la información adicional que se le suministre al algoritmo, se pueden diferenciar dos tipos diferentes de modelos de aprendizaje automático:

- **Aprendizaje automático supervisado.**

En el aprendizaje supervisado se le va indicando al algoritmo si la salida que ha generado para un cierto caso (una predicción) es correcta o no. Para ello, se necesitan unos datos de entrada y salida etiquetados durante la fase de entrenamiento, luego se realizan pruebas a partir de datos nuevos e invisibles para predecir los resultados.

Este tipo de entrenamiento se usa para clasificar archivos como imágenes, documentos o palabras escritas y para pronosticar tendencias y resultados a través de patrones en los datos. Se le llama aprendizaje supervisado porque se requiere supervisión humana para etiquetar con precisión los datos disponibles (*Universidad Europea*, 2022).

- **Aprendizaje automático no supervisado.**

En el aprendizaje no supervisado, la única información que se le entrega al algoritmo son muestras de datos, sin detalles. En este tipo de aprendizaje los datos están sin procesar ni etiquetar, no necesitan intervención humana, excepto a la hora de establecer los parámetros del modelo y la cantidad de puntos de clúster (agrupaciones de datos con características similares).

El aprendizaje automático no supervisado es capaz de procesar grandes conjuntos de datos de forma efectiva. Se suele usar principalmente para clúster, para realizar análisis de datos iniciales, y para entender la relación entre diferentes puntos de datos (*Universidad Europea, 2022*).

Para entender un poco mejor el aprendizaje no supervisado, se expone un ejemplo donde se tienen un conjunto de datos sobre información de frutas, como manzanas, melones y naranjas. El aprendizaje no supervisado agrupa automáticamente estas frutas en clústers basados en sus similitudes, como el tamaño, el color y la textura. El algoritmo podría descubrir que las manzanas y las naranjas se agrupan en un clúster debido a su tamaño similar, mientras que los melones se agruparían en otro.

- **Diferencias.**

A continuación, se expone una serie de ideas principales que ayudan a comprender las diferencias de ambos tipos de aprendizaje supervisado (*Universidad Europea, 2022*).

- Los modelos en el aprendizaje supervisado necesitan datos etiquetados.
- Los modelos del aprendizaje supervisado requieren muchos más recursos debido a dicha necesidad.
- El modelo de aprendizaje supervisado se usa para clasificar datos o hacer predicciones mientras que el aprendizaje no supervisado se usa para comprender las relaciones dentro de los conjuntos de datos.

- **Tipos de aprendizaje supervisado.**

Se puede hacer una subdivisión del aprendizaje supervisado, ya que existen dos tipos: el aprendizaje supervisado por clasificación y el aprendizaje supervisado por regresión (*De Ceupe*, s. f.).

A continuación, se explican ambos aprendizajes supervisados:

a) Aprendizaje supervisado por regresión.

Es un entrenamiento de los algoritmos que se usa para predecir un rango de valores posibles (*De Ceupe*, s. f.). Permite predecir un valor real basándose en las entradas pasadas de dicho entrenamiento. Un ejemplo de este tipo de aprendizaje puede ser un modelo de regresión lineal donde se busque determinar la relación entre una variable dependiente, con respecto a otras variables.

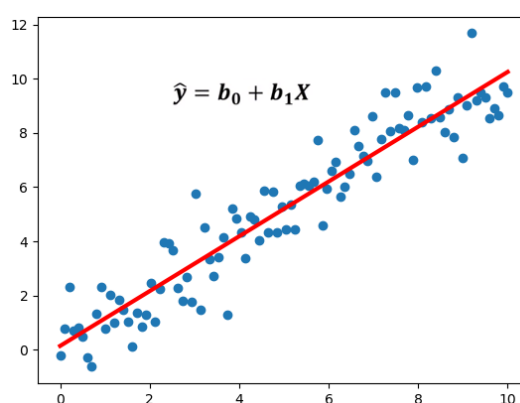


Ilustración 12. Ejemplo de aprendizaje supervisado por regresión

Fuente: Roldán, (2022)

b) Aprendizaje supervisado por clasificación.

Los algoritmos de este tipo de aprendizaje supervisado tratan de etiquetar a los ejemplos eligiendo entre dos o más clases (*De Ceupe*, s. f.). Usa información aprendida de unos datos de entrenamiento para elegir la etiqueta correcta. Un ejemplo de aprendizaje supervisado por clasificación sería el de la siguiente imagen.

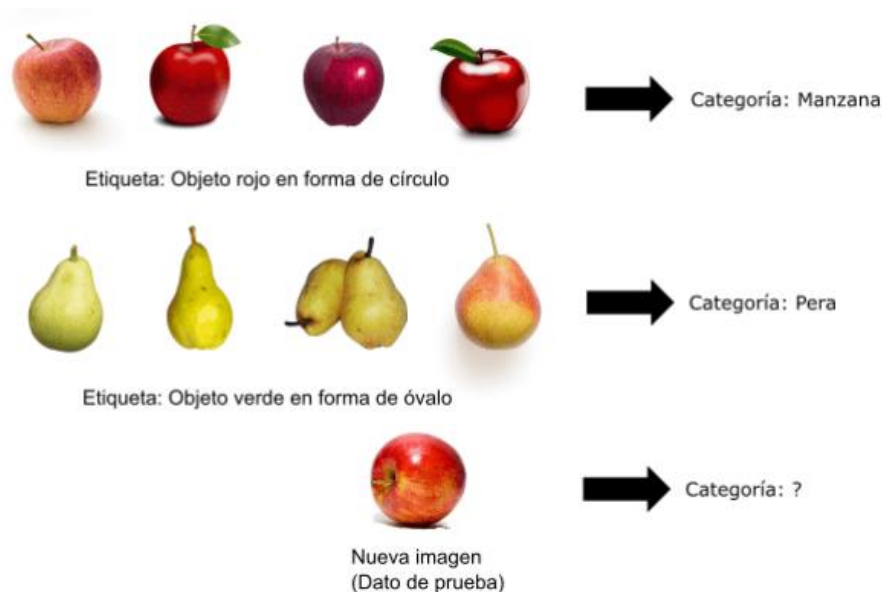


Ilustración 13. Ejemplo de aprendizaje supervisado por clasificación.

Fuente: De Ceupe, (s. f.)

En los datos de entrenamiento encontramos que el primer grupo de elementos tiene categoría “manzana” con etiqueta “objeto rojo en forma de círculo”; y en el segundo grupo de elemento tiene categoría “pera” con etiqueta “objeto verde en forma de óvalo”. Por lo tanto, al hacer la prueba y detectar un objeto rojo en forma de círculo, se le dará el nombre de “manzana” (*De Ceupe, s. f.*).

En este proyecto se hace uso de un modelo de aprendizaje supervisado ya que se realizan predicciones de tiempo. Exactamente el proyecto se encuentra en un aprendizaje supervisado por regresión, ya que predecimos un valor basándonos en unos datos de entrenamiento (*De Ceupe, s. f.*).

7. INTRODUCCIÓN AL CONTENIDO DEL PROYECTO

Para explicar el proyecto se divide el contenido del mismo en tres partes, que vendrán diferenciadas por las bases de datos usadas en cada una de ellas. Cada parte corresponde a una carpeta. Los nombres de las carpetas son los siguientes:

- “python_tfg”: en esta carpeta se incluyen los datos de la instalación en los meses de invierno.
- “python_tfg_verano”: en esta carpeta se incluyen los datos de la instalación en los meses de verano.
- “python_tfg_conjunto”: en esta carpeta se incluyen los datos de ambas carpetas anteriores, tanto los datos de verano como los de invierno.

El contenido de los archivos de cada una de las carpetas es prácticamente el mismo, lo único que cambia en ellas son las bases de datos usadas. El contenido de cada una de estas tres carpetas se puede dividir en tres secciones, que son “Preprocesamiento y Split”, “Selección de variables” y “Ejecución de modelos y evaluación”.



Ilustración 14. División de contenido

Fuente: Elaboración propia

A continuación, se realiza una breve introducción de los tres bloques que componen el proyecto.

a) Preprocesamiento y Split.

Al disponer de los datos en brutos y en formato JSON, se necesita realizar un preprocesamiento que permita su uso de forma sencilla y eficaz. Para ello, se han eliminado datos no necesarios y se han creado otras variables a partir de otras definidas

en las bases de datos, se han normalizado y se han creado varios archivos en formato TSV que se explicarán más adelante.

b) Selección de variables.

Tras realizar un estudio de correlación de las variables de X, se ha detectado que dos de las variables pueden ser cofactores. Por lo tanto, en este bloque, se van a crear varios archivos en formato TSV donde se escojan distintas variables que vendrán dadas de tres formas: según un estudio de correlación, escogiendo la totalidad de las variables y aplicando método de selección de variables llamado Lasso (Least Absolute Shrinkage and Selection Operator).

c) Ejecución de modelos y evaluación.

En este bloque se realiza usa cada uno de los archivos TSV del bloque anterior para realizar una ejecución de los modelos objetos de estudio: Regresión multivariante, Árbol de decisión, Bosques aleatorios y XGBoost. Tras este entrenamiento se someten a un test donde se evalúan los errores por medio del coeficiente de determinación (R^2), el Error Cuadrático Medio (MSE), el Error de Raíz Cuadrada Media (RMSE) y el Error Absoluto Medio (MAE).

Los siguientes esquemas muestran cada uno de los tres bloques (A, B y C, D). En ellos, se presenta el recorrido del proyecto, los archivos que se han ido creando y lo que devuelven cada uno de ellos en formato resumen, dichos archivos estarán incluidos en cada una de las carpetas (“python_tfg”, “python_tfg_verano”, “python_tfg_conjunto”). Para entender de forma clara y esquemática los puntos 9, 10 y 11 del proyecto, es conveniente considerar las tres imágenes que se muestran a continuación.

Más adelante, se justificará detenidamente la función desarrollada por cada código creado y sus respectivos archivos de salida. Para ello se explicará principalmente la carpeta de “python_tfg” con las bases de datos de invierno, ya que es la carpeta origen creada desde un principio y con la que se han creado las otras dos.

A PREPROCESAMIENTO Y SPLIT

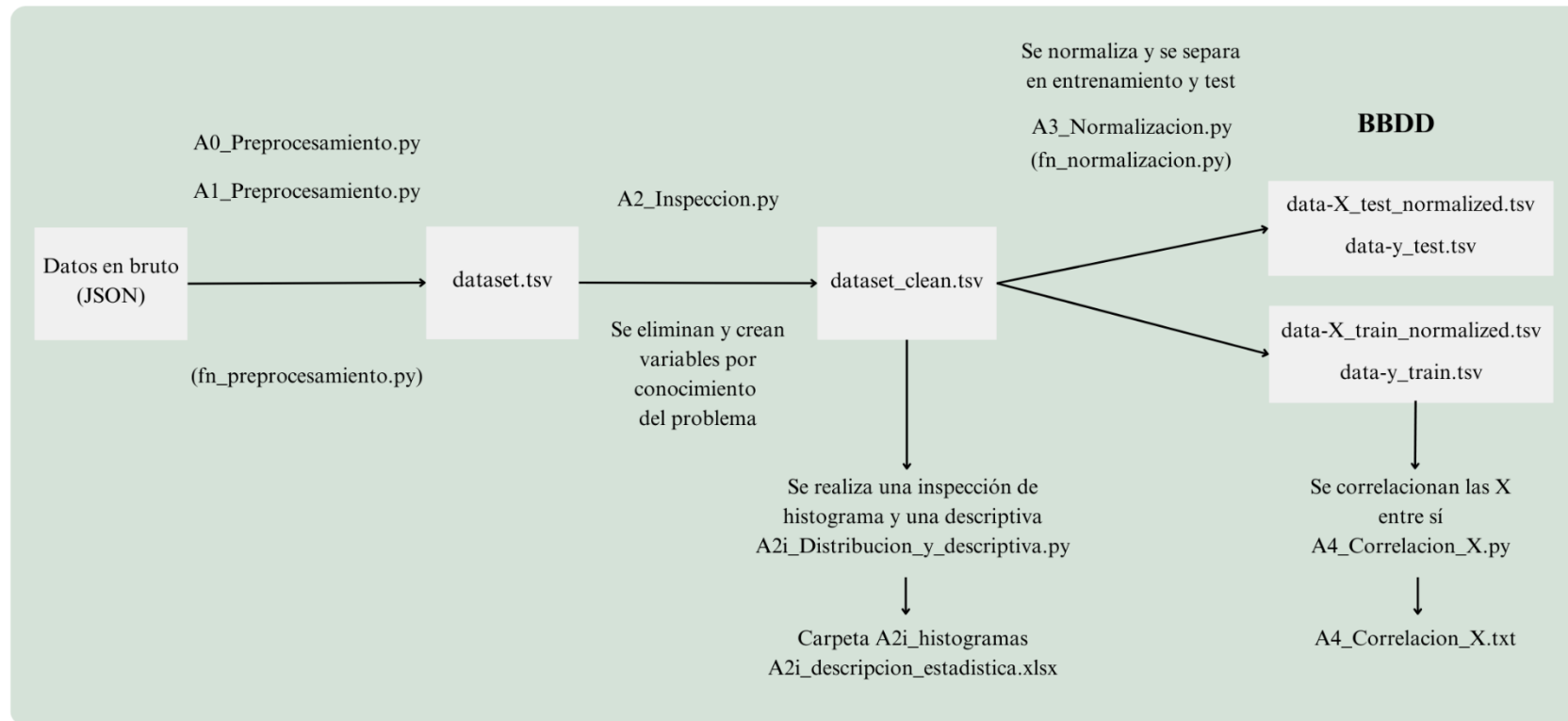


Ilustración 15. Esquema A - Preprocesamiento y Split

Fuente: Elaboración propia

B y C SELECCIÓN DE VARIABLES

'feels_like_activeTrue' y
'temp_activeTrue' tienen
una correlación de
0.997.
Es muy cercana a 1.
Pueden ser cofactores.



Se realizan 2 estudios:
B) BBDD: base de datos
al completo.
C) BBDD_corr_filter:
base de datos sin
'feels_like_activeTrue'

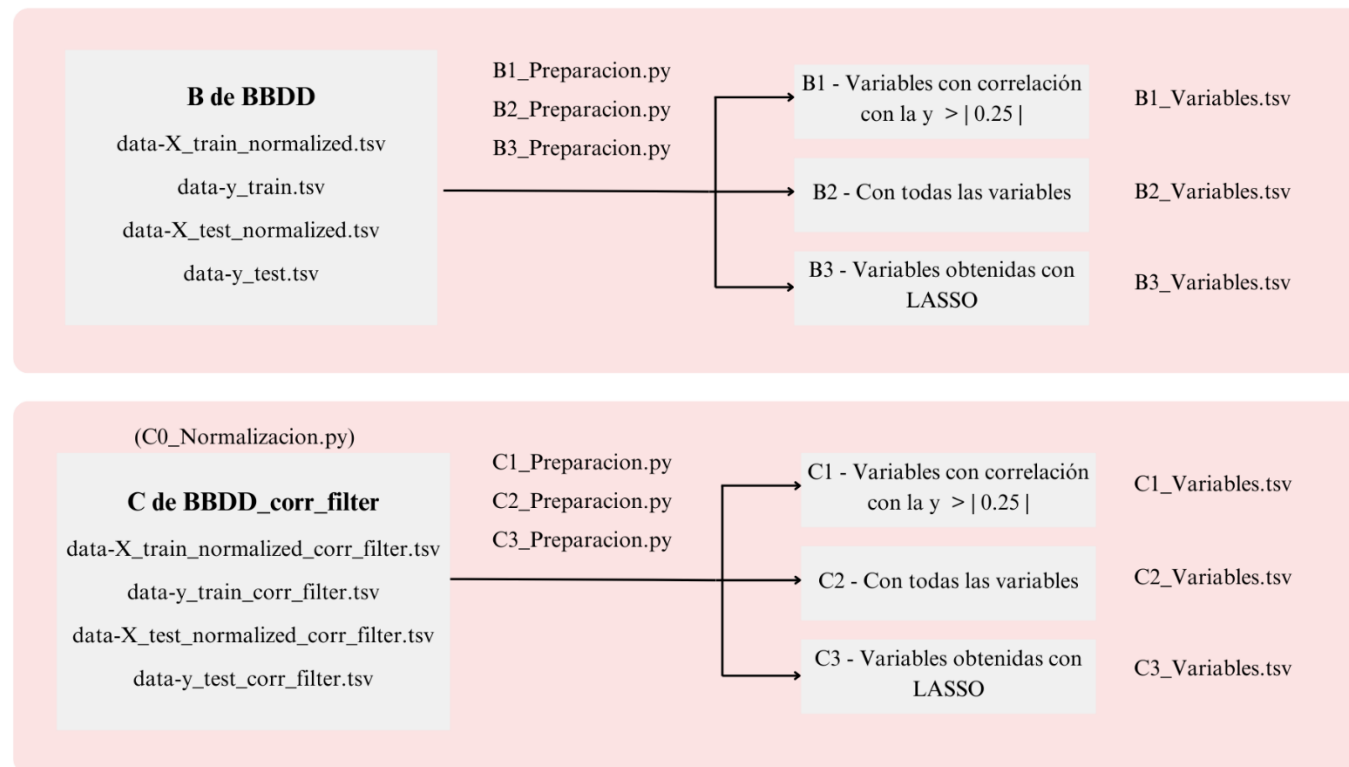


Ilustración 16. Esquema B y C - Selección de variables

Fuente: Elaboración propia

D

EJECUCIÓN DE MODELOS Y EVALUACIÓN

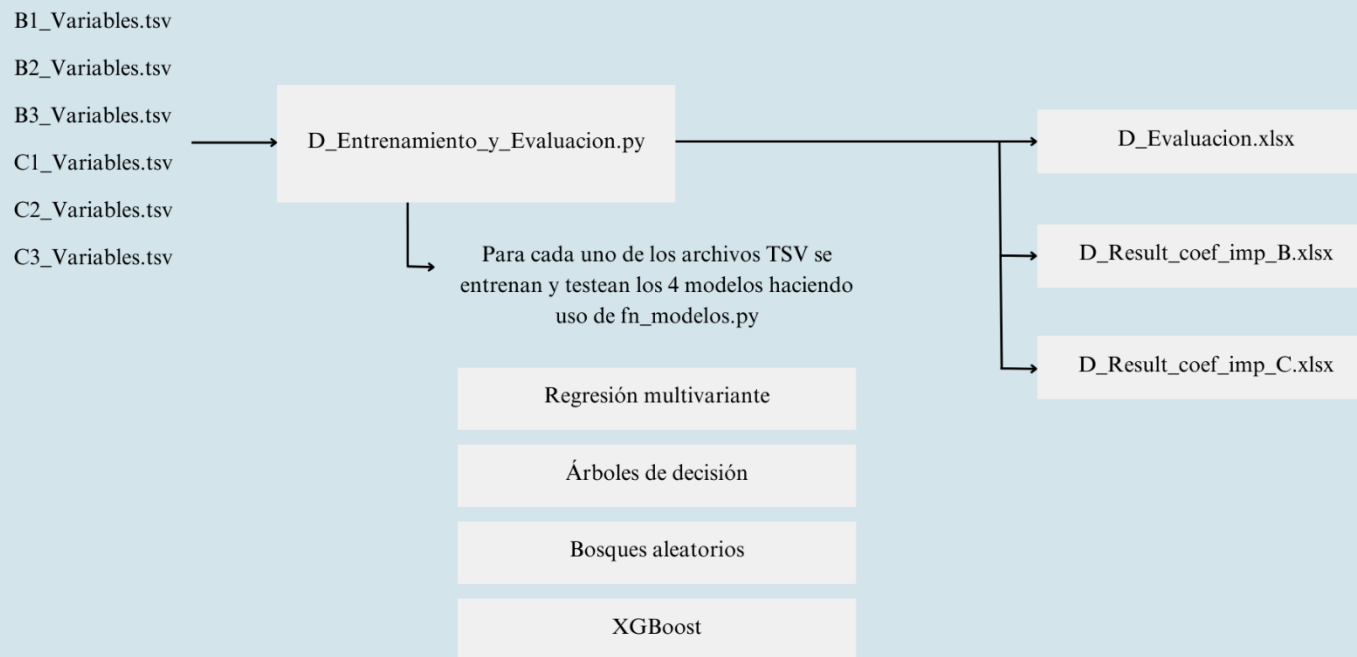


Ilustración 17. Esquema D - Ejecución y Evaluación

Fuente: Elaboración propia

8. PASO PREVIO - ARCHIVOS DE TEXTO

Como ya se ha mencionado en el punto anterior, la estructura de las tres carpetas va a ser la misma, lo único que cambian en ellas son las bases de datos. Para poder incluir, eliminar o cambiar estas bases de datos se va a hacer uso de dos archivos de texto (.txt) que mostrarán las rutas de las bases de datos de DeviceEvents.json y de Weather.json, respectivamente.

A continuación, se muestran algunas de las rutas relativas de la base de datos DeviceEvents.json.

```
listado-DeviceEvents.txt
1  datos/3a1b0c80058f5a04e07fc0a4/2022-12-15T08_42_07.538_2022-12-15T11_53_06.579/DeviceEvents.json
2  datos/3a1b0c80058f5a04e07fc0a4/2022-12-18T06_36_26.558_2022-12-18T12_10_01.835/DeviceEvents.json
3  datos/3a1b0c80058f5a04e07fc0a4/2022-12-20T11_38_50.115_2022-12-20T13_33_53.522/DeviceEvents.json
4  datos/3a1b0c80058f5a04e07fc0a4/2022-12-22T17_05_01.853_2022-12-22T19_29_40.582/DeviceEvents.json
5  datos/3a1b0c80058f5a04e07fc0a4/2022-12-23T05_00_19.633_2022-12-23T08_28_07.753/DeviceEvents.json
6  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T08_15_35.516_2022-12-25T10_41_27.719/DeviceEvents.json
7  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T15_38_00.574_2022-12-25T17_40_43.284/DeviceEvents.json
8  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T05_00_49.972_2022-12-28T06_52_29.793/DeviceEvents.json
9  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T10_49_39.071_2022-12-28T11_33_45.886/DeviceEvents.json
```

Ilustración 18. Listado-DeviceEvents.txt

Fuente: Elaboración propia

De la misma forma, se muestra algunas de las rutas de Weather.json.

```
listado-weather.txt
1  datos/3a1b0c80058f5a04e07fc0a4/2022-12-15T08_42_07.538_2022-12-15T11_53_06.579/weather.json
2  datos/3a1b0c80058f5a04e07fc0a4/2022-12-18T06_36_26.558_2022-12-18T12_10_01.835/weather.json
3  datos/3a1b0c80058f5a04e07fc0a4/2022-12-20T11_38_50.115_2022-12-20T13_33_53.522/weather.json
4  datos/3a1b0c80058f5a04e07fc0a4/2022-12-22T17_05_01.853_2022-12-22T19_29_40.582/weather.json
5  datos/3a1b0c80058f5a04e07fc0a4/2022-12-23T05_00_19.633_2022-12-23T08_28_07.753/weather.json
6  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T08_15_35.516_2022-12-25T10_41_27.719/weather.json
7  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T15_38_00.574_2022-12-25T17_40_43.284/weather.json
8  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T05_00_49.972_2022-12-28T06_52_29.793/weather.json
9  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T10_49_39.071_2022-12-28T11_33_45.886/weather.json
```

Ilustración 19. Listado-Weather.txt

Fuente: Elaboración propia

Con ello, se consiguen una variedad de ventajas, seguidamente se muestran algunas de ellas:

- Una mayor organización de las rutas.

- Una mayor automatización del proceso, lo cual permite la automatización de tareas como la manipulación, procesamiento o iteración sobre cada una de las rutas, en lugar de tenerlas que escribir manualmente en el código.
- Una mayor flexibilidad en la configuración: ya que al almacenar todas las rutas en los dos archivos separados permite modificarlas o añadir otras nuevas rutas de nuevas bases de datos sin tener que modificar directamente el código del programa.
- Un ahorro de tiempo ya que se define la ruta una sola vez, y al abrir y cerrar el archivo se evitan problemas de memoria o conflictos al intentar acceder al archivo posteriormente.

Además, de todas las ventajas expuestas, cabe destacar, que el establecimiento de estos dos archivos de texto hace más ameno el proyecto ya que, una vez se tenga uno de los tres estudios realizado y completo, sólo se tiene que copiar la misma carpeta, hacer unos cambios en las rutas en los nuevos archivos e incluir las nuevas bases de datos en la carpeta para crear los otros dos estudios faltantes.

Es decir, al acabar el estudio con las bases de datos de invierno, se podrían obtener los estudios de verano y mixto cambiando los viejos archivos JSON por los nuevos, realizando unos breves cambios en las rutas del código y los dos archivos de texto explicados (Listado-DeviceEvents.txt y Listado-Weather.txt).

En el Anexo A. listado-DeviceEvents.txt y listado-weather.txt, se muestra el contenido de los archivos de texto de Device-Events y de Weather, de invierno, de verano, y del conjunto de estos.

Con todo esto, se puede concluir que este paso previo proporciona beneficios de organización, automatización y flexibilidad, además de ahorrar tiempo y evitar posibles errores futuros.

9. SECCIÓN A: PREPROCESAMIENTO Y SPLIT

De acuerdo con las bases de datos proporcionadas por la empresa, es necesario el desarrollo de un preprocesamiento de cada uno de los tres bloques (invierno, verano y mixto) que permita el posterior trabajo de los datos. Para ello, se ha hecho uso de Python, tal y como se ha comentado en el punto 5. Lenguaje de Programación.

Para poder trabajar con los datos proporcionados realizamos una extracción de los mismos.

9.1. Tipos y recuento de parámetros en DeviceEvents.json

Primeramente, se procede a identificar los diferentes tipos del atributo “*param*” para las bases de datos DeviceEvents.json., además de realizar un recuento de cada uno de ellos. Para ello se crea un programa de Python junto con sus correspondientes explicaciones tal y como se muestra en el Anexo B. A0_Preprocesamiento.py – “python_tfg”.

Dicho programa devuelve el siguiente diccionario para las bases de datos de invierno:

```
{'power': 12, 'active': 24, 'local_temp': 248, 'humidity': 176, 'setpoint_air_heat': 15}
```

Con este resultado se identifica que el parámetro “power” se encuentra 12 veces en las bases de datos de DeviceEvents.json; “active” 24 veces; “local_temp” 248 veces, “humidity” 176 veces; y “setpoint_air_heat” 15 veces.

Al haber más “setpoint_air_heat” que bases de datos se puede concluir que en algunas de las bases existen dos o más temperaturas objetivo. Esto se debe a que puede que el usuario haya cambiado la temperatura objetivo sin permitir que la habitación llegue a dicha temperatura. Sin embargo, otras bases de datos no incluyen este parámetro.

Por lo tanto, se va a definir la temperatura objetivo como aquella más cercana a la parada de la máquina, Active → False.

Usando el mismo código en las otras carpetas, se obtienen los siguientes resultados para verano y para el conjunto de invierno y verano, respectivamente.

Para verano: {'power': 17, 'setpoint_air_cool': 13, 'active': 34, 'humidity': 297, 'local_temp': 331}

Para conjunto: {'power': 29, 'active': 58, 'local_temp': 579, 'humidity': 473, 'setpoint_air_heat': 15, 'setpoint_air_cool': 13}

De igual forma, se elige como temperatura objetivo la más cercana a la primera parada de la máquina cuando Active es False.

9.2. Funciones de preprocesamiento

En el archivo de Python que se muestra en el Anexo C se crean dos funciones auxiliares que servirán para el posterior preprocesamiento de ambas bases de datos. Estas funciones se encargan de mostrar por resultado los datos a usar de los ficheros JSON. Ambas funciones reciben como parámetro una cadena que indica la ruta del archivo que contiene los conjuntos de datos JSON.

Estas funciones no devuelven nada por pantalla porque en ningún momento se le ha ordenado la imprimación de los datos, simplemente es un paso previo al siguiente fichero mostrado en el Anexo D, son funciones auxiliares que se van a usar posteriormente.

Los datos devueltos por resultado en la primera función de preprocesamiento_DeviceEvents son los siguientes:

- “zona”: La zona donde se trabaja.
- “fecha”: La fecha de la toma de datos.
- “temp_inicial”: La temperatura inicial del interior.
- “humidity_inicial”: La temperatura humedad del interior.
- “temp_final”: La temperatura final del interior.
- “humidity_final”: La humedad final del interior.
- “temp_objetivo”: La temperatura objetivo.
- “ts_activeTrue”: La fecha correspondiente a Active → True.

- “ts_activeFalse”: La fecha correspondiente a Active → False.

Los datos que devueltos por resultado en la segunda función de preprocesamiento _Weather son los siguientes:

- “zona”: Al igual que en preprocesamiento _DeviceEvents, nos devuelve la zona.
- “fecha”: Al igual que en preprocesamiento _DeviceEvents, nos devuelve la fecha.
- “tiempo_lista”: Una lista con los tiempos.
- “temp_lista”: Una lista con las temperaturas del exterior.
- “feels_like_lista”: Una lista con las sensaciones térmicas del exterior.
- “pressure_lista”: Una lista con las presiones del exterior.
- “humidity_lista”: Una lista con las humedades del exterior.
- “dew_point_lista”: Una lista con las temperaturas atmosféricas correspondientes al punto de rocío.
- “uvi_lista”: Una lista con las intensidades de radiación ultravioletas.
- “clouds_lista”: Una lista con los porcentajes de nubosidad.
- “visibility_lista”: Una lista con el número de visibilidad media.
- “wind_speed_lista”: Una lista con las velocidades medias del viento.
- “wind_deg_lista”: Una lista con las direcciones del viento en grados.
- “wind_gust_lista”: Una lista con las velocidades de rachas de viento.
- “pop_lista”: Una lista con las probabilidades de lluvia.

Es decir, ambas funciones procesan los archivos en formato JSON y devuelven una serie de datos que se usan posteriormente en el siguiente punto 9.3. Procesamiento de los JSON y obtención de los datos.

9.3. Procesamiento de los JSON y obtención de los datos

En este punto se crea el programa correspondiente con el que se obtienen los datos en un dataset. Un dataset representa un conjunto completo de datos tabulados. Estos datos serán almacenados en un archivo TSV con el que se podrá trabajar de forma cómoda y eficiente.

Además, en vez de pasar las listas de la función `preprocesamiento_Weather`, se va a hacer uso de los valores más cercanos al arranque y parada de la máquina, es decir, cuando `Active` $\rightarrow$ `True`, y `Active` $\rightarrow$ `False`.

En el Anexo D, se muestra el programa encargado de procesar los datos contenidos en archivos de texto haciendo uso de las funciones específicas definidas en el punto anterior 9.2. Funciones de preprocesamiento, este archivo se encarga de guardar todos los resultados de la función de `preprocesamiento_DeviceEvents` y los más cercanos al arranque y parada de la máquina de la función `preprocesamiento_Weather` en un archivo TSV.

Al compilar el código del Anexo D ya mencionado, se crea el siguiente archivo en formato TSV mostrado en la imagen. En la imagen únicamente aparece una parte recortada de 13 bases de datos de invierno. Esto se repite de forma simultánea en las otras dos carpetas, con las bases de datos de verano y con las bases de datos de ambas estaciones del año.

```
dataset.tsv
1 zona fecha temp_inicial temp_final humidity_inicial humidity_final temp_objetivo tiempo_activeTr
2 3a1b0c80058f5a04e07fc0a4 2022-12-15T08_42_07.538 2022-12-15T11_53_06.579 17.7 20.6 51 53 None 167
3 3a1b0c80058f5a04e07fc0a4 2022-12-18T06_36_26.558 2022-12-18T12_10_01.835 15.1 21 43 48 20.5 1671345
4 3a1b0c80058f5a04e07fc0a4 2022-12-20T11_38_50.115 2022-12-20T13_33_53.522 18 21 59 52 20.5 16715363301
5 3a1b0c80058f5a04e07fc0a4 2022-12-22T17_05_01.853 2022-12-22T19_29_40.582 16.8 20.3 56 52 None 167
6 3a1b0c80058f5a04e07fc0a4 2022-12-23T05_00_19.633 2022-12-23T08_28_07.753 16.8 20.1 59 55 19.5 167
7 3a1b0c80058f5a04e07fc0a4 2022-12-25T08_15_35.516 2022-12-25T10_41_27.719 16.4 20.8 54 59 20.5 167
8 3a1b0c80058f5a04e07fc0a4 2022-12-25T15_38_00.574 2022-12-25T17_40_43.284 20.8 22.1 56 55 21.5 167
9 3a1b0c80058f5a04e07fc0a4 2022-12-28T05_00_49.972 2022-12-28T06_52_29.793 17.8 19.6 47 43 19 1672203
10 3a1b0c80058f5a04e07fc0a4 2022-12-28T10_49_39.071 2022-12-28T11_33_45.886 18.5 19.4 47 46 None 167
11 3a1b0c80058f5a04e07fc0a4 2023-02-17T10_58_30.262 2023-02-17T13_23_37.626 17.2 21.6 44 37 21 1676631
12 3a1b0c80058f5a04e07fc0a4 2023-02-24T13_56_16.612 2023-02-24T18_06_05.010 19.2 21.5 44 41 21 1677247
13 3a1b0c80058f5a04e07fc0a4 2023-02-25T08_43_43.267 2023-02-25T11_00_58.488 16.9 21 50 45 22 16773146232
```

Ilustración 20. Dataset almacenado en TSV

Fuente: Elaboración propia

Por lo tanto, ya se puede trabajar de forma fácil en Python con este `dataset.tsv`. A partir de ahora, no se volverá a hacer uso de los archivos en formato JSON

proporcionados por la empresa Airzone, y se usará el conjunto de datos extraídos y tabulados para trabajar.

9.4. Creación y eliminación de variables por conocimiento del problema.

Una vez que se ha realizado la extracción de los datos, es necesario realizar una limpieza de las variables que se van a usar en el estudio, ya que muchas de las seleccionadas en el dataset.tsv no influirán en los modelos y no es conveniente almacenarlas. Además, es conveniente crear una serie de variables por conocimiento del problema.

Como ya se ha mencionado a lo largo del documento, el objetivo de este proyecto es realizar un estudio para poder predecir el tiempo que tarda una habitación en alcanzar una temperatura. Por lo que se van a descartar los datos de cuando Active pasa a ser False, ya que estos datos no se tendrán cuando se vaya a realizar la predicción de dicho tiempo. Se estudiarán las distintas variables con la diferencia de tiempo que ocurre entre el arranque de la máquina (Active → True) y la parada de la misma al llegar a la temperatura consigna (Active → False). Esta diferencia de tiempo será la variable dependiente a predecir, se le asignará el nombre “diff_tiempo” a esta nueva variable creada.

A continuación, se muestran todas las variables independientes del dataset_clean.tsv del estudio. Algunas las obtenemos directamente del dataset.tsv y otras nuevas son creadas al considerarse interesantes en el estudio.

- Diferencia “temp_final” y “temp_inicial”: Es interesante estudiar la diferencia de temperatura existente entre la temperatura interior final a la que se quiere llegar y la temperatura interior inicial en la que se encuentra la estancia, por lo que se crea esta nueva variable cuyo nombre asignado será “diff_temp_interior”.
- Diferencia “temp_inicial” y “temp_activeTrue”: También es interesante estudiar la diferencia de temperatura que existe entre la temperatura inicial del interior de la habitación y la temperatura del exterior de la misma, por lo que se crea esta nueva variable cuyo nombre asignado será “diff_temp_int_ext”.
- “humidity_inicial”: Se estudia la humedad inicial del interior de la estancia.

- “feels_like_activeTrue”: Se estudia la sensación térmica inicial del exterior.
- “pressure_activeTrue”: Se estudia la presión inicial del exterior.
- “humidity_activeTrue”: Se estudia la humedad inicial del exterior.
- “dew_point_activeTrue”: Se estudia la temperatura atmosférica inicial del exterior.
- “uvi_activeTrue”: Se estudia la intensidad de radiación ultravioleta inicial.
- “clouds_activeTrue”: Se estudia el porcentaje inicial de nubosidad.
- “visibility_activeTrue”: Se estudia el número de visibilidad media del exterior.
- “wind_speed_activeTrue”: Se estudia la velocidad media del viento.
- “wind_deg_activeTrue”: Se estudia la dirección del viento.
- “wind_gust_activeTrue”: Se estudia la velocidad de las rachas de viento.
- “pop_activeTrue”: Se estudia la probabilidad de lluvia.

Para ello, se usa el código mostrado en el Anexo E, donde se crean y se eliminan variables para obtener un nuevo dataset_clean.tsv. En el código mostrado en este anexo, se muestran en columnas_interes todas las variables mencionadas anteriormente. Estos datos variarán según las bases de datos con la que se esté trabajando.

Para crear el dataset_clean.tsv también interesa realizar una inspección y exploración de los datos de aquellas variables que se van a usar en el proyecto, realizando un análisis de distribución de cada una de ellas y obteniendo además una descriptiva de las mismas (Caballero, s. f.). De esta forma, se identificarán valores atípicos y ayudará a comprender la naturaleza de los datos con los que se están trabajando (*ANÁLISIS EXPLORATORIO DE DATOS*, s. f.). Esta inspección se realiza en el código del Anexo F, que da lugar a un Excel y una serie de histogramas, que variarán según la base de datos con la que se esté trabajando.

Los resultados de la descripción estadística de las bases de datos de invierno se muestran en el Anexo G, de las bases de datos de verano en el Anexo H, y del conjunto de ambas en el Anexo I. Se usan los siguientes estadísticos descriptivos comunes:

- “count”: indica el número de valores que se tienen de cada variable, es el número total de datos.

- “mean” (promedio): la media es el promedio aritmético de todos los valores en la variable. Se suman todos los valores y se divide por el número de observaciones. Es por así decirlo como el “centro” de la distribución de los datos.
- “std” (desviación estándar): Mide la dispersión de los datos con respecto a la media. Si es alta, los valores están dispersos. Si es baja, están muy agrupados alrededor de la media.
- “min” (valor mínimo): El valor mínimo presente en cada variable.
- “25%” (primer cuartil): Divide los datos en cuatro partes iguales y corresponde al valor que está en el 25% inferior a la distribución.
- “50%” (segundo cuartil o mediana): La mediana es el valor que divide los datos en dos partes iguales. Es una medida de tendencia central que no se ve afectada por valores atípicos de la misma forma que la media.
- “75%” (tercer cuartil): También divide los datos en cuatro partes iguales como en el primer cuartil, pero ahora corresponde en el 75% superior de la distribución.
- “max” (valor máximo): El valor máximo presente en cada variable.

Con todo esto se obtiene una visión completa de la distribución de los datos y de las características más importantes que tienen.

Por otro lado, en los Anexos J, K y L se encuentran las imágenes de los histogramas de cada una de las variables.

Gracias a este estudio de las variables, se observa que “*visibility_activeTrue*” proporciona siempre el mismo dato tanto en las bases de datos de invierno como en las de verano. Por lo tanto, se elimina esta variable para evitar ruido en los análisis, ya que al no tener variabilidad y permanecer constante puede afectar negativamente a los modelos de aprendizaje automático, en la precisión y en la capacidad de generalización de los análisis y modelos.

Esto también ocurre con “*pop_activeTrue*” en la base de datos de verano, por lo que la eliminamos del estudio de verano.

Para que estas variables no aparezcan en nuestro dataset_clean.tsv se tendrá que compilar de nuevo A2_Inspeccion.py eliminándolas de la lista de “*columns_interes*”.

9.5. Normalización y división de los datos en entrenamiento y test

Normalizar las bases de datos es importante ya que permite una mayor organización y estructura de los datos de forma lógica. De esta forma se podrá realizar mejores análisis de los datos al extraer información útil (*KeepCoding*, 2023).

Como en el proyecto se quiere encontrar el modelo que mejor encaje con los datos, se dividen las bases de datos en dos categorías: “train” y “testing”. Aproximadamente se dejará el 80% de los datos para entrenar el modelo, y el 20% restante para testarlo. Esto es importante ya que, si se entrenase el modelo usando de la totalidad de los datos sin dejar una parte de los mismos para su posterior testeo, en los resultados se obtendría un grado de accuracy (exactitud) enorme, en torno a un 99%. Sin embargo, si se cambia el modelo, este fracasaría vergonzosamente al exponerle un nuevo dataset. Esto ocurre debido al data leakage, ya que la información del test pasa al training dando lugar a resultados muy poco realistas (*Avb & Avb*, 2020). Esta es la razón por la que se va a realizar una diferencia de categorías entre aquellos datos que se usarán para el entrenamiento y los que se usarán para testear el modelo, así se evita que se pase información al set de prueba que ya se ha estudiado previamente en el set de entrenamiento, provocando que se realice el testeo con un conocimiento previo.

Hay que destacar que la normalización de los datos también se efectúa separando los conjuntos en prueba y entrenamiento. Es crucial que el modelo nunca antes haya tenido acceso a la información de los datos de prueba durante el entrenamiento, incluyendo información de la normalización de los datos. Si se normalizaran los datos de prueba y entrenamiento de forma conjunta se estaría filtrando información del conjunto de prueba en el proceso de normalización, lo que daría lugar a resultados poco realistas al evaluar el rendimiento del modelo en el conjunto de prueba.

Para realizar la normalización de los datos, se usan los valores mínimos y máximos de las columnas del dataset_clean.tsv, y se aplica la siguiente ecuación:

$$\text{Dato normalizado} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Esta ecuación se usa en el Anexo M, donde se crea una función auxiliar para realizar la normalización de los datos, teniendo por entrada los datos de entrenamiento y test la función devuelve estas dos categorías normalizadas.

Esta función se usa en el Anexo N, donde se separa en entrenamiento y prueba tanto la X como la Y. Se incorporan cuatro archivos en formato TSV dentro de una carpeta llamada “BBDD”:

- data-X_train_normalized.tsv: Se almacenan el 80% de los datos de entrenamiento normalizados de las variables.
- data-X_test_normalized.tsv: Se almacenan los datos de las variables que se usarán para testear el modelo.
- data-y_train.tsv: Se almacenan el 80% de los datos de entrenamiento de la diferencia de tiempo existente entre el apagado y encendido del dispositivo (“diff_tiempo”).
- data-y_test.tsv: Se almacenan los datos restantes de “diff_tiempo” para poder realizar el test.

En los datos de la Y, tanto de train como de test, se tiene la diferencia de tiempo que transcurre desde el arranque de la máquina hasta la parada de la misma, se le denomina “diff_tiempo”, no es necesario normalizar estos datos, a diferencia de los datos de X, donde las variables son normalizadas en el Anexo N usando la función auxiliar de normalización definida en el Anexo M.

En conclusión, ya se tienen los datos separados en X e Y, además de tenerlos en categorías de entrenamiento y test, con ello, se procura obtener mejores resultados. Todo este proceso realizado para las bases de datos de invierno se repite paralelamente y de idéntica forma para las bases de datos de verano y para la conjunta.

9.6. Análisis de correlación entre variables X

Tras hacer distinción de las variables (X e y) objetos de estudio con respecto a la diferencia de tiempo, es conveniente realizar un estudio de correlación entre las variables de X que determine la relación existente entre cada par de variables X (*Análisis de correlación*, 2017).

Para ello, se ha realizado el código que se muestra en el Anexo Ñ, donde se realiza un análisis de correlación entre las variables de X, estas correlaciones pueden tomar valores que oscilan entre -1 y +1. Si el signo es positivo se puede decir que existe una

relación positiva entre las variables, cuando la magnitud de una variable incrementa, la magnitud de la otra también lo hace. Si el signo es negativo se puede decir que existe una relación negativa, mientras los valores de una variable incrementan, los de la segunda disminuyen, y viceversa. (*Análisis de correlación*, 2017).

La fuerza de la relación lineal incrementa si el coeficiente de correlación se aproxima a -1 o a +1. Mientras tanto, cuanto más cercano a 0, más débil es la correlación entre el par de variables (*Análisis de correlación*, 2017).

El código del Anexo Ñ devuelve un archivo en TXT donde se muestra la correlación de cada par de variables. Además de un mapa de calor donde se muestra de forma gráfica y comprensible dichas correlaciones.

Una vez se obtengan los valores de las correlaciones, es necesario estudiar qué variables pueden ser cofactores. Dos variables son cofactores cuando están muy relacionadas entre sí y pueden influir en la variable dependiente del proyecto, pudiendo influir también en el resultado de los modelos creados. Por ello, es importante considerarlos en el análisis, ya que de esta manera se podrá comprender mejor las relaciones ocultas entre el par de variables X y se podrá estudiar las influencias del mismo.

En los Anexos O, P y Q se muestran las distintas correlaciones de las variables X de las bases de datos de invierno, verano, y el conjunto de ambas, respectivamente. Los valores de las tablas de los anexos se han sacado de los archivos TXT, que han sido representados en formato de tabla para mayor comprensión, ordenando la columna “Correlation” de mayor a menor.

En los mapas de calor se muestran de forma gráfica los valores de las correlaciones de cada par de variables X objetos de estudio. Los mapas se ven representados con una escala de colores que oscila desde -1 con un color azul intenso, a +1 con un color rojo intenso, siendo 0 el color blanco grisáceo.

A continuación, se muestran los mapas de calor creados para la base de datos de invierno, verano y del conjunto, respectivamente.

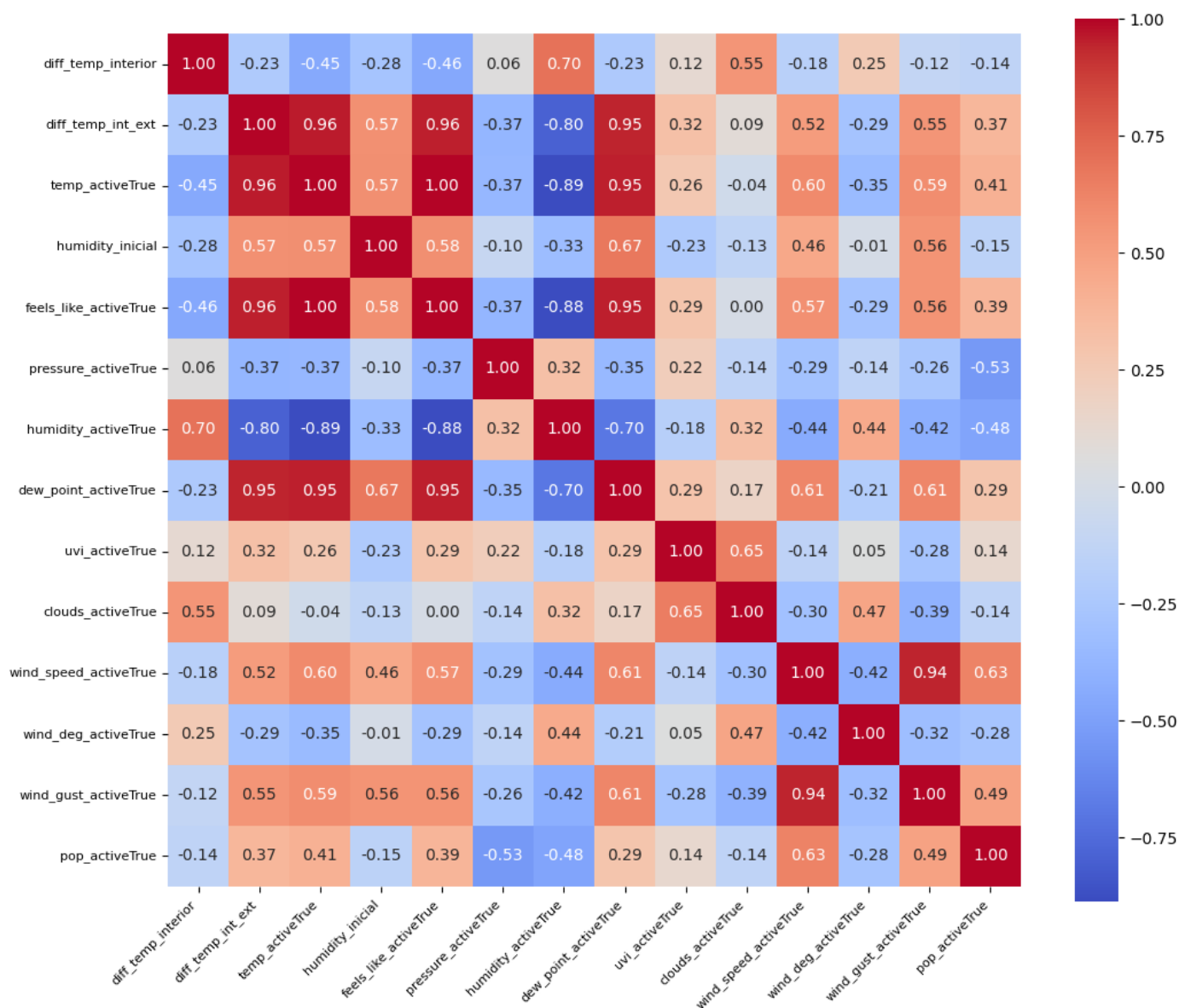


Ilustración 21. Mapa de calor correlaciones X de invierno

Fuente: Elaboración propia

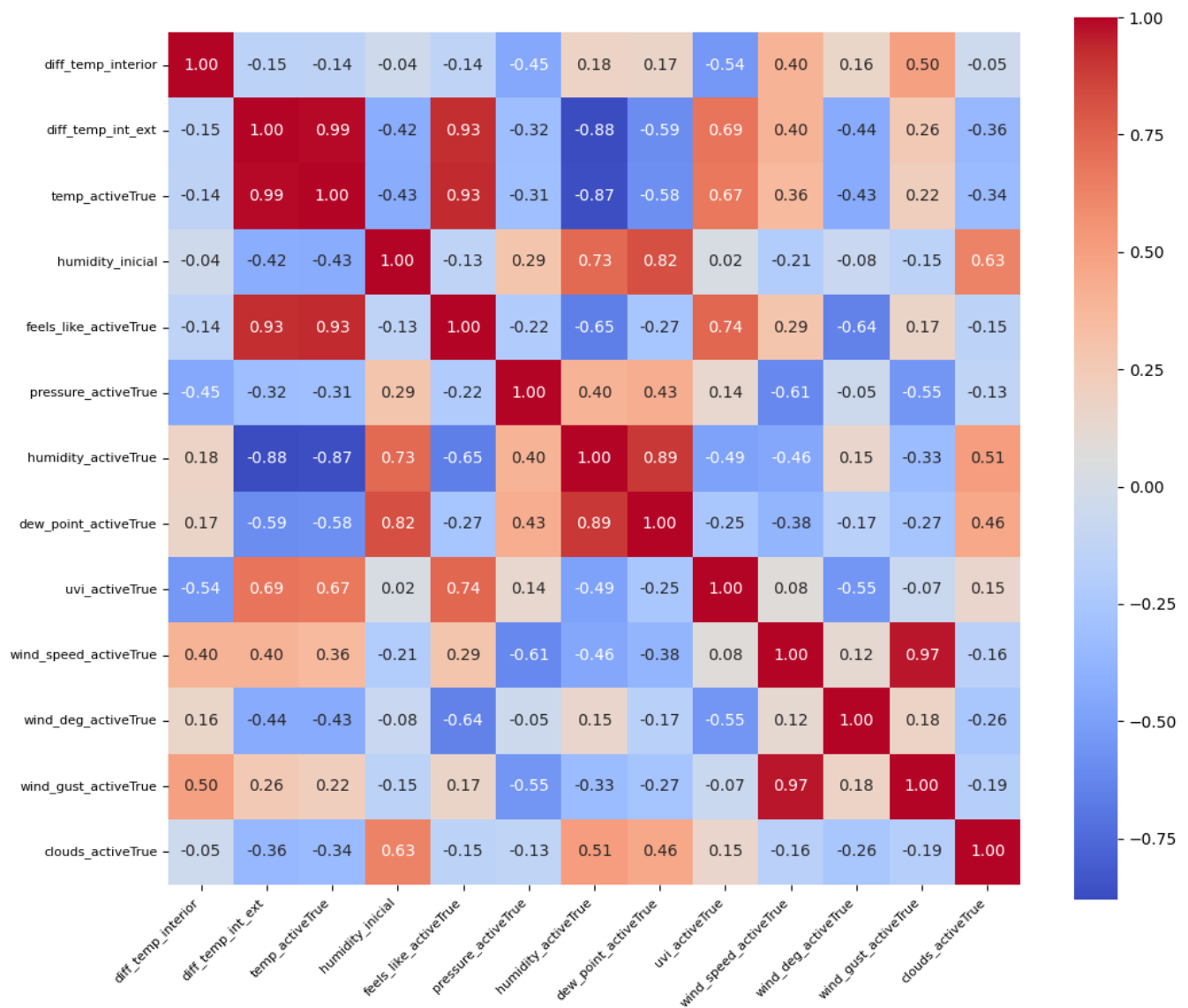


Ilustración 22. Mapa de calor correlaciones X de verano

Fuente: Elaboración propia

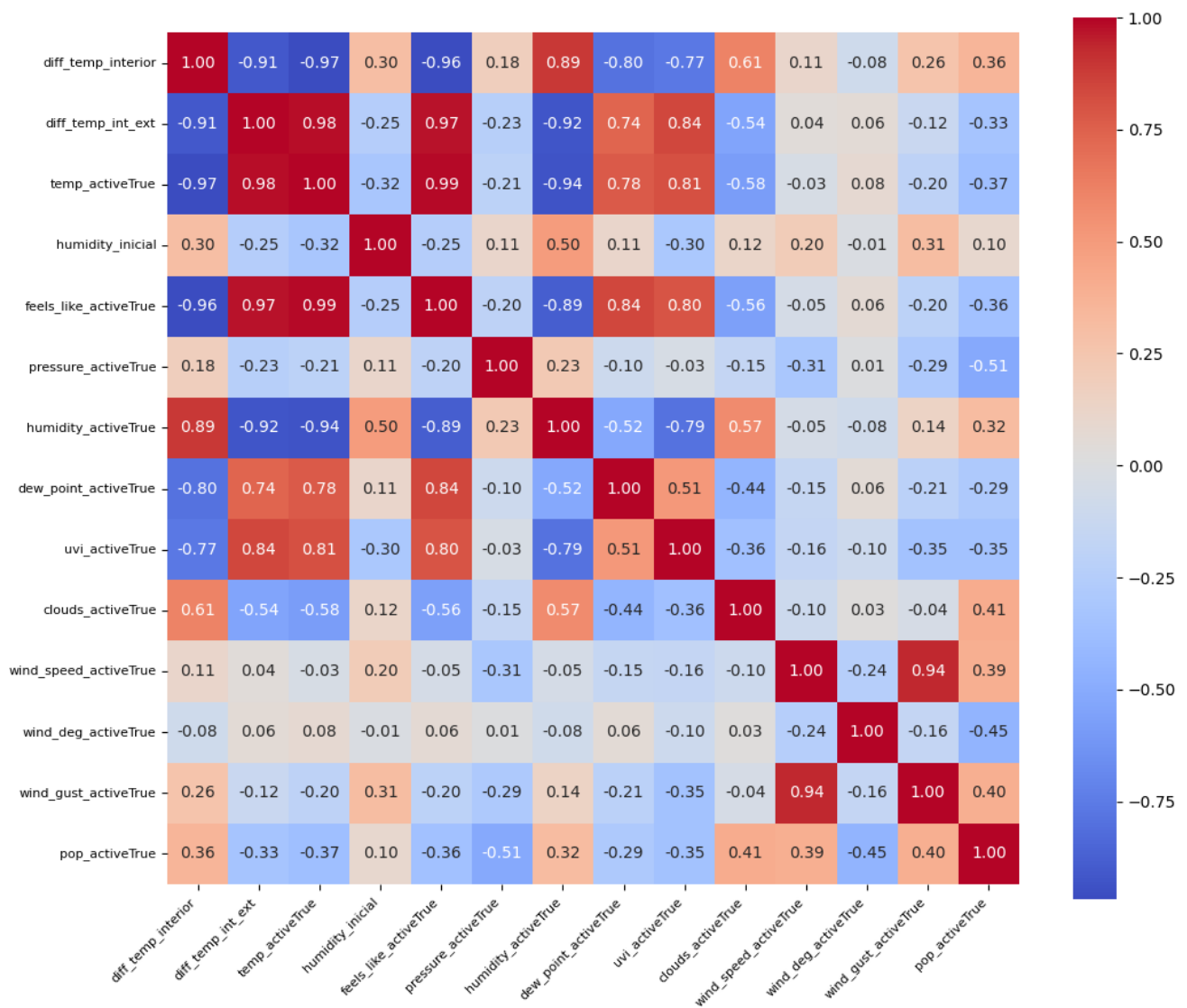


Ilustración 23. Mapa de calor correlaciones X del conjunto

Fuente: Elaboración propia

Se puede observar que, en las bases de datos de invierno las variables “feels_like_activeTrue” y “temp_activeTrue” tienen una correlación muy alta, siendo prácticamente 1, exactamente del valor de 0,997676 (Anexo O). La correlación entre estas dos variables también es muy alta en las bases de datos del conjunto de invierno y verano, con un valor de 0,993556 (Anexo Q). Es cierto que en las bases de datos de verano dicha correlación no es tan cercana a 1, siendo 0,894 (Anexo P), y no tienen una fuerza tan sumamente grande como en las dos bases de datos anteriores, aunque aun así sigue siendo muy alta.

Estas dos variables tienen un alto potencial de ser cofactores al ser prácticamente 1 la correlación entre ambas en dos de los tres estudios. Esto tiene sentido ya que son dos variables muy relacionadas entre sí. La temperatura es el grado de calor existente en la atmósfera y la sensación térmica es el grado de calor que experimentan los cuerpos que dependen de otros factores meteorológicos, como son la humedad o el viento, (*Sensación térmica - concepto, cálculo y factores que la afectan*, s. f.). Por consiguiente, cuando una aumenta, la otra tiende a aumentar, y viceversa.

Tras este análisis de correlación entre las variables X, se ha visto conveniente realizar dos estudios paralelos, uno de ellos incluyendo la sensación térmica en el conjunto de datos (“feels_like_activeTrue”), que corresponderá al estudio B, y otro de ellos sin la sensación térmica, que corresponderá al estudio C. Se ha elegido eliminar la sensación térmica y no la temperatura exterior porque se considera que la temperatura exterior es una variable que debe permanecer en el modelo, ya que la sensación térmica es una variable que dependen de otras como la humedad y el viento, cuyas dependencias ya están reflejadas en otras variables.

Una vez obtenidos los resultados de los distintos modelos, podremos ver y comparar cómo se comporta el modelo con y sin dicha variable, y evaluar si realmente su eliminación representa una diferencia elevada en los resultados de los modelos.

10. SECCIÓN B Y C: SELECCIÓN DE VARIABLES

En esta sección del proyecto se realizará un estudio previo de la importancia que tienen las variables en el modelo, y se seleccionarán aquellas que sean más relevantes. La selección de características es un proceso crucial en la construcción de modelos de aprendizaje automático y análisis de datos.

Al realizar el análisis de correlación entre las variables de X del punto anterior (9.6. Análisis de correlación entre variables X) se ha detectado que sería interesante realizar un estudio paralelo sin la variable de sensación térmica del exterior, “feels_like_activeTrue”. De este modo se tendrían dos estudios en esta sección:

- Estudio de la sección B: tiene por entrada las variables divididas de la carpeta “BBDD” que incluye la sensación térmica.
- Estudio de la sección C: se realizará el mismo estudio que en la sección B, pero sin la sensación térmica.

Para el estudio de la sección C se deberá de crear otra carpeta a la que se le denominará “BBDD_corr_filter”. Esta carpeta estará compuesta por los mismos archivos en formato TSV que en la sección B, pero eliminando dicha variable. Estos archivos TSV se crearán por medio del código mostrado en el Anexo R, donde se vuelve a normalizar los datos y a dividir las variables en “entrenamiento” y “test”, y en “X” e “y”, prescindiendo de la sensación térmica.

Los nombres de los archivos del interior de carpeta son los mismos que los usados en la sección B, pero le incorporamos al final “_corr_filter”, de esta forma tenemos los siguientes cuatro archivos que se crean por medio del código mostrado en el Anexo R:

- data-X_train_normalized_corr_filter.tsv
- data-X_test_normalized_corr_filter.tsv
- data-y_train_corr_filter.tsv
- data-y_test_corr_filter.tsv

Por lo tanto, se tienen dos secciones, B y C, y las dos carpetas que dividen el proyecto, “BBDD” y “BBDD_corr_filter”. Para cada una de estas dos secciones se crean tres archivos de Python donde se va a realizar un estudio de las variables, así se

podrá comparar y analizar cómo funcionan los distintos modelos según las variables que tomen de entrada los modelos.

El código de estos archivos se puede ver en los anexos S, X y BB para la sección B; y FF, JJ y NN para la sección C. Estos códigos se repiten para cada una de las carpetas; de invierno, verano y conjunto.

Existen varios métodos para evaluar el peso de las variables en los modelos. A continuación, se exponen qué métodos se usan y cómo están divididas las secciones B y C de los 3 estudios de invierno, verano y del conjunto de ambos:

- B1 y C1: En los primeros archivos de Python tanto para B como para C, se va a realizar un estudio de correlación de las variables, ya que habrá muchas de ellas que no interfieran en los modelos. Los códigos se muestran en los Anexos S y FF.
- B2 y C2: En los segundos archivos de Python, se van a usar la totalidad de las variables filtradas de las carpetas. Los códigos se muestran en los Anexos X y JJ.
- B3 y C3: En los terceros archivos de Python, se va a usar un método de selección de variables llamado LASSO, cuyas siglas del inglés corresponden a Least Absolute Shrinkage and Selection Operator (*Regression shrinkage and selection via the lasso on JSTOR*, s. f.). Los códigos se muestran en los Anexos BB y NN.

Estos códigos dan lugar a una serie de archivos en formato TSV donde se guardan los nombres de aquellas variables seleccionadas. En los segundos archivos se guardan los nombres de la totalidad de las variables filtradas. Además, devuelven una serie de gráficos donde se representa la correlación de cada una de las variables.

Por lo tanto, en total se obtendrán 6 archivos TSV con el nombre de las variables para cada una de las carpetas de invierno, verano y conjunto, siendo en total 18 archivos TSV representados en los Anexos T, U W, Y, Z, AA, CC, DD, EE, GG, HH, II, KK, LL, MM, ÑÑ, OO y PP.

En los apartados que vienen a continuación se exponen los resultados de cada método de selección de variables y de cada una de las bases de datos de invierno, verano y del conjunto de ambas.

10.1. Análisis de correlación y sus variables

Un análisis de correlación puede ser una parte esencial en el proceso de selección de características, ya que con ello se puede tomar decisiones que nos indiquen qué características incluir en el modelo.

Este análisis de correlación va a ser de gran ayuda para eliminar aquellas características que sean irrelevantes en el modelo, que serán aquellas que representen una correlación muy baja o nula con la variable objetivo. Gracias a esta información, se podrán eliminar estas variables del conjunto de datos, ya que son poco probables que aporten una información significativa al modelo. Si se reducen el número de características, se podrá simplificar el modelo, y esto dará lugar a una eficiencia mayor.

10.1.1. Estudio de invierno

A continuación, se muestran los gráficos de invierno obtenidos por los códigos mostrados en los Anexos S y FF. El primero de ellos es el perteneciente a la sección B y el segundo a la sección C sin la variable de sensación térmica.

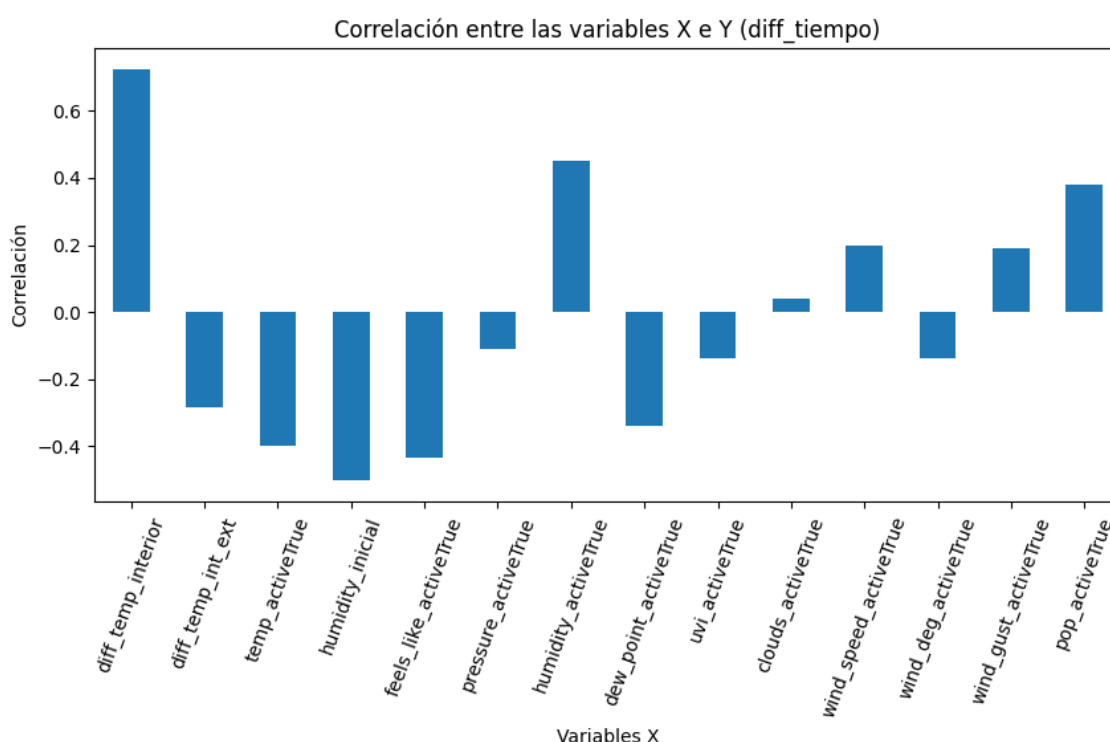


Ilustración 24. Gráfico de barras invierno B1

Fuente: Elaboración propia

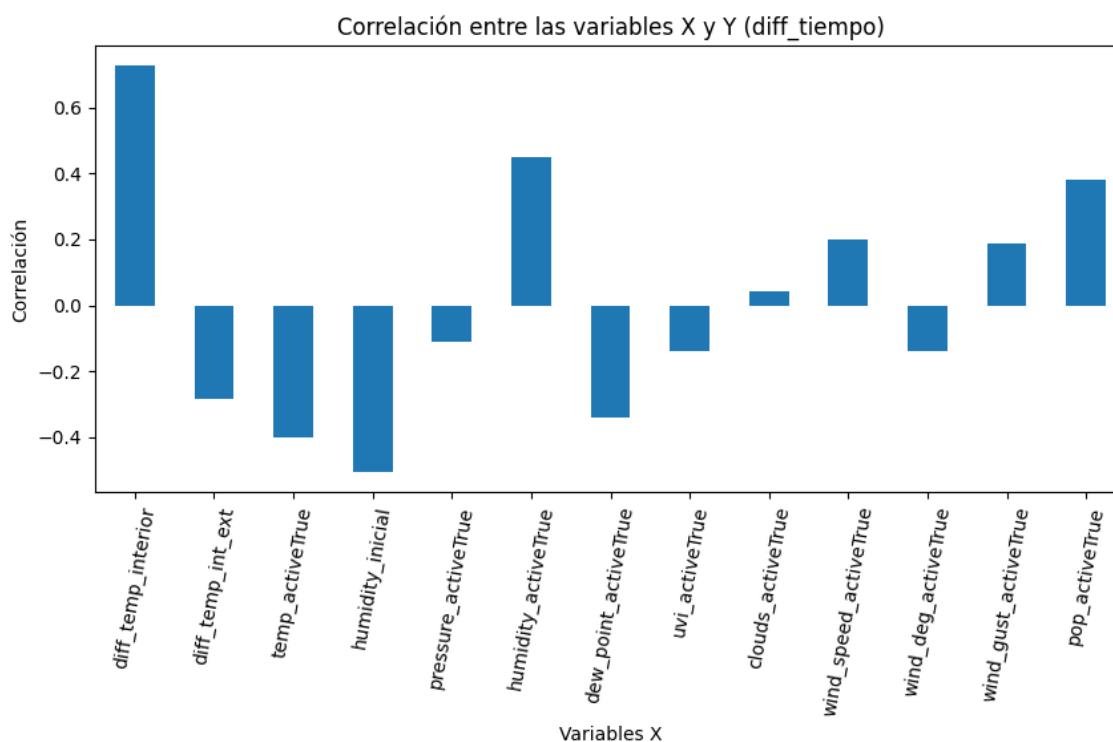


Ilustración 25. Gráfico de barras invierno C1

Fuente: Elaboración propia

En el eje de abscisas se ve representada cada una de las características o variables a estudiar, y en el eje de ordenadas el número correspondiente a la correlación que van a tener las características con la variable objetivo.

Se observa que ambos gráficos son prácticamente iguales, lo único que cambia en ellos es la ausencia de la variable “feels_like_activeTrue”.

Se van a seleccionar aquellas variables que tengan una correlación mayor a 0.25 en valor absoluto, estas van a quedar escritas en un archivo TSV mostrado en el Anexo T y en el anexo GG.

Las variables correspondientes al estudio de invierno de la sección B son un total de 8 y son las siguientes:

- “diff_temp_interior”,
- “diff_temp_int_ext”,
- “temp_activeTrue”,
- “humidity_inicial”,

- “feels_like_activeTrue”,
- “humidity_activeTrue”,
- “dew_point_activeTrue”
- y “pop_activeTrue”.

Las variables correspondientes al estudio de invierno de la sección C son un total de 7, son las mismas que las mencionadas en el párrafo anterior, pero con la ausencia de “feels_like_activeTrue”.

10.1.2. Estudio de verano

De igual forma se repite el proceso para el estudio de verano.

A continuación, se muestran los gráficos de verano obtenidos por los códigos mostrados en los Anexos S y FF, es el mismo código que para la base de datos de invierno, lo único que cambia es los datos de entrada. El primero de ellos es el perteneciente a la sección B y el segundo a la sección C sin la variable de sensación térmica.

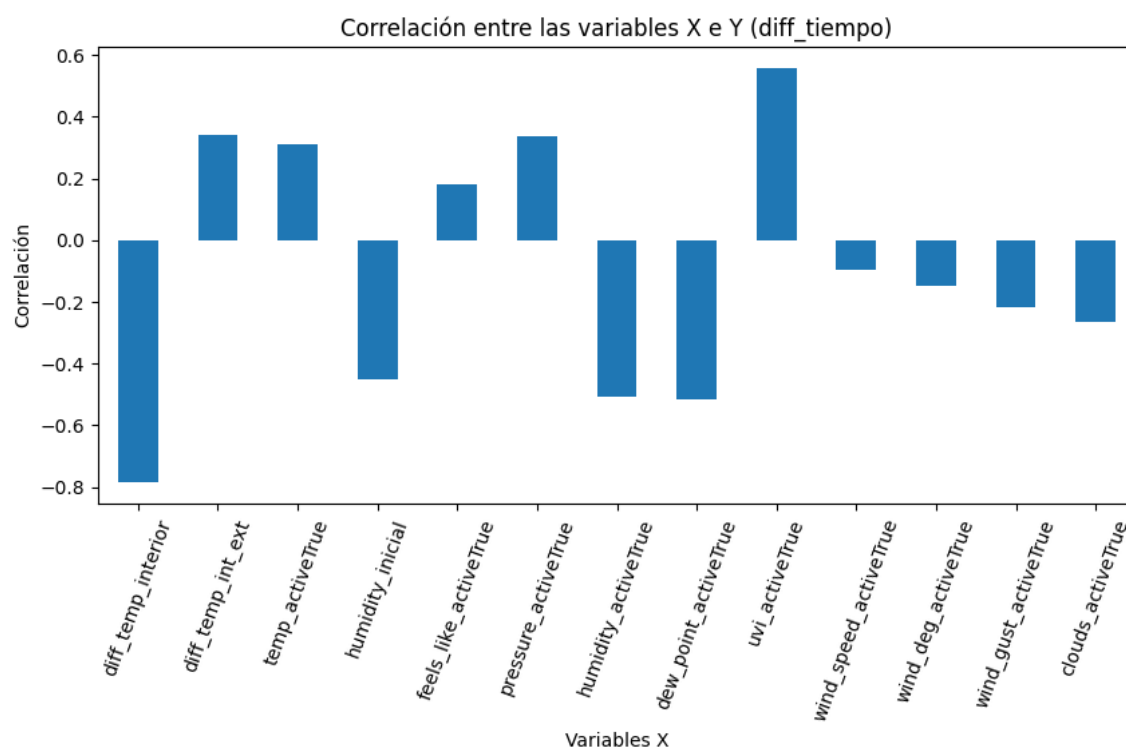


Ilustración 26. Gráfico de barras verano B1

Fuente: Elaboración propia

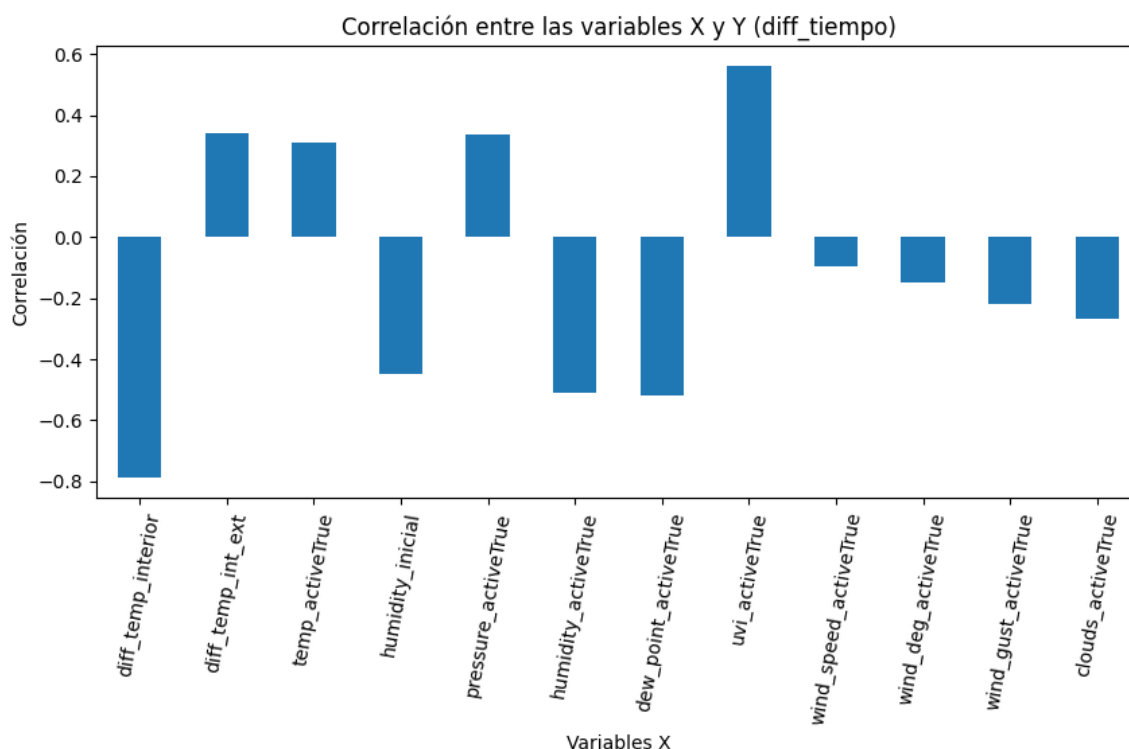


Ilustración 27. Gráfico de barras verano C1

Fuente: Elaboración propia

Al igual que en invierno, se seleccionan aquellas variables que tengan una correlación mayor a 0.25 en valor absoluto, estas van a quedar escritas en un archivo TSV mostrado en el Anexo U y en el anexo HH.

Las variables correspondientes al estudio de verano de la sección B son:

- “diff_temp_interior”,
- “diff_temp_int_ext”,
- “temp_activeTrue”,
- “humidity_inicial”,
- “pressure_activeTrue”
- “humidity_activeTrue”,
- “dew_point_activeTrue”
- “uvi_activeTrue”
- “wind_gust_activeTrue”

- y “clouds_activeTrue”.

Resulta interesante ver como no se ha obtenido la sensación térmica aquí. Para el estudio de verano de la sección C se tienen las mismas variables excepto la variable “wind_gust_activeTrue”, cuya variable representa la velocidad de las rachas de viento del exterior al arranque de la máquina.

10.1.3. Estudio conjunto

Se repite lo mismo con las bases de datos conjunta que incluyen las de invierno y las de verano. A continuación, se muestran ambos gráficos de barras.

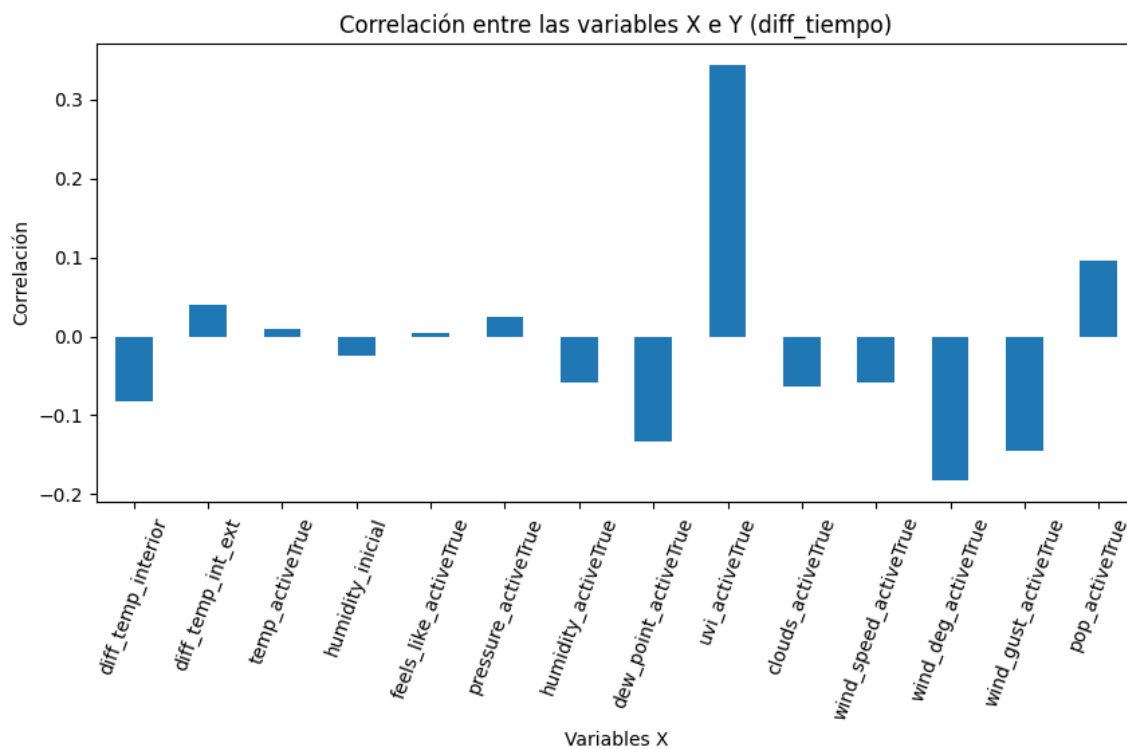


Ilustración 28. Gráfico de barras conjunto B1

Fuente: Elaboración propia

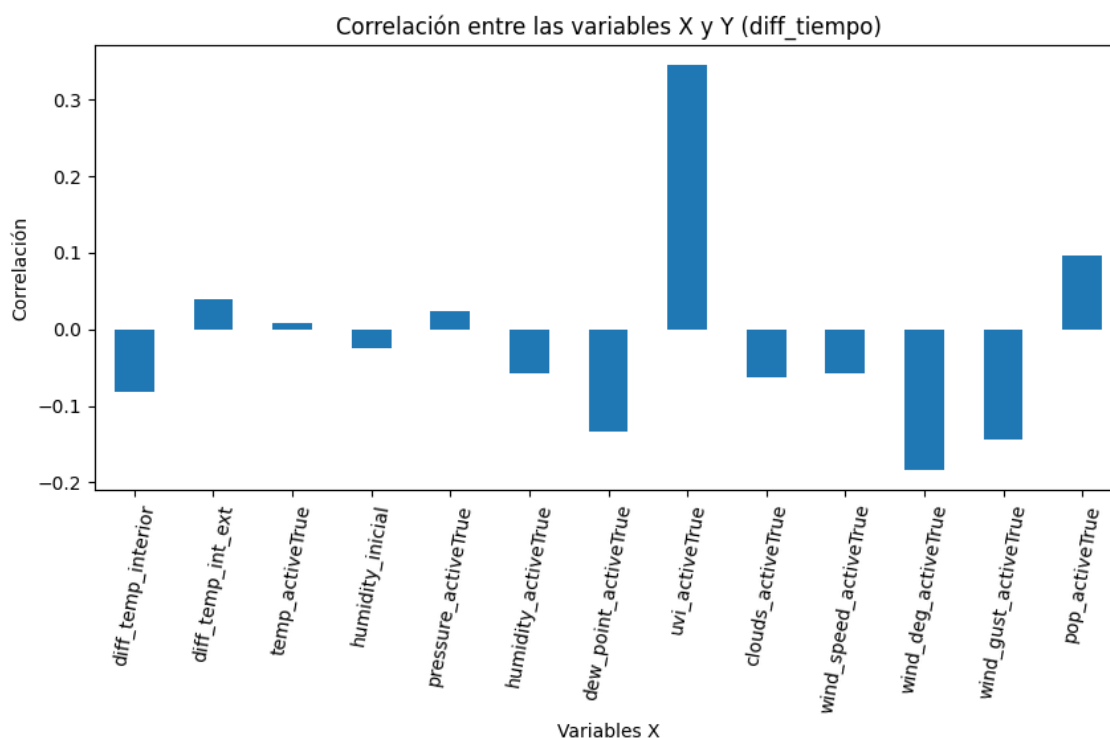


Ilustración 29. Gráfico de barras conjunto C1

Fuente: Elaboración propia

Como las correlaciones del estudio conjunto son tan pequeñas, se ha decidido que el límite de corte 0.1 en valor absoluto. Con ello se obtienen las siguientes características para el estudio de la sección B, que son también representadas en el archivo en formato TSV expuesto en el Anexo W:

- “dew_point_activeTrue”,
- “uvi_activeTrue”,
- “wind_deg_activeTrue”,
- y “wind_gust_activeTrue”

Las mismas variables han sido seleccionadas para el estudio de de la seccion C del proyecto, cuyos resultados en formato TSV se muestran en el Anexo HH.

10.2. Todas las variables previamente filtradas

En este apartado se crean los archivos en formato TSV con la totalidad de las variables que hayan sido filtradas previamente. En el Anexo X y JJ se puede observar los códigos usados para este apartado, el primero de ellos corresponde a la sección B y el segundo a la C.

Tanto en el Anexo Y como en el KK se observan las variables obtenidas para las bases de datos de invierno. Ambas solo difieren entre sí en que en la sección B se incluye la sensación térmica y en la C no.

En los Anexos Z y LL se observan las variables obtenidas para las bases de datos de verano. De nuevo, la diferencia entre este par de Anexos es la misma que la mostrada anteriormente.

Por último, en los Anexos AA y MM se observan las variables obtenidas para el conjunto de ambas bases de datos. Una vez más, ambos solo difieren en la sensación térmica.

Se puede observar que en ninguno de los anexos se ve la visibilidad del exterior (“visibility_activeTrue”) ya que es una variable que se ha eliminado anteriormente en el punto 9. Otra diferencia que tienen los archivos es que en las bases de datos de verano la probabilidad de lluvia (“pop_activeTrue”) se elimina también en el punto 9, mientras que en las de invierno y las conjuntas permanece. Esto es porque en las bases de datos de verano dicha probabilidad permanece constante en el tiempo, por lo que no aplica información al modelo.

10.3. LASSO y sus variables

El método LASSO es un acrónimo que significa “Least Absolute Shrinkage and Selection Operator”, en español sería “Operador de Encogimiento y Selección Menor Absoluto”. Es un método propuesto por Robert Tibshirani en 1996 y se utiliza en estadísticas y aprendizaje automático en la selección y regularización de variables. Tiene gran capacidad para manejar problemas de alta dimensionalidad y gran habilidad para identificar las características más relevantes. Por ello, se convierte en una

herramienta valiosa en el campo de la estadística y el aprendizaje automático (*Regression shrinkage and selection via the lasso on JSTOR*, s. f.-b).

LASSO se introduce una penalización en la función objetivo del modelo, que está basada en la suma de los valores absolutos de los coeficientes de las variables. Esta penalización encoge los coeficientes hacia el cero, provocando la eliminación de algunas variables del modelo ya que sus coeficientes son nulos. Con esta característica se pueden manejar problemas de gran dimensionalidad donde el número de las variables predictoras sea mayor que de número de observaciones. LASSO eliminaría variables menos relevantes y mantendría aquellas con un impacto significativo.

LASSO tiene un hiperparámetro llamado lambda (λ) que controla el nivel de penalización. El valor de lambda determina la cantidad de encogimiento aplicada en los coeficientes.

Con una lambda con un valor pequeño algunas variables empezarán a tener coeficientes pequeños, pero no nulos. A medida que lambda va aumentando, más coeficientes se van volviendo cero y las variables menos relevantes se eliminan del modelo, simplificando el modelo. Con lambda alto la mayoría de los coeficientes se acercarán a cero o se volverán cero, convirtiendo el modelo en un modelo muy simplificado, lo que también puede llevar a una pérdida de información.

Para hallar el mejor hiperparámetro se realiza una validación cruzada (cross-validation), en el siguiente punto se explica qué es y cómo funciona.

10.3.1. Validación cruzada

La validación cruzada, también conocida como cross-validation, es un método que consiste en evaluar y probar el rendimiento de un modelo de aprendizaje automático, con el fin de encontrar un mejor modelo rápidamente (*Team*, 2022).

Para realizar el cross-validation primero se tiene que hacer una división de los datos en entrenamiento y prueba. Con el conjunto de entrenamiento se ajustan los modelos LASSO con diferentes valores de lambda, mientras el de prueba se usa para evaluar el rendimiento de los modelos.

Tras ello, se divide el conjunto de entrenamiento en k subconjuntos de aproximadamente igual tamaño. Para el valor de lambda que se desea averiguar se repite

el conjunto k veces, una por subconjunto. Tal y como viene representado en la siguiente imagen.

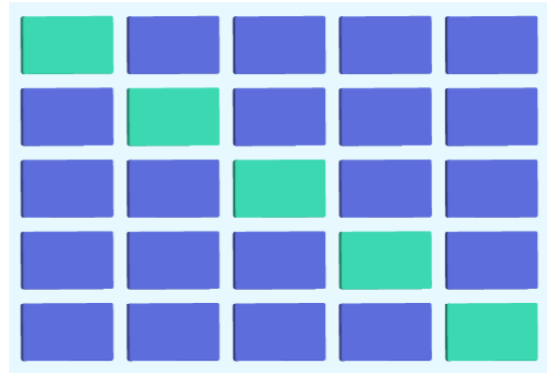


Ilustración 30. Validación cruzada

Fuente: Team (2020b)

Se elige el subconjunto de validación (en la imagen sería el rectángulo verde) y el resto sería el conjunto de entrenamiento (rectángulos azules), se entrena con la λ específica y se evalúa el rendimiento del modelo con el subconjunto de validación.

Una vez que el modelo se haya evaluado k veces, una por cada subconjunto, se calcula el promedio de las métricas de rendimiento obtenidas en cada iteración, lo que proporcionará una visión global del rendimiento del modelo con ese valor particular de λ (Team, 2022).

Se repite el proceso de validación cruzada para varios valores de λ . El valor de λ que resulte en el menor promedio de la métrica de rendimiento en las iteraciones de cross-validation se considera la mejor λ para el modelo.

Con este método se garantiza que la evaluación del rendimiento no dependa solamente de una partición de los datos en conjunto de entrenamiento y prueba, por lo que resulta una forma más confiable de cómo se desempeña el modelo en datos no vistos.

Al seleccionar el hiperparámetro λ que minimiza el error promedio en diferentes conjuntos de datos de entrenamiento y prueba, se tienen más probabilidades de obtener un modelo LASSO que generalice mejor a la hora de ponerlo en práctica con nuevos datos.

Es importante mencionar que la técnica de validación cruzada también se extiende a la ejecución de los modelos de Árboles de Decisión, Random Forest y XGBoost que veremos más adelante, ya que cuentan con una variedad de hiperparámetros ajustables. Estos hiperparámetros desempeñan un papel crucial en la configuración y el rendimiento de estos modelos.

10.3.2. Variables obtenidas

Los códigos mostrados en los Anexos BB y NN muestran los gráficos, ya sea de invierno, verano o del conjunto de ambos, de los coeficientes del modelo LASSO de la sección B y de la sección C respectivamente.

En el código de los anexos aparece esto:

```
model = LassoCV(cv=5, alphas = [2, 5]).
```

Quiere decir que k va a ser igual a 5, es decir se va a dividir los datos en 5 partes iguales y el proceso de entrenamiento y evaluación se repetirá 5 veces con distintas combinaciones, como en la imagen del punto anterior. Con el parámetro `alphas = [ 2, 5 ]` se indican los valores de lambda que se probaran en el proceso de validación cruzada.

Estos datos se usarán en los estudios de invierno y verano ya que se disponen de menos bases de datos. Sin embargo, para el estudio del conjunto de ambos se tendrá que ampliar el número de lambdas y sus valores. Por lo que se tendría la siguiente línea de código:

```
model = LassoCV(cv=5, alphas=[2, 5, 7, 10])
```

Además de los parámetros 2 y 5, se añaden otros más altos como 7 y 10.

Se seleccionará automáticamente aquel hiperparámetro que mejor funcione y cuyo rendimiento sea mayor, y se ejecutarán los modelos y el LASSO con ellos.

Como ya se ha mencionado al inicio del punto se seleccionarán aquellas variables cuyos coeficientes no sean nulos. A continuación, se muestran los gráficos de invierno, verano y del conjunto de ambos, tanto para la sección B como para la C.

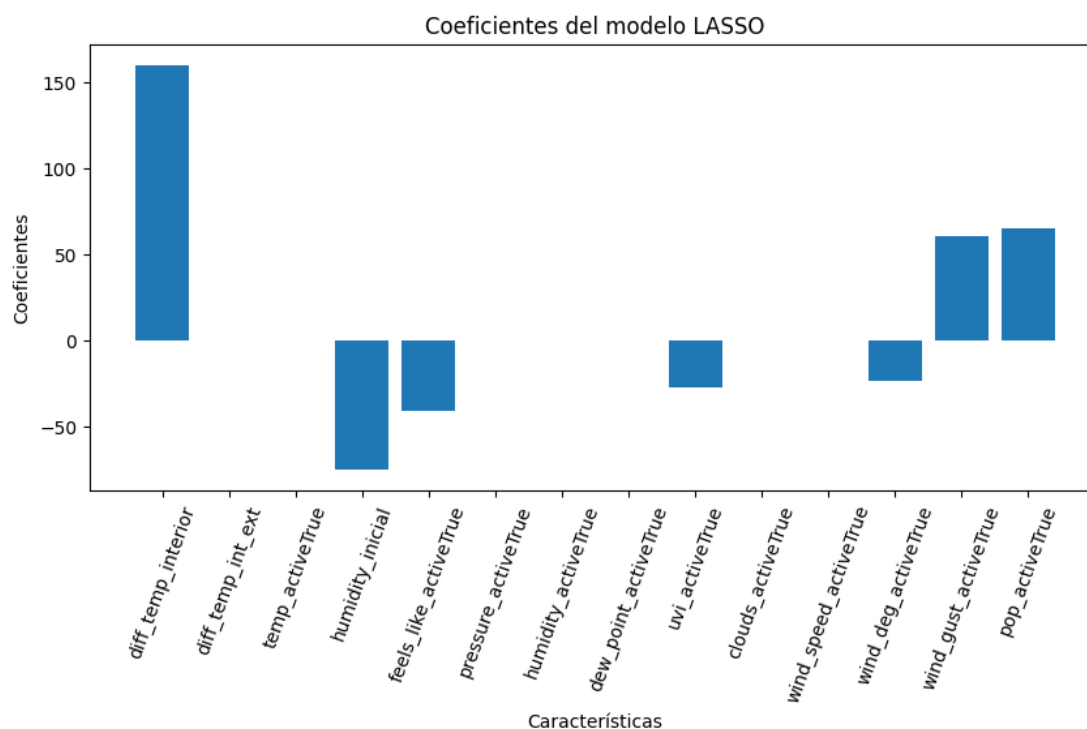


Ilustración 31. Gráfico de barras invierno B3

Fuente: Elaboración propia

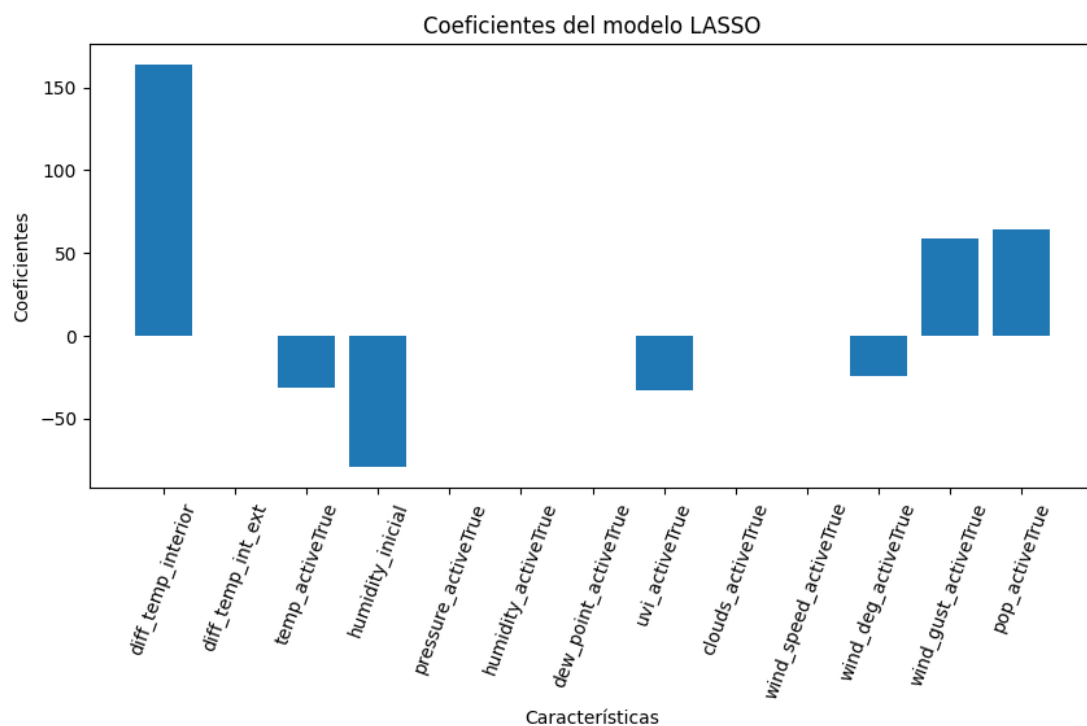


Ilustración 32. Gráfico de barras invierno C3

Fuente: Elaboración propia

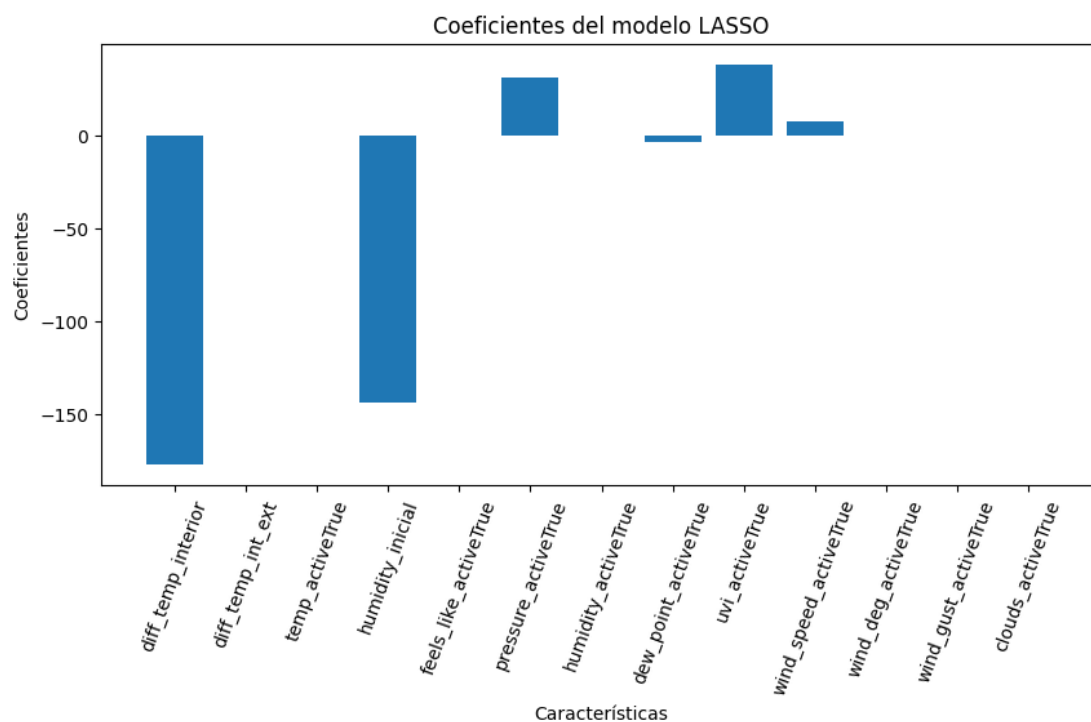


Ilustración 33. Gráfico de barras verano B3

Fuente: Elaboración propia

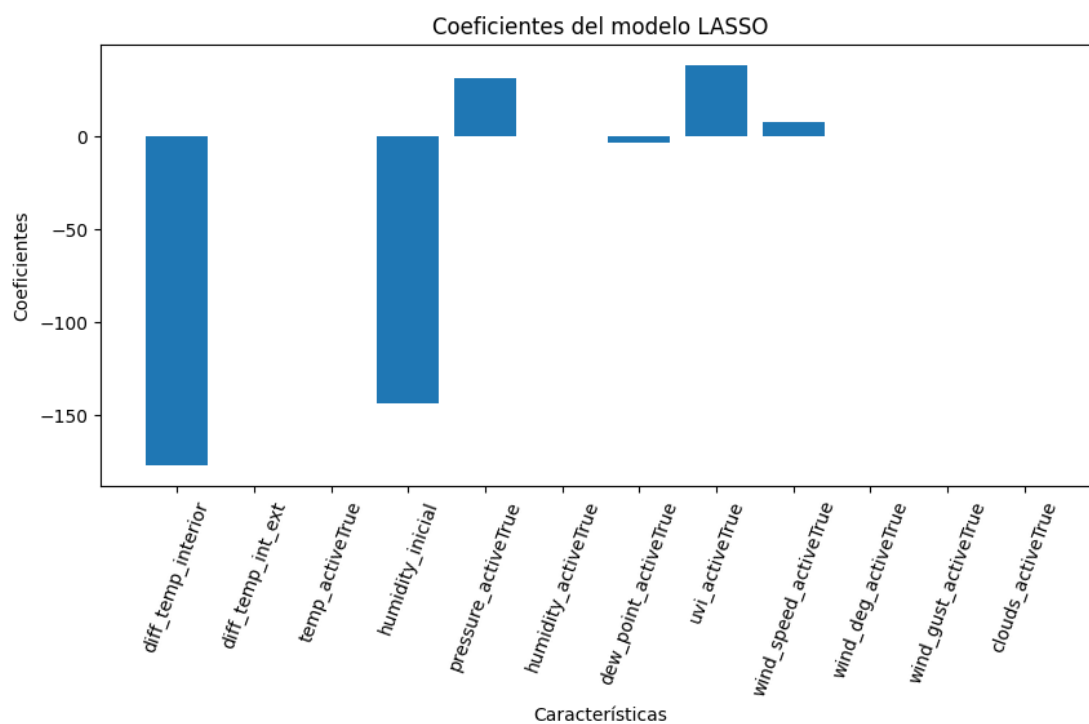


Ilustración 34. Gráfico de barras verano C3

Fuente: Elaboración propia

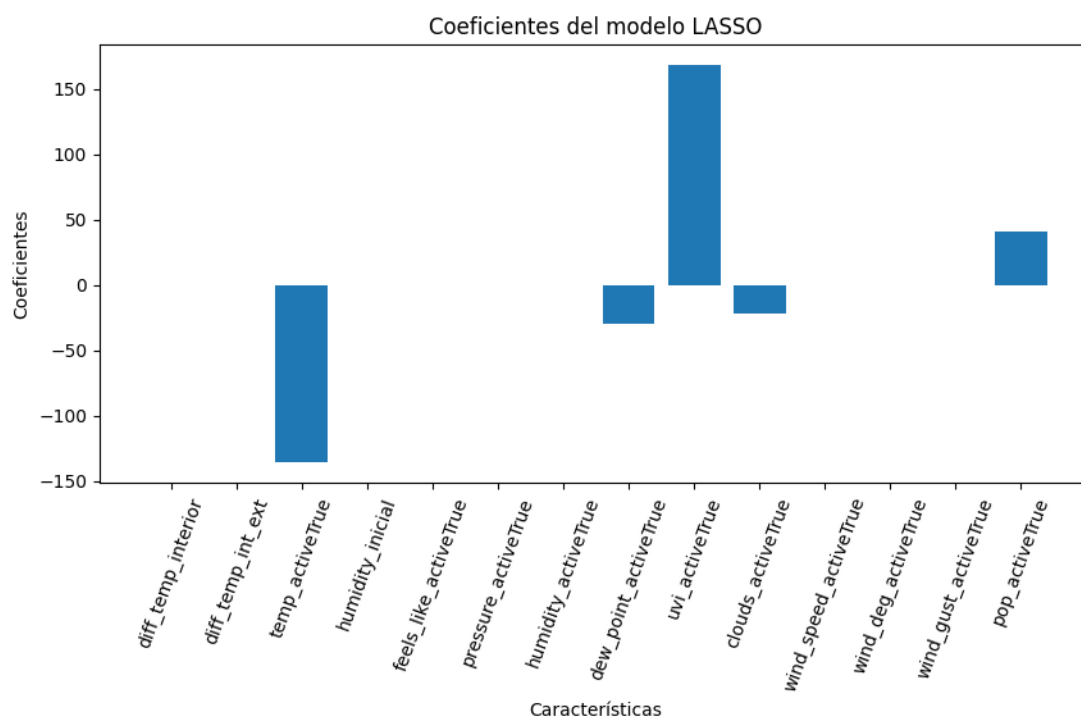


Ilustración 35. Gráfico de barras conjunto B3

Fuente: Elaboración propia

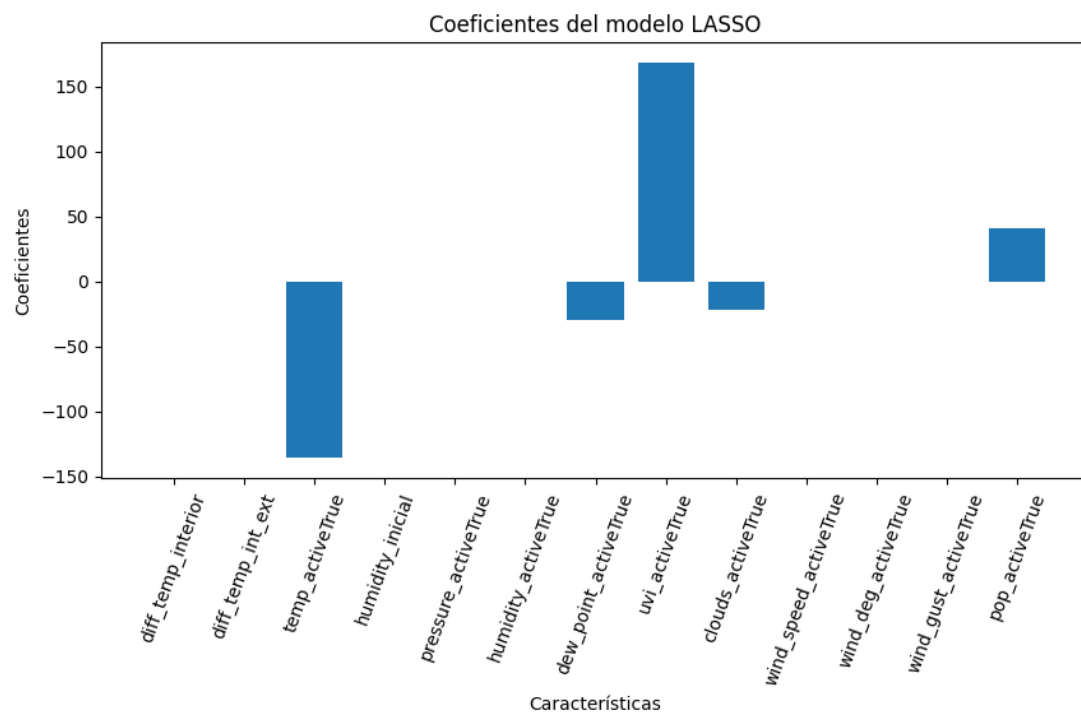


Ilustración 36. Gráfico de barras conjunto C3

Fuente: Elaboración propia

Del estudio de invierno se van a considerar las variables nombradas en los archivos en formato TSV de los Anexos CC y NN para la sección B y para la sección C respectivamente. Para la sección B se tienen las siguientes características:

- “diff_temp_interior”,
- “humidity_inicial”,
- “feels_like_activeTrue”,
- “uvi_activeTrue”,
- “wind_deg_activeTrue”,
- “wind_gust_activeTrue”
- y “pop_activeTrue”.

Mientras que para la sección C se tienen las siguientes características:

- “diff_temp_interior”,
- “temp_activeTrue”
- “humidity_inicial”,
- “uvi_activeTrue”,
- “wind_deg_activeTrue”,
- “wind_gust_activeTrue”
- y “pop_activeTrue”.

Para el estudio de verano, se consideran las variables nombradas en los archivos en formato TSV de los Anexos DD y OO para la sección B y para la sección C respectivamente. En ambas secciones se seleccionan las mismas variables, son las siguientes:

- “diff_temp_interior”,
- “humidity_inicial”,
- “pressure_activeTrue”,
- “dew_point_activeTrue”,

- “uvi_activeTrue”,
- “wind_speed_activeTrue”

Lo mismo ocurre para el estudio del conjunto de las bases de datos de invierno y verano, que la ausencia de la variable de sensación térmica no altera los resultados al realizar la selección por medio del método LASSO. Los archivos TSV para el conjunto vienen representados en los Anexos EE y PP. Las variables seleccionadas de ambos archivos son las siguientes:

- “temp_activeTrue”,
- “dew_point_activeTrue”,
- “uvi_activeTrue”,
- “clouds_activeTrue”,
- y “pop_activeTrue”

10.4. Conclusiones de la selección de variables

Para concluir la sección de selección de variables del proyecto, se ha realizado una tabla modo resumen de las variables que tenemos en cada uno de los casos: B1, C1, B2, C2, B3, C3 para invierno, verano y el conjunto de ambos.

En verde están representadas aquellas variables que se incluyen en el modelo que corresponda, y en rojo las que son eliminadas.

Se puede observar que en el estudio mixto que usa ambas bases de datos, se tiende a eliminar un mayor número de variables que cuando se estudia por separado invierno y verano.

	INVIERNO	INVIERNO	VERANO	VERANO	CONJUNTO	CONJUNTO	INVIERNO	INVIERNO	VERANO	VERANO	CONJUNTO	CONJUNTO	INVIERNO	INVIERNO	VERANO	VERANO	CONJUNTO	CONJUNTO
VARIABLES	B1	C1	B1	C1	B1	C1	B2	C2	B2	C2	B2	C2	B3	C3	B3	C3	B3	C3
diff_tiempo	variable objetivo																	
diff_temp_interior																		
diff_temp_int_ext																		
temp_activeTrue																		
humidity_inicial																		
feels_like_activeTrue																		
pressure_activeTrue																		
humidity_activeTrue																		
dew_point_activeTrue																		
uvi_activeTrue																		
clouds_activeTrue																		
wind_speed_activeTrue																		
wind_deg_activeTrue																		
wind_gust_activeTrue																		
pop_activeTrue																		
visibility_activeTrue																		

Ilustración 37. Resumen selección de variables

Fuente: Elaboración propia

Para cada selección de variables se estudiarán los 4 modelos: Regresión, Árbol de decisión, Random Forest y XGBoost. Por lo tanto, se tendrá para cada estudio de invierno, verano y conjunto, 24 posibilidades diferentes. Esto se verá más detenidamente en el siguiente punto (11. SECCIÓN D: EJECUCIÓN DE MODELOS Y EVALUACIÓN).

11. SECCIÓN D: EJECUCIÓN DE MODELOS Y EVALUACIÓN

En esta sección del proyecto se procede a ejecutar y evaluar los modelos de Regresión multivariante, Árbol de decisión, Bosques aleatorios y XGBoost con las variables de cada uno de los archivos TSV mostrados en los anexos. Para entender mejor cada uno de estos cuatro modelos se han explicado detenidamente en el apartado 11.1 titulado “Explicación de modelos”.

Para ejecutar y evaluar dichos modelos se crea una función auxiliar, que se puede consultar en el Anexo QQ. Esta función auxiliar recibe por entrada los datos de entrenamiento y test tanto de X como de Y, además de un archivo que proporcione el orden de las variables, es decir los archivos en formato TSV con la selección de variables obtenidos en el punto anterior.

En el código de la función se muestra, para los modelos de Árbol de decisión, Random Forest y XGBoost, una serie de hiperparámetros que se explicarán más detenidamente en el punto 11.2. Hiperparámetros.

La función devuelve los resultados de las métricas de evaluación de entrenamiento y test de cada uno de los cuatro modelos. Es decir, los resultados de R², MSE, MAE, RMSE e importancias y coeficientes de los modelos de Regresión multivariante, Árbol de decisión, Random Forest y XGBoost. Las métricas de evaluación mencionadas serán exhaustivamente descritas en la sección 11.3 titulada “Métricas de evaluación”. En este apartado también se llevará a cabo un análisis para discernir qué métrica demuestra ser la más adecuada para interpretar y comunicar los resultados obtenidos por el modelo.

Esta función en el código mostrado en el Anexo RR, donde se resuelve el resultado de todas las posibles combinaciones de modelos. El código crea un archivo en formato Excel donde se evalúa para cada modelo los resultados de R², MAE, MSE y RMSE, tanto de entrenamiento como test. Además, se crean otros dos archivos en formato Excel, uno por cada sección B y C, donde se reflejan los coeficientes e importancias de las variables según el modelo. Cada uno de estos dos archivos Excel tendrá tres hojas, una que use las variables obtenidas por correlación, otra que use la totalidad de las variables y otra que use las variables obtenidas por el método LASSO. Por lo que se obtendrían B1, B2 y B3 por medio de las bases de datos de la carpeta “BBDD”, y C1, C2 y C3 por medio de las bases de datos de la carpeta “BBDD_corr_filter”.

Se realizará esto para cada uno de los estudios invierno, verano y conjunto, de esta forma se obtendrán tres archivos Excel de evaluaciones (que se pueden ver en los Anexos SS, TT y UU) y seis archivos Excel de coeficientes e importancias de las variables, donde en cada uno de estos seis archivos habrá 3 hojas (que se pueden ver en los anexos WW, XX e YY).

11.1. Explicación de modelos

Cada vez más se está usando la ciencia de datos y el aprendizaje automático para tomar decisiones precisas, útiles e informadas. En este contexto, hay una gran variedad de modelos que se han desarrollado para abordar gran variedad de problemas de predicción. En este punto, se explicarán cuatro de ellos, que son los que se van a usar en el proyecto y constituyen los modelos más ampliamente utilizados en el campo del aprendizaje automático y la ciencia de datos. Estos cuatro modelos son: Regresión multivariante, Árboles de decisión, Bosques aleatorios y XGBoost.

11.1.1. Regresión multivariante

La regresión multivariante es una extensión de la regresión lineal simple donde se considera la relación existente entre una variable dependiente, en nuestro caso la diferencia de tiempo entre la parada y el arranque de la máquina, y múltiples variables independientes, todas las demás (Montero Granados, 2016). En la regresión lineal simple, se investiga como una única variable independiente se relaciona con una variable dependiente. Sin embargo, en el proyecto y en otras situaciones del mundo real, se necesitan varias variables independientes que puedan influir en la variable dependiente de forma conjunta.

El objetivo seguirá siendo modelar la relación entre la variable dependiente y las independientes, pero ahora el sistema es más complejo ya que variables independientes interactúan para predecir la dependiente.

De esta forma, la regresión multivariante toma la siguiente ecuación:

$$y = \beta + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p + \varepsilon$$

Donde:

- “y”: es la variable dependiente que se quiere predecir.
- “ $x_1, x_2, \dots, x_p$ ”: son las variables independientes (predictores) que influyen en la variable dependiente “y”.
- “ $\beta_0, \beta_1, \beta_2, \dots, \beta_p$ ”: son los coeficientes que representan las relaciones entre las variables independientes y la variable dependiente.
- “ ε ”: es el término de error que captura la variación no explicada por el modelo.

La regresión multivariante es útil cuando hay varios factores que pueden influir en la variable dependiente y cuando se busca comprender cómo estos factores interactúan para afectar las predicciones.

En la siguiente imagen se muestra un modelo de regresión lineal multivariante con una variable dependiente y dos variables independientes. Si solo se usan dos variables independientes, sigue siendo posible representar la relación gráficamente por medio de un plano. En lugar de una línea como en la regresión simple, se necesita un plano tridimensional para visualizar como influyen las variables independientes.

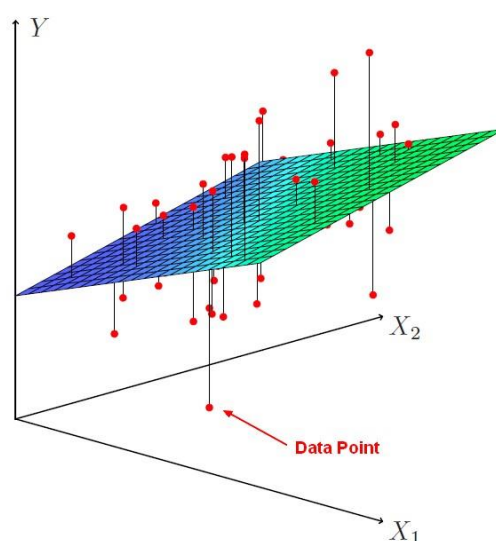


Ilustración 38. Regresión multivariante

Fuente: Statistics - multiple linear regression (2021)

Los problemas objetos de trabajo de este proyecto tienen muchas más de dos variables independientes por lo que son imposibles de representar gráficamente. Sin embargo, la imagen sirve de gran ayuda para hacerse una idea del funcionamiento de la regresión multivariante.

11.1.2. Árboles de decisión (Decision Trees)

Un árbol de decisión es un algoritmo de aprendizaje supervisado que se utiliza tanto para tareas de clasificación como de regresión. Su estructura jerárquica de árbol consta de un nodo raíz, ramas, nodos internos y nodos hoja (*¿Qué es un árbol de decisión? | IBM, s. f.*).

Esta técnica se usa en el aprendizaje automático para tomar decisiones o para realizar predicciones basadas en una serie de condiciones. Es una representación visual de un flujo de decisiones que se asemeja a un árbol donde cada nodo interno representa una característica o atributo, cada rama representa una posible decisión o acción, y cada hoja representa un resultado final o una predicción. (*¿Qué es un árbol de decisión? | IBM, s. f.*).

En la siguiente imagen podemos ver lo explicado gráficamente:

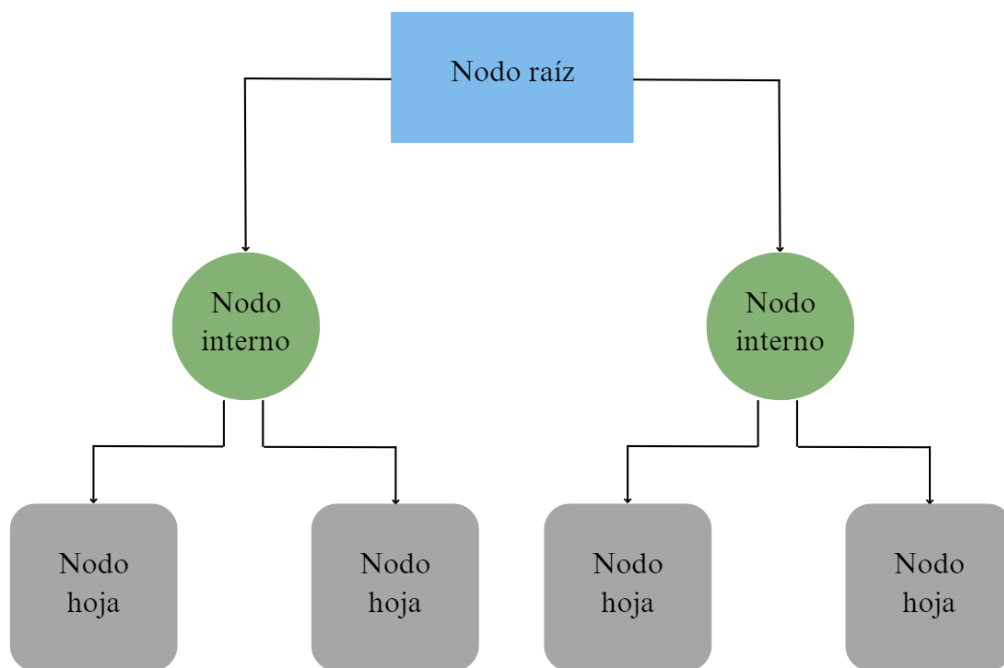


Ilustración 39. Árbol de decisión

Fuente: Elaboración propia

Los árboles de decisión tienen gran capacidad para manejar tanto datos numéricos como categóricos, su facilidad de interpretación y su flexibilidad para problemas de clasificación y regresión, son un elemento de fácil comprensión para la toma de decisiones en las empresas. Es cierto que a veces son propensos al sobreajuste, es decir,

se adaptan demasiado a los datos de entrenamiento y no generalizan bien al trabajar con nuevos datos.

11.1.3. Bosques aleatorios (Random Forest)

Un Random Forest, o bosque aleatorio en español, es un grupo de Árboles de decisión que se combinan mediante la técnica de bagging, lo que les brindan una capacidad muy alta para abordar muchos problemas. (Heras, 2020)

El bagging es una técnica de ensamble que consiste en combinar múltiples modelos de aprendizaje (árboles de decisión) para mejorar la precisión y la robustez de las predicciones. Con el bagging se crean varios conjuntos de datos de entrenamiento diferentes a partir del conjunto de datos original mediante muestreo con reemplazo. Cada conjunto de datos de entrenamiento utiliza para entrenar un modelo separado, y luego las predicciones de estos modelos individuales se combinan de alguna manera para obtener una predicción final.

La idea detrás de un Random Forest es construir un conjunto, o bosque, de árboles de decisión independientes y luego combinar sus resultados para obtener una predicción más precisa.

Se construyen los árboles individuales, cada uno entrenado en diferentes subconjuntos de datos (esto se logra mediante el bagging). Se generan varias muestras de entrenamiento a partir del conjunto de datos original mediante muestreo con reemplazo y se entrena el árbol de decisión en cada una de estas muestras.

Cada árbol se construye de manera independiente, lo que conduce a una diversidad de árboles generados. Esto es importante, porque si todos los árboles fuesen idénticos, el rendimiento podría no ser mucho mejor que el de un solo árbol.

Una vez contruidos los árboles, las precisiones de cada uno se combinan para formar una predicción final. En el caso de clasificación, suele realizarse una votación entre los árboles para determinar la clase más común. En la regresión, se promedian las predicciones numéricas de los árboles.

Los bosques aleatorios son muy buenos a la hora de trabajar con problemas propensos al sobreajuste (overfitting), ya que al haber tantos árboles individuales diferentes se reduce la probabilidad de que el modelo se ajuste demasiado a detalles

específicos de los datos de entrenamiento, este es un problema que presentaban los árboles de decisión. Al compensar los distintos errores de las predicciones de los distintos árboles de decisión, se consigue una mejora en la generalización. Como ventaja se puede decir que pueden aprender cualquier correspondencia entre datos de entrada y resultado a predecir, y como desventaja se puede decir que no son buenos extrapolando ya que no siguen un modelo conocido (Heras, 2020)

En la siguiente imagen se representa de forma gráfica el funcionamiento de un Random Forest, donde se cogen varias muestras aleatorias de los datos de entrenamiento, se realizan distintos árboles de decisión, y se hacen unos resultados de cada uno de los árboles de decisión. Al ser un problema de clasificación, el modelo se queda con el resultado más votado, en este caso 1.

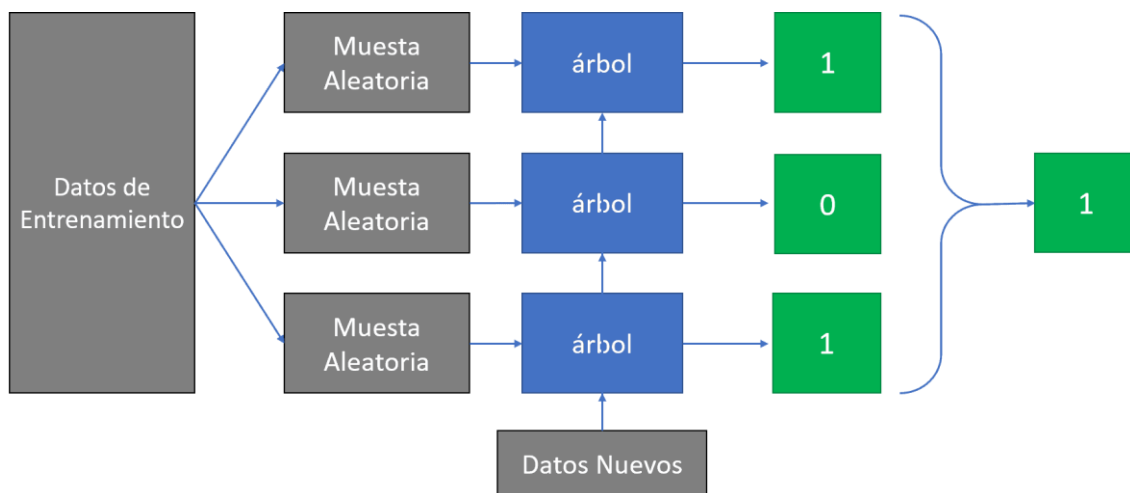


Ilustración 40. Random Forest

Fuente: Heras (2020)

11.1.4. XGBoost

XGBoost es la abreviatura de las palabras inglesas “Extreme Gradient Boosting”, en español quiere decir refuerzo de gradientes extremo. Este es un método que se basa en árboles de decisión y que supone una mejora sobre otros métodos como el Random Forest o el refuerzo de gradientes. Funciona con grandes y complejos datasets al usar varios métodos de optimización (Cómo funciona el algoritmo XGBOOST—ArcGIS Pro | Documentación, s. f.).

XGBoost funciona mediante el proceso de Boosting, que es una técnica de ensamblaje de modelos en la que se combinan varios modelos más simples y débiles para formar un modelo más robusto y preciso. Aquí hay una descripción general de cómo funciona XGBoost:

- Inicialización del modelo base: se comienza con un modelo base simple, por lo general un árbol de decisión poco profundo, también llamado "árbol débil" o "estimador débil". Este árbol resuelve el problema de predicción inicial, pero no es muy preciso por sí solo.
- Cálculo de residuos y ajuste del siguiente árbol: Se calculan los residuos (diferencias entre las predicciones del modelo actual y las etiquetas reales) del modelo base. Luego, se entrena un nuevo árbol para predecir estos residuos. Este árbol se ajusta para minimizar la suma de los cuadrados de los residuos.
- Actualización del modelo: Se combina el modelo base inicial con el nuevo árbol ajustado, dando como resultado un modelo mejorado que intenta corregir los errores cometidos por el modelo anterior.
- Regularización y control de sobreajuste: XGBoost implementa regularización L1 (Lasso) y L2 (Ridge) en cada paso del proceso de ajuste para evitar el sobreajuste. Esto significa que se penaliza a los coeficientes de los árboles para evitar que crezcan demasiado y se adapten excesivamente a los datos de entrenamiento.
- Actualización de pesos: Cada árbol construido tiene un peso asociado, que determina su contribución a la predicción final del modelo. Los árboles que tienen un rendimiento mejor contribuyen más a la predicción final, mientras que los árboles con un rendimiento inferior contribuyen menos.
- Iteración del proceso: Los pasos 2 a 5 se repiten varias veces. En cada iteración se ajusta un nuevo árbol para predecir los residuos del modelo actual y se actualizan los pesos de los árboles anteriores. Esto permite que el modelo mejore gradualmente y se acerque más a las verdaderas relaciones subyacentes en los datos.
- Predicción final: La predicción final del modelo se calcula sumando las predicciones de todos los árboles ajustados ponderados por sus respectivos pesos.

El proceso continúa mediante un número determinado de iteraciones o hasta que se alcance cierto criterio de parada, como la convergencia de las predicciones.

En la siguiente imagen se muestra un ejemplo que sirve como analogía a este modelo de aprendizaje automático, donde se quiere formar a un grupo de expertos con habilidades diferentes, para resolver una serie de adivinanzas desconocidas.

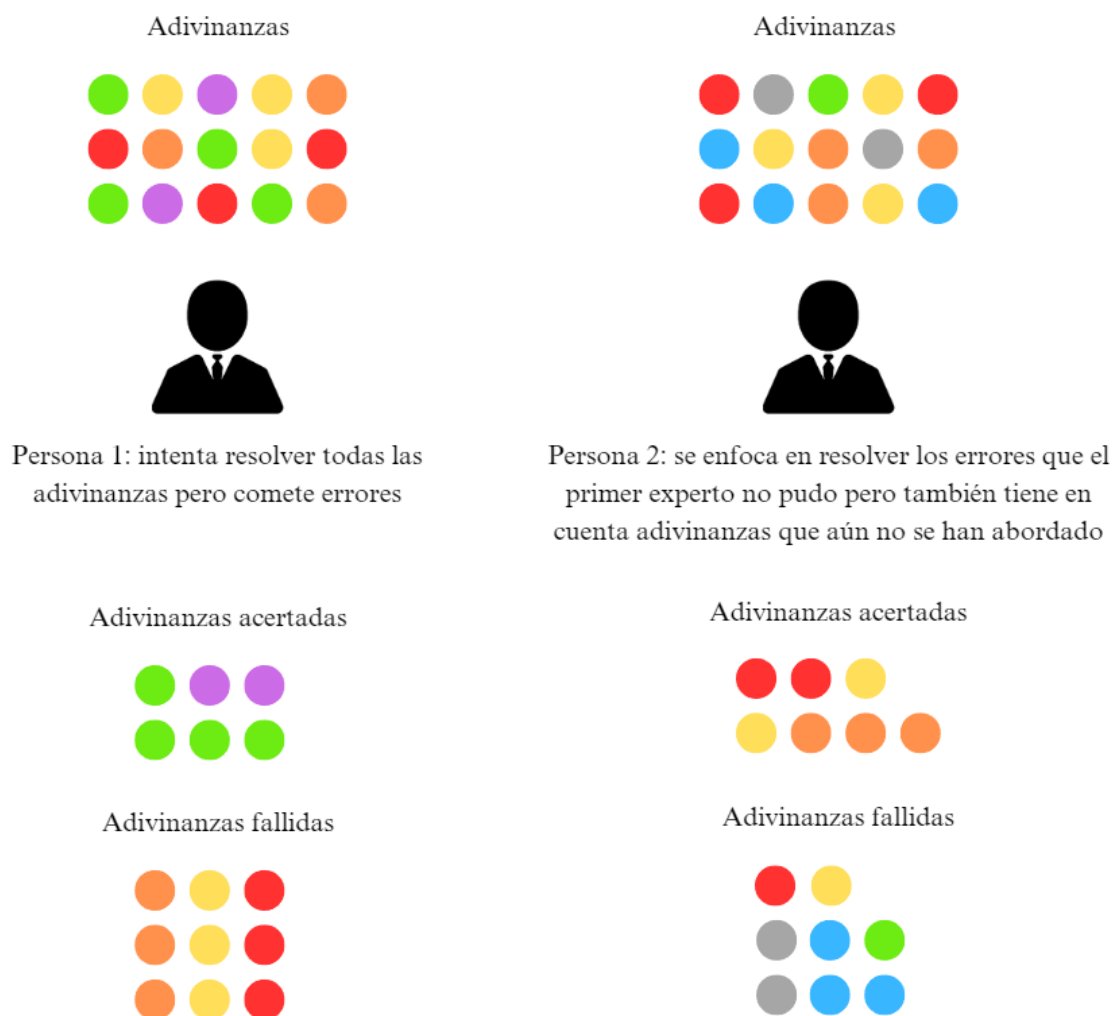


Ilustración 41. Ejemplo adivinanzas XGBoost

Fuente: Elaboración propia

El primer experto intenta resolver todas las adivinanzas, pero comete errores en algunas de ellas. Se identifican las adivinanzas en las que el primer experto falló (residuos del primer experto). Se incorpora un segundo experto con un conocimiento diferente, por lo que puede ser bueno en resolver las adivinanzas fallidas del primero. Se combinan las respuestas de ambos expertos dándole más peso al segundo experto en las adivinanzas que el primero falló. El equipo combinado tiene una mejor probabilidad de

adivinar correctamente una mayor cantidad de adivinanzas. Se repite el proceso agregando más expertos y ajustando los pesos en función de las adivinanzas no resultas correctamente. Cada nuevo experto se especializa en resolver problemas restantes (GeeksforGeeks, 2023).

11.2. Hiperparámetros

Los hiperparámetros son parámetros ajustables con los que se puede controlar el proceso de entrenamiento de un modelo. Estos se configuran manualmente antes de entrenar un modelo.

El código mostrado en el Anexo QQ usa un proceso de validación cruzada (cross-validation) para ajustar los hiperparámetros de los modelos de Árbol de decisión, Random Forest y XGBoost, no se mencionan los modelos de Regresión ya que estos no tienen hiperparámetros. La validación cruzada se realiza por medio de la función “GridSearchCV”, que divide automáticamente los datos de entrenamiento en varias partes y realiza múltiples entrenamientos y evaluaciones para diferentes combinaciones de hiperparámetros.

El proceso de validación cruzada ya ha sido explicado anteriormente en el punto 10.3.1 titulado “Validación cruzada”, ya que también fue un método utilizado en LASSO para obtener el mejor hiperparámetro lambda a la hora de seleccionar las características.

A continuación, se va a explicar de forma detallada los hiperparámetros utilizados en cada uno de los tres modelos de Árbol de decisión, Random Forest y XGBoost (3.2. *Tuning the hyper-parameters of an estimator*, s. f.).

- Hiperparámetros del Árbol de decisión

Se consideran dos hiperparámetros para el modelo de Árbol de decisión.

“max_depth”: Representa la profundidad máxima del árbol de decisión. Controla cuántos niveles de decisiones puede tomar el árbol antes de llegar a las hojas. Valores más bajos previenen el sobreajuste. Los valores en la lista [None, 5, 10] del código indican que se probarán árboles con profundidad ilimitada (sin restricción de profundidad), así como árboles con profundidades 5 y 10.

“min_samples_split”: Es el número mínimo de muestras requeridas para dividir un nodo interno. Valores más altos previenen el sobreajuste dividiendo los nodos solo cuando tienen suficientes muestras. Los valores de la lista [2, 5, 6] del código indican que se probarán diferentes umbrales para la división, permitiendo la exploración de diferentes niveles de complejidad.

- **Hiperparámetros de Random Forest**

Se consideran tres hiperparámetros para el modelo de Random Forest:

“n_estimators”: Representa la cantidad de árboles en el bosque. Más árboles generalmente mejoran la capacidad de generalización del modelo, hasta cierto punto. Los valores de la lista [100, 200, 300, 500] del código indican diferentes cantidades de árboles para explorar.

“max_depth”: Igual que en el árbol de decisión, controla la profundidad máxima de cada árbol dentro del bosque.

“min_samples_split”: Similar al árbol de decisión, determina el número mínimo de muestras requeridas para dividir un nodo.

- **Hiperparámetros de XGBoost**

Se consideran tres hiperparámetros para el modelo de XGBoost:

“n_estimators”: Nuevamente, la cantidad de árboles en el modelo XGBoost.

“max_depth”: La profundidad máxima de cada árbol.

“learning_rate”: Controla la magnitud de los ajustes que se hacen en cada paso para corregir los errores. Valores más bajos requieren más árboles para lograr la misma capacidad predictiva, pero pueden ayudar a prevenir el sobreajuste. Los valores de la lista [0.01, 0.1, 0.3] del código indican diferentes tasas de aprendizaje para explorar.

La variedad de los valores de las listas refleja la intención de explorar diferentes configuraciones con las que se puedan mejorar el equilibrio entre la capacidad de ajustes y la de generalización del modelo en los datos no vistos. Con la validación cruzada se seleccionarán los hiperparámetros que mejor funcionan y serán los usados para realizar la evaluación de los modelos.

11.3. Métricas de evaluación

Para medir el rendimiento de los modelos se van a usar una serie de métricas de evaluación. Las métricas de evaluación son herramientas fundamentales en el ámbito de la evaluación de modelos, especialmente en la estadística y el aprendizaje automático. Su función es medir el rendimiento de un modelo predictivo en función de cómo se ajustan sus predicciones a los valores reales.

Las métricas usadas para evaluar los rendimientos son el Coeficiente de Determinación (R^2), el Error Absoluto Medio (MAE), el Error Cuadrático Medio (MSE) y la Raíz del Error Cuadrático Medio (RMSE).

En los apartados posteriores se explica cada métrica detenidamente.

11.3.1. Coeficiente de Determinación (R^2)

El Coeficiente de Determinación, conocido como R^2 , es una métrica que proporciona información sobre el ajuste de las predicciones de un modelo a los valores reales de los datos. Su valor oscila entre cero y uno, donde uno es un ajuste perfecto del modelo a los datos, y cero no ofrece un ajuste mejor que la media de los valores reales.

El coeficiente evalúa la fuerza de la relación lineal entre dos variables y es muy utilizado a la hora de analizar tendencias (Estrategias de Inversión, s. f.-b). Éste muestra lo bien o lo mal que se ajusta un modelo de regresión a un conjunto de datos, indica el porcentaje explicado por el modelo de regresión.

El R^2 representa cuánta variabilidad en los datos puede ser explicado por las predicciones del modelo. La ecuación del Coeficiente de Determinación es la siguiente (Estadística P. Y., 2023):

$$R^2 = 1 - \frac{\sigma_r^2}{\sigma^2} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2}$$

Donde:

- σ_r^2 : es la varianza residual
- σ^2 : es la varianza de la variable dependiente Y.
- y_i : el valor de la variable dependiente de la observación i.

- $\hat{y}_i$: el valor aproximado por el modelo de regresión para la observación i .
- $\bar{y}_i$: la medida de la variable dependiente de todas las observaciones.

Hay que destacar que al comparar dos modelos de regresión diferentes no siempre es mejor el modelo con un coeficiente de regresión mayor, ya que, por ejemplo, un modelo de regresión puede tener un coeficiente de regresión de 1, lo más alto posible, porque se han añadido variables explicativas al modelo y puede explicar todas las observaciones perfectamente. Sin embargo, seguramente el modelo haga una predicción muy mala con un nuevo valor que no se hay usado para la construcción del mismo (*Estadística P. Y., 2023*).

Si la variabilidad de los residuos del modelo no es constante, el modelo resultará inútil, a pesar de que se tenga un Coeficiente de Determinación muy alto (*Estadística P. Y., 2023*).

Cuanto más variables explicativas tenga un modelo de regresión, más complicado será el modelo, pero mejor explicará los datos observados, por lo que mayor será su R^2 .

11.3.2. Error Absoluto Medio (MAE)

El Error Absoluto Medio (MAE por sus siglas en inglés, Mean Absolute Error) es una medida de evaluación común usada en análisis de datos y estadísticas que se encarga de medir la precisión de un modelo o predicción numérica. Se usa para cuantificar la discrepancia promedio entre los valores reales y los predichos por el modelo.

El Error Absoluto Medio se calcula sumando todos los errores absolutos y dividiéndolos por el número de observaciones. La ecuación del Error Absoluto Medio es la mostrada a continuación (*IBM documentation, s. f.*):

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

Donde:

- n : es el número de observaciones
- y_i : los valores reales u observados

- $\hat{y}_i$: los valores predichos por el modelo para los mismos puntos de datos.

El Error Absoluto Medio calcula el promedio de la diferencia absoluta entre la predicción del modelo y el valor de destino.

A diferencia del coeficiente de determinación, una tendencia alta indica que la métrica del modelo se está deteriorando. Por lo tanto, cuanto más bajo sea el Error Absoluto Medio mejor.

Además, el Error Absoluto Medio proporciona los resultados en valores absolutos por lo que no considera si la dirección de la diferencia es positiva o negativa, solo la magnitud de la diferencia en sí.

El MAE es un método fácil de entender y calcular ya que se expresa en las mismas unidades de medida que se estén utilizando, en este proyecto, sería en minutos (*Data*, 2023). También hay que destacar que no es sensible a valores atípicos, a diferencia del Error Cuadrático Medio (MSE), por lo que presenta una ventaja en el estudio.

11.3.3. Error Cuadrático Medio (MSE)

El Error Cuadrático Medio (MSE por sus siglas del inglés, Mean Squared Error) es otra métrica común usada para evaluar la precisión de los modelos de regresión o de predicción numérica. Este error penaliza de forma más significativa las diferencias entre los valores reales y las predicciones porque eleva al cuadrado dichas diferencias.

El MSE se halla calculando la media de las diferencias al cuadrado entre los valores reales y las predicciones. Esta es la ecuación que representa el Error Cuadrático Medio (*Torres*, 2021):

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Donde al igual que el MAE:

- n: es el número de observaciones
- y_i : los valores reales u observados
- $\hat{y}_i$: los valores predichos por el modelo para los mismos puntos de datos.

Los valores al cuadrado crecen mucho más rápidamente que los valores lineales, por lo que aquí se les da más peso a las diferencias más grandes entre los valores reales y las predicciones. Es útil para destacar los errores más grandes, pero es cierto que es muy probable que se vea influido por valores atípicos y que sea una métrica difícil de interpretar directamente.

Al igual que en MAE, cuanto más pequeño sea el valor de MSE mejor ya que indica un mejor rendimiento del modelo.

Por otro lado, cabe destacar que al estar al cuadrado la unidad de medida sería también al cuadrado, es decir, en el caso de este proyecto tendríamos minutos al cuadrado, y esto dificulta la interpretación de los resultados obtenidos.

11.3.4. Raíz del Error Cuadrático Medio (RMSE)

La Raíz del Error Cuadrático Medio (RMSE por sus siglas en inglés Root Mean Squared Error) es una métrica muy similar al MSE (Error Cuadrático Medio), pero con una diferencia muy importante: que RMSE hace la raíz cuadrada del valor calculado en el MSE.

Esto se hace para no cambiar la métrica de los valores originales, lo que lo hace más fácil de interpretar y comparar con los datos reales.

El RMSE es la desviación estándar de los valores residuales (errores de predicción). Se halla calculando la raíz cuadrada de la media de las diferencias al cuadrado entre los valores reales y las predicciones (*Oracle® Fusion Cloud EPM Trabajo con Planning*, s. f.). La ecuación del RMSE es la siguiente:

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

Donde al igual que MAE y MSE:

- n: es el número de observaciones
- y_i : los valores reales u observados
- $\hat{y}_i$: los valores predichos por el modelo para los mismos puntos de datos.

De la misma forma que MAE y MSE, cuanto más pequeño sea RMSE, mejor está funcionando el modelo.

Por otro lado, al realizar la raíz cuadrada del valor obtenido del Error Cuadrático Medio se confirma que la métrica está en la misma unidad de medida que los valores originales, en el caso de este proyecto en minutos, por lo que esto representa una ventaja frente al MSE y facilita la interpretación y la comparación con los datos reales.

11.3.5. Otras consideraciones sobre las métricas de evaluación

Todas las métricas de evaluación explicadas desempeñan un papel importante a la hora de realizar las predicciones de los modelos.

El R2 es un buen indicador para medir la el porcentaje de variabilidad de los datos capturados por el modelo, pero no ofrece una medida directa en minutos, que es la magnitud con la que se trabaja en el proyecto (*Kumar, 2023*). El R2 es muy sensible a características irrelevantes y tiende a aumentar a medida que se agregan más variables predictoras al modelo, incluso si esas variables no son realmente informativas, esto es un problema ya que puede representar una visión inflada del modelo al no mejorar significativamente la calidad de los modelos al aumentar. Tampoco es recomendable usarlo para comparaciones entre modelos diferentes si tienen diferentes conjuntos de datos o diferentes rangos de valores, ya que el valor de R2 siempre oscila entre 0 y 1. R2 se enfoca en la variabilidad explicada en lugar de la precisión de los modelos.

El MSE y el RMSE son técnicas que penalizan los errores más grandes haciendo que su valor sea más elevado aún, por lo que puede ser útil para identificar aquellos errores que sobresalen. Además, el MSE es difícil de interpretar porque sus valores están elevados al cuadrado y los resultados estarían dados en minutos elevados al cuadrado. Es cierto que el RMSE si tiene la misma escala que los errores originales y es más intuitivo de interpretar, pero sigue pudiendo ser influenciado por los valores atípicos.

El MAE ofrece una evaluación comprensible del error medio en valor absoluto entre las predicciones y los valores reales. Al no elevar los errores al cuadrado no es sensible a valores atípicos y representa una medida robusta de la discrepancia promedio en valor absoluto (*Kumar, 2023*). Por lo que a la hora de interpretar los datos podría ser una buena opción.

12. RESULTADOS OBTENIDOS

En los Anexos SS, TT y UU se observan los resultados de las métricas de evaluación de las bases de datos de invierno, verano y conjuntas para cada posible modelo.

Tal y como se muestra en la imagen posterior, la columna “Modelo” está compuesta por lo siguiente:

- Una letra B o C: que depende de si se ha seleccionado la sensación térmica, letra B, o no, letra C.
- Un número, 1, 2 o 3: que será 1 con las variables obtenidas del análisis de correlación, 2 con la totalidad de las variables, y 3 con las variables obtenidas por el método LASSO.
- Un modelo de aprendizaje automático: que será un modelo de Regresión multivariante, de Árbol de decisión, de Random Forest o de XGBoost.

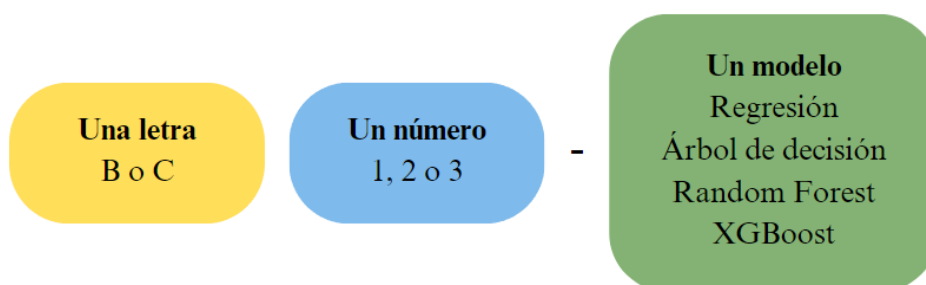


Ilustración 42. Posibles modelos

Fuente: Elaboración propia

El resto de las columnas, representan los valores obtenidos de las métricas de evaluación en el entrenamiento y en el test.

Si se examina con detenimiento los resultados obtenidos de la base de datos de invierno, se puede observar que el modelo que mayor R^2 tiene es el C1-Regresión. Se podría decir que es un buen modelo al no representar variabilidad, pero si se observa el MAE, se puede ver que tiene un valor altísimo con respecto a los demás modelos. A continuación, se muestran dos representaciones visuales donde esto se ve de forma clara. En el Anexo SS se puede ver que los valores de B1-Regresión destacan demasiado frente a los demás (MAE de 3369,46 minutos y MSE de 15779937,12

minutos) por lo que se va a considerar un valor atípico que no se tendrá en cuenta en las representaciones.

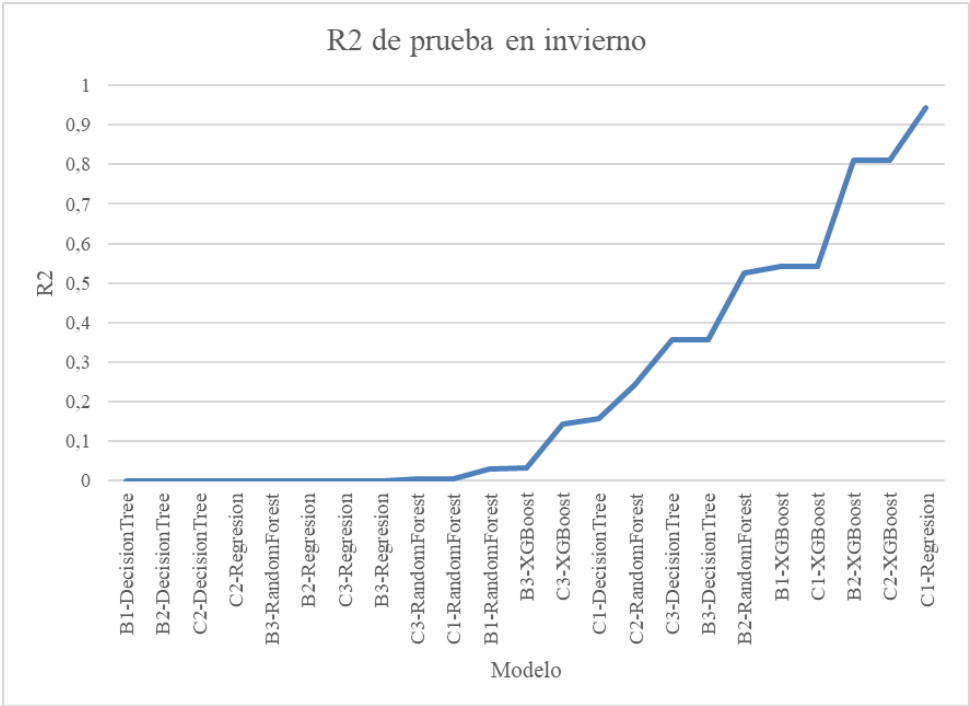


Ilustración 43. R2 prueba - invierno

Fuente: Elaboración propia

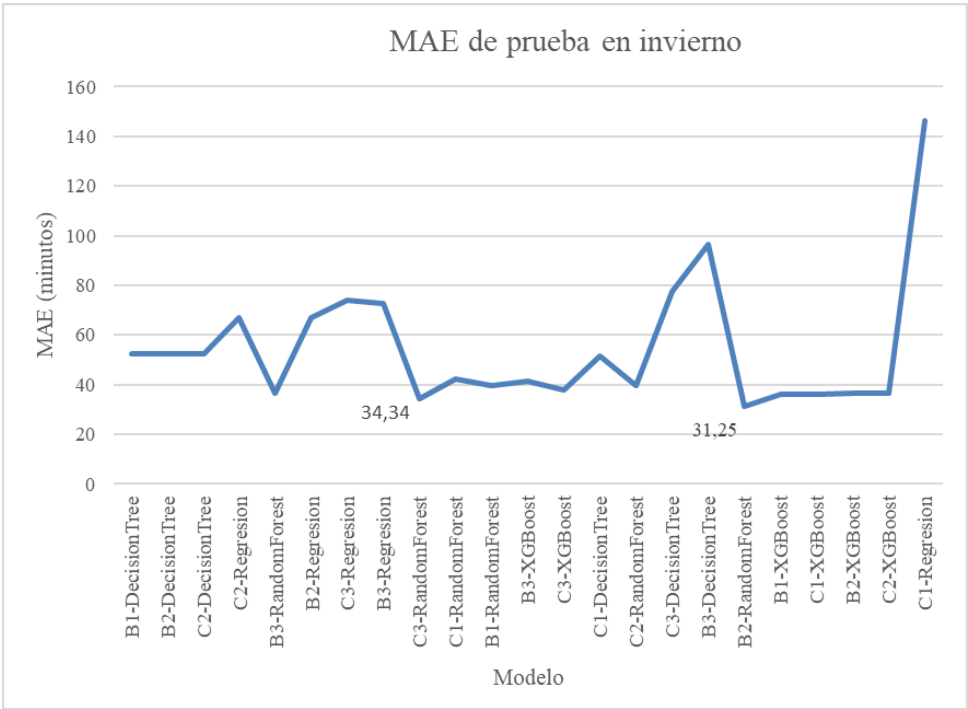


Ilustración 44. MAE prueba - invierno

Fuente: Elaboración propia

Si se analizan con atención ambos gráficos, se puede observar que modelos más complejos como son el Random Forest o el XGBoost mejoran el rendimiento de la predicción, y otros más sencillos como regresión multivariante o el árbol de decisión degradan el desempeño de los modelos, siendo los peores modelos, en cualquier caso.

El modelo que mejor funciona en las bases de datos de invierno es el B2- Random Forest con un MAE de 31,25 minutos y R2. Es decir, se consigue un error aproximadamente de media hora, lo que indica un error pequeño y un buen rendimiento del modelo para las pocas bases de datos de las que se disponían.

Es cierto que el número 2 del nombre del modelo indica que es mejor seleccionar todas las variables, pero si observamos los resultados de las importancias de las características del Random Forest (Ver Variables del B2 Anexo VV), se puede ver que hay 4 de ellas que indican una importancia mucho mayor frente a las demás, que son muy cercanas a 0. Estas cuatro características son: la humedad inicial del interior de la estancia, la diferencia de temperatura al arranque y a la parada de la máquina en el interior de la estancia, la dirección del viento en grados y la humedad inicial del exterior de la estancia. Aparentemente las humedades juegan un papel importante en el cálculo de la diferencia de tiempo.

Si se continúa llevando a cabo observaciones en ambas gráficas, el segundo modelo que mejor funciona es el C3-Random Forest con un error en valor absoluto de 34,33 minutos, por lo que podemos decir que los bosques aleatorios se desempeñan muy bien en este estudio. Si miramos los niveles de importancias de las características, también destacan frente a las demás cuatro de ellas, la diferencia de temperatura que existe entre el arranque y la parada de la máquina en el interior de la estancia, la humedad inicial del interior, la dirección del viento en grados y la probabilidad de lluvia. Las importancias de las tres primeras características también son las más relevantes en el modelo que mejor funciona B2- Random Forest (Ver Variables C3 del Anexo VV).

Tras estos dos ya se encuentran los modelos ambos modelos de XGBoost tanto de la sección B como de la sección C con las características tomadas en el análisis de correlación (B1 y C1). Seguidamente se tienen los modelos de XGBoost también, pero con la selección total de las características (B2 y C2).

Parece curioso observar como en los modelos de XGBoost se contempla que la ausencia de la sensación térmica no beneficia en nada al rendimiento, provocando que el número de minutos permanezca constante en las secciones B y C.

Indudablemente, el estudio que mejores resultados ha obtenido ha sido el estudio realizado con las bases de datos de invierno ya que representa unos errores de media 31 minutos aproximadamente. A continuación, se muestra una representación gráfica que ilustra el Error Absoluto Medio (MAE) de los tres estudios: invierno, verano y del conjunto de ambos.

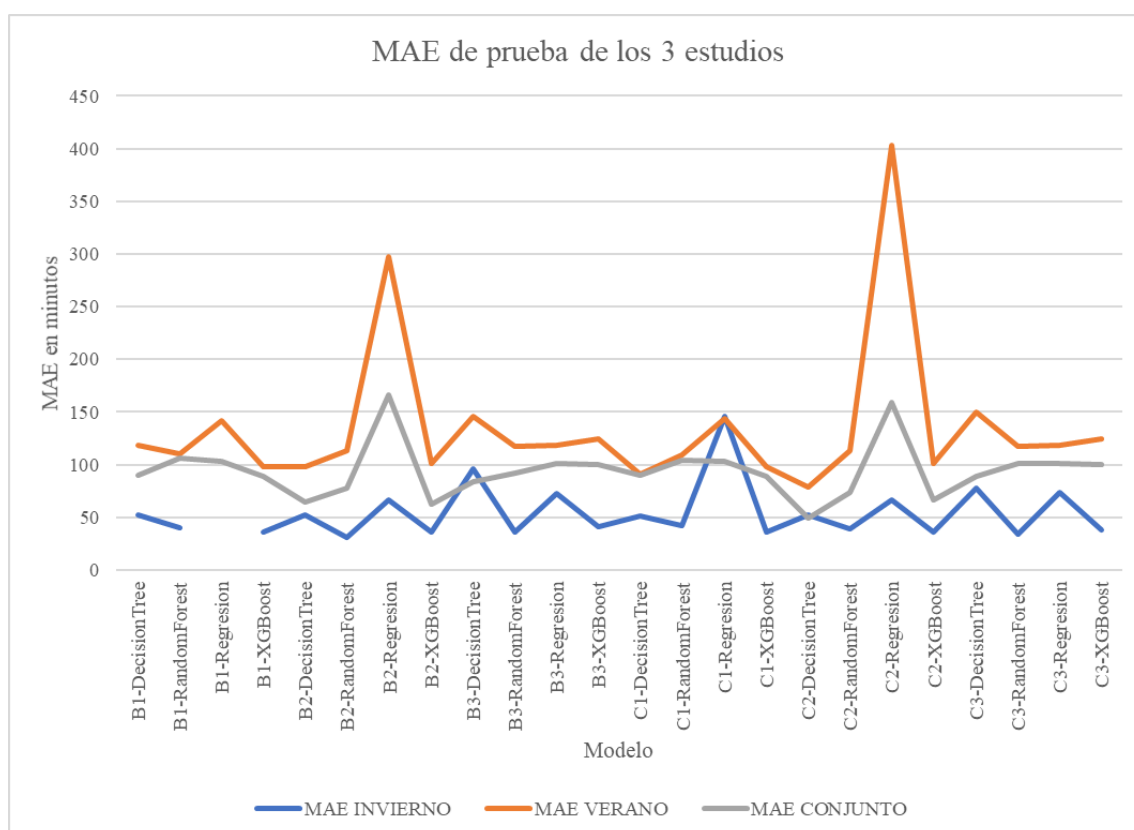


Ilustración 45. MAE prueba de los 3 estudios

Fuente: Elaboración propia

Como ya se ha mencionado, se elimina el valor de B1-Regresión de invierno al considerarse un valor atípico, el Error Cuadrático Medio (MSE) ayuda a determinar este tipo de valores, ya que al elevar al cuadrado el error sale un valor muy chocante con respecto a los demás.

En la representación gráfica del Error Absoluto Medio de los tres estudios se puede observar a simple vista que el estudio de verano es el que peores resultados a obtenido,

siendo el error mínimo 78 minutos, (Ver Anexo TT), lo que supone un error de más de una hora, más del doble que en el estudio de invierno. Por lo que sería interesante proporcionar más bases de datos para poder disminuir los errores y mejorar la calidad de los modelos de aprendizaje automáticos. Además, se mejoraría la detección de patrones al tener más variedad de ejemplos y se reduciría el sobreajuste a detalles específicos que pueden haber ocurrido al entrenar los modelos.

A diferencia del estudio de invierno, los mejores modelos de aprendizaje automático tanto para verano como para el conjunto de invierno y verano son el modelo de Árbol de decisión y el modelo XGBoost. En las bases de datos conjunta se consigue un error de 49 minutos, que con respecto al estudio de verano el error aumenta unos 20 minutos aproximadamente (Ver Anexo UU).

Por lo tanto, para los estudios paralelos de verano y conjunto se tiene:

- Para verano el mejor modelo de aprendizaje automático es C2-DecisionTree (Ver Anexo TT) cuyas características con mayor importancia son la humedad inicial del interior de la habitación y a la diferencia de temperatura que hay en el interior entre el arranque y la parada del equipo de climatización.
- Para el conjunto de las bases de datos el mejor modelo de aprendizaje automático es el mismo que para verano, C2-DecisionTree (Ver Anexo UU) cuyas características con mayor importancia son la diferencia de temperatura que hay en el interior entre el arranque y la parada del equipo de climatización, la humedad del exterior de la habitación y la velocidad del viento.

En cuanto a las variables se puede concluir, que tanto la humedad como la diferencia de temperatura existente en el interior de la habitación entre el arranque del equipo de climatización y la parada del mismo, son las variables más importantes del estudio ya que representan valores de importancia y coeficientes que sobresalen descaradamente con respecto a los otros.

Por otro lado, en cuanto a los modelos de aprendizaje automático, se da por sentado que el modelo de regresión es notablemente el que peor funciona en los tres estudios, tanto en invierno, como en verano y en el conjunto de ambos. Además, resulta curioso como otros modelos mucho más complejos, como es el modelo de aprendizaje

automático XGBoost, han representado unos errores que tienen aspectos favorables en los tres estudios.

Si se tuviese que decidir la forma en la que entrenar y testear los datos para dar resultados, sería conveniente que en la estación de invierno se usen solo las bases de datos de invierno, ya que ha presentado unos resultados muy favorables con respecto a los demás.

13. CONCLUSIONES

Como conclusión del proyecto, ha resultado interesante ver cómo se comporta el rendimiento de los modelos al realizar cambios en las variables de entrada a los mismos. Resulta curioso que la humedad del interior de la estancia salga como una de las variables más importantes en prácticamente todos los estudios. Era importante el estudio de esta variable porque muchos de los termostatos de los que dispone la empresa Airzone no constan de sensores que midan esta humedad. También existe una notable relevancia de la diferencia de temperatura entre el valor inicial al arrancar la máquina y el valor objetivo al apagarla, no obstante, esta observación si era de esperar y estaba dentro de las expectativas previas al estudio.

Inesperadamente, el modelo de regresión multivariante ha sido el que peores rendimientos ha dado de forma notoria. Esto ha resultado sorprendente ya que se tenía la suposición de que se iba a seguir un modelo lineal de regresión, y sin embargo ha sido el que peor ha funcionado.

Como ingenieros, siempre se buscará aquel modelo que proporcione los mejores resultados con la mayor precisión y exactitud en los mismos. Sin embargo, en un contexto empresarial, es muy importante considerar el costo computacional que suponen los mejores modelos. Los modelos más precisos, siempre suelen ser aquellos que requieran costes computacionales mayores, largos tiempos de entrenamiento y requerimientos de hardware con costos altos de implementación. Por ello, es importante y crucial buscar un equilibrio entre la mayor precisión y la viabilidad del modelo en la empresa, ya que en las empresas no solo se tienen en cuenta la calidad de los resultados que un modelo puede ofrecer, sino también la eficiencia que se le va a sacar a ese modelo con respecto a otros más simples.

En este proyecto, modelos más simples desde el punto de vista computacional como la regresión multivariante ha dado unos resultados bastante insatisfactorios en prácticamente todos los estudios realizados. Contrariamente ocurre con otros modelos más complejos que usan iteraciones no lineales como Random Forest y XGBoost, que han demostrado ser mucho más efectivos en términos predictivos.

Cada análisis requiere un punto de equilibrio adecuado. Cuando la diferencia de los errores de los distintos modelos es pequeña o insignificante, es común y lógico que la elección del modelo sea el más simple. Sin embargo, en este proyecto no resulta

conveniente hacer uso del modelo más simple (regresión multivariante), ya que presenta unos errores que duplican, y en ocasiones incluso triplican, los errores de otros modelos con mayor rendimiento.

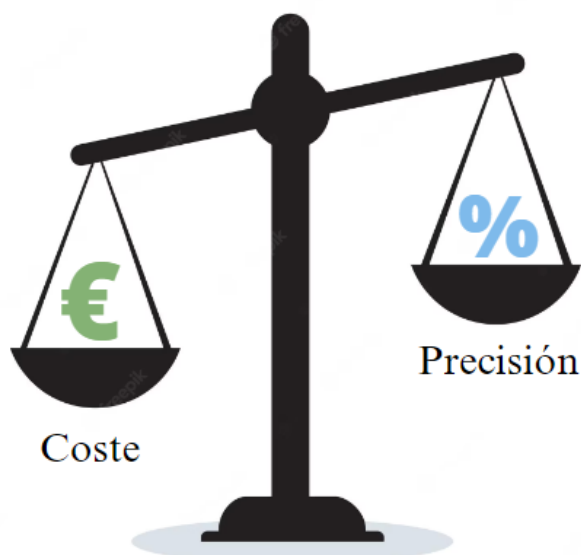


Ilustración 46. Balance coste y precisión

Fuente: Elaboración propia

Si se tuviese que elegir un modelo para el estudio inicial de las bases de datos de invierno, sería conveniente elegir el modelo de bosques aleatorios (Random Forest) ya que es el que mejores resultados proporciona. Dado que el conjunto de las bases de datos no es excesivamente extenso, la carga computacional asociada no sería una preocupación significativa para la empresa.

Al hallar predicciones de tiempo, este proyecto puede servir para hallar a modo informativo el nivel de aislamiento de una casa, por ejemplo, o para determinar el coste energético sabiendo el consumo de la máquina.

Además, si se ven fallos muy atípicos en estas predicciones de tiempos, el sistema podría estar informando sobre posibles fugas o malos comportamientos. Con esto, los residentes de la habitación podrían tomar medidas como revisiones del sistema de climatización y un mantenimiento correctivo del mismo.

Por otro lado, se considera que el desarrollo de este proyecto tiene gran importancia en la actualidad, ya que usa el aprendizaje automático para mejorar la eficiencia energética de los edificios, que es un tema de gran relevancia debido a su impacto en el consumo de energía y en la mitigación del cambio climático. Los edificios son los responsables de una parte significativa del consumo energético global, ya sea en términos de electricidad como de combustibles usados para la calefacción, ventilación y refrigeración. Mejorar la eficiencia energética en edificios implica implementar medidas y tecnologías que reduzcan el consumo de energía sin comprometer el confort y la funcionalidad de los espacios, como las que se han implementado en el contenido de este proyecto.

14. LÍNEAS FUTURAS

Como líneas de desarrollo del proyecto, se plantea la ampliación de la cantidad de bases de datos del estudio. Esta expansión permitiría alcanzar mejores resultados y conclusiones aún más certeras y precisas. Cuantas más bases de datos tengan los modelos para entrenar, mayor confiabilidad habrá en los resultados y mayor capacidad habrá para detectar patrones o relaciones más significativas.

Asimismo, resultaría interesante la exploración de otros enfoques analíticos más avanzados, como podría ser el modelo de redes neuronales artificiales. Este tipo de modelos son modelos más complejos que requieren un mayor coste computacional, ya que capturan relaciones no lineales que normalmente provocan una mayor precisión en los estudios.

Otra posible línea futura del proyecto podría ser la implementación de un sistema de soporte en la nube (Cloud). Ya se ha mencionado anteriormente que los usuarios de Airzone disponen de un sistema llamado “Airzone Cloud”, que es una aplicación móvil donde pueden controlar la temperatura del sistema de climatización donde y cuando los residentes de la habitación lo deseen. Por lo que se podría implementar la opción de saber en cuanto tiempo la habitación va a llegar a la temperatura consigna, de esta forma el usuario podría saber y programar la hora del encendido y arranque del sistema de climatización.

15. BIBLIOGRAFÍA DE CONTENIDO

Airzone Comunicación. (2022c, septiembre 13). Airzone - La Climatización

Inteligente. Airzone. <https://www.airzone.es/>

Airzonecontrol | HVAC Control Solutions. (2023).

<https://www.airzonecontrol.com/ib/es/eficiencia/>

Análisis de correlación. (2017, 25 mayo). Conogasi.

<https://conogasi.org/articulos/analisis-de-correlacion-2/>

ANÁLISIS EXPLORATORIO DE DATOS. (s. f.).

http://www.ub.edu/aplica_infor/spss/cap2-3.htm

Avb, & Avb. (2020). Data Leakage Machine Learning. *Solo para Entendidos*.

<https://www.soloentendidos.com/data-leakage-machine-learning-2337>

Bahit, E. (2020). *Python para Principiantes: Edición 2020* (Vol. 1). EBRC Publisher.

[https://www.google.es/books/edition/Python_para_Principiantes/aVF-](https://www.google.es/books/edition/Python_para_Principiantes/aVF-EAAAQBAJ?hl=es&gbpv=1&printsec=frontcover)

[EAAAQBAJ?hl=es&gbpv=1&printsec=frontcover](https://www.google.es/books/edition/Python_para_Principiantes/aVF-EAAAQBAJ?hl=es&gbpv=1&printsec=frontcover)

Caballero, M. M. (s. f.). Descubriendo patrones ocultos: introducción al análisis multivariante y exploración de datos. *es.linkedin.com*.

[https://es.linkedin.com/pulse/descubriendo-patrones-ocultos-](https://es.linkedin.com/pulse/descubriendo-patrones-ocultos-introducci%C3%B3n-al-y-de-mora-caballero)

[introducci%C3%B3n-al-y-de-mora-caballero](https://es.linkedin.com/pulse/descubriendo-patrones-ocultos-introducci%C3%B3n-al-y-de-mora-caballero)

Charte, F. (s. f.). *Qué tipos de problemas podemos resolver con técnicas de aprendizaje automático (Machine learning)* - *campusMVP.es*. *campusMVP.es*.

[https://www.campusmvp.es/recursos/post/que-tipos-de-problemas-podemos-](https://www.campusmvp.es/recursos/post/que-tipos-de-problemas-podemos-resolver-con-tecnicas-de-aprendizaje-automatico-machine-learning.aspx)

[resolver-con-tecnicas-de-aprendizaje-automatico-machine-learning.aspx](https://www.campusmvp.es/recursos/post/que-tipos-de-problemas-podemos-resolver-con-tecnicas-de-aprendizaje-automatico-machine-learning.aspx)

colaboradores de Wikipedia. (2023). LASSO (estadística). *Wikipedia, la enciclopedia libre*. [https://es.wikipedia.org/wiki/LASSO_\(estad%C3%ADstica\)](https://es.wikipedia.org/wiki/LASSO_(estad%C3%ADstica))

Cómo aprenden los termostatos Nest - Ayuda de Google Nest. (s. f.).

<https://support.google.com/googlenest/answer/9247510?hl=es>

Cómo funciona el algoritmo XGBOOST—ArcGIS Pro | Documentación. (s. f.).

<https://pro.arcgis.com/es/pro-app/latest/tool-reference/geoai/how-xgboost-works.htm>

Corporativa, Iberdrola. (s. f.). *Descubre los principales beneficios del Machine*

Learning. Iberdrola. [https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-](https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatizado#:~:text=El%20Machine%20Learning%20es%20una,elaborar%20predicciones%20(an%C3%A1lisis%20predictivo).)

[automatizado#:~:text=El%20Machine%20Learning%20es%20una,elaborar%20predicciones%20\(an%C3%A1lisis%20predictivo\).](https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatizado#:~:text=El%20Machine%20Learning%20es%20una,elaborar%20predicciones%20(an%C3%A1lisis%20predictivo).)

Dahlberg, E. (2020). *Replacing Setpoint Control with Machine Learning : Model*

Predictive Control Using Artificial Neural Networks. DIVA. [https://www.diva-](https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1439659&dswid=1709)

[portal.org/smash/record.jsf?pid=diva2%3A1439659&dswid=1709](https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1439659&dswid=1709)

datos.gob.es. (2022, 3 mayo). 11 Librerías para crear visualizaciones de datos.

[datos.gob.es.](https://datos.gob.es/es/blog/11-librerias-para-crear-visualizaciones-de-datos) <https://datos.gob.es/es/blog/11-librerias-para-crear-visualizaciones-de-datos>

De Ceupe, B. (s. f.). Aprendizaje supervisado: qué es, tipos y ejemplo. *Ceupe.*

[https://www.ceupe.com/blog/aprendizaje-](https://www.ceupe.com/blog/aprendizaje-supervisado.html#:~:text=Aprendizaje%20supervisado%20por%20regresi%C3%B3n%3A%20Es,real%20bas%C3%A1ndose%20en%20entradas%20pasadas.)
[supervisado.html#:~:text=Aprendizaje%20supervisado%20por%20regresi%C3%B3n%3A%20Es,real%20bas%C3%A1ndose%20en%20entradas%20pasadas.](https://www.ceupe.com/blog/aprendizaje-supervisado.html#:~:text=Aprendizaje%20supervisado%20por%20regresi%C3%B3n%3A%20Es,real%20bas%C3%A1ndose%20en%20entradas%20pasadas.)

Energetico, C., & Energetico, C. (2021). Ventajas de obtener el certificado energético |

Blog de Certificado de Eficiencia Energética. Blog de Certificado de Eficiencia Energética | Consigue tu certificado energético al mejor precio.

https://certificadodeeficienciaenergetica.com/blog/ventajas-obtener-certificado-energetico/#Ayuda_a_tener_un_mundo_mas_sostenible

- Estadística, P. Y. (2023). Coeficiente de determinación (R cuadrado). *Probabilidad y Estadística*. https://www.probabilidadyestadistica.net/coeficiente-de-determinacion-r-cuadrado/#google_vignette
- Estrategias de Inversión. (s. f.-b). *Dividendos más rentables de la bolsa española*. <https://www.estrategiasdeinversion.com/herramientas/diccionario/fondos/r2-r-cuadrado-o-coeficiente-de-determinacion-t-1163>
- Europa Press. (s. f.-b). *El control por zonas de la climatización del hogar permite un ahorro energético de hasta el 64%*. [europapress.es. https://www.europapress.es/economia/noticia-control-zonas-climatizacion-hogar-permite-ahorro-energetico-64-20220623162233.html](https://www.europapress.es/economia/noticia-control-zonas-climatizacion-hogar-permite-ahorro-energetico-64-20220623162233.html)
- GeeksforGeeks. (2023). XGBoost. *GeeksforGeeks*. <https://www.geeksforgeeks.org/xgboost/>
- Heras, J. M. (2020, 18 septiembre). Random Forest (Bosque aleatorio): Combinando árboles - IArtificial.net. *IArtificial.net*. <https://www.iartificial.net/random-forest-bosque-aleatorio/#:~:text=combinar%20varios%20%C3%A1rboles,-,%20BFQu%C3%A9%20es%20un%20Random%20Forest%3F,todos%20los%20datos%20de%20entrenamiento>.
- Huang, K., Hallinan, K. P., Lou, R., Alanezi, A., Alshatshati, S., & Sun, Q. (2020). Self Learning Algorithm to Predict Indoor Temperature and Cooling Demand from Smart WiFi Thermostat in a Residential Building. *Sustainability*, 12(17), 7110. <https://doi.org/10.3390/su12177110>
- json* — Codificador y decodificador JSON. (s. f.). Python documentation. <https://docs.python.org/es/3/library/json.html>

- KeepCoding, R. (2022, 20 enero). ¿Por qué la programación en Python es la mejor opción para aprender? *KeepCoding Bootcamps*.
<https://keepcoding.io/blog/programacion-en-python-aprender/#:~:text=%C2%BFEsto%20qu%C3%A9%20quiere%20decir%3F,de%20leer%2C%20escribir%20y%20aprender.>
- KeepCoding, R. (2023, 3 mayo). Dividir los datos en train y testing. *KeepCoding Bootcamps*. <https://keepcoding.io/blog/dividir-los-datos-en-train-y-testing/>
- Kumar, A. (2023, 4 abril). *Mean squared error or R-Squared - which one to use?* - Data analytics. Data Analytics. <https://vitalflux.com/mean-square-error-r-squared-which-one-to-use/>
- Las 11 técnicas más utilizadas en el modelado de análisis predictivos - Insight | Keyrus.* (s. f.). <https://keyrus.com/sp/es/insights/las-11-tecnicas-mas-utilizadas-en-el-modelado-de-analisis-predictivos>
- Li, W., Lork, C., Li, W., Yuen, C., & Keow, Y. M. (2019b). Benchmarking air-conditioning energy performance of residential rooms based on regression and clustering techniques. *Applied Energy*, 253, 113548.
<https://doi.org/10.1016/j.apenergy.2019.113548>
- Ltd, A. P. (2023). API Python de código abierto para hojas de cálculo de Excel - Leer y crear hojas de cálculo. *Aspose Pty. Ltd*.
[https://products.fileformat.com/es/spreadsheet/python/openpyxl/#:~:text=Openpyxl%20es%20una%20API%20de,XLSM%2FXLTX%2FXLTM\).](https://products.fileformat.com/es/spreadsheet/python/openpyxl/#:~:text=Openpyxl%20es%20una%20API%20de,XLSM%2FXLTX%2FXLTM).)
- Matplotlib — Visualization with Python.* (s. f.).
<https://matplotlib.org/>

Ministerio de Energía, Turismo y Agenda Digital. (2013). *Reglamento de Instalaciones Térmicas en los Edificios, aprobado por Real Decreto 1027/2007, de 20 de julio.*

<https://www.boe.es/boe/dias/2013/04/13/pdfs/BOE-A-2013-3905.pdf>

MongoDB. (s. f.). *MongoDB: La Plataforma De Datos Para Aplicaciones.*

<https://www.mongodb.com/es>

Montero Granados, R. (2016). *Modelos de regresión lineal múltiple.* Roberto Montero Granados.

NumPy. (s. f.).

<https://numpy.org/>

Oracle® Fusion Cloud EPM Trabajo con Planning. (s. f.).

https://docs.oracle.com/cloud/help/es/pbcs_common/PFUSU/insights_metrics_RMSE.htm#PFUSU-GUID-FD9381A1-81E1-4F6D-8EC4-82A6CE2A6E74

OS — interfaces misceláneas del sistema operativo — documentación de Python -

3.10.12. (s. f.). <https://docs.python.org/es/3.10/library/os.html>

pandas - Python Data Analysis Library. (s. f.-b).

<https://pandas.pydata.org/>

Promokit.eu. (s. f.). *¿Qué es Inverter? ¿Qué ventajas tiene el sistema y la tecnología inverter?* Toshiba aire acondicionado y calefacción - Toshiba Aire.

https://www.toshiba-aire.es/que-es-inverter/#inverter_01

Qué es un árbol de decisión | IBM. (s. f.). [https://www.ibm.com/es-es/topics/decision-](https://www.ibm.com/es-es/topics/decision-trees#:~:text=Un%C3%A1rbol%20de%20decisi%C3%B3n%20es,nodos%20internos%20y%20nodos%20hoja.)

[trees#:~:text=Un%C3%A1rbol%20de%20decisi%C3%B3n%20es,nodos%20internos%20y%20nodos%20hoja.](https://www.ibm.com/es-es/topics/decision-trees#:~:text=Un%C3%A1rbol%20de%20decisi%C3%B3n%20es,nodos%20internos%20y%20nodos%20hoja.)

Recio, Á. (2023, 3 marzo). «Málaga será la segunda capital de España, pero hay que controlar el crecimiento». *El Español*.

https://www.elespanol.com/malaga/economia/20230115/malaga-segunda-capital-espana-controlar-crecimiento/733176837_0.html

Regression shrinkage and selection via the lasso on JSTOR. (s. f.-b).

<https://www.jstor.org/stable/2346178>

Team, D. (2022, 1 agosto). *Cross-Validation : Definición e importancia en Machine learning*. Formation Data Science | DataScientest.com.

<https://datascientest.com/es/cross-validation-definicion-e-importancia>

scikit-learn: machine learning in Python — scikit-learn 0.16.1 documentation. (s. f.).

<https://scikit-learn.org/>

seaborn: statistical data visualization — seaborn 0.12.2 documentation. (s. f.).

<https://seaborn.pydata.org/>

Sensación térmica - concepto, cálculo y factores que la afectan. (s. f.). Concepto.

<https://concepto.de/sensacion->

[termica/#:~:text=Dicho%20de%20un%20modo%20muy,debido%20a%20factores%20meteorol%C3%B3gicos%20determinados.](https://concepto.de/sensacion-termica/#:~:text=Dicho%20de%20un%20modo%20muy,debido%20a%20factores%20meteorol%C3%B3gicos%20determinados.)

Smart Thermostats & Smart home Devices | EcoBee. (s. f.).

<https://www.ecobee.com/en-us/>

The Weather Impact on Efficient Home Heating with Smart Thermostats. (2019, 5 mayo). IEEE Conference Publication | IEEE Xplore.

<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8861778>

Torres, A. (2021). Aprendizaje automático: una introducción al error cuadrático medio y las líneas de regresión. *freeCodeCamp.org*.

<https://www.freecodecamp.org/espanol/news/aprendizaje-automatico-una-introduccion-al-error-cuadratico-medio-y-las-lineas-de-regresion/>

Triguero, N. (2022, 23 junio). Airzone invierte 8,5 millones para poner en marcha una nueva fábrica en Málaga. *Diario Sur*.

https://www.diariosur.es/economia/empresas-malaguenas/airzone-inversion-millonaria-frabrica-malaga-20220623113858-nt.html?gig_actions=sso.login&gig_enteredFromComponent=fromLoginClick

Tuning the hyper-parameters of an estimator. (s. f.). scikit-learn. https://scikit-learn.org/stable/modules/grid_search.html

Ueno, T., & Meier, A. (2020). A method to generate heating and cooling schedules based on data from connected thermostats. *Energy and Buildings*, 228, 110423. <https://doi.org/10.1016/j.enbuild.2020.110423>

Universidad Europea. (2022). Aprendizaje supervisado y no supervisado. *Universidad Europea*. <https://universidadeuropea.com/blog/aprendizaje-supervisado-no-supervisado/>

XGBoost Documentation — xgboost 1.7.5 documentation. (s. f.). <https://xgboost.readthedocs.io/>

16. BIBLIOGRAFÍA DE IMÁGENES

Airzone Comunicación. (2022b, septiembre 13). *Airzone - La Climatización Inteligente*.

Airzone. <https://www.airzone.es/>

BREEAM®. *El Certificado de Construcción Sostenible*. (2023b, enero 20). BREEAM®

ES. <https://breeam.es/>

Data, S. B. (2023, 30 agosto). *Aprendizaje automatico y las metricas de regresión -*

Sitiobigdata.com. sitiobigdata.com.

<https://sitiobigdata.com/2018/08/27/machine-learning-metricas-regresion-mse/>

GBCe | *Certificación VERDE*. (s. f.-b). <https://gbce.es/certificacion-verde/>

Méndez, J., & Méndez, J. (2009b, septiembre 24). El campus más sostenible. *El País*.

https://elpais.com/diario/2009/09/24/andalucia/1253744525_850215.html

Moon. (2022b, diciembre 27). *Qué es el WELL Building Standard. Certificación WELL*

| *ITG WELL*. ITG Market Partner WELL en España.

<https://wellservices.itg.es/certificado->

[well/#:~:text=La%20certificaci%C3%B3n%20WELL%20es%20un,el%20bienestar%20de%20los%20ocupantes](https://wellservices.itg.es/certificado-well/#:~:text=La%20certificaci%C3%B3n%20WELL%20es%20un,el%20bienestar%20de%20los%20ocupantes).

Roldán, P. N. (2022). Modelo de regresión. *Economipedia*.

<https://economipedia.com/definiciones/modelo-de-regresion.html>

RPubs - Introducción a los Modelos de Regresión en R. (s. f.-b).

<https://rpubs.com/rdelgado/395717>

Statistics - multiple linear regression. (2021, 6 abril). Datacadamia - Data and Co.

https://datacadamia.com/data_mining/multiple_regression

Welcome to Python.org. (2023b, mayo 22). Python.org. <https://www.python.org/>

17. ANEXOS

A continuación, se mostrarán los anexos usados a lo largo del documento, cuyo contenido principal son las escrituras de código realizadas para la ejecución de los programas y otras tablas y representaciones gráficas de interés. La denominación de los anexos son principalmente el nombre que se le ha dado al archivo concreto junto a la carpeta a la que pertenece.

Anexo A. Listados listado-DeviceEvents.txt y listado-weather.txt

Los listados de las rutas relativas de los archivos de DeviceEvents.json y de Weather.json que se encuentran en la carpeta de las bases de datos de invierno, python_tfg, son los siguientes:

listado-DeviceEvents.txt

```
1  datos/3a1b0c80058f5a04e07fc0a4/2022-12-15T08_42_07.538_2022-12-15T11_53_06.579/DeviceEvents.json
2  datos/3a1b0c80058f5a04e07fc0a4/2022-12-18T06_36_26.558_2022-12-18T12_10_01.835/DeviceEvents.json
3  datos/3a1b0c80058f5a04e07fc0a4/2022-12-20T11_38_50.115_2022-12-20T13_33_53.522/DeviceEvents.json
4  datos/3a1b0c80058f5a04e07fc0a4/2022-12-22T17_05_01.853_2022-12-22T19_29_40.582/DeviceEvents.json
5  datos/3a1b0c80058f5a04e07fc0a4/2022-12-23T05_00_19.633_2022-12-23T08_28_07.753/DeviceEvents.json
6  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T08_15_35.516_2022-12-25T10_41_27.719/DeviceEvents.json
7  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T15_38_00.574_2022-12-25T17_40_43.284/DeviceEvents.json
8  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T05_00_49.972_2022-12-28T06_52_29.793/DeviceEvents.json
9  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T10_49_39.071_2022-12-28T11_33_45.886/DeviceEvents.json
10 datos/3a1b0c80058f5a04e07fc0a4/2023-02-17T10_58_30.262_2023-02-17T13_23_37.626/DeviceEvents.json
11 datos/3a1b0c80058f5a04e07fc0a4/2023-02-24T13_56_16.612_2023-02-24T18_06_05.010/DeviceEvents.json
12 datos/3a1b0c80058f5a04e07fc0a4/2023-02-25T08_43_43.267_2023-02-25T11_00_58.488/DeviceEvents.json
```

listado-weather.txt

```
1  datos/3a1b0c80058f5a04e07fc0a4/2022-12-15T08_42_07.538_2022-12-15T11_53_06.579/weather.json
2  datos/3a1b0c80058f5a04e07fc0a4/2022-12-18T06_36_26.558_2022-12-18T12_10_01.835/weather.json
3  datos/3a1b0c80058f5a04e07fc0a4/2022-12-20T11_38_50.115_2022-12-20T13_33_53.522/weather.json
4  datos/3a1b0c80058f5a04e07fc0a4/2022-12-22T17_05_01.853_2022-12-22T19_29_40.582/weather.json
5  datos/3a1b0c80058f5a04e07fc0a4/2022-12-23T05_00_19.633_2022-12-23T08_28_07.753/weather.json
6  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T08_15_35.516_2022-12-25T10_41_27.719/weather.json
7  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T15_38_00.574_2022-12-25T17_40_43.284/weather.json
8  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T05_00_49.972_2022-12-28T06_52_29.793/weather.json
9  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T10_49_39.071_2022-12-28T11_33_45.886/weather.json
10 datos/3a1b0c80058f5a04e07fc0a4/2023-02-17T10_58_30.262_2023-02-17T13_23_37.626/weather.json
11 datos/3a1b0c80058f5a04e07fc0a4/2023-02-24T13_56_16.612_2023-02-24T18_06_05.010/weather.json
12 datos/3a1b0c80058f5a04e07fc0a4/2023-02-25T08_43_43.267_2023-02-25T11_00_58.488/weather.json
```

Los listados de las rutas relativas de los archivos de DeviceEvents.json y de Weather.json que se encuentran en la carpeta de las bases de datos de verano, python_tfg_verano, son los siguientes:

≡ listado-DeviceEvents.txt

```
1  datos/3a1b0c80058f5a04e07fc0a4/2023-07-08T11_51_37.049_2023-07-08T14_37_25.836/DeviceEvents.json
2  datos/3a1b0c80058f5a04e07fc0a4/2023-07-09T20_07_20.520_2023-07-09T20_54_04.835/DeviceEvents.json
3  datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T08_48_38.663_2023-07-11T10_37_03.296/DeviceEvents.json
4  datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T13_07_19.416_2023-07-11T15_18_28.795/DeviceEvents.json
5  datos/3a1b0c80058f5a04e07fc0a4/2023-07-12T09_35_19.516_2023-07-12T13_30_05.477/DeviceEvents.json
6  datos/3a1b0c80058f5a04e07fc0a4/2023-07-13T09_22_54.860_2023-07-13T12_35_58.132/DeviceEvents.json
7  datos/3a1b0c80058f5a04e07fc0a4/2023-07-14T11_26_23.128_2023-07-14T17_44_38.185/DeviceEvents.json
8  datos/3a1b0c80058f5a04e07fc0a4/2023-07-17T10_04_28.711_2023-07-17T13_28_20.140/DeviceEvents.json
9  datos/3a1b0c80058f5a04e07fc0a4/2023-07-18T13_48_10.213_2023-07-18T15_57_24.183/DeviceEvents.json
10 datos/3a1b0c80058f5a04e07fc0a4/2023-07-19T12_11_45.626_2023-07-19T19_34_44.965/DeviceEvents.json
11 datos/3a1b0c80058f5a04e07fc0a4/2023-07-20T12_33_42.775_2023-07-20T13_59_21.915/DeviceEvents.json
12 datos/3a1b0c80058f5a04e07fc0a4/2023-07-21T15_56_21.900_2023-07-21T17_07_31.581/DeviceEvents.json
13 datos/3a1b0c80058f5a04e07fc0a4/2023-07-23T10_27_04.855_2023-07-23T13_41_19.256/DeviceEvents.json
14 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T11_36_04.310_2023-07-29T12_54_23.894/DeviceEvents.json
15 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T17_21_57.418_2023-07-29T18_18_46.773/DeviceEvents.json
16 datos/3a1b0c80058f5a04e07fc0a4/2023-07-30T20_06_53.419_2023-07-30T21_28_34.605/DeviceEvents.json
17 datos/3a1b0c80058f5a04e07fc0a4/2023-07-31T13_53_16.940_2023-07-31T15_07_47.481/DeviceEvents.json
```

≡ listado-weather.txt

```
1  datos/3a1b0c80058f5a04e07fc0a4/2023-07-08T11_51_37.049_2023-07-08T14_37_25.836/weather.json
2  datos/3a1b0c80058f5a04e07fc0a4/2023-07-09T20_07_20.520_2023-07-09T20_54_04.835/weather.json
3  datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T08_48_38.663_2023-07-11T10_37_03.296/weather.json
4  datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T13_07_19.416_2023-07-11T15_18_28.795/weather.json
5  datos/3a1b0c80058f5a04e07fc0a4/2023-07-12T09_35_19.516_2023-07-12T13_30_05.477/weather.json
6  datos/3a1b0c80058f5a04e07fc0a4/2023-07-13T09_22_54.860_2023-07-13T12_35_58.132/weather.json
7  datos/3a1b0c80058f5a04e07fc0a4/2023-07-14T11_26_23.128_2023-07-14T17_44_38.185/weather.json
8  datos/3a1b0c80058f5a04e07fc0a4/2023-07-17T10_04_28.711_2023-07-17T13_28_20.140/weather.json
9  datos/3a1b0c80058f5a04e07fc0a4/2023-07-18T13_48_10.213_2023-07-18T15_57_24.183/weather.json
10 datos/3a1b0c80058f5a04e07fc0a4/2023-07-19T12_11_45.626_2023-07-19T19_34_44.965/weather.json
11 datos/3a1b0c80058f5a04e07fc0a4/2023-07-20T12_33_42.775_2023-07-20T13_59_21.915/weather.json
12 datos/3a1b0c80058f5a04e07fc0a4/2023-07-21T15_56_21.900_2023-07-21T17_07_31.581/weather.json
13 datos/3a1b0c80058f5a04e07fc0a4/2023-07-23T10_27_04.855_2023-07-23T13_41_19.256/weather.json
14 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T11_36_04.310_2023-07-29T12_54_23.894/weather.json
15 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T17_21_57.418_2023-07-29T18_18_46.773/weather.json
16 datos/3a1b0c80058f5a04e07fc0a4/2023-07-30T20_06_53.419_2023-07-30T21_28_34.605/weather.json
17 datos/3a1b0c80058f5a04e07fc0a4/2023-07-31T13_53_16.940_2023-07-31T15_07_47.481/weather.json
```

Los listados de las rutas relativas de los archivos de DeviceEvents.json y de Weather.json que se encuentran en la carpeta de las bases de datos tanto de invierno como de verano, python_tfg_conjunto, son los siguientes:

listado-DeviceEvents.txt

```
1  datos/3a1b0c80058f5a04e07fc0a4/2022-12-15T08_42_07.538_2022-12-15T11_53_06.579/DeviceEvents.json
2  datos/3a1b0c80058f5a04e07fc0a4/2022-12-18T06_36_26.558_2022-12-18T12_10_01.835/DeviceEvents.json
3  datos/3a1b0c80058f5a04e07fc0a4/2022-12-20T11_38_50.115_2022-12-20T13_33_53.522/DeviceEvents.json
4  datos/3a1b0c80058f5a04e07fc0a4/2022-12-22T17_05_01.853_2022-12-22T19_29_40.582/DeviceEvents.json
5  datos/3a1b0c80058f5a04e07fc0a4/2022-12-23T05_00_19.633_2022-12-23T08_28_07.753/DeviceEvents.json
6  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T08_15_35.516_2022-12-25T10_41_27.719/DeviceEvents.json
7  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T15_38_00.574_2022-12-25T17_40_43.284/DeviceEvents.json
8  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T05_00_49.972_2022-12-28T06_52_29.793/DeviceEvents.json
9  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T10_49_39.071_2022-12-28T11_33_45.886/DeviceEvents.json
10 datos/3a1b0c80058f5a04e07fc0a4/2023-02-17T10_58_30.262_2023-02-17T13_23_37.626/DeviceEvents.json
11 datos/3a1b0c80058f5a04e07fc0a4/2023-02-24T13_56_16.612_2023-02-24T18_06_05.010/DeviceEvents.json
12 datos/3a1b0c80058f5a04e07fc0a4/2023-02-25T08_43_43.267_2023-02-25T11_00_58.488/DeviceEvents.json
13 datos/3a1b0c80058f5a04e07fc0a4/2023-07-08T11_51_37.049_2023-07-08T14_37_25.836/DeviceEvents.json
14 datos/3a1b0c80058f5a04e07fc0a4/2023-07-09T20_07_20.520_2023-07-09T20_54_04.835/DeviceEvents.json
15 datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T08_48_38.663_2023-07-11T10_37_03.296/DeviceEvents.json
16 datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T13_07_19.416_2023-07-11T15_18_28.795/DeviceEvents.json
17 datos/3a1b0c80058f5a04e07fc0a4/2023-07-12T09_35_19.516_2023-07-12T13_30_05.477/DeviceEvents.json
18 datos/3a1b0c80058f5a04e07fc0a4/2023-07-13T09_22_54.860_2023-07-13T12_35_58.132/DeviceEvents.json
19 datos/3a1b0c80058f5a04e07fc0a4/2023-07-14T11_26_23.128_2023-07-14T17_44_38.185/DeviceEvents.json
20 datos/3a1b0c80058f5a04e07fc0a4/2023-07-17T10_04_28.711_2023-07-17T13_28_20.140/DeviceEvents.json
21 datos/3a1b0c80058f5a04e07fc0a4/2023-07-18T13_48_10.213_2023-07-18T15_57_24.183/DeviceEvents.json
22 datos/3a1b0c80058f5a04e07fc0a4/2023-07-19T12_11_45.626_2023-07-19T19_34_44.965/DeviceEvents.json
23 datos/3a1b0c80058f5a04e07fc0a4/2023-07-20T12_33_42.775_2023-07-20T13_59_21.915/DeviceEvents.json
24 datos/3a1b0c80058f5a04e07fc0a4/2023-07-21T15_56_21.900_2023-07-21T17_07_31.581/DeviceEvents.json
25 datos/3a1b0c80058f5a04e07fc0a4/2023-07-23T10_27_04.855_2023-07-23T13_41_19.256/DeviceEvents.json
26 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T11_36_04.310_2023-07-29T12_54_23.894/DeviceEvents.json
27 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T17_21_57.418_2023-07-29T18_18_46.773/DeviceEvents.json
28 datos/3a1b0c80058f5a04e07fc0a4/2023-07-30T20_06_53.419_2023-07-30T21_28_34.605/DeviceEvents.json
29 datos/3a1b0c80058f5a04e07fc0a4/2023-07-31T13_53_16.940_2023-07-31T15_07_47.481/DeviceEvents.json
```

listado-weather.txt

```
1  datos/3a1b0c80058f5a04e07fc0a4/2022-12-15T08_42_07.538_2022-12-15T11_53_06.579/weather.json
2  datos/3a1b0c80058f5a04e07fc0a4/2022-12-18T06_36_26.558_2022-12-18T12_10_01.835/weather.json
3  datos/3a1b0c80058f5a04e07fc0a4/2022-12-20T11_38_50.115_2022-12-20T13_33_53.522/weather.json
4  datos/3a1b0c80058f5a04e07fc0a4/2022-12-22T17_05_01.853_2022-12-22T19_29_40.582/weather.json
5  datos/3a1b0c80058f5a04e07fc0a4/2022-12-23T05_00_19.633_2022-12-23T08_28_07.753/weather.json
6  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T08_15_35.516_2022-12-25T10_41_27.719/weather.json
7  datos/3a1b0c80058f5a04e07fc0a4/2022-12-25T15_38_00.574_2022-12-25T17_40_43.284/weather.json
8  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T05_00_49.972_2022-12-28T06_52_29.793/weather.json
9  datos/3a1b0c80058f5a04e07fc0a4/2022-12-28T10_49_39.071_2022-12-28T11_33_45.886/weather.json
10 datos/3a1b0c80058f5a04e07fc0a4/2023-02-17T10_58_30.262_2023-02-17T13_23_37.626/weather.json
11 datos/3a1b0c80058f5a04e07fc0a4/2023-02-24T13_56_16.612_2023-02-24T18_06_05.010/weather.json
12 datos/3a1b0c80058f5a04e07fc0a4/2023-02-25T08_43_43.267_2023-02-25T11_00_58.488/weather.json
13 datos/3a1b0c80058f5a04e07fc0a4/2023-07-08T11_51_37.049_2023-07-08T14_37_25.836/weather.json
14 datos/3a1b0c80058f5a04e07fc0a4/2023-07-09T20_07_20.520_2023-07-09T20_54_04.835/weather.json
15 datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T08_48_38.663_2023-07-11T10_37_03.296/weather.json
16 datos/3a1b0c80058f5a04e07fc0a4/2023-07-11T13_07_19.416_2023-07-11T15_18_28.795/weather.json
17 datos/3a1b0c80058f5a04e07fc0a4/2023-07-12T09_35_19.516_2023-07-12T13_30_05.477/weather.json
18 datos/3a1b0c80058f5a04e07fc0a4/2023-07-13T09_22_54.860_2023-07-13T12_35_58.132/weather.json
19 datos/3a1b0c80058f5a04e07fc0a4/2023-07-14T11_26_23.128_2023-07-14T17_44_38.185/weather.json
20 datos/3a1b0c80058f5a04e07fc0a4/2023-07-17T10_04_28.711_2023-07-17T13_28_20.140/weather.json
21 datos/3a1b0c80058f5a04e07fc0a4/2023-07-18T13_48_10.213_2023-07-18T15_57_24.183/weather.json
22 datos/3a1b0c80058f5a04e07fc0a4/2023-07-19T12_11_45.626_2023-07-19T19_34_44.965/weather.json
23 datos/3a1b0c80058f5a04e07fc0a4/2023-07-20T12_33_42.775_2023-07-20T13_59_21.915/weather.json
24 datos/3a1b0c80058f5a04e07fc0a4/2023-07-21T15_56_21.900_2023-07-21T17_07_31.581/weather.json
25 datos/3a1b0c80058f5a04e07fc0a4/2023-07-23T10_27_04.855_2023-07-23T13_41_19.256/weather.json
26 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T11_36_04.310_2023-07-29T12_54_23.894/weather.json
27 datos/3a1b0c80058f5a04e07fc0a4/2023-07-29T17_21_57.418_2023-07-29T18_18_46.773/weather.json
28 datos/3a1b0c80058f5a04e07fc0a4/2023-07-30T20_06_53.419_2023-07-30T21_28_34.605/weather.json
29 datos/3a1b0c80058f5a04e07fc0a4/2023-07-31T13_53_16.940_2023-07-31T15_07_47.481/weather.json
```

Anexo B. Código A0_Preprocesamiento.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Identificar los diferentes tipos del atributo 'param'
en DeviceEvents.json

# Se importa la biblioteca json para poder trabajar con los archivos
.json
import json
# Se importa la biblioteca os, que permite acceder y manipular un sistema
de archivos
import os

# Se define la ruta absoluta de la carpeta con la que estamos trabajando
ruta_absoluta = r"C:/Users/Marina/Desktop/python_tfg"
# Función para cambiar el directorio del trabajo actual al especificado
por la ruta_absoluta,
# Así no tenemos que escribir la ruta completa de cada archivo
os.chdir(ruta_absoluta)

# Se define la variable para acabar la ruta
archivo_listado_DeviceEvents = 'listado-DeviceEvents.txt'
# Se abre el archivo
archivo = open(archivo_listado_DeviceEvents)
# Se lee el archivo, splitlines separa una línea de otra
ruta_archivos = archivo.read().splitlines()
# Se cierra el archivo de DeviceEvents.txt
archivo.close()

# Se crea una función que lee los parámetros
def read_param(ruta_archivo, diccionario_param):
    fichero = open(ruta_archivo) # Se abre un .json
    data = json.load(fichero) # Se lee el fichero en formato lista de
diccionarios
    fichero.close() #Se cierra el archivo de ruta_fichero

    for elemento in data: # Se crea un bucle for que recorre los ficheros
        param = elemento['meta']['param'] # Coge el parámetro
        # Si el parámetro ya se encuentra en el diccionario que suma 1
        if param in diccionario_param:
            diccionario_param[param] = diccionario_param[param]+1
        else: # Si no se encuentra, es 1
            diccionario_param[param] = 1

    return diccionario_param # Se devuelve el diccionario
```

```
diccionario_param = {} # Se crea un diccionario vacío
for ruta_archivo in ruta_archivos: # Donde para cada ruta se haga uso de
    la función
    diccionario_param = read_param(ruta_archivo, diccionario_param)

print(diccionario_param) # Se imprime por pantalla el diccionario
```

Anexo C. Código fn_preprocesamiento.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Funciones auxiliares para realizar el preprocesamiento
##### de DeviceEvents.json y Weather.json

import json # Se importa la biblioteca json para poder trabajar con los
archivos .json

#### PREPROCESAMIENTO 1 - DeviceEvents.json

# Se define la función que nos va a mostrar por resultado los datos del
fichero .json
# Esta función recibe como parámetro una cadena que indica la ruta del
archivo que contiene un conjunto de datos JSON

def preprocesamiento_DeviceEvents (ruta_archivo_1):

    fichero = open(ruta_archivo_1) # Se abre el archivo en modo lectura y
se almacena en la variable fichero
    data = json.load(fichero) # Se cargan los datos del archivo JSON en
la variable data.
    fichero.close() # Se cierra el archivo de listado-DeviceEvents.txt

    # Se inician dos diccionarios vacíos, uno donde se almacenará la
fecha y otro el parámetro correspondiente
    diccionario_param = {}
    diccionario_param_fecha = {}

    for elemento in data: # Se recorre cada elemento de los datos
almacenados en data
        param = elemento['meta']['param'] # Se extrae el valor param y se
almacena en param.
        value = elemento['value'] # Se extrae el valor correspondiente a
value y se almacena en value.
        fecha = int(elemento['ts']['$date']['$numberLong']) # Se extrae
la fecha, se almacena y se convierte a un número entero

        if type(value) is dict: # Si el valor de value es un diccionario
            if 'celsius' in value: # y en el se encuentra la palabra
"Celsius"
                value = value['celsius'] # que almacene en value el valor
de Celsius

        if param in diccionario_param: # Si el parámetro ya está en el
diccionario_param
```

```

        diccionario_param[param].append(value) #que añada el valor
        diccionario_param_fecha[param].append(fecha) #y que añada
también la fecha
    else: #sino
        diccionario_param[param] = [value] # Inicializa la clave
'param' en el diccionario diccionario_param
        diccionario_param_fecha[param] = [fecha] # y se incluye
también la fecha

    posicion_active_true = diccionario_param['active'].index(True) # Se
encuentra la posicion en la lista donde "Active" sea igual a "True" y se
almacena en esta variable
    fecha_posicion_active_true =
diccionario_param_fecha['active'][posicion_active_true] # Se obtiene la
fecha correspondiente a la posición de active_true y se almacena

    diferencia_fechaActive_fechaTemp = [abs(a-fecha_posicion_active_true)
for a in diccionario_param_fecha['local_temp']] # Se calcula la
diferencia de segundos que hay entre fecha_posicion_active_true y cada
una de las fechas en diccionario_param_fecha['local_temp'] y se almacena
en esta lista
    index_temp_inicial =
diferencia_fechaActive_fechaTemp.index(min(diferencia_fechaActive_fechaTe
mp)) # Se busca la posicion donde dicha diferencia es mínima
    temp_inicial = diccionario_param['local_temp'][index_temp_inicial] #
Se obtiene el valor de temperatura inicial correspondiente y se almacena

    diferencia_fechaActive_fechaHum = [abs(a-fecha_posicion_active_true)
for a in diccionario_param_fecha['humidity']] # Se repite con la humedad
inicial (Active --> True)
    index_hum_inicial =
diferencia_fechaActive_fechaHum.index(min(diferencia_fechaActive_fechaHum
))
    humidity_inicial = diccionario_param['humidity'][index_hum_inicial]

    # Se vuelve a repetir todo para cuando active sea false
    posicion_active_false = diccionario_param['active'].index(False)
    fecha_posicion_active_false =
diccionario_param_fecha['active'][posicion_active_false]

    diferencia_fechaActive_fechaTemp = [abs(a-
fecha_posicion_active_false) for a in
diccionario_param_fecha['local_temp']]
    index_temp_final =
diferencia_fechaActive_fechaTemp.index(min(diferencia_fechaActive_fechaTe
mp))
    temp_final = diccionario_param['local_temp'][index_temp_final]

```

```

    diferencia_fechaActive_fechaHum = [abs(a-fecha_posicion_active_false)
for a in diccionario_param_fecha['humidity']]
    index_hum_final =
diferencia_fechaActive_fechaHum.index(min(diferencia_fechaActive_fechaHum
))
    humidity_final = diccionario_param['humidity'][index_hum_final]

    temp_objetivo = None # Se inicializa una variable de temp_objetivo
    if 'setpoint_air_heat' in diccionario_param: # Si setpoint_air_heat
se encuentra dentro del diccionario
        temp_objetivo = diccionario_param['setpoint_air_heat'] # Se le
asigna el valor a temp_objetivo

    zona = ruta_archivo_1.split("/")[1] # Define la zona del archivo
haciendo uso del nombre de la ruta, que se encuentra a partir de la
primera barra "/"
    fecha = ruta_archivo_1.split("/")[2] # Define la fecha del archivo
haciendo uso del nombre de la ruta, que se encuentra a partir de la
segunda barra "/"

    # POR DEFECTO LA TEMPERATURA OBJETIVO ES LA ÚLTIMA DADA
    if temp_objetivo is not None:
        temp_objetivo = temp_objetivo[-1]

    #Se devuelven los valores en un diccionario
    return {'zona':zona, 'fecha':fecha,
            'temp_inicial': temp_inicial, 'humidity_inicial':
humidity_inicial,
            'temp_final': temp_final, 'humidity_final': humidity_final,
            'temp_objetivo' : temp_objetivo,
            'ts_activeTrue': fecha_posicion_active_true,
            'ts_activeFalse': fecha_posicion_active_false }

```

PREPROCESAMIENTO 2 - Weather.json

```

# Se define la función que nos va a mostrar por resultado los datos del
fichero .json
# Esta función recibe como parámetro una cadena que indica la ruta del
archivo que contiene un conjunto de datos JSON

```

```

def preprocesamiento_Weather (ruta_archivo_2):

    #Abre el archivo json y carga su contenido en data
    fichero = open(ruta_archivo_2)
    data = json.load(fichero)
    fichero.close()

```

```

#Se crean doce listas vacías
tiempo_lista = []
temp_lista = []
feels_like_lista = []
pressure_lista = []
humidity_lista = []
dew_point_lista = []
uvi_lista = []
clouds_lista = []
visibility_lista = []
wind_speed_lista = []
wind_deg_lista = []
wind_gust_lista = []
pop_lista = []

# Se recorren todos los elementos del archivo .json, se extraen los
valores que se vayan a usar y se almacenan en las listas creadas
anteriormente
for elemento in data:
    tiempo_lista.append(elemento['ts']['$date']['$numberLong'])
    temp_lista.append(elemento['weather']['temp']['celsius'])
    feels_like_lista.append(elemento['weather']['feels_like']['celsius'])
    pressure_lista.append(elemento['weather']['pressure'])
    humidity_lista.append(elemento['weather']['humidity'])
    dew_point_lista.append(elemento['weather']['dew_point']['celsius'])
    uvi_lista.append(elemento['weather']['uvi'])
    clouds_lista.append(elemento['weather']['clouds'])
    visibility_lista.append(elemento['weather']['visibility'])
    wind_speed_lista.append(elemento['weather']['wind_speed'])
    wind_deg_lista.append(elemento['weather']['wind_deg'])
    wind_gust_lista.append(elemento['weather']['wind_gust'])
    pop_lista.append(elemento['weather']['pop'])

zona = ruta_archivo_2.split("/")[1] #Define la zona del archivo
haciendo uso del nombre de la ruta, que se encuentra a partir de la
primera barra "/"
fecha = ruta_archivo_2.split("/")[2] #Define la fecha del archivo
haciendo uso del nombre de la ruta, que se encuentra a partir de la
segunda barra "/"

#Se crea un diccionario que contiene los datos
resultado_2 = {'zona': zona,
               'fecha': fecha,
               'tiempo_lista': tiempo_lista,
               'temp_lista': temp_lista,

```

```
    'feels_like_lista': feels_like_lista,  
    'pressure_lista': pressure_lista,  
    'humidity_lista': humidity_lista,  
    'dew_point_lista': dew_point_lista,  
    'uvi_lista': uvi_lista,  
    'clouds_lista': clouds_lista,  
    'visibility_lista': visibility_lista,  
    'wind_speed_lista': wind_speed_lista,  
    'wind_deg_lista': wind_deg_lista,  
    'wind_gust_lista': wind_gust_lista,  
    'pop_lista': pop_lista}  
  
    return (resultado_2) # Se devuelve el resultado
```

Anexo D. Código A1_Preprocesamiento.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Procesar todos los json y obtener un dataset
almacenado en tsv

# Variables a construir:
# - V1: Temperatura inicial del interior (Primera temperatura tras active
== True)
# - V2: Humedad inicial del interior (Primera humedad tras active ==
True)
# - V3: Temperatura final del interior (Primera temperatura tras active
== False)
# - V4: Humedad final del interior (Primera temperatura tras active ==
False)
# - V5-V16: Datos de weather más cercanos al active-True
# - V17-V28: Datos de weather más cercanos al active-False
# - Variable a predecir: tiempo entre active-True y active-False

#\n significa salto de línea
#\t significa tabulación

# Se importa las bibliotecas json y os
import json
import os

from fn_preprocesamiento import preprocesamiento_DeviceEvents # Importa
la función preprocesamiento_DeviceEvents del archivo preprocesamiento
from fn_preprocesamiento import preprocesamiento_Weather # Importa la
función preprocesamiento_Weather del archivo preprocesamiento

# Se define la ruta absoluta de la carpeta con la que se está trabajando
ruta_absoluta = r"C:/Users/Marina/Desktop/python_tfg"
os.chdir(ruta_absoluta)

archivo_listado_DeviceEvents = 'listado-DeviceEvents.txt' # Se define la
variable para acabar la ruta
archivo_1 = open(archivo_listado_DeviceEvents) # Se abre el archivo
ruta_archivos_1 = archivo_1.read().splitlines() # Se lee el archivo,
splitlines separa una línea de la otra
archivo_1.close() # Se cierra el archivo de listado-DeviceEvents.txt

archivo_listado_weather = "listado-weather.txt" # Se repite lo mismo con
listado-Weather.txt
archivo_2 = open(archivo_listado_weather)
```

```

ruta_archivos_2 = archivo_2.read().splitlines()
archivo_2.close()

# Crear archivo TSV y escribir encabezados de columnas
with open("dataset.tsv", "w") as archivo_tsv: #Abre el archivo en modo
escritura y lo asigna a la variable archivo_tsv.

    #Escribimos el encabezado de la funcion
    archivo_tsv.write("zona\tfecha\ttemp_inicial\ttemp_final\thumidity_in
    icial\thumidity_final\ttemp_objetivo\ttiempo_activeTrue\ttiempo_activeFal
    se\ttemp_activeTrue\ttemp_activeFalse\tfeels_like_activeTrue\tfeels_like_
    activeFalse\tpressure_activeTrue\tpressure_activeFalse\thumidity_activeTr
    ue\thumidity_activeFalse\tdew_point_activeTrue\tdew_point_activeFalse\tuv
    i_activeTrue\tuvi_activeFalse\tclouds_activeTrue\tclouds_activeFalse\tvis
    ibility_activeTrue\tvisibility_activeFalse\twind_speed_activeTrue\twind_s
    peed_activeFalse\twind_deg_activeTrue\twind_deg_activeFalse\twind_gust_ac
    tiveTrue\twind_gust_activeFalse\tpop_activeTrue\tpop_activeFalse\n")

    # Procesar cada archivo y escribir los valores en el archivo TSV
    # Este bucle for itera simultáneamente sobre los elementos de
    ruta_archivos_1 y ruta_archivos_2 haciendo uso de la función zip(),
    haciendo que se combinen las rutas de las dos listas.
    for ruta_archivo_1, ruta_archivo_2 in zip(ruta_archivos_1,
    ruta_archivos_2):

        # Llama a la función preprocesamiento_1 y le asigna el resultado
        de la misma a la variable resultado_1
        resultado_1 = preprocesamiento_DeviceEvents(ruta_archivo_1)
        # Llama a la función preprocesamiento_1 y le asigna el resultado
        de la misma a la variable resultado_2
        resultado_2 = preprocesamiento_Weather(ruta_archivo_2)

        # Explicación por partes:
        ### Se crea una línea de texto separada por tabulaciones usando
        los valores devueltos por el dict. resultado_1.
        ### los cuales se convierten en stream y se unen haciendo uso de
        join()
        ### En cada iteración del bucle for se toma el valor de uno de
        los elementos de esa lista, que son
        ### nombres de columnas que se esperan encontrar en el
        diccionario resultado_1
        linea = "\t".join(str(resultado_1[columna]) for columna in
        ["zona", "fecha", "temp_inicial", "temp_final", "humidity_inicial",
        "humidity_final", "temp_objetivo", "ts_activeTrue", "ts_activeFalse"])

        # Desglose de resultado_2 para conseguir las métricas más
        cercanas a active-True y active-False

```

```

        diferencia_fechaActive_fechaWeather =
[abs(resultado_1['ts_activeTrue']-int(a)) for a in
resultado_2['tiempo_lista']]
        diferencia_fechaFalse_fechaWeather =
[abs(resultado_1['ts_activeFalse']-int(a)) for a in
resultado_2['tiempo_lista']]
        index_weather_active =
diferencia_fechaActive_fechaWeather.index(min(diferencia_fechaActive_fechaWeather))
        index_weather_false =
diferencia_fechaFalse_fechaWeather.index(min(diferencia_fechaFalse_fechaWeather))

        # Se hace lo mismo que antes, pero con los datos de la función
preprocesamiento_Weather (resultado_2)
        # concatenándolos anteriormente con la línea de
preprocesamiento_DeviceEvents y el tabulador
        for variable in ["temp_lista", "feels_like_lista",
"pressure_lista", "humidity_lista", "dew_point_lista", "uvi_lista",
"clouds_lista", "visibility_lista", "wind_speed_lista", "wind_deg_lista",
"wind_gust_lista", "pop_lista"]:
            linea = linea + "\t" +
str(resultado_2[variable][index_weather_active]) + "\t" +
str(resultado_2[variable][index_weather_false])

        #Escribe la línea y la concatena con un salto de línea antes de
pasar al siguiente archivo
        archivo_tsv.write(linea + "\n")

```

Anexo E. Código A2_Inspeccion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Se eliminan y crean variables por conocimiento del
problema
##### Se crea un dataset_clean.tsv con las variables que
vamos a usar en el proyecto.

# Tras hacer A2i_histogramas, se eliminan de columnas_interes aquellas
# variables que nos proporciona el mismo dato en todas las bases de
# datos. En este código están todas escritas.

# Se importa la librería pandas
import pandas as pd

# Se cargan los datos del archivo dataset.tsv en un DataFrame
data = pd.read_csv('dataset.tsv', sep='\t')

# Se crean las siguientes variables:
#### Calcular la diferencia de tiempo en mins. Esta en milisegundos,
dividimos entre 1000, para pasarlo a minutos dividimos entre 60. /60x1000
data['diff_tiempo'] = (data['tiempo_activeFalse'] -
data['tiempo_activeTrue']) / 60000
#### Calcular la diferencia de temperatura final e inicial del interior
data['diff_temp_interior'] = data['temp_final'] - data['temp_inicial']
#### Calcular la diferencia de temperatura inicial del interior y la
temperatura inicial del exterior
data['diff_temp_int_ext'] = data['temp_activeTrue'] -
data['temp_inicial']

# Seleccionar las columnas de interés
columnas_interes = ['diff_tiempo', 'diff_temp_interior',
'diff_temp_int_ext', 'temp_activeTrue',
                    'humidity_inicial', 'feels_like_activeTrue',
                    'pressure_activeTrue', 'humidity_activeTrue',
'dew_point_activeTrue',
                    'uvi_activeTrue', 'clouds_activeTrue',
                    'wind_speed_activeTrue', 'wind_deg_activeTrue',
                    'wind_gust_activeTrue', 'pop_activeTrue']

dataset_clean = data[columnas_interes]

# Guardar el nuevo dataset en un archivo
dataset_clean.to_csv('dataset_clean.tsv', sep='\t', index=False)
```

Anexo F. Código A2i_distribucion_y_descriptiva.py – python_tfg

```
##### Proyecto: TFG. Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Se realiza una inspección de las variables que vamos a
usar en el proyecto.
##### Realizamos un análisis de distribución (histograma) y
una descriptiva.

# Se importan las librerías necesarias
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os
from openpyxl import Workbook

# DESCRIPTIVA ESTADÍSTICA

# Cargar el archivo CSV clean
df = pd.read_csv('dataset_clean.tsv', delimiter='\t')
print(df.describe())

# Crear un nuevo libro de Excel y seleccionar la hoja activa
wb = Workbook()
ws = wb.active

# Obtener la descripción estadística
description = df.describe()

# Obtener los datos y nombres de las variables del DataFrame
variable_names = list(df.columns)
data = description.values.tolist()

# Agregar los nombres de las variables en la primera fila (B1 en Excel)
ws.append([""] + variable_names)

# Agregar las etiquetas de las estadísticas en la primera columna (A2 en
Excel)
stat_labels = ["count", "mean", "std", "min", "25%", "50%", "75%", "max"]
for i in range(len(stat_labels)):
    ws.cell(row=i + 2, column=1, value=stat_labels[i])

# Agregar los datos en las celdas correspondientes (B2 en Excel)
for i in range(len(data)):
    for j in range(len(data[i])):
        ws.cell(row=i + 2, column=j + 2, value=data[i][j])
```

```

# Guardar el archivo de Excel
wb.save('A2i_Descripcion_estadistica.xlsx')

# ANÁLISIS DE DISTRIBUCIÓN. HISTOGRAMAS.

# Seleccionar las columnas numéricas
numeric_columns = df.select_dtypes(include=['float64', 'int64'])

# Obtener la ruta absoluta del directorio actual
ruta_absoluta = r"C:/Users/Marina/Desktop/python_tfg"
os.chdir(ruta_absoluta)

# Construir la ruta completa de la carpeta A2i_histogramas
ruta_completa = ruta_absoluta + r"/A2i_histogramas"

# Calcular la distribución de cada columna numérica y guardar el
# histograma como imagen
for column in numeric_columns:
    plt.figure() # Crear una nueva figura para cada histograma
    sns.histplot(
        data = df,
        x = column,
        stat = "count",
        kde = True,
        line_kws= {'linewidth': 2},
        alpha = 0.3
    )
    ruta = os.path.join(ruta_completa, f"{column}.png")
    plt.savefig(ruta,dpi=300) # Guarda la figura como imagen en formato
    PNG
    plt.close() # Cerrar la figura para liberar memoria

print("Los histogramas se han guardado correctamente en la carpeta
A2i_histogramas")

```

Anexo G. Excel A2i_Descripcion_estadistica.xlsx – python_tfg

	diff_tiempo	diff_temp_i nterior	diff_temp_i nt_ext	temp_active True	humidity_i nicial	feels_like_ac tiveTrue	pressure_a ctiveTrue	humidity_a ctiveTrue
count	12,000	12,000	12,000	12,000	12,000	12,000	12,000	12,000
mean	153,490	3,150	-5,800	11,800	50,833	10,925	1017,667	81,417
std	78,007	1,441	3,814	4,464	5,890	5,210	10,671	10,664
min	34,114	0,900	-14,800	3,000	43,000	-0,100	997,000	66,000
25%	111,708	2,175	-5,725	11,250	46,250	10,650	1014,500	75,250
50%	136,382	3,150	-4,700	12,900	50,500	12,500	1019,000	81,000
75%	191,924	4,175	-3,650	14,075	56,000	13,300	1025,250	88,000
max	325,588	5,900	-1,800	16,300	59,000	15,900	1031,000	100,000

	dew_point_a ctiveTrue	uvi_activeT rue	clouds_acti veTrue	wind_speed_ activeTrue	wind_deg_ activeTrue	wind_gust_a ctiveTrue	pop_active True	visibility_ac tiveTrue
count	12,000	12,000	12,000	12,000	12,000	12,000	12,000	12,000
mean	8,592	0,592	30,000	5,125	199,500	7,701	0,196	10000,000
std	3,166	0,839	43,901	2,841	113,231	4,748	0,368	0,000
min	2,000	0,000	0,000	0,900	76,000	1,020	0,000	10000,000
25%	8,075	0,000	0,000	2,353	108,750	2,743	0,000	10000,000
50%	9,150	0,125	0,000	5,460	133,500	9,440	0,000	10000,000
75%	10,875	0,968	55,000	7,803	311,000	10,468	0,165	10000,000
max	12,000	2,730	100,000	8,250	353,000	14,760	1,000	10000,000

Anexo H. Excel A2i_Descripcion_estadistica.xlsx – python_tfg_verano

	diff_tiempo	diff_temp_in terior	diff_temp_in t_ext	temp_activeTr ue	humidity_ini cial	feels_like_acti veTrue	pressure_ac tiveTrue	humidity_ac tiveTrue
count	17,000	17,000	17,000	17,000	17,000	17,000	17,000	17,000
mean	157,186	-2,453	4,447	30,971	45,176	31,635	1013,588	44,353
std	111,587	0,716	2,809	2,924	7,316	3,649	2,785	12,505
min	45,753	-3,900	-0,900	26,200	31,000	26,200	1008,000	23,000
25%	78,326	-2,900	2,700	28,500	40,000	29,500	1013,000	34,000
50%	128,760	-2,200	4,600	31,300	46,000	31,200	1014,000	44,000
75%	194,240	-1,900	6,900	33,400	48,000	32,800	1016,000	54,000
max	442,989	-1,500	8,600	36,100	58,000	40,100	1018,000	65,000

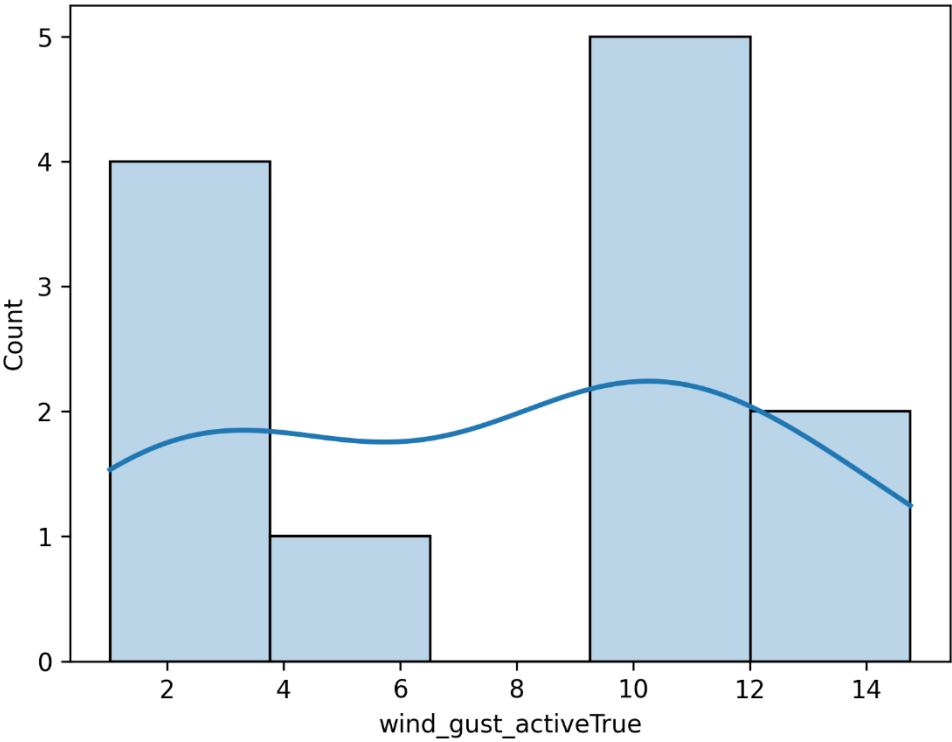
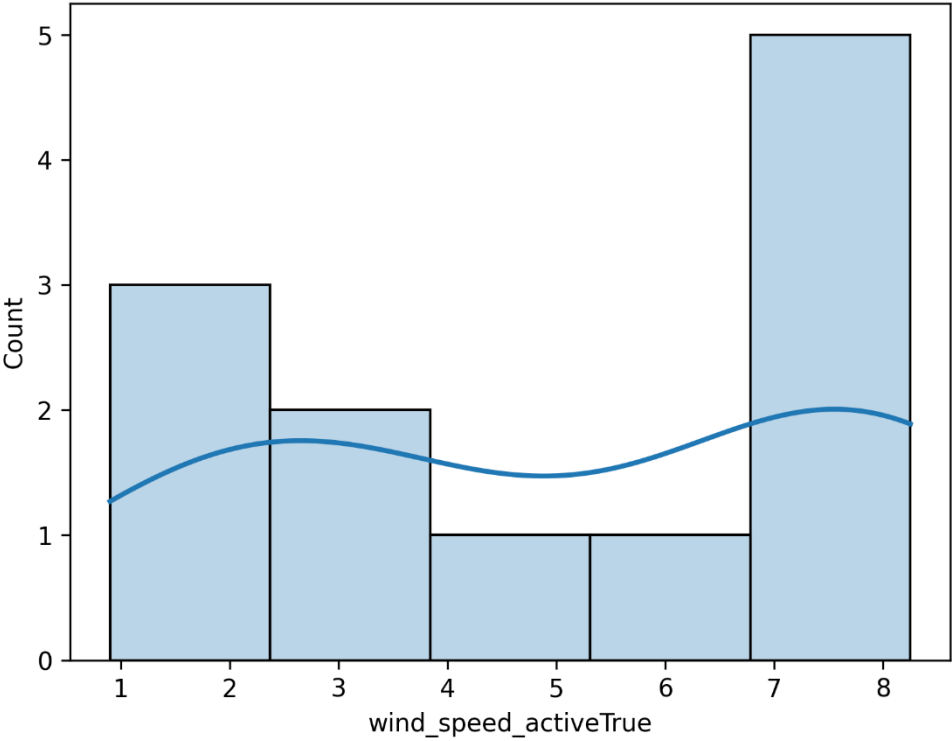
	dew_point_ac tiveTrue	uvi_activeTr ue	wind_speed _activeTrue	wind_deg_acti veTrue	wind_gust_a ctiveTrue	clouds_activeT rue	visibility_acti veTrue	pop_activeTr ue
count	17,000	17,000	17,000	17,000	17,000	17,000	17,000	17,000
mean	16,812	5,917	4,745	231,882	6,029	5,588	10000,000	0,000
std	3,477	3,195	2,493	62,844	3,626	18,531	0,000	0,000
min	10,400	0,000	0,210	156,000	0,790	0,000	10000,000	0,000
25%	15,000	5,290	2,990	181,000	3,490	0,000	10000,000	0,000
50%	17,000	6,820	4,260	221,000	5,370	0,000	10000,000	0,000
75%	19,100	8,570	6,920	270,000	9,480	0,000	10000,000	0,000
max	23,600	9,320	9,300	350,000	12,470	75,000	10000,000	0,000

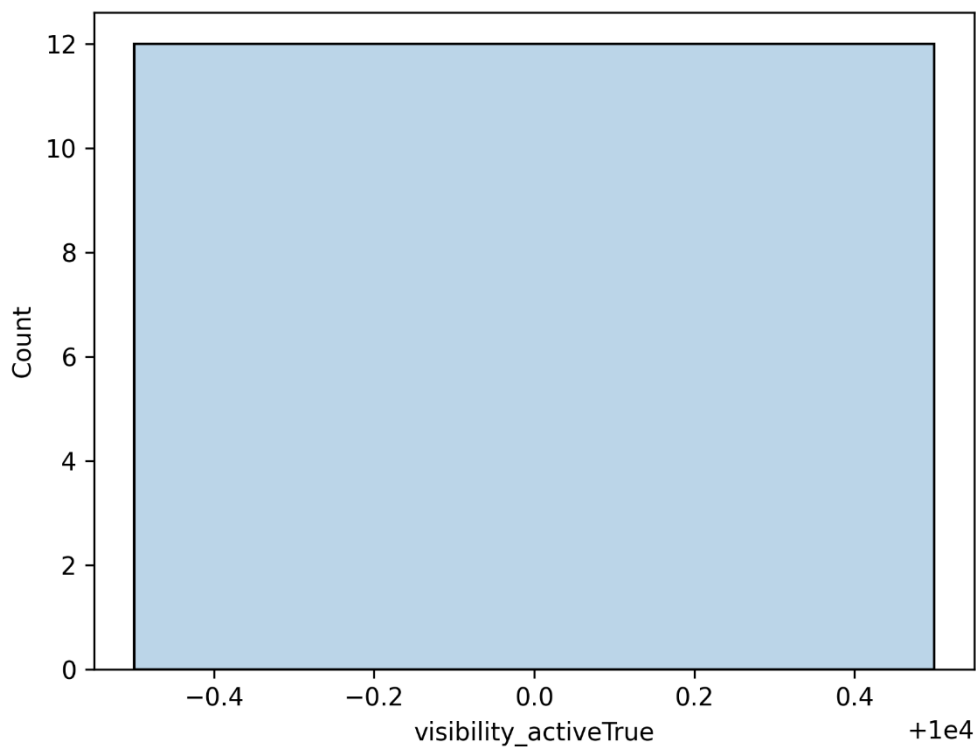
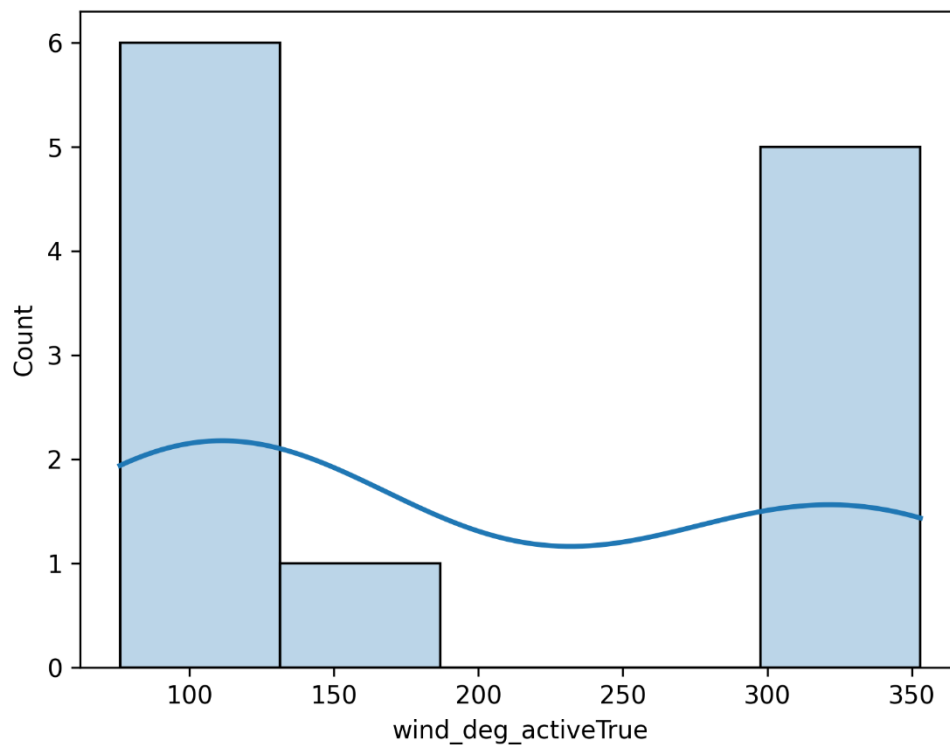
Anexo I. Excel A2i_Descripcion_estadistica.xlsx – python_tfg_conjunto

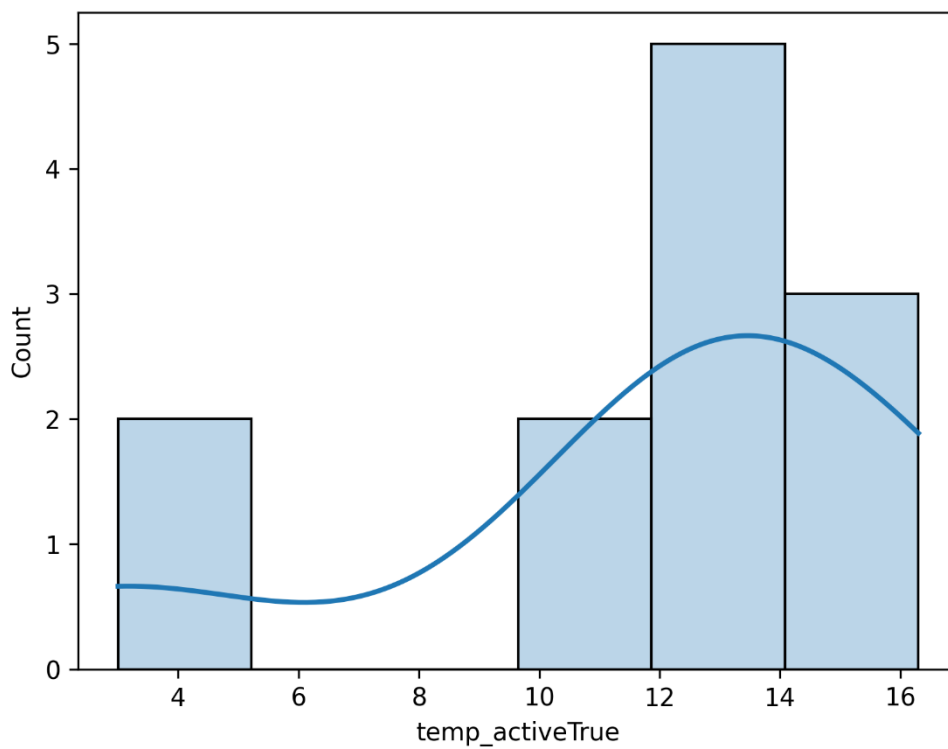
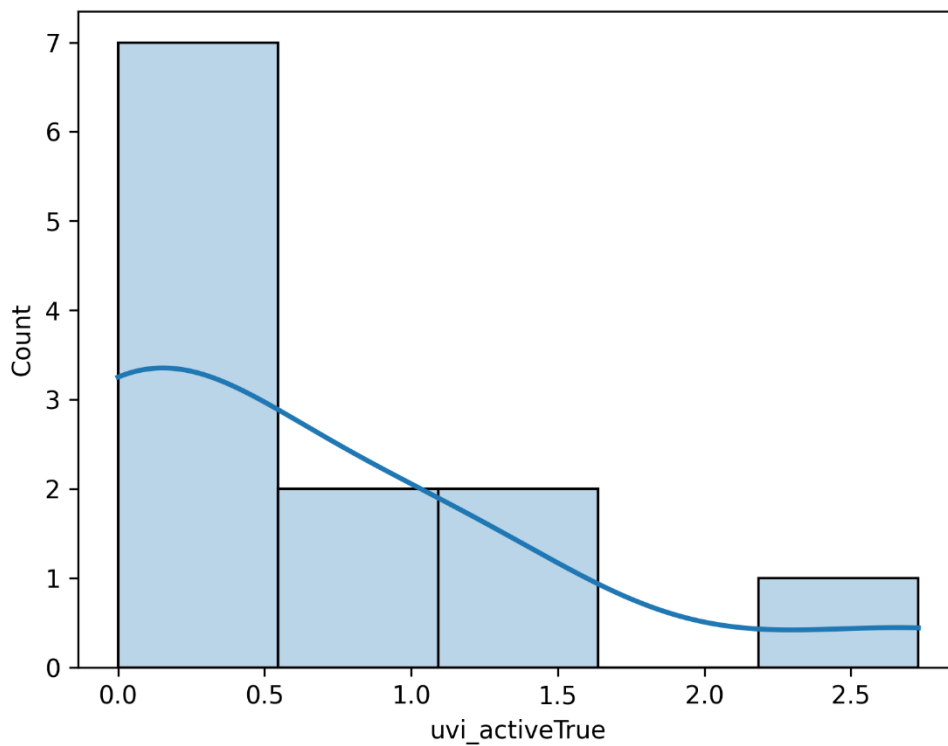
	diff_tiempo	diff_temp_in terior	diff_temp_in t_ext	temp_activeTr ue	humidity_ini cial	feels_like_acti veTrue	pressure_ac tiveTrue	humidity_ac tiveTrue
count	29,000	29,000	29,000	29,000	29,000	29,000	29,000	29,000
mean	155,657	-0,134	0,207	23,038	47,517	23,066	1015,276	59,690
std	97,515	2,999	6,050	10,249	7,229	11,226	7,304	21,890
min	34,114	-3,900	-14,800	3,000	31,000	-0,100	997,000	23,000
25%	81,309	-2,500	-3,900	13,400	44,000	12,800	1013,000	40,000
50%	133,498	-1,800	1,100	27,700	47,000	28,400	1015,000	56,000
75%	194,240	2,900	4,700	31,400	53,000	31,500	1019,000	77,000
max	442,989	5,900	8,600	36,100	59,000	40,100	1031,000	100,000

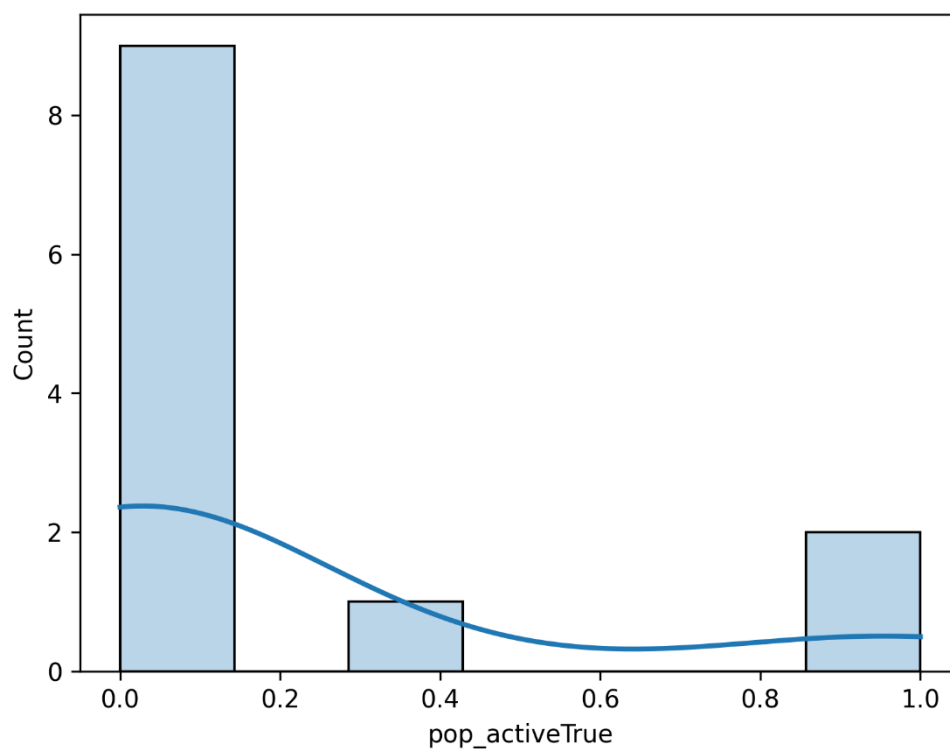
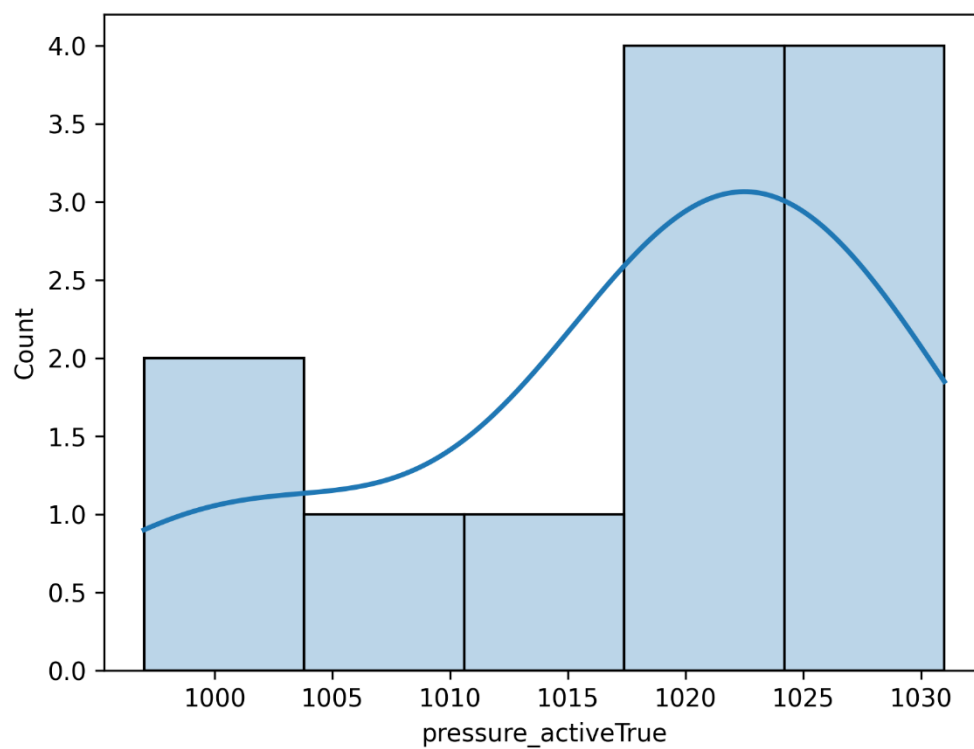
	dew_point_ac tiveTrue	uvi_activeTr ue	clouds_activ eTrue	wind_speed_a ctiveTrue	wind_deg_a ctiveTrue	wind_gust_ac tiveTrue	pop_activeTr ue	visibility_acti veTrue
count	29,000	29,000	29,000	29,000	29,000	29,000	29,000	29,000
mean	13,410	3,713	15,690	4,902	218,483	6,721	0,081	10000,000
std	5,274	3,638	33,213	2,600	86,932	4,132	0,251	0,000
min	2,000	0,000	0,000	0,210	76,000	0,790	0,000	10000,000
25%	9,500	0,100	0,000	2,480	156,000	2,760	0,000	10000,000
50%	13,900	2,200	0,000	4,360	198,000	6,930	0,000	10000,000
75%	17,900	7,360	0,000	7,430	304,000	9,900	0,000	10000,000
max	23,600	9,320	100,000	9,300	353,000	14,760	1,000	10000,000

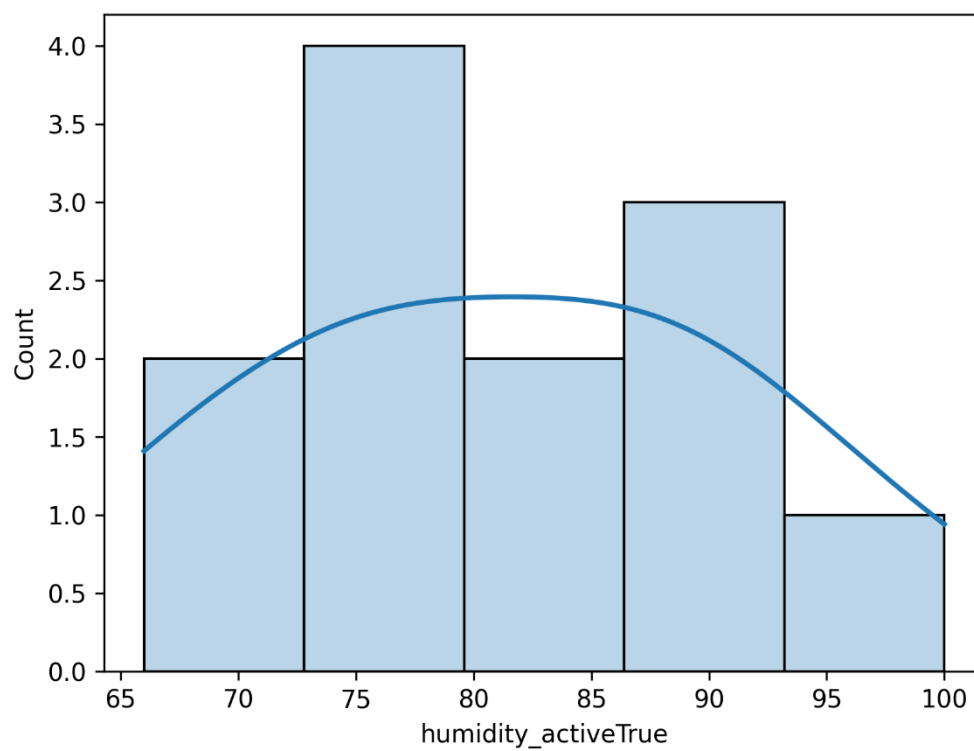
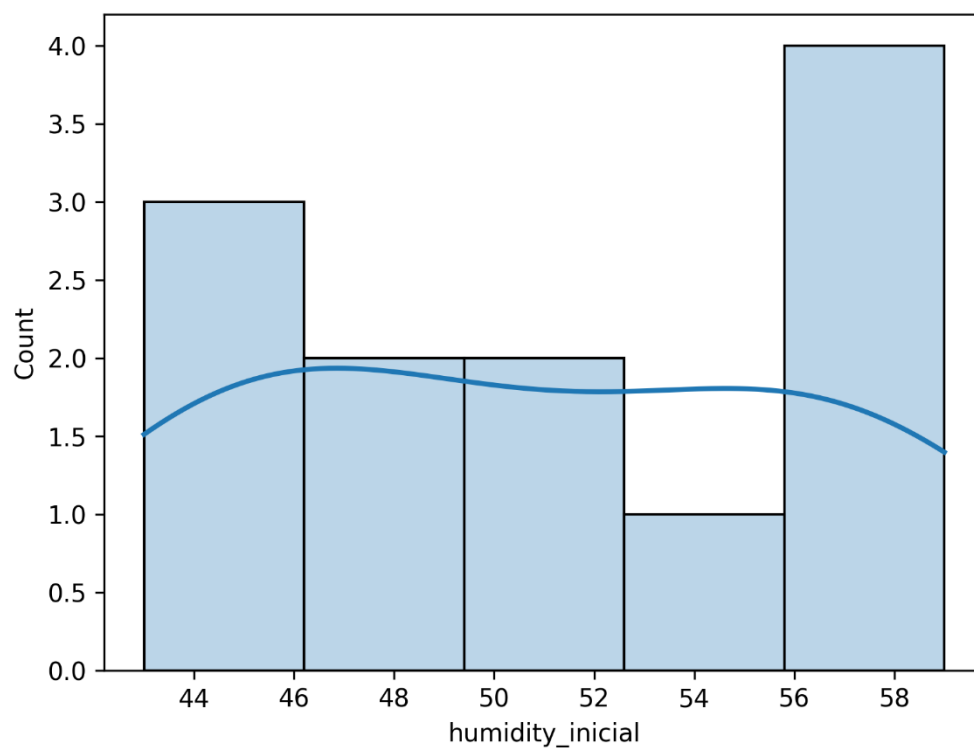
Anexo J. Imágenes de la carpeta A2i_histogramas – python_tfg

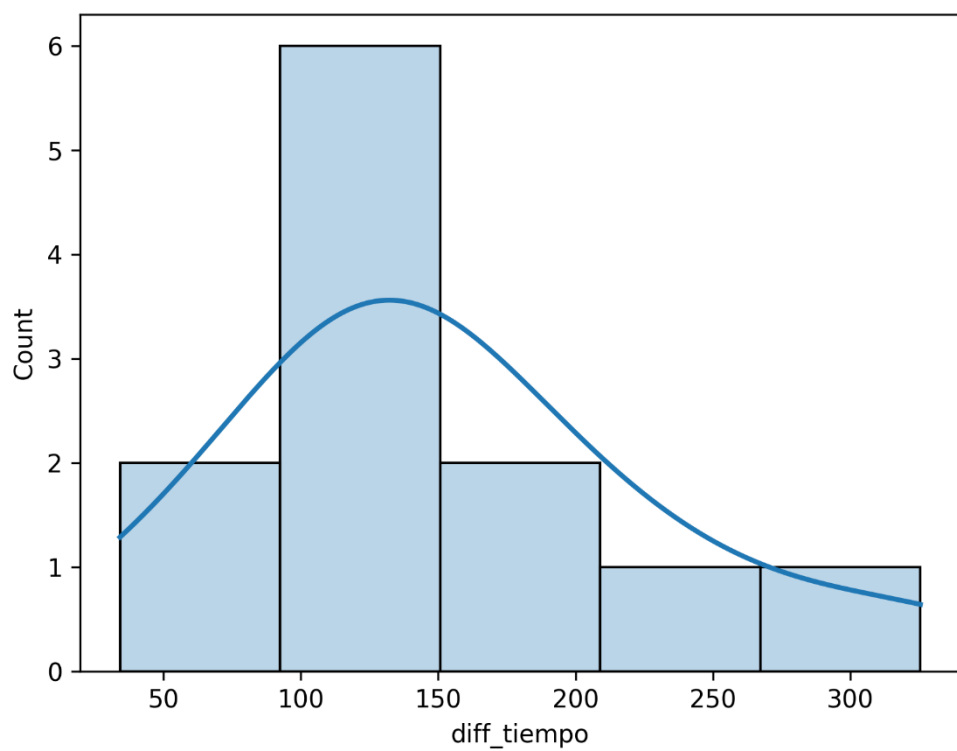
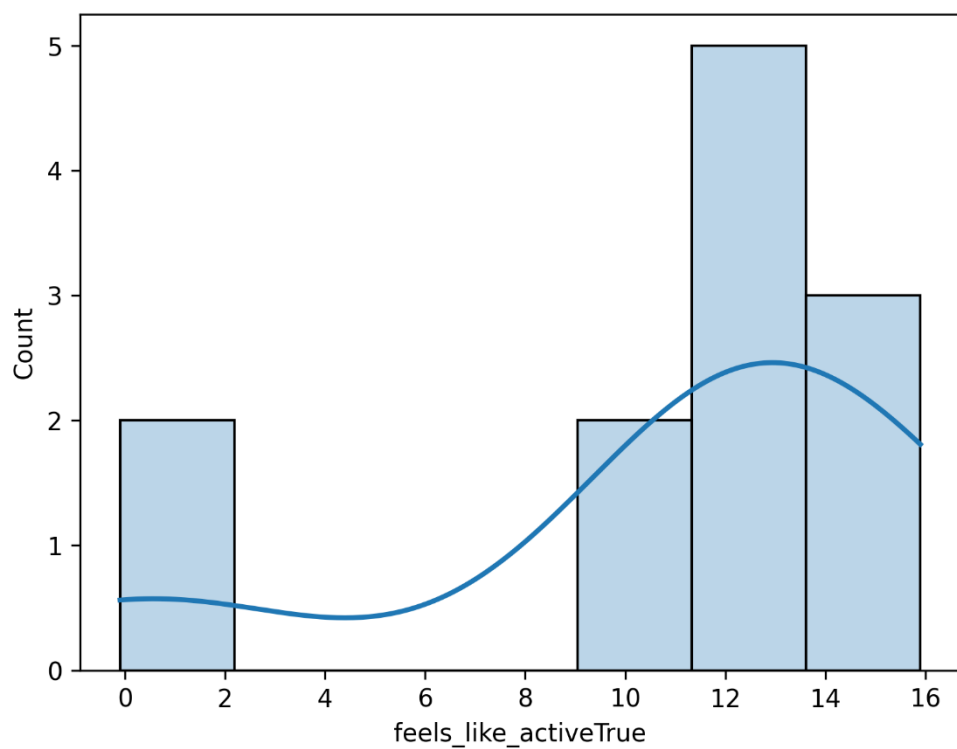


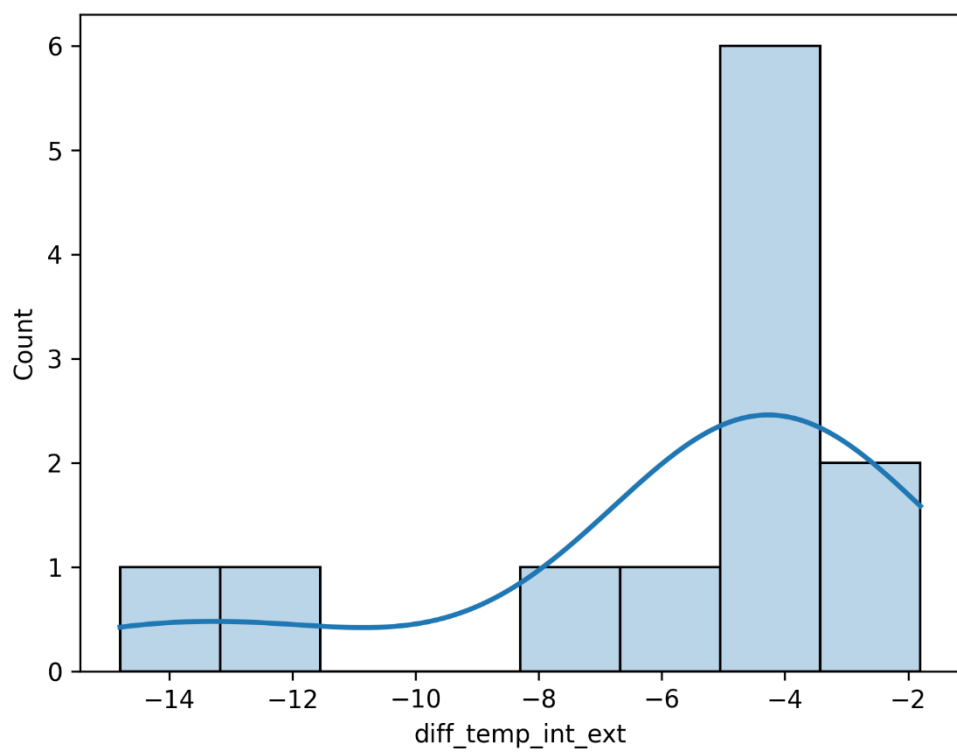
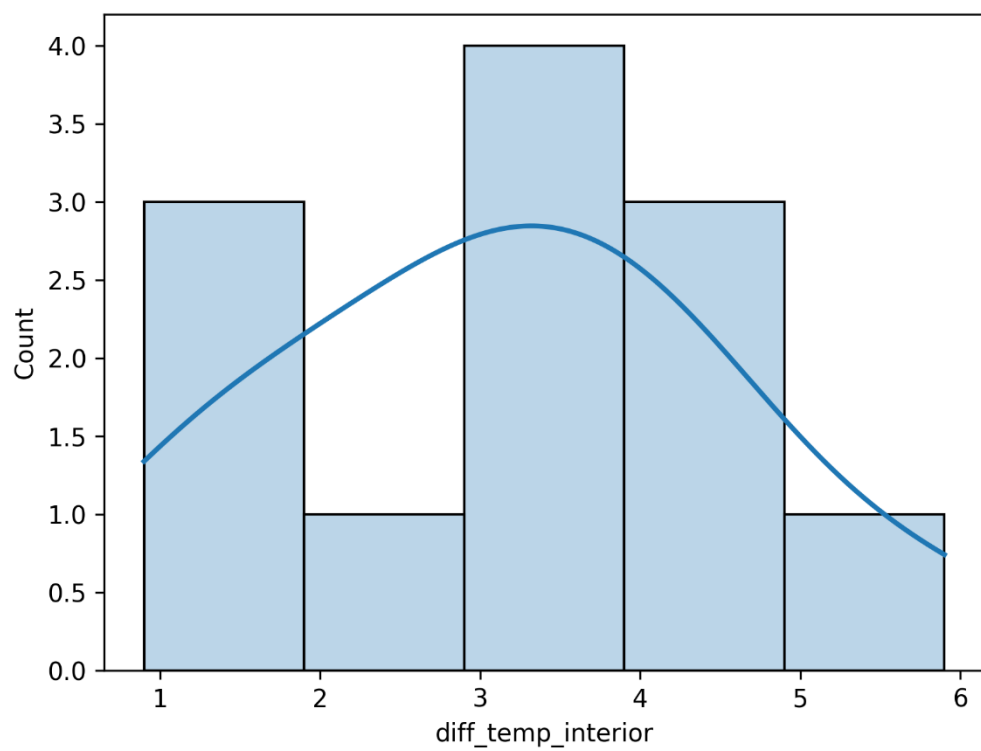


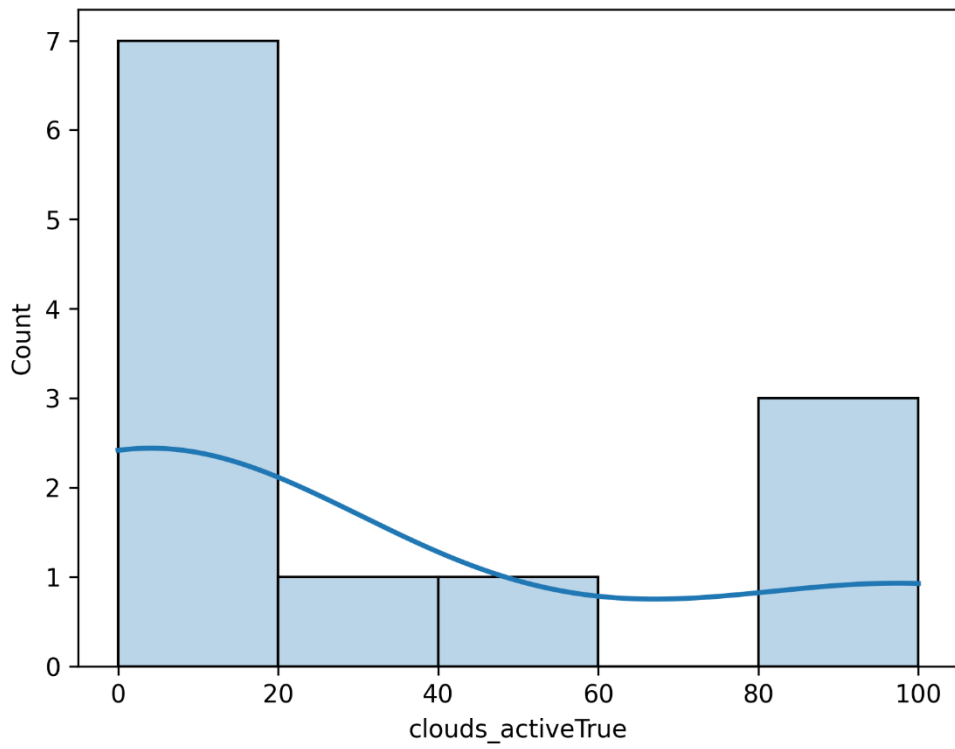
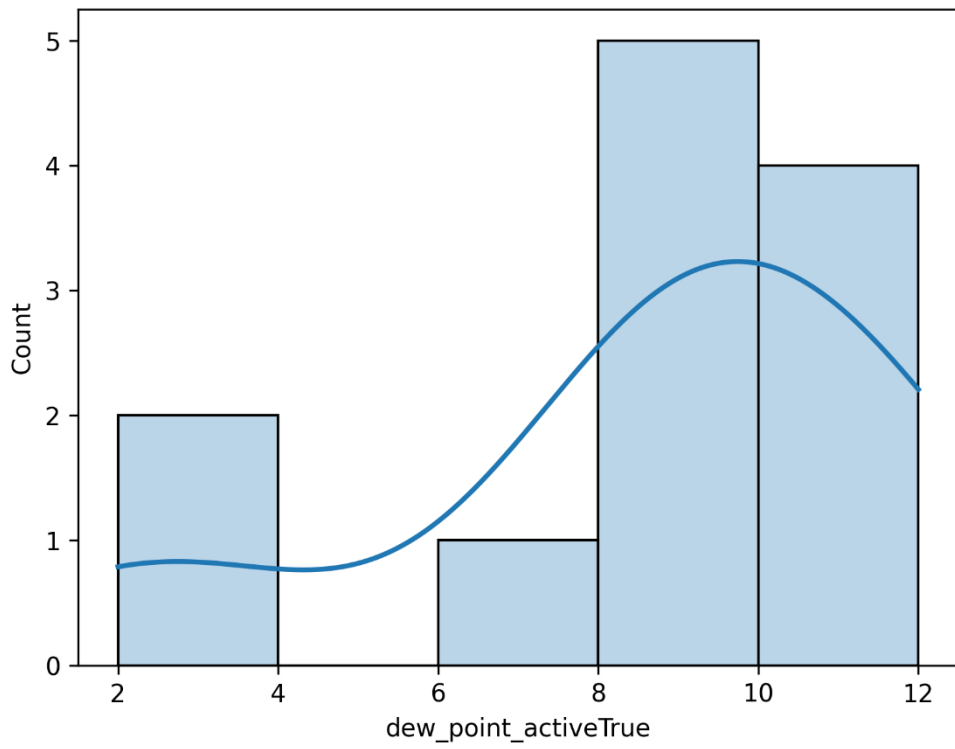




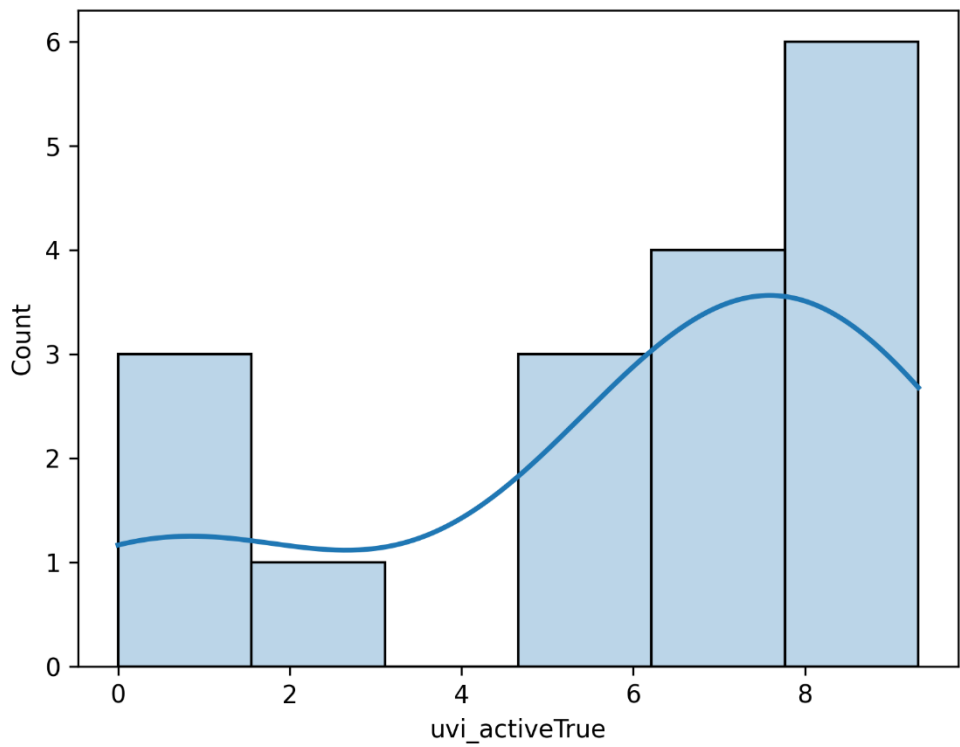
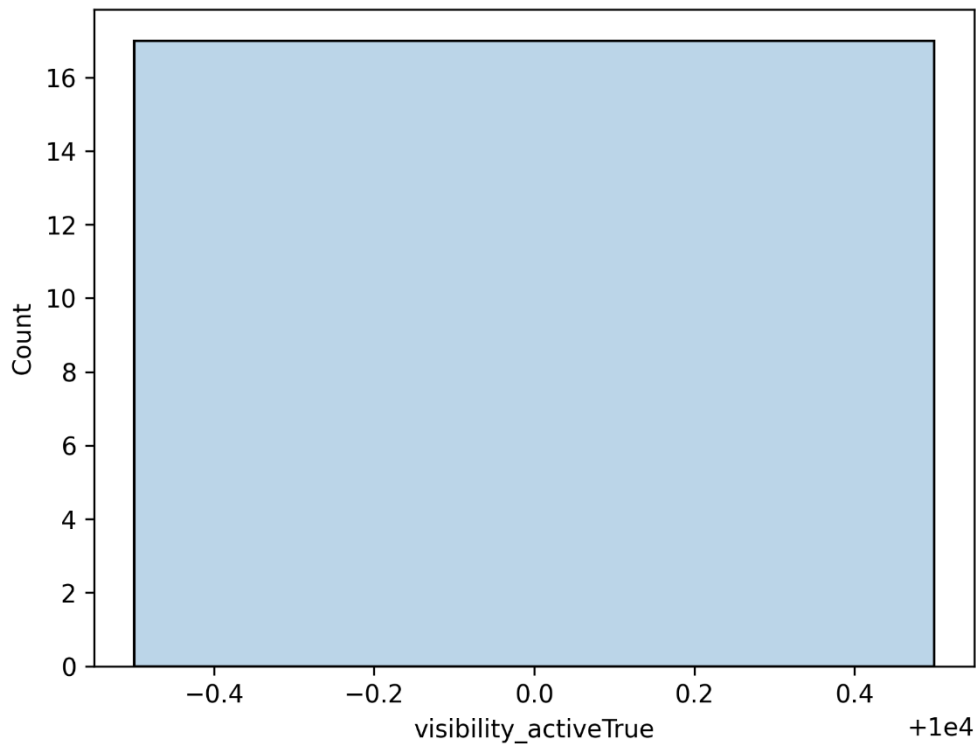


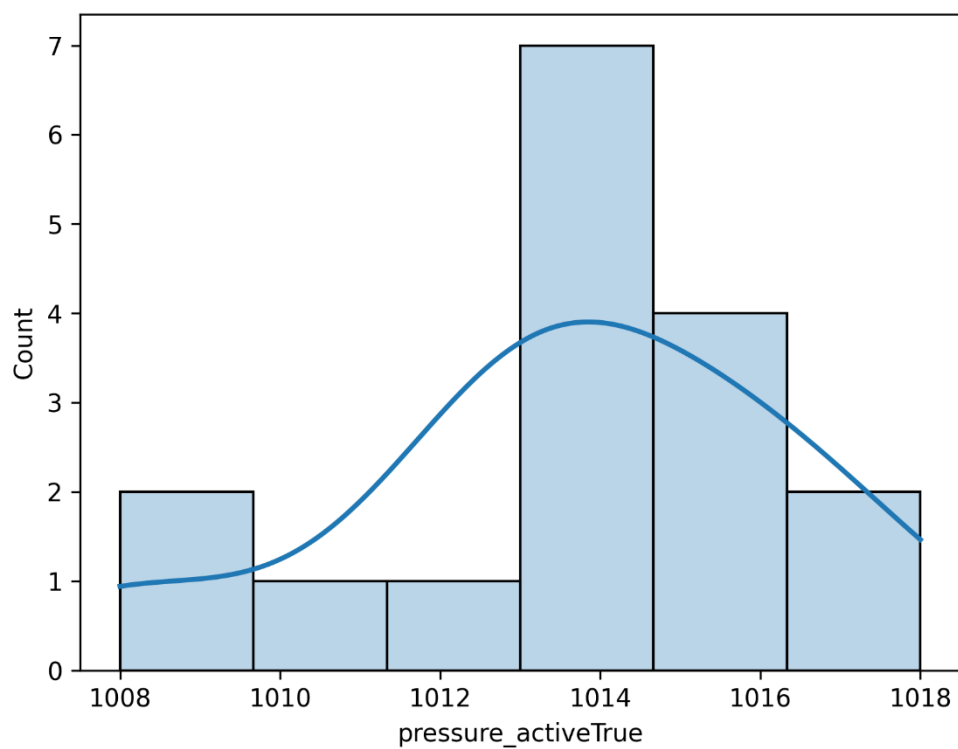
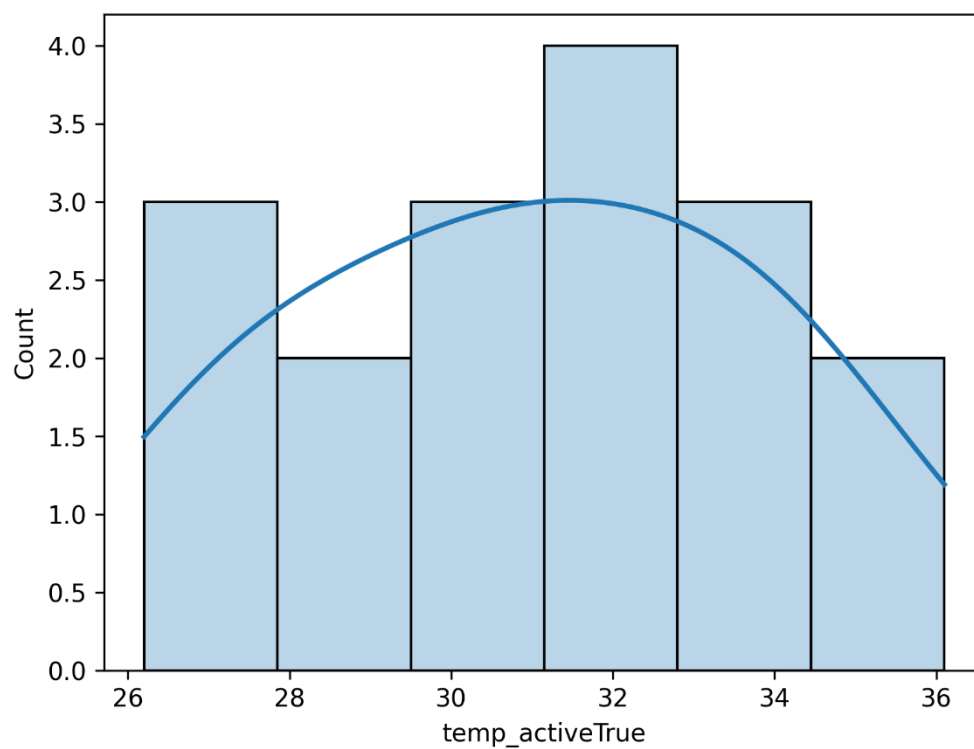


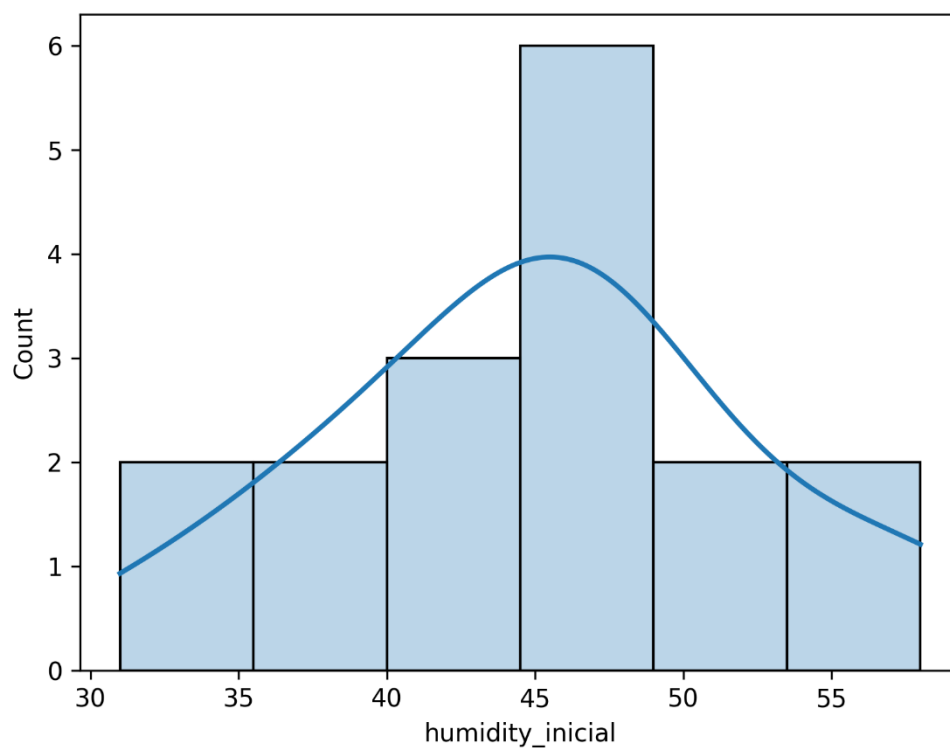
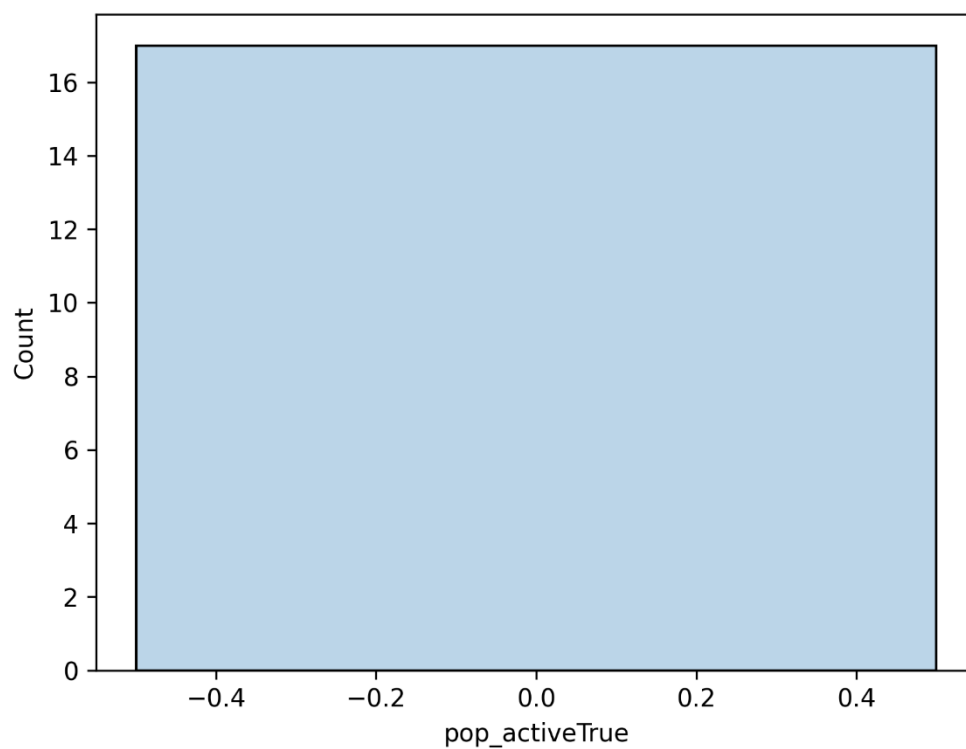


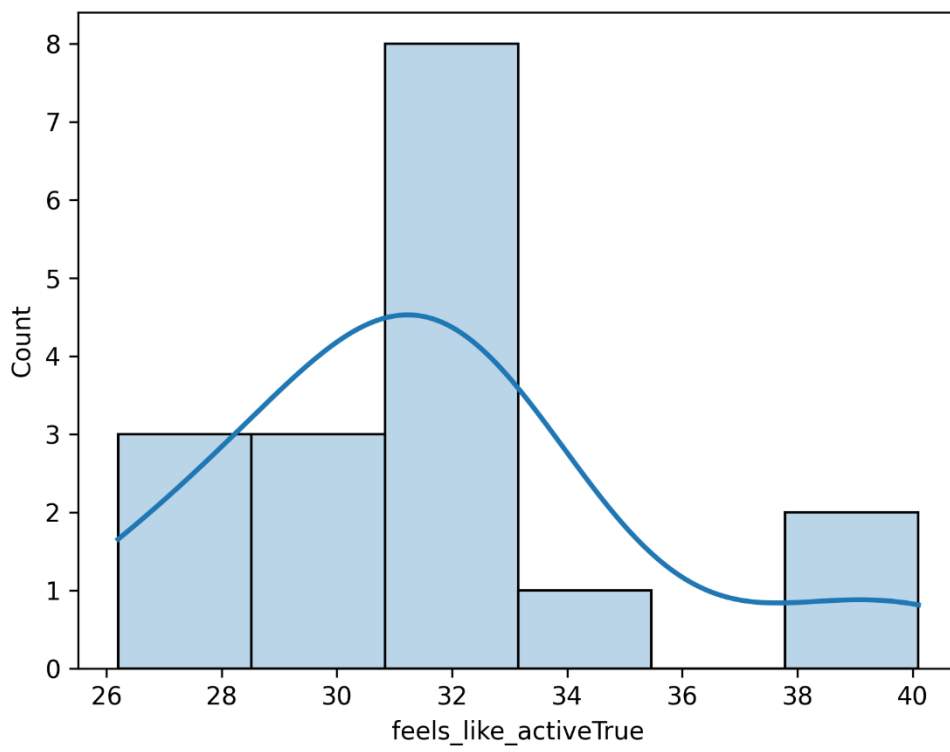
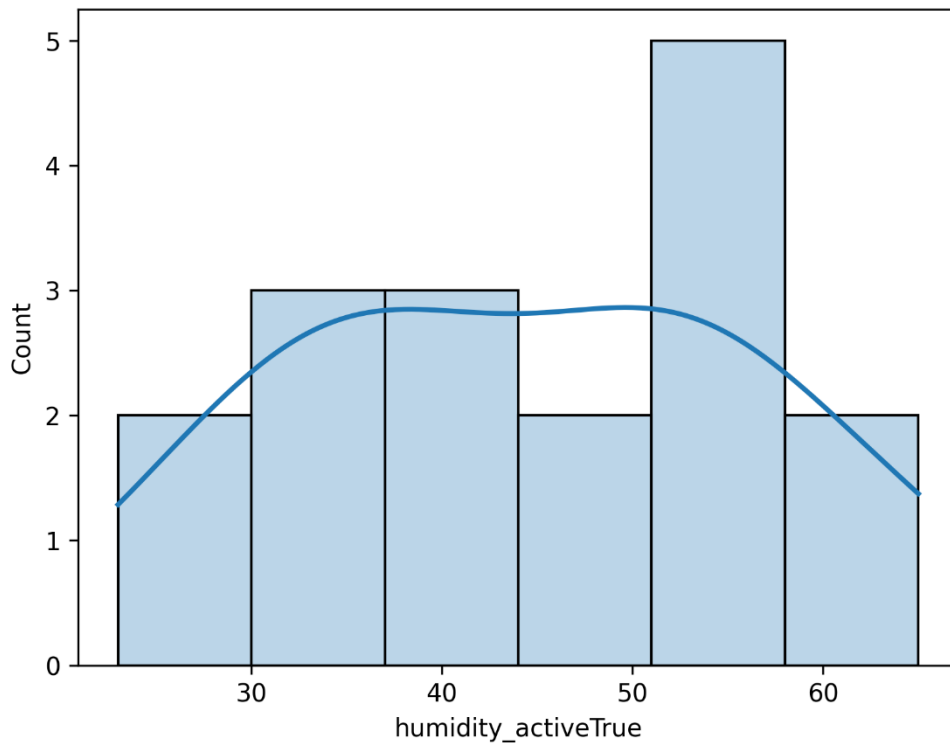


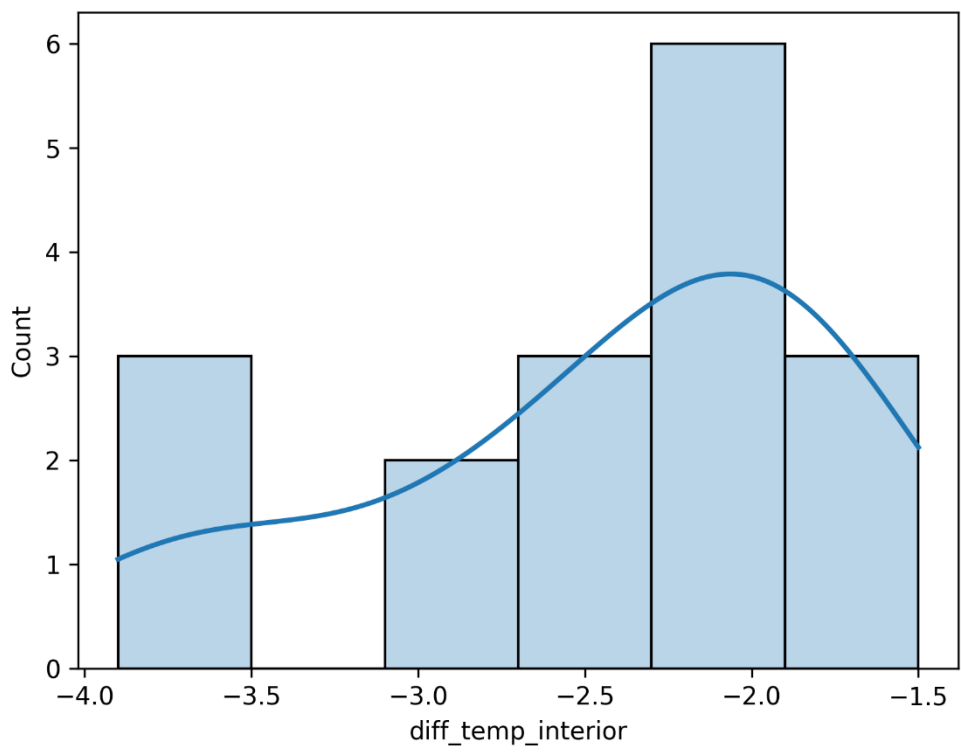
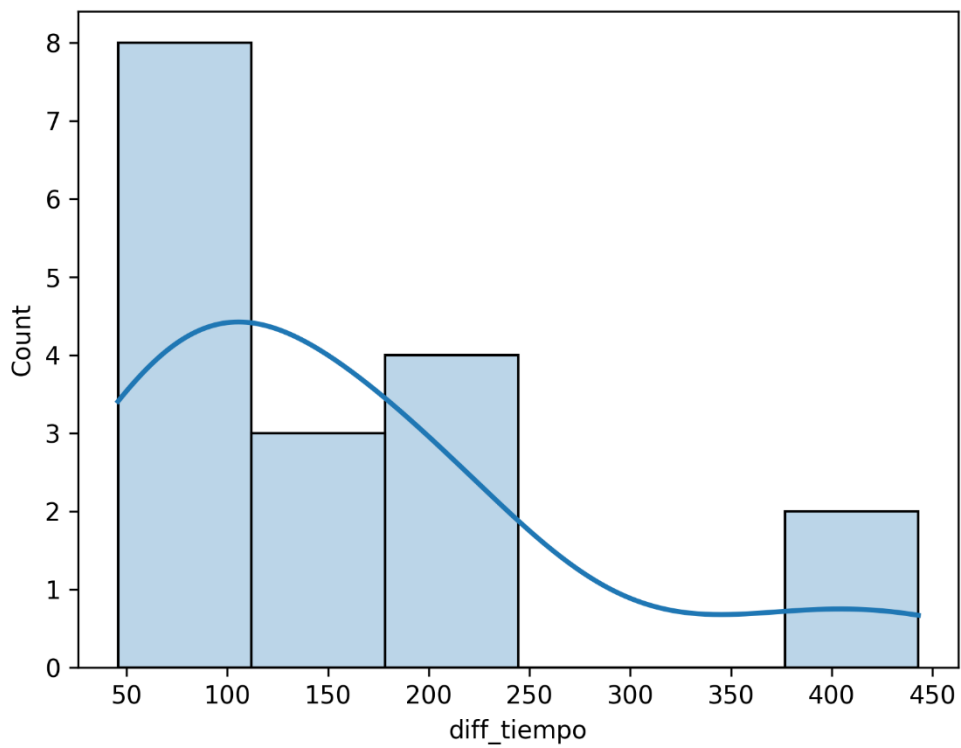
Anexo K. Imágenes de la carpeta A2i_histogramas – python_tfg_verano

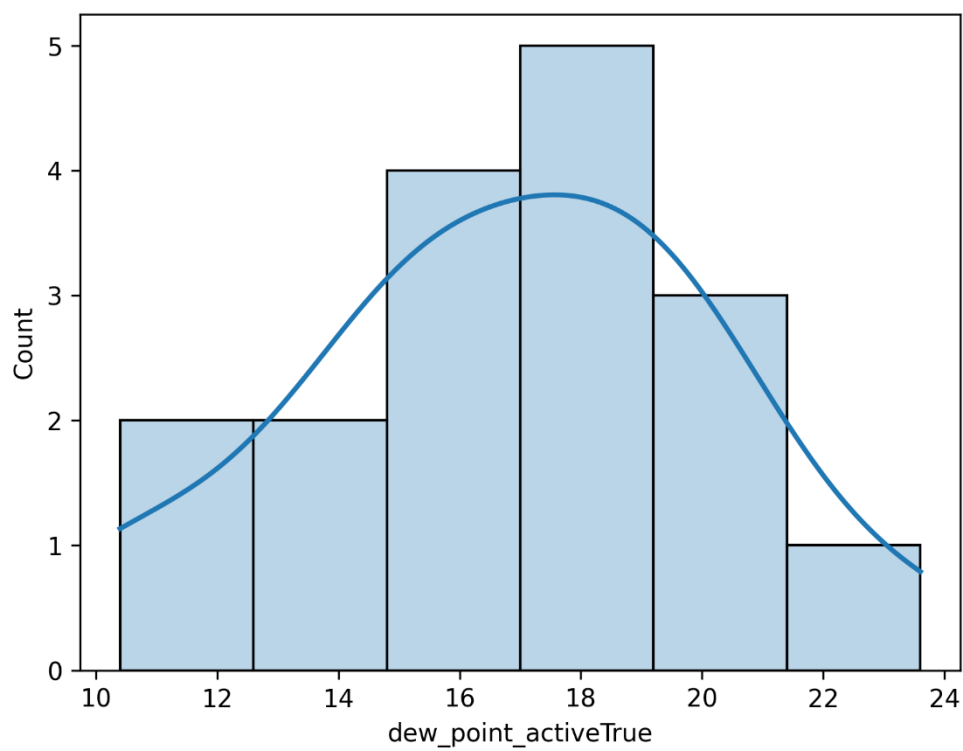
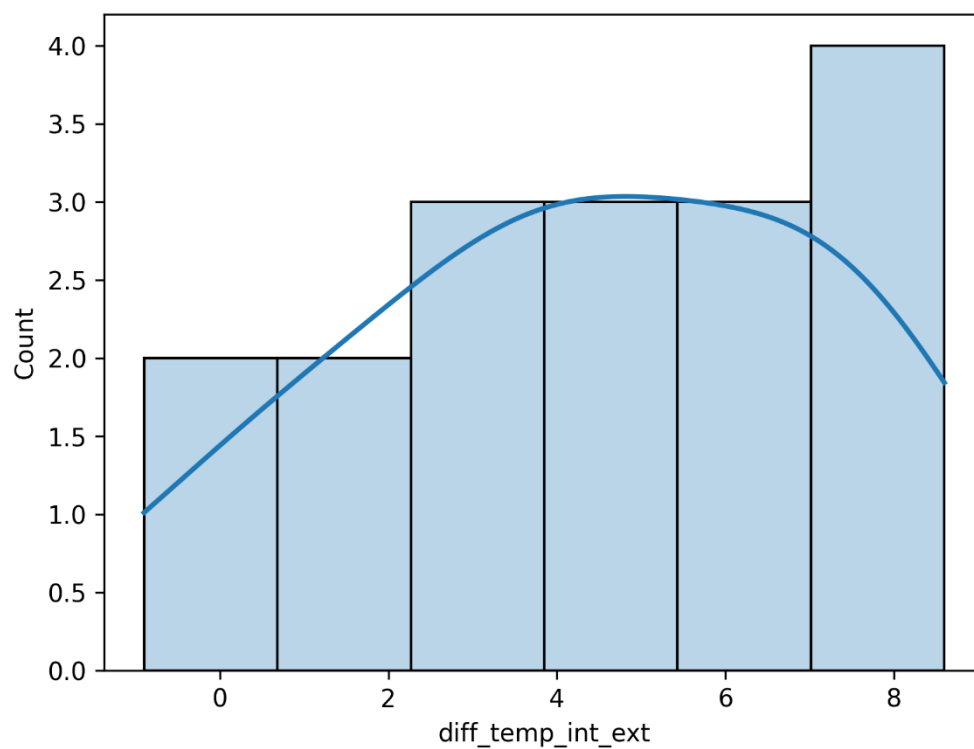


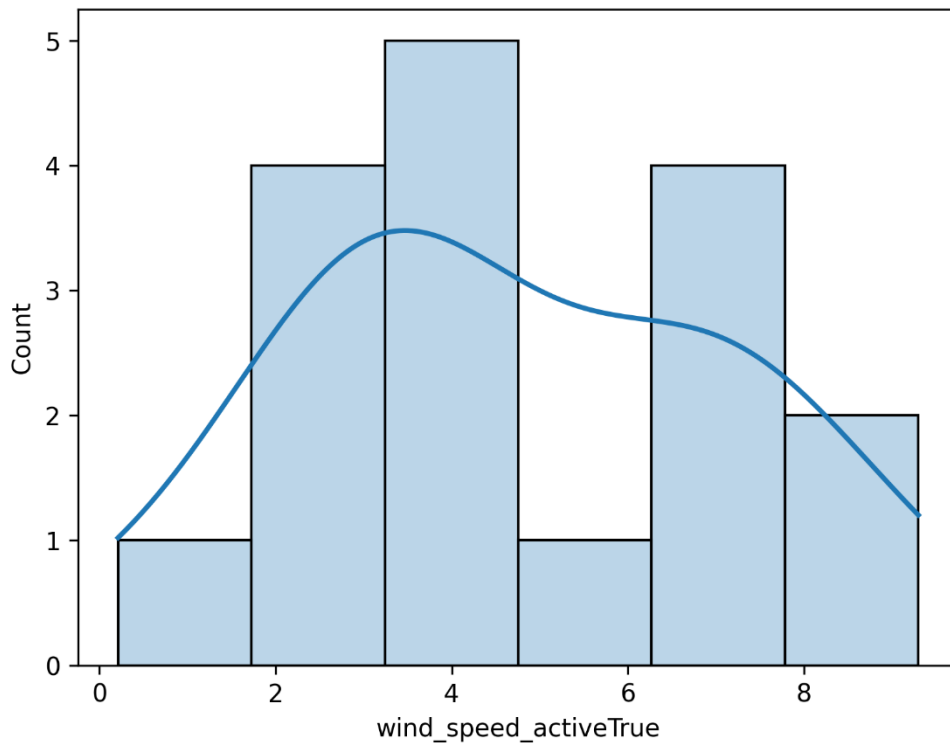
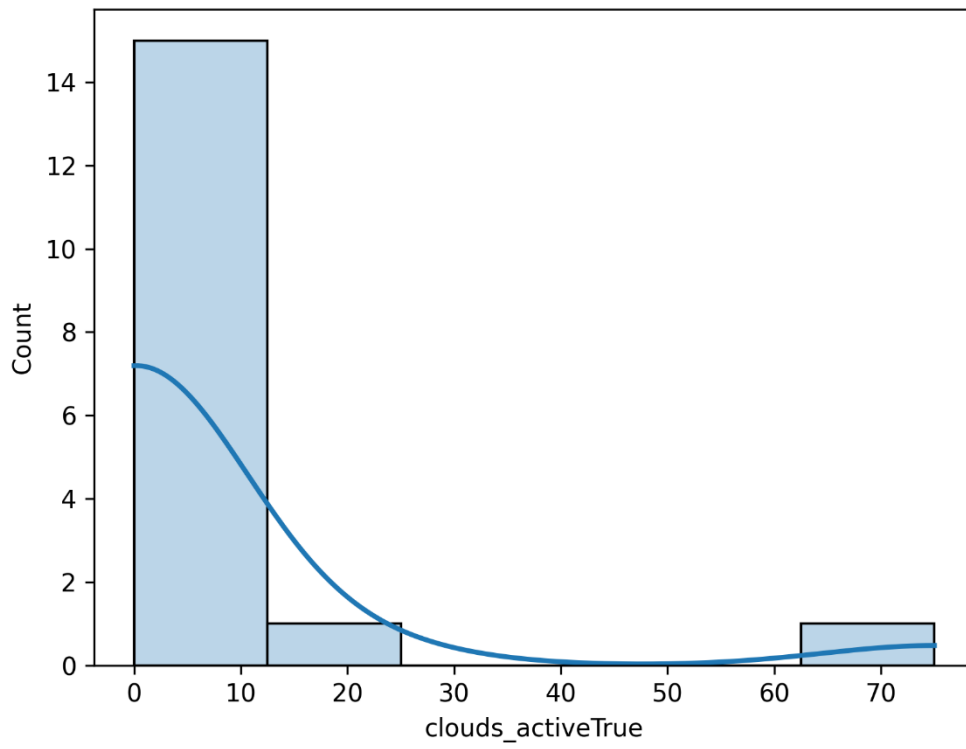


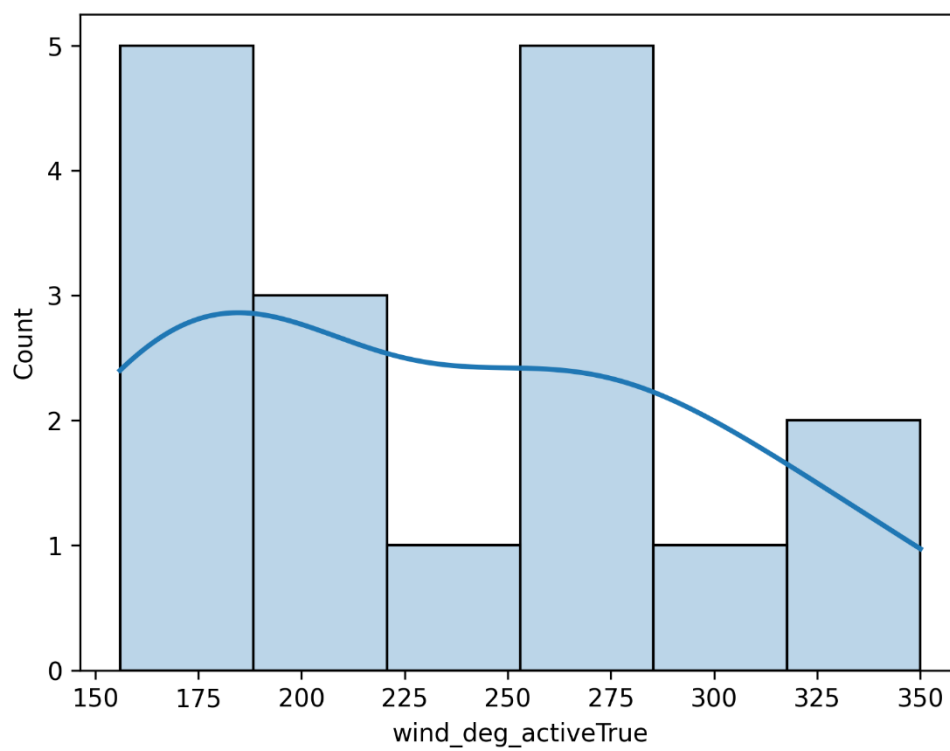
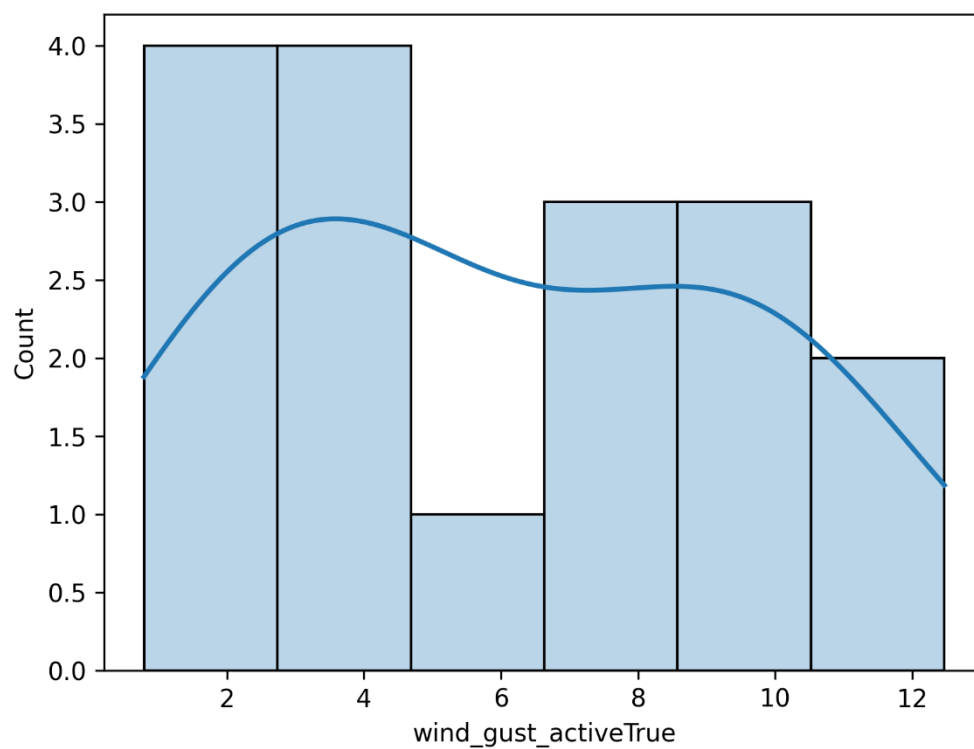




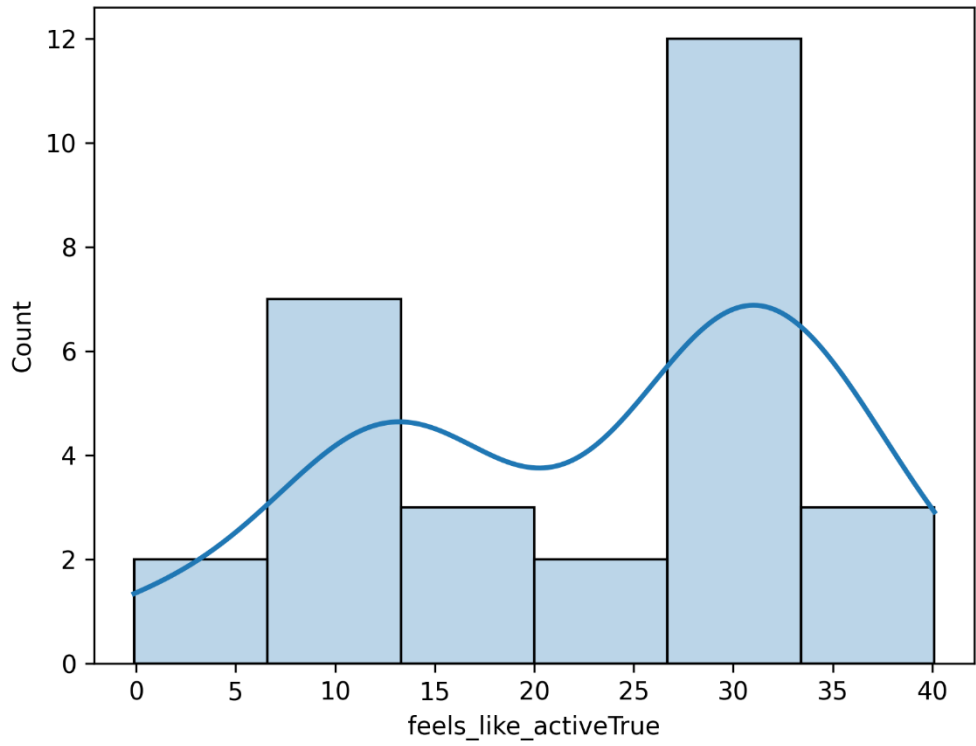
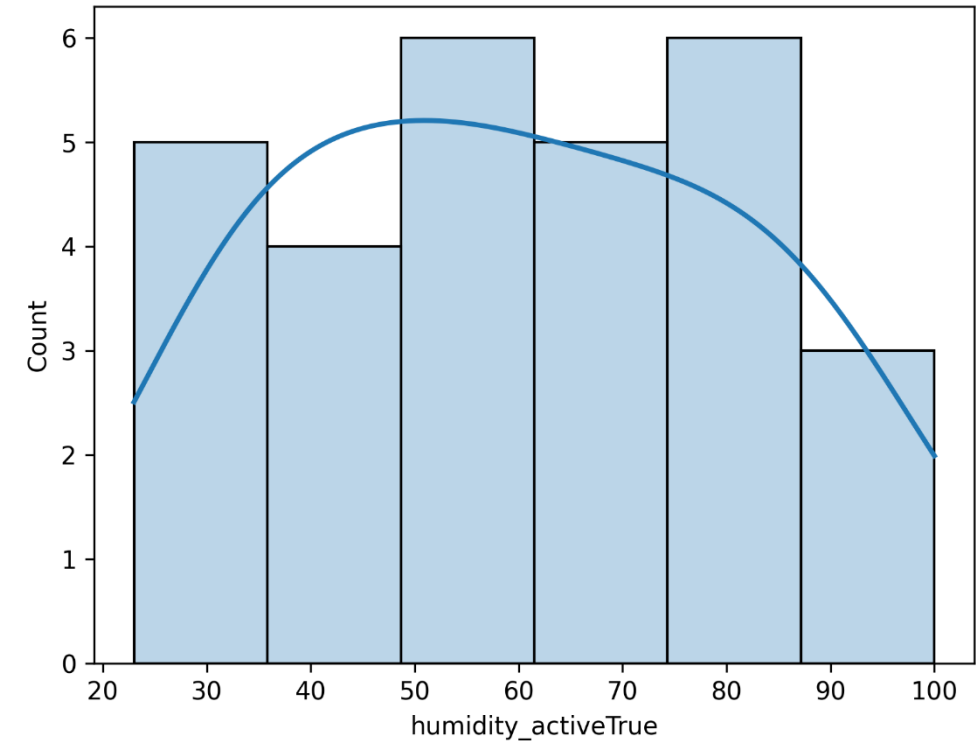


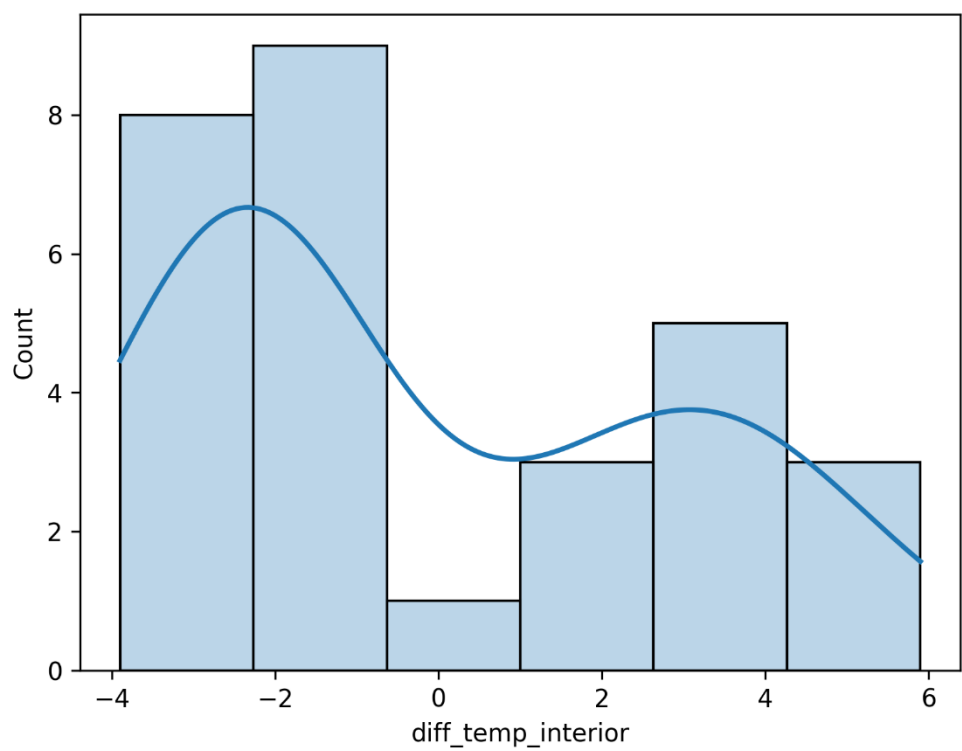
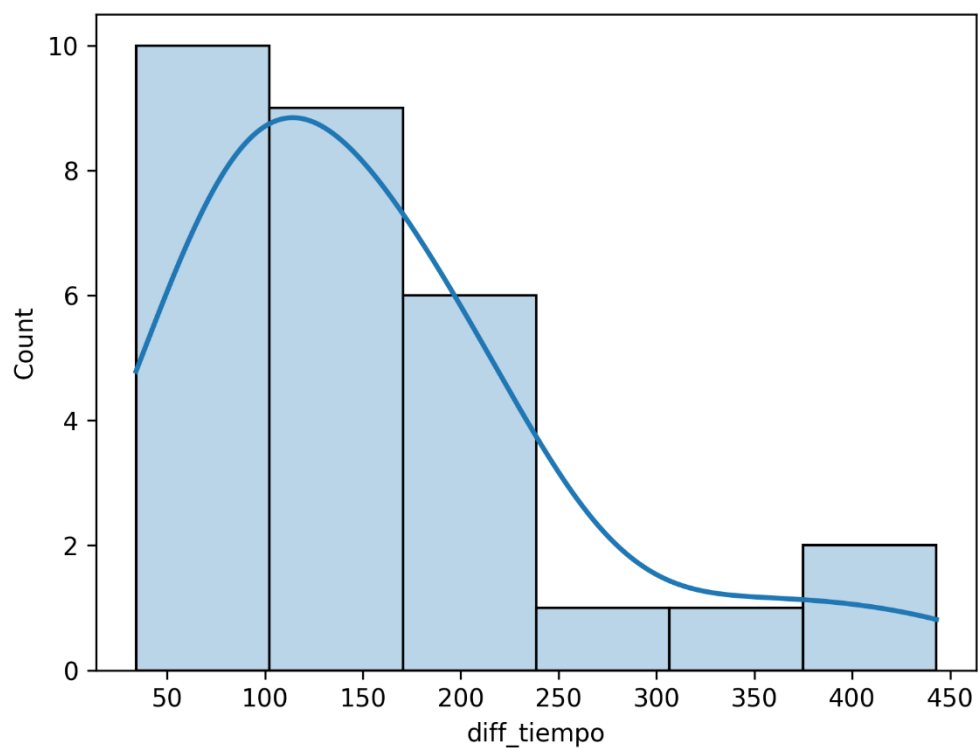


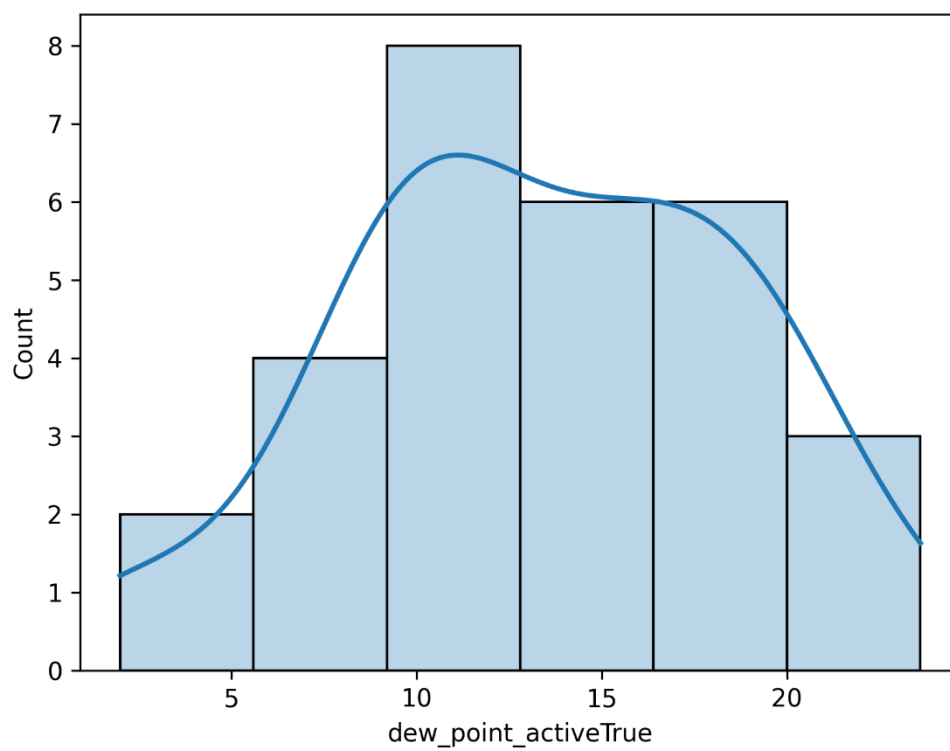
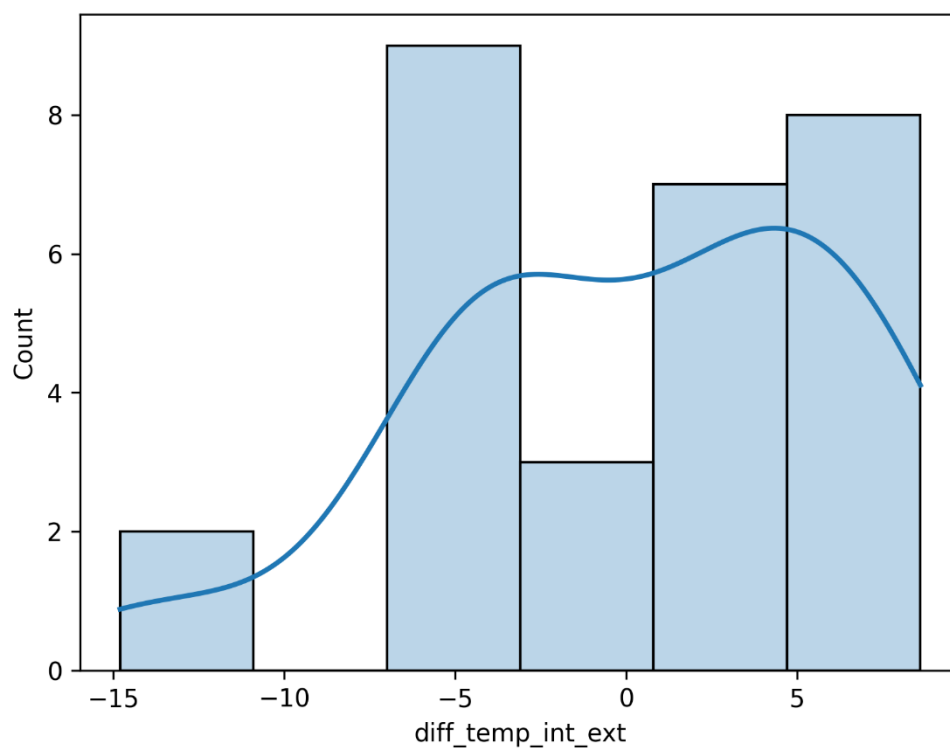


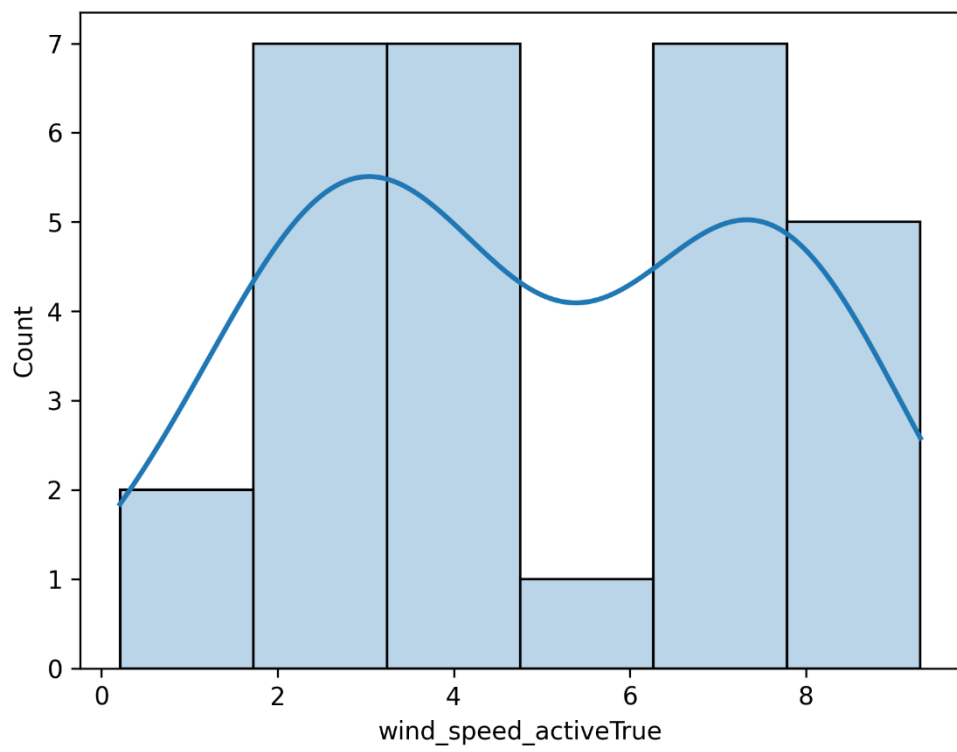
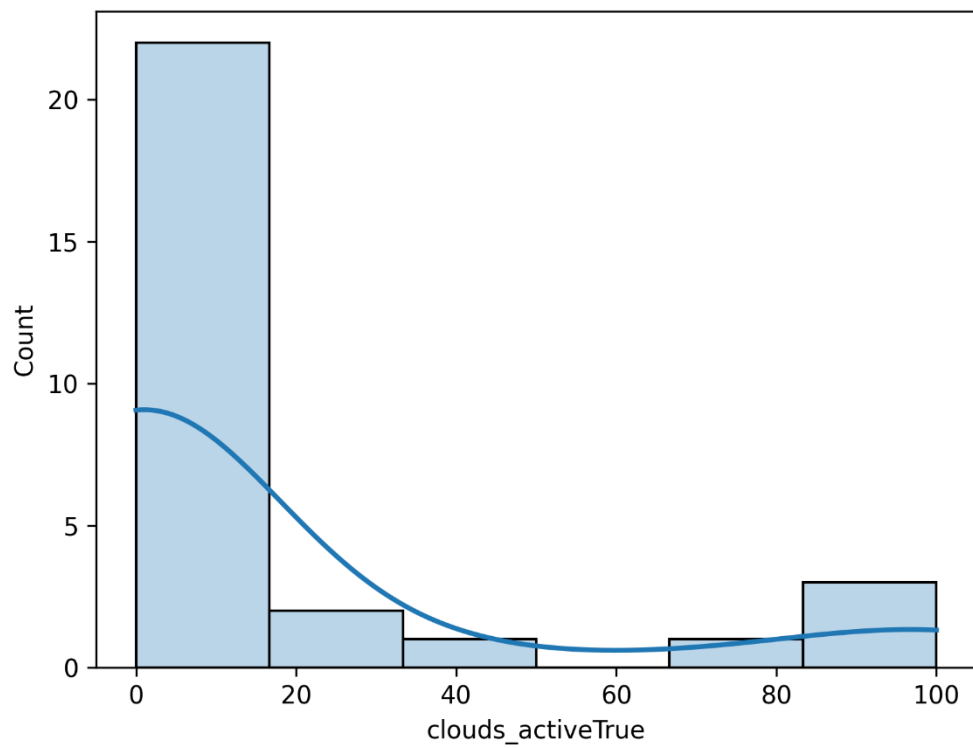


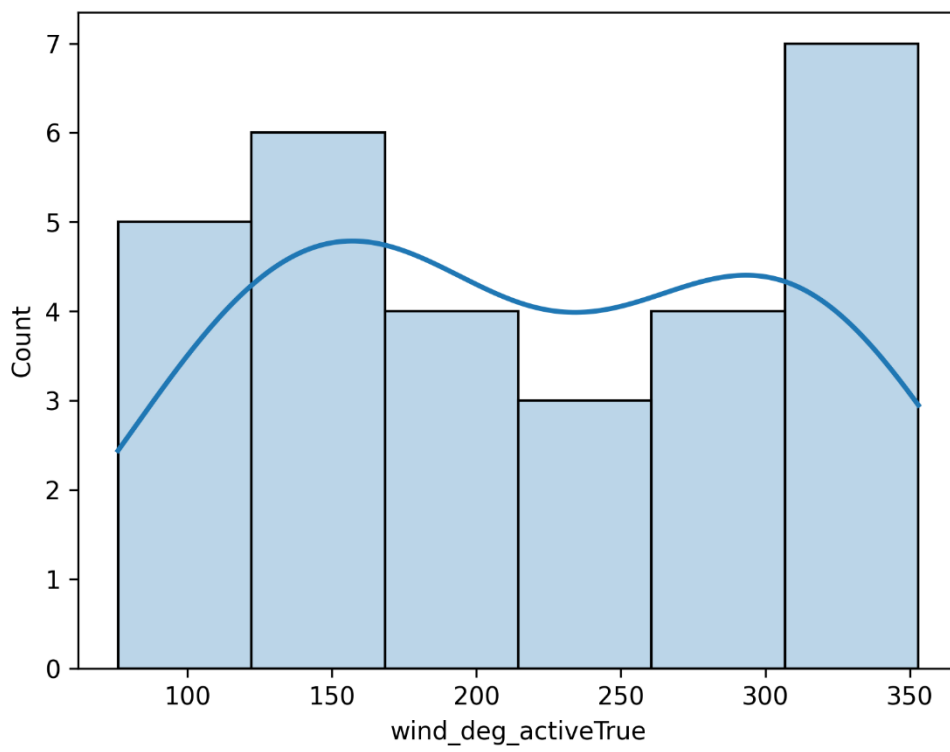
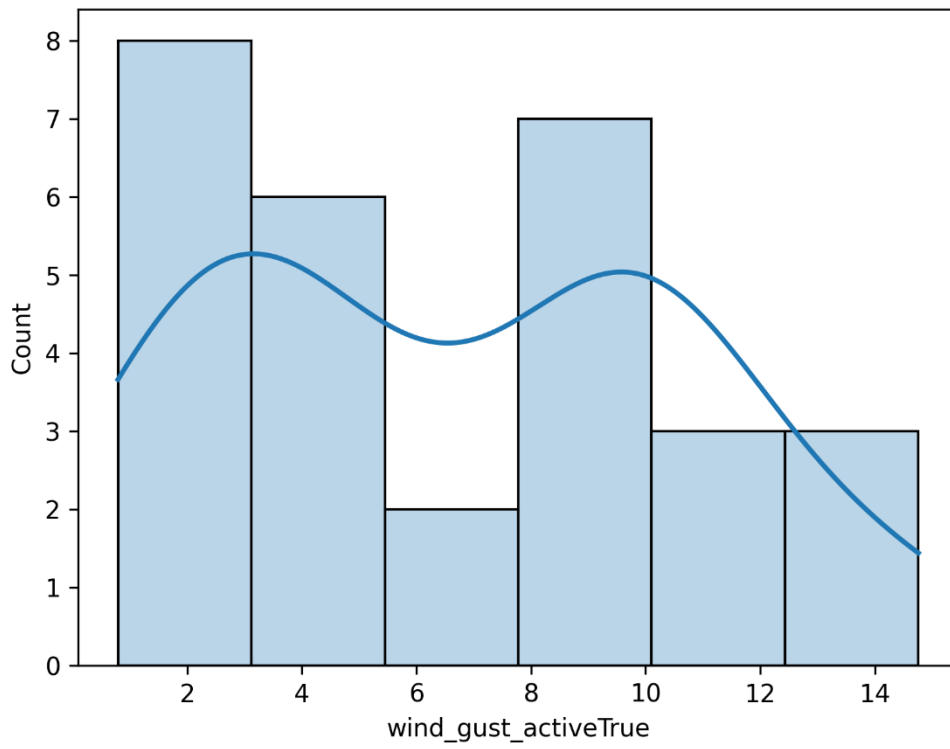
Anexo L. Imágenes de la carpeta A2i_histogramas – python_tfg_conjunto

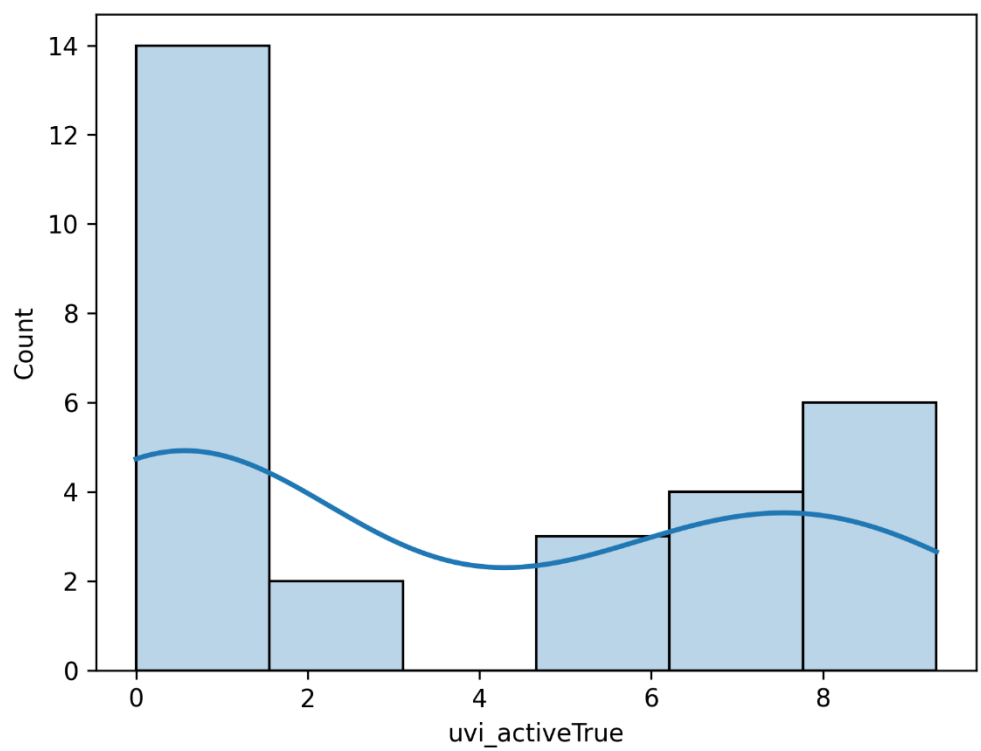
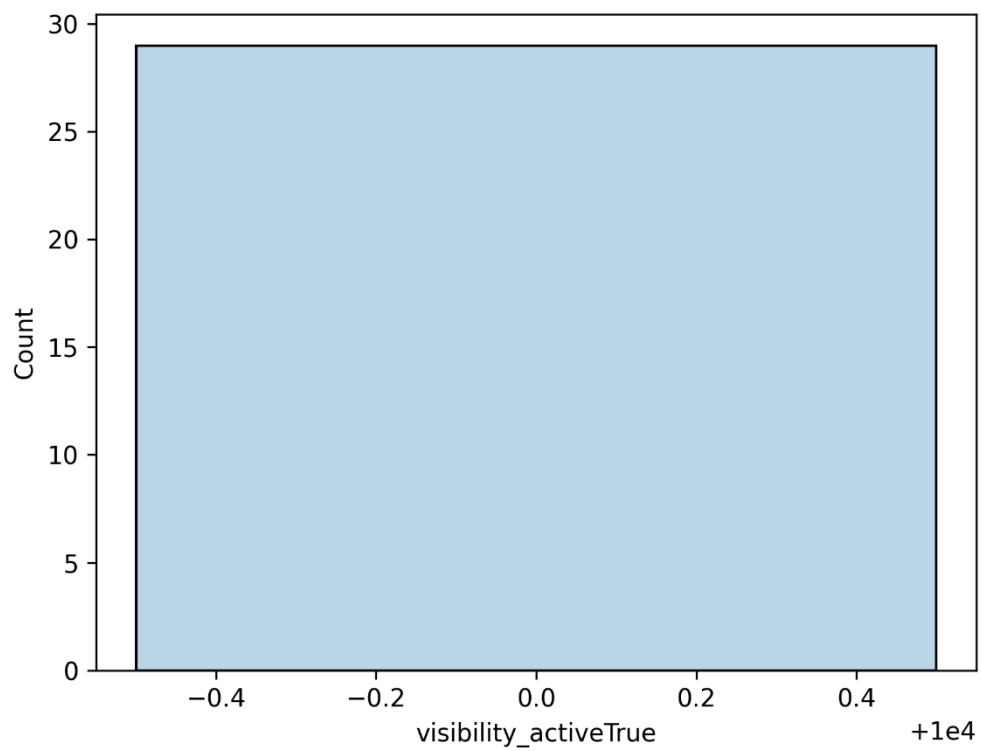


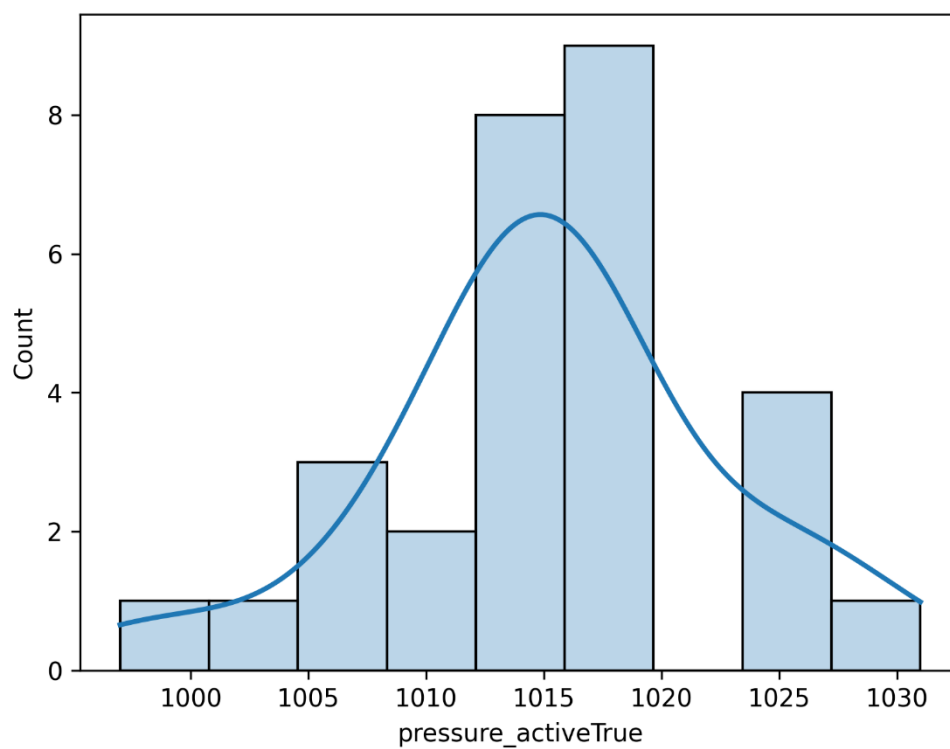
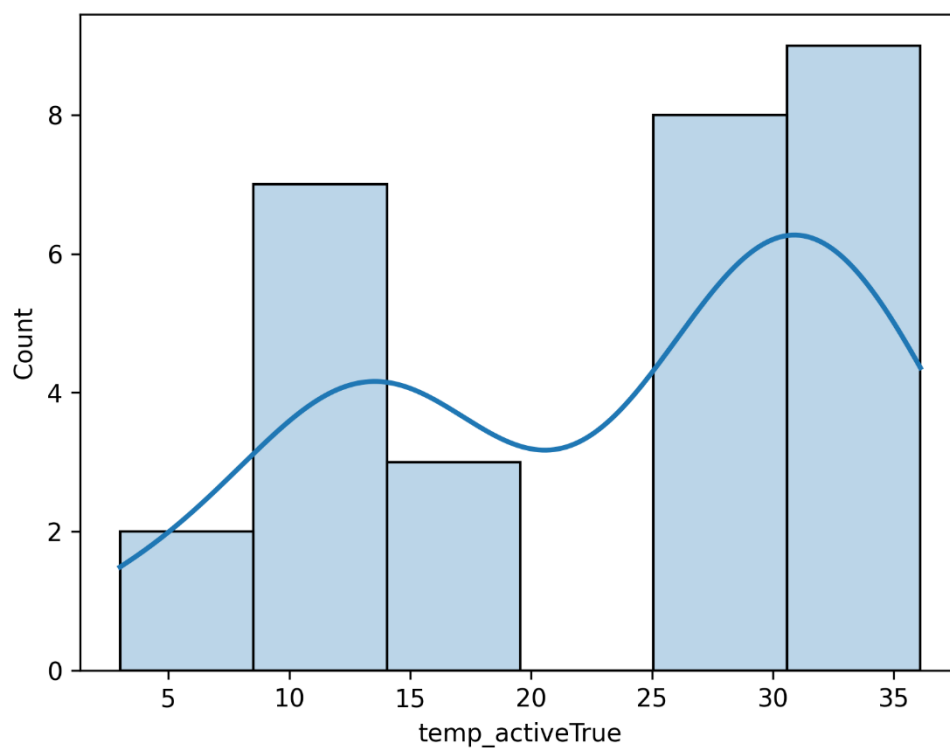


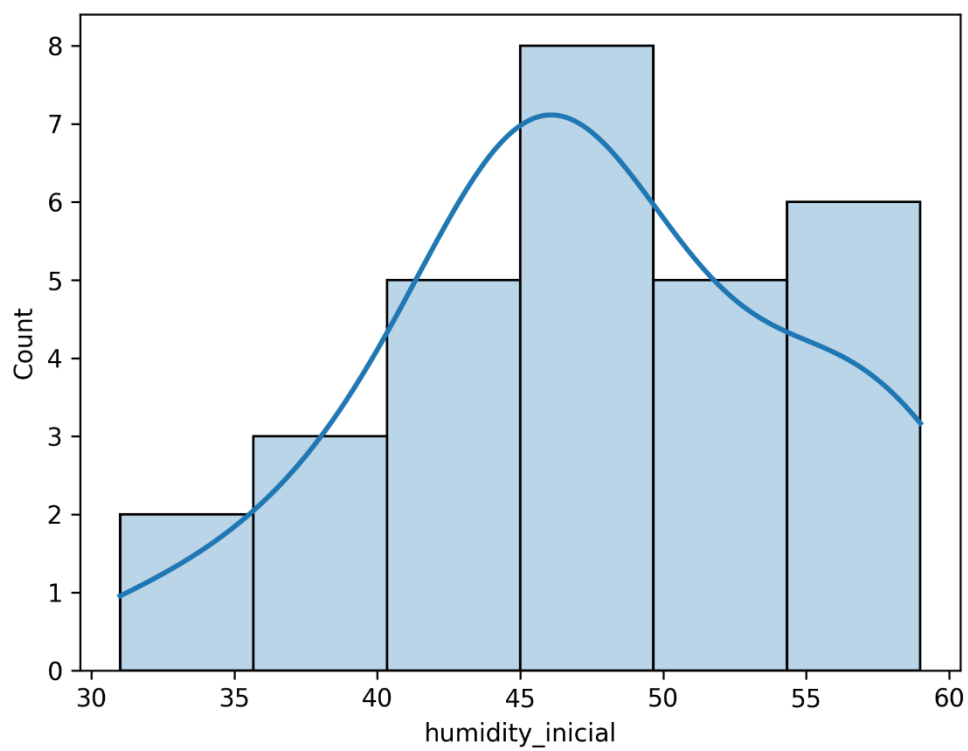
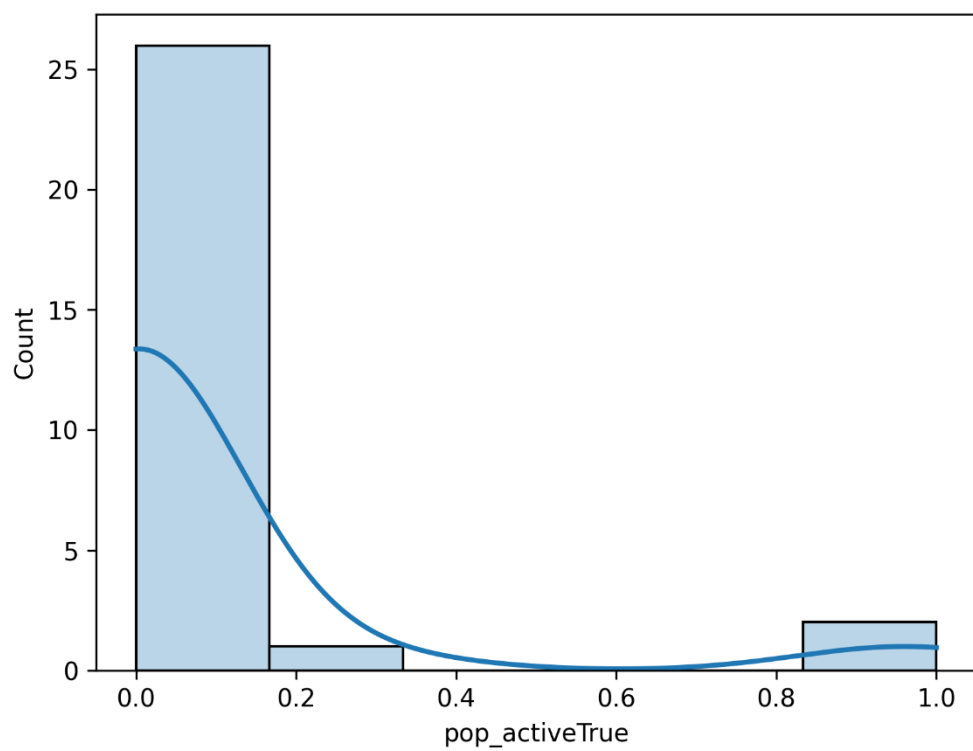












Anexo M. Código fn_normalizacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Función auxiliar para realizar la normalización

# Se importan las bibliotecas necesarias
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

# Se define la función de normalización MinMax, teniendo en cuenta el
data leakage entre entrenamiento y test
def normalize_data(data_train, data_test):

    # Seleccionar las columnas numéricas que incluyan los float y los int
    numeric_columns = data_train.select_dtypes(include=[float,
int]).columns

    # Normalizar los datos
    for column in numeric_columns:
        min_value = data_train[column].min()
        max_value = data_train[column].max()
        data_train[column] = (data_train[column] - min_value) /
(max_value - min_value)
        data_test[column] = (data_test[column] - min_value) / (max_value
- min_value)

    return data_train, data_test
```

Anexo N. Código A3_Normalizacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Normalización y split en entrenamiento y test en un
20%

# Se importan las bibliotecas necesarias
from sklearn.model_selection import train_test_split
from fn_normalizacion import normalize_data
import pandas as pd

# Se cargan los datos del archivo dataset_clean.tsv en un DataFrame
data = pd.read_csv('dataset_clean.tsv', sep='\t')

# Definir la columna objetivo (Y)
columna_objetivo = 'diff_tiempo'

# Obtener las características (X) excluyendo la columna objetivo
columnas_interes = data.columns[data.columns != columna_objetivo]
X = data[columnas_interes]
y = data[columna_objetivo]

# Separar en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=123)

# Se aplica la normalización a las columnas numéricas en X_train y X_test
X_train_normalized, X_test_normalized = normalize_data(X_train, X_test)

# Convertir los datos normalizados en DataFrames de Pandas
X_train_normalized_df = pd.DataFrame(X_train_normalized,
columns=columnas_interes)
X_test_normalized_df = pd.DataFrame(X_test_normalized,
columns=columnas_interes)

# Guardar los DataFrames normalizados en archivos tsv
X_train_normalized_df.to_csv('BBDD/data-X_train_normalized.tsv',
sep="\t", index=False)
X_test_normalized_df.to_csv('BBDD/data-X_test_normalized.tsv', sep="\t",
index=False)

# Guardar las columnas objetivo (Y) en archivos tsv
y_train_df = pd.DataFrame(y_train, columns=[columna_objetivo])
y_test_df = pd.DataFrame(y_test, columns=[columna_objetivo])
y_train_df.to_csv('BBDD/data-y_train.tsv', sep="\t", index=False)
y_test_df.to_csv('BBDD/data-y_test.tsv', sep="\t", index=False)
```

Anexo Ñ. Código A4_Correlacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Análisis de correlación entre las variables X.
##### Representación de la matriz de correlación con "mapa
de calor"
##### Creación del archivo .TXT con nombre
"A4_Correlacion.txt"

"""
Cuánto más cercano a 0 más débil es la correlación entre las variables.

Determinar qué variables tienen un R cercano 1 y pueden ser cofactores.

De este análisis vemos que feels_like_activeTrue tiene una correlación de
0.99 y se presenta
potencialmente como variable a eliminar de nuestro conjunto de datos. Por
lo tanto, vamos a
desarrollar de forma paralela unos modelos con esta variable y otros sin
ella.

temp_activeTrue feels_like_activeTrue    0.9976763232011605

"""

# Importar bibliotecas a usar
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Cargar el dataset en un DataFrame
df = pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg\\BBDD\\data-
X_train_normalized.tsv', delimiter='\\t')

# Obtener todas las variables del DataFrame
variables = df.columns

# Obtener las columnas numéricas correspondientes a las variables
seleccionadas
numeric_columns =
df[variables].select_dtypes(include=[np.number]).columns

# Crear un nuevo DataFrame con las variables seleccionadas
df_selected = df[variables]
```

```

# Calcular la matriz de correlación
correlation_matrix = df_selected[numeric_columns].corr()

# Crear un nuevo DataFrame con los pares de variables y coeficientes de correlación
correlation_df = pd.DataFrame(columns=['Variable 1', 'Variable 2', 'Correlation'])

# Recorrer la matriz de correlación y guardar los pares de variables y coeficientes
for i in range(len(correlation_matrix.columns)):
    for j in range(i+1, len(correlation_matrix.columns)):
        var1 = correlation_matrix.columns[i]
        var2 = correlation_matrix.columns[j]
        correlation = correlation_matrix.iloc[i, j]
        correlation_df.loc[len(correlation_df)] = [var1, var2, correlation]

# Imprimir el DataFrame con los resultados
print(correlation_df)

# Guardar el DataFrame en un archivo de texto
correlation_df[['Variable 1', 'Variable 2', 'Correlation']].to_csv('A4_Correlacion_X.txt', sep='\t', index=False)

# Crear un mapa de calor de las correlaciones
plt.figure(figsize=(12, 10))
heatmap = sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f", square=True)

# Ajustar el tamaño de las etiquetas de los ejes x
heatmap.set_xticklabels(correlation_matrix.columns, rotation=45,
ha='right', fontsize=8)

# Ajustar el tamaño de las etiquetas de los ejes y
heatmap.set_yticklabels(correlation_matrix.columns, rotation=0,
ha='right', fontsize=8)

# Guardar el gráfico como una imagen PNG
plt.savefig('A4_Mapa_de_calor.png')

# Mostrar el mapa de calor
plt.show()

```

Anexo O. Excel A4_Correlacion_X.xlsx – python_tfg

Variable 1	Variable 2	Correlation
temp_activeTrue	feels_like_activeTrue	0,99767632320116
diff_temp_int_ext	temp_activeTrue	0,95879043959531
diff_temp_int_ext	feels_like_activeTrue	0,95575128522750
feels_like_activeTrue	dew_point_activeTrue	0,95243897458360
temp_activeTrue	dew_point_activeTrue	0,94877155611823
diff_temp_int_ext	dew_point_activeTrue	0,94664811455441
wind_speed_activeTrue	wind_gust_activeTrue	0,94192859790235
diff_temp_interior	humidity_activeTrue	0,69696842025514
humidity_inicial	dew_point_activeTrue	0,67473137259990
uvi_activeTrue	clouds_activeTrue	0,64762131457277
wind_speed_activeTrue	pop_activeTrue	0,63096281226012
dew_point_activeTrue	wind_speed_activeTrue	0,61422089251265
dew_point_activeTrue	wind_gust_activeTrue	0,60950143658916
temp_activeTrue	wind_speed_activeTrue	0,60029195871026
temp_activeTrue	wind_gust_activeTrue	0,58578307087617
humidity_inicial	feels_like_activeTrue	0,57689417178203
temp_activeTrue	humidity_inicial	0,57288291106874
feels_like_activeTrue	wind_speed_activeTrue	0,57054810237960
diff_temp_int_ext	humidity_inicial	0,56624908044412
humidity_inicial	wind_gust_activeTrue	0,55747859845340
feels_like_activeTrue	wind_gust_activeTrue	0,55685629577324
diff_temp_interior	clouds_activeTrue	0,54717406085341
diff_temp_int_ext	wind_gust_activeTrue	0,54648192291773
diff_temp_int_ext	wind_speed_activeTrue	0,52088890776201
wind_gust_activeTrue	pop_activeTrue	0,49094846228604
clouds_activeTrue	wind_deg_activeTrue	0,47124289435661
humidity_inicial	wind_speed_activeTrue	0,45690927109271
humidity_activeTrue	wind_deg_activeTrue	0,44000985691088
temp_activeTrue	pop_activeTrue	0,40670806388072
feels_like_activeTrue	pop_activeTrue	0,38644702701848
diff_temp_int_ext	pop_activeTrue	0,37429289813460

humidity_activeTrue	clouds_activeTrue	0,32183150461976
pressure_activeTrue	humidity_activeTrue	0,31959066686936
diff_temp_int_ext	uvi_activeTrue	0,31870899631680
dew_point_activeTrue	uvi_activeTrue	0,29363493444225
feels_like_activeTrue	uvi_activeTrue	0,29132577347215
dew_point_activeTrue	pop_activeTrue	0,28895368196461
temp_activeTrue	uvi_activeTrue	0,26333782567000
diff_temp_interior	wind_deg_activeTrue	0,25139429869368
pressure_activeTrue	uvi_activeTrue	0,22365236953452
dew_point_activeTrue	clouds_activeTrue	0,17083107998607
uvi_activeTrue	pop_activeTrue	0,13684154902596
diff_temp_interior	uvi_activeTrue	0,12142939414216
diff_temp_int_ext	clouds_activeTrue	0,08985655516790
diff_temp_interior	pressure_activeTrue	0,05795792737568
uvi_activeTrue	wind_deg_activeTrue	0,05384684109561
feels_like_activeTrue	clouds_activeTrue	0,00010852355621
humidity_inicial	wind_deg_activeTrue	-0,00763496512303
temp_activeTrue	clouds_activeTrue	-0,03830947366248
humidity_inicial	pressure_activeTrue	-0,09768563107164
diff_temp_interior	wind_gust_activeTrue	-0,11531361486399
humidity_inicial	clouds_activeTrue	-0,12895897573331
diff_temp_interior	pop_activeTrue	-0,13621795376034
pressure_activeTrue	wind_deg_activeTrue	-0,13841147385744
pressure_activeTrue	clouds_activeTrue	-0,14023875640537
clouds_activeTrue	pop_activeTrue	-0,14100173341527
uvi_activeTrue	wind_speed_activeTrue	-0,14439536483404
humidity_inicial	pop_activeTrue	-0,15064431576260
diff_temp_interior	wind_speed_activeTrue	-0,17627418821194
humidity_activeTrue	uvi_activeTrue	-0,18158672816722
dew_point_activeTrue	wind_deg_activeTrue	-0,20613775580728
diff_temp_interior	diff_temp_int_ext	-0,22518390519151
humidity_inicial	uvi_activeTrue	-0,23038129113291
diff_temp_interior	dew_point_activeTrue	-0,23414633795574

pressure_activeTrue	wind_gust_activeTrue	-0,26139696540348
diff_temp_interior	humidity_inicial	-0,27773676069844
wind_deg_activeTrue	pop_activeTrue	-0,27994825495134
uvi_activeTrue	wind_gust_activeTrue	-0,28199479980222
diff_temp_int_ext	wind_deg_activeTrue	-0,28735875457455
feels_like_activeTrue	wind_deg_activeTrue	-0,28883683867636
pressure_activeTrue	wind_speed_activeTrue	-0,29345833001424
clouds_activeTrue	wind_speed_activeTrue	-0,30068512986473
wind_deg_activeTrue	wind_gust_activeTrue	-0,32146040611338
humidity_inicial	humidity_activeTrue	-0,32651073507063
temp_activeTrue	wind_deg_activeTrue	-0,34598575206175
pressure_activeTrue	dew_point_activeTrue	-0,35439839760094
temp_activeTrue	pressure_activeTrue	-0,36921046552720
feels_like_activeTrue	pressure_activeTrue	-0,37150116817660
diff_temp_int_ext	pressure_activeTrue	-0,37378074605266
clouds_activeTrue	wind_gust_activeTrue	-0,39312598395343
humidity_activeTrue	wind_gust_activeTrue	-0,41828061412844
wind_speed_activeTrue	wind_deg_activeTrue	-0,42092992041896
humidity_activeTrue	wind_speed_activeTrue	-0,43842769377014
diff_temp_interior	temp_activeTrue	-0,45012411504030
diff_temp_interior	feels_like_activeTrue	-0,45598750043884
humidity_activeTrue	pop_activeTrue	-0,48024017654848
pressure_activeTrue	pop_activeTrue	-0,53290416488727
humidity_activeTrue	dew_point_activeTrue	-0,69611194410317
diff_temp_int_ext	humidity_activeTrue	-0,79512719740015
feels_like_activeTrue	humidity_activeTrue	-0,87954179412369
temp_activeTrue	humidity_activeTrue	-0,88654681496990

Anexo P. Excel A4_Correlacion_X.xlsx – python_tfg_verano

Variable 1	Variable 2	Correlation
diff_temp_int_ext	temp_activeTrue	0,98515704987531
wind_speed_activeTrue	wind_gust_activeTrue	0,97867560779028
temp_activeTrue	feels_like_activeTrue	0,89391549897352
diff_temp_int_ext	feels_like_activeTrue	0,87975280083547
diff_temp_int_ext	uvi_activeTrue	0,77939985583159
humidity_inicial	dew_point_activeTrue	0,73707065563648
humidity_activeTrue	dew_point_activeTrue	0,73282364790752
temp_activeTrue	uvi_activeTrue	0,73155628477676
feels_like_activeTrue	dew_point_activeTrue	0,57195180817163
feels_like_activeTrue	uvi_activeTrue	0,54908551077454
humidity_inicial	feels_like_activeTrue	0,53517708684660
pressure_activeTrue	humidity_activeTrue	0,52620802217159
diff_temp_interior	humidity_activeTrue	0,40436964343933
diff_temp_int_ext	wind_speed_activeTrue	0,39242710656020
humidity_inicial	humidity_activeTrue	0,38579958958156
temp_activeTrue	humidity_inicial	0,34864874767765
temp_activeTrue	wind_speed_activeTrue	0,33057879172821
humidity_activeTrue	wind_deg_activeTrue	0,31137403475311
pressure_activeTrue	dew_point_activeTrue	0,30313030956857
diff_temp_int_ext	wind_gust_activeTrue	0,29902087143142
diff_temp_int_ext	humidity_inicial	0,29827472567862
temp_activeTrue	wind_gust_activeTrue	0,23516141286826
feels_like_activeTrue	wind_speed_activeTrue	0,23211235752609
diff_temp_interior	wind_gust_activeTrue	0,23052865729713
humidity_inicial	wind_gust_activeTrue	0,21670270912005
temp_activeTrue	dew_point_activeTrue	0,18719343642292
uvi_activeTrue	wind_speed_activeTrue	0,18363671882705
humidity_inicial	wind_speed_activeTrue	0,18059100194479
humidity_inicial	pressure_activeTrue	0,17708714758995
feels_like_activeTrue	wind_gust_activeTrue	0,17158303472012
diff_temp_int_ext	dew_point_activeTrue	0,15010112174264

diff_temp_interior	wind_deg_activeTrue	0,13570919291793
humidity_inicial	uvi_activeTrue	0,10120868330491
uvi_activeTrue	wind_gust_activeTrue	0,10105464684617
diff_temp_interior	wind_speed_activeTrue	0,09357000538037
humidity_inicial	wind_deg_activeTrue	0,09031309491537
pressure_activeTrue	uvi_activeTrue	0,07200121195229
wind_deg_activeTrue	wind_gust_activeTrue	0,06364694141548
wind_speed_activeTrue	wind_deg_activeTrue	0,04618600105612
diff_temp_interior	dew_point_activeTrue	0,03546509203383
dew_point_activeTrue	wind_gust_activeTrue	-0,03264098007055
pressure_activeTrue	wind_deg_activeTrue	-0,10091297380994
dew_point_activeTrue	wind_speed_activeTrue	-0,10294006571022
diff_temp_interior	pressure_activeTrue	-0,11444871445910
feels_like_activeTrue	pressure_activeTrue	-0,11649548786330
feels_like_activeTrue	humidity_activeTrue	-0,11814516957126
dew_point_activeTrue	wind_deg_activeTrue	-0,12273813867768
dew_point_activeTrue	uvi_activeTrue	-0,16491949280059
diff_temp_interior	humidity_inicial	-0,21851958814996
humidity_activeTrue	wind_gust_activeTrue	-0,26853803934278
temp_activeTrue	pressure_activeTrue	-0,32093175400662
diff_temp_int_ext	pressure_activeTrue	-0,32747328916484
humidity_activeTrue	wind_speed_activeTrue	-0,39163799432318
temp_activeTrue	humidity_activeTrue	-0,52259108031343
diff_temp_interior	feels_like_activeTrue	-0,54315107625525
feels_like_activeTrue	wind_deg_activeTrue	-0,54991672384898
diff_temp_int_ext	humidity_activeTrue	-0,55767205228618
diff_temp_interior	diff_temp_int_ext	-0,57523462204421
temp_activeTrue	wind_deg_activeTrue	-0,59360646900169
diff_temp_interior	temp_activeTrue	-0,62230019229603
pressure_activeTrue	wind_gust_activeTrue	-0,63042979944244
diff_temp_int_ext	wind_deg_activeTrue	-0,64583556054576
humidity_activeTrue	uvi_activeTrue	-0,64895377853596
diff_temp_interior	uvi_activeTrue	-0,64958317747896

uvi_activeTrue	wind_deg_activeTrue	-0,67485046037277
pressure_activeTrue	wind_speed_activeTrue	-0,70230809981817

Anexo Q. Excel A4_Correlacion_X.xlsx – python_tfg_conjunto

Variable 1	Variable 2	Correlation
temp_activeTrue	feels_like_activeTrue	0,99355631063017
diff_temp_int_ext	temp_activeTrue	0,97323679997758
diff_temp_int_ext	feels_like_activeTrue	0,97302246005712
wind_speed_activeTrue	wind_gust_activeTrue	0,95147904017859
feels_like_activeTrue	dew_point_activeTrue	0,94484905969551
temp_activeTrue	dew_point_activeTrue	0,91838918656962
diff_temp_int_ext	dew_point_activeTrue	0,90262055342811
diff_temp_interior	humidity_activeTrue	0,87857691646468
feels_like_activeTrue	uvi_activeTrue	0,80487880471360
temp_activeTrue	uvi_activeTrue	0,80428122410819
diff_temp_int_ext	uvi_activeTrue	0,79250480122932
dew_point_activeTrue	uvi_activeTrue	0,69172016144428
diff_temp_interior	clouds_activeTrue	0,55800764884301
humidity_activeTrue	clouds_activeTrue	0,48951389519979
wind_speed_activeTrue	pop_activeTrue	0,46409203745346
wind_gust_activeTrue	pop_activeTrue	0,45099016336955
humidity_inicial	humidity_activeTrue	0,43068762449519
clouds_activeTrue	pop_activeTrue	0,39518776175329
humidity_inicial	wind_gust_activeTrue	0,31049745409821
diff_temp_interior	pop_activeTrue	0,29846357561583
diff_temp_interior	humidity_inicial	0,27956430544273
diff_temp_interior	wind_gust_activeTrue	0,24973178370398
pressure_activeTrue	humidity_activeTrue	0,23440631333150
humidity_activeTrue	pop_activeTrue	0,22159440362336
humidity_inicial	wind_speed_activeTrue	0,17994520333072
diff_temp_interior	pressure_activeTrue	0,16859440666194
diff_temp_interior	wind_speed_activeTrue	0,13200492466955
diff_temp_int_ext	wind_speed_activeTrue	0,12157679509237
humidity_activeTrue	wind_gust_activeTrue	0,09031010055624
humidity_activeTrue	wind_deg_activeTrue	0,08238137043933
humidity_inicial	clouds_activeTrue	0,08015829935849

humidity_inicial	pop_activeTrue	0,06750574236429
clouds_activeTrue	wind_deg_activeTrue	0,06473314710869
humidity_inicial	pressure_activeTrue	0,06218977988762
pressure_activeTrue	wind_deg_activeTrue	0,04969463402256
temp_activeTrue	wind_speed_activeTrue	0,03134406466738
humidity_inicial	wind_deg_activeTrue	0,02744449499452
clouds_activeTrue	wind_gust_activeTrue	0,02489455915638
diff_temp_int_ext	wind_gust_activeTrue	0,02277040766080
feels_like_activeTrue	wind_speed_activeTrue	0,02108781203184
clouds_activeTrue	wind_speed_activeTrue	-0,00741691035325
diff_temp_interior	wind_deg_activeTrue	-0,01041893005270
dew_point_activeTrue	wind_speed_activeTrue	-0,01049879637747
dew_point_activeTrue	wind_deg_activeTrue	-0,02069433873940
humidity_inicial	dew_point_activeTrue	-0,03219690956194
temp_activeTrue	wind_deg_activeTrue	-0,04268366344789
feels_like_activeTrue	wind_deg_activeTrue	-0,05323503386154
pressure_activeTrue	uvi_activeTrue	-0,05737931050824
humidity_activeTrue	wind_speed_activeTrue	-0,06272176311236
dew_point_activeTrue	wind_gust_activeTrue	-0,07352751804096
temp_activeTrue	wind_gust_activeTrue	-0,08411843702285
feels_like_activeTrue	wind_gust_activeTrue	-0,08845883300402
diff_temp_int_ext	wind_deg_activeTrue	-0,10131517375797
uvi_activeTrue	wind_deg_activeTrue	-0,11025322256003
wind_deg_activeTrue	wind_gust_activeTrue	-0,14328779308472
uvi_activeTrue	wind_speed_activeTrue	-0,14908992476732
diff_temp_int_ext	pop_activeTrue	-0,17706853534639
diff_temp_int_ext	humidity_inicial	-0,18062111913652
dew_point_activeTrue	pop_activeTrue	-0,18801670682586
wind_speed_activeTrue	wind_deg_activeTrue	-0,21383540602303
feels_like_activeTrue	pop_activeTrue	-0,23236811998210
humidity_inicial	feels_like_activeTrue	-0,23512771909379
temp_activeTrue	pop_activeTrue	-0,24055508868283
pressure_activeTrue	clouds_activeTrue	-0,24596820694902

uvi_activeTrue	wind_gust_activeTrue	-0,27451198820586
feels_like_activeTrue	pressure_activeTrue	-0,27690707298982
temp_activeTrue	pressure_activeTrue	-0,27852778189498
temp_activeTrue	humidity_inicial	-0,27900170208293
pressure_activeTrue	dew_point_activeTrue	-0,28063549942565
uvi_activeTrue	clouds_activeTrue	-0,28139391444381
humidity_inicial	uvi_activeTrue	-0,28706421754402
uvi_activeTrue	pop_activeTrue	-0,29486936395122
dew_point_activeTrue	clouds_activeTrue	-0,31453534698538
diff_temp_int_ext	pressure_activeTrue	-0,32226287019815
diff_temp_int_ext	clouds_activeTrue	-0,35050462362433
pressure_activeTrue	wind_gust_activeTrue	-0,37673078976178
pressure_activeTrue	wind_speed_activeTrue	-0,39168955205886
wind_deg_activeTrue	pop_activeTrue	-0,39365822001053
feels_like_activeTrue	clouds_activeTrue	-0,40990612739377
temp_activeTrue	clouds_activeTrue	-0,42867812540631
pressure_activeTrue	pop_activeTrue	-0,57076731344489
humidity_activeTrue	dew_point_activeTrue	-0,71715690948221
humidity_activeTrue	uvi_activeTrue	-0,76968862357186
diff_temp_interior	uvi_activeTrue	-0,77563476108722
diff_temp_interior	dew_point_activeTrue	-0,82689555280110
diff_temp_interior	diff_temp_int_ext	-0,82742321354105
feels_like_activeTrue	humidity_activeTrue	-0,89512112214683
diff_temp_int_ext	humidity_activeTrue	-0,89747563739804
diff_temp_interior	feels_like_activeTrue	-0,91280620262089
diff_temp_interior	temp_activeTrue	-0,91930537416081
temp_activeTrue	humidity_activeTrue	-0,92844288771750

Anexo R. C0_Normalizacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Hacemos lo mismo que en A3_Normalizacion.py pero
eliminando feels_like_activeTrue
##### al tener una correlación muy cercana a 1 con
temp_activeTrue.
##### Normalización y split en entrenamiento y test en un
20%

# Se importan las bibliotecas necesarias
from sklearn.model_selection import train_test_split
from fn_normalizacion import normalize_data
import pandas as pd

# Se cargan los datos del archivo dataset_clean.tsv en un DataFrame
data = pd.read_csv('dataset_clean.tsv', sep='\t')

# Definir la columna objetivo (Y)
columna_objetivo = 'diff_tiempo'

# Obtener las características (X) excluyendo la columna objetivo
columnas_interes = data.columns[(data.columns != columna_objetivo) &
(data.columns != 'feels_like_activeTrue')]
X = data[columnas_interes]
y = data[columna_objetivo]

# Separar en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=123)

# Se aplica la normalización a las columnas numéricas en X_train y X_test
X_train_normalized, X_test_normalized = normalize_data(X_train, X_test)

# Convertir los datos normalizados en DataFrames de Pandas
X_train_normalized_df = pd.DataFrame(X_train_normalized,
columns=columnas_interes)
X_test_normalized_df = pd.DataFrame(X_test_normalized,
columns=columnas_interes)

# Guardar los DataFrames normalizados en archivos tsv
X_train_normalized_df.to_csv('BBDD_corr_filter/data-
X_train_normalized_corr_filter.tsv', sep="\t", index=False)
X_test_normalized_df.to_csv('BBDD_corr_filter/data-
X_test_normalized_corr_filter.tsv', sep="\t", index=False)
```

```
# Guardar las columnas objetivo (Y) en archivos tsv
y_train_df = pd.DataFrame(y_train, columns=[columna_objetivo])
y_test_df = pd.DataFrame(y_test, columns=[columna_objetivo])
y_train_df.to_csv('BBDD_corr_filter/data-y_train_corr_filter.tsv',
sep="\t", index=False)
y_test_df.to_csv('BBDD_corr_filter/data-y_test_corr_filter.tsv',
sep="\t", index=False)
```

Anexo S. Código B1_Preparacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Análisis de Correlación para Evaluación las variables
haciendo solo uso de los datos entrenados
##### y normalizados. Se seleccionan aquellas cuya
correlación tiene un valor absoluto mayor a 0.25.
##### Se muestra en un gráfico
##### Se crea un archivo TSV con los nombres de las
variables seleccionadas

# Se importan las bibliotecas necesarias
import pandas as pd
import matplotlib.pyplot as plt
import os

# Leen los archivos (separados por tabulaciones '\t') y los almacenan en
DataFrames
X_train = pd.read_csv('BBDD/data-X_train_normalized.tsv', sep='\t')
Y_train = pd.read_csv('BBDD/data-y_train.tsv', sep='\t')

# Se calcula la correlación entre la columna X_train y "diff_tiempo" de
Y_train y se imprimen
correlation = X_train.corrwith(Y_train['diff_tiempo'])
print("Los valores de todas correlaciones son:", correlation)

# Se filtran los valores de correlación mayores a 0.25 en valor absoluto
y se imprimen
correlation_filtrada = correlation[correlation.abs() > 0.25]
print("Nos quedamos con las siguientes variables cuya correlación es
mayor a |0.25|:", correlation_filtrada)

# Para mostrar en un gráfico de barras las correlaciones en función de
"diff_tiempo" de Y_train
plt.figure(figsize=(10, 6))
correlation.plot(kind='bar')
plt.title('Correlación entre las variables X e Y (diff_tiempo)')
plt.xlabel('Variables X')
plt.ylabel('Correlación')
plt.xticks(rotation=70)
plt.subplots_adjust(bottom=0.31) # Ajustar los márgenes inferiores del
gráfico
plt.savefig('B1_Grafico_barras.png') # Guardar el gráfico como una
imagen PNG
plt.show()
```

```
# Crear archivo TSV con los nombres de las variables (filtradas) cuya
correlación es mayor a 0.25
variables_filtradas = correlacion_filtrada.index.tolist()
df_variabels_filtradas = pd.DataFrame(variables_filtradas,
columns=['Variables'])
df_variabels_filtradas.to_csv('B1_Variables.tsv', sep='\t', index=False)
```

Anexo T. B1_Variables.tsv– python_tfg

≡ B1_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 feels_like_activeTrue
7 humidity_activeTrue
8 dew_point_activeTrue
9 pop_activeTrue
```

Anexo U. B1_Variables.tsv– python_tfg_verano

≡ B1_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 pressure_activeTrue
7 humidity_activeTrue
8 dew_point_activeTrue
9 uvi_activeTrue
10 wind_gust_activeTrue
11 clouds_activeTrue
```

Anexo W. B1_Variables.tsv– python_tfg_conjunto

≡ B1_Variables.tsv

```
1 Variables
2 dew_point_activeTrue
3 uvi_activeTrue
4 wind_deg_activeTrue
5 wind_gust_activeTrue
```

Anexo X. Código B2_Preparacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Análisis de Correlación para Evaluación de todas las
variables sin separar en train y test3
#####          Nos quedamos con todas las variables filtradas
#####          Guardamos dichas variables en un archivo TSV

# Se importan las bibliotecas necesarias
import pandas as pd
import matplotlib.pyplot as plt

# Leen los archivos (separados por tabulaciones '\t') y los almacenan en
DataFrames
X_train = pd.read_csv('BBDD/data-X_train_normalized.tsv', sep='\t')
Y_train = pd.read_csv('BBDD/data-y_train.tsv', sep='\t')

# Se calcula la correlacion entre la columna X_train y "diff_tiempo" de
Y_train y se imprimen
correlation = X_train.corrwith(Y_train['diff_tiempo'])
print("Los valores de todas correlaciones son:",correlation)

# Para mostrar en un gráfico de barras las correlaciones en función de
"diff_tiempo" de Y_train
plt.figure(figsize=(10, 6))
correlation.plot(kind='bar')
plt.title('Correlación entre las variables X y Y (diff_tiempo)')
plt.xlabel('Variables X')
plt.ylabel('Correlación')
plt.xticks(rotation=70)
plt.subplots_adjust(bottom=0.31) # Ajustar los márgenes inferiores del
gráfico
plt.savefig('B2_Grafico_barras.png') # Guardar el gráfico como una
imagen PNG
plt.show()

# Crear archivo TSV con los nombres de todas las variables filtradas
variables = correlation.index.tolist()
df_variables = pd.DataFrame(variables, columns=['Variables'])
df_variables.to_csv('B2_Variables.tsv', sep='\t', index=False)
```

Anexo Y. B2_Variables.tsv– python_tfg

≡ B2_Variables.tsv

```
1  Variables
2  diff_temp_interior
3  diff_temp_int_ext
4  temp_activeTrue
5  humidity_inicial
6  feels_like_activeTrue
7  pressure_activeTrue
8  humidity_activeTrue
9  dew_point_activeTrue
10 uvi_activeTrue
11 clouds_activeTrue
12 wind_speed_activeTrue
13 wind_deg_activeTrue
14 wind_gust_activeTrue
15 pop_activeTrue
```

Anexo Z. B2_Variables.tsv– python_tfg_verano

≡ B2_Variables.tsv

```
1  Variables
2  diff_temp_interior
3  diff_temp_int_ext
4  temp_activeTrue
5  humidity_inicial
6  feels_like_activeTrue
7  pressure_activeTrue
8  humidity_activeTrue
9  dew_point_activeTrue
10 uvi_activeTrue
11 wind_speed_activeTrue
12 wind_deg_activeTrue
13 wind_gust_activeTrue
14 clouds_activeTrue
```

Anexo AA. B2_Variables.tsv- python_tfg_conjunto

≡ B2_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 feels_like_activeTrue
7 pressure_activeTrue
8 humidity_activeTrue
9 dew_point_activeTrue
10 uvi_activeTrue
11 clouds_activeTrue
12 wind_speed_activeTrue
13 wind_deg_activeTrue
14 wind_gust_activeTrue
15 pop_activeTrue
```

Anexo BB. Código B3_Preparacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: LASSO con datos de entrenamiento Least Absolute
Shrinkage and Selection Operator

# Importar las bibliotecas necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LassoCV
from sklearn.preprocessing import StandardScaler

# Paso 1: Preparación de datos
# Carga los datos de X en un DataFrame de Pandas
datos_X = pd.read_csv("BBDD/data-X_train_normalized.tsv", sep="\t")

# Carga los datos de Y en un DataFrame de Pandas
datos_y = pd.read_csv("BBDD/data-y_train.tsv", sep="\t")

# Exploración inicial de los datos
print(datos_X.head())
print(datos_X.describe())
print(datos_y.head())
print(datos_y.describe())

# Paso 2: Preprocesamiento de datos
# Elimina las columnas con valores faltantes
datos_X = datos_X.dropna()
datos_y = datos_y.dropna()

# Paso 3: Instancia un objeto de la clase LassoCV para selección
automática del parámetro de regularización
# Hacemos un CV para buscar la mejor lambda
model = LassoCV(cv=5, alphas = [2, 5])

print(datos_X)

# Ajusta el modelo utilizando los datos de entrenamiento
model.fit(datos_X, datos_y)

# Paso 4: Obtener coeficientes y características seleccionadas
# Obtén los coeficientes del modelo
coefficients = model.coef_

# Obtén las características seleccionadas
```

```

selected_features = datos_X.columns[model.coef_ != 0]

print("Coeficientes:", coefficients)
print("Características seleccionadas:", selected_features)

# Paso 5: Visualización de resultados
# Visualiza los coeficientes
plt.figure(figsize=(10, 6))
plt.bar(datos_X.columns, coefficients)
plt.xticks(rotation=70)
plt.xlabel('Características')
plt.ylabel('Coeficientes')
plt.title('Coeficientes del modelo LASSO')
plt.subplots_adjust(bottom=0.31) # Ajustar los márgenes inferiores del
gráfico
plt.savefig('B3_Grafico_barras.png') # Guardar el gráfico como una
imagen PNG
plt.show()

# Se filtran los valores de correlación
print("Nos quedamos con las siguientes variables tras aplicar LASSO:",
selected_features)

# Crear archivo TSV con los nombres de las variables (filtradas)
variables_filtradas = selected_features.tolist()
df_variables_filtradas = pd.DataFrame(variables_filtradas,
columns=['Variables'])
df_variables_filtradas.to_csv('B3_Variables.tsv', sep='\t', index=False)

```

Anexo CC. B3_Variables.tsv– python_tfg

≡ B3_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 humidity_inicial
4 feels_like_activeTrue
5 uvi_activeTrue
6 wind_deg_activeTrue
7 wind_gust_activeTrue
8 pop_activeTrue
```

Anexo DD. B3_Variables.tsv– python_tfg_verano

≡ B3_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 humidity_inicial
4 pressure_activeTrue
5 dew_point_activeTrue
6 uvi_activeTrue
7 wind_speed_activeTrue
```

Anexo EE. B3_Variables.tsv– python_tfg_conjunto

≡ B3_Variables.tsv

```
1 Variables
2 temp_activeTrue
3 dew_point_activeTrue
4 uvi_activeTrue
5 clouds_activeTrue
6 pop_activeTrue
```

Anexo FF. Código C1_Preparacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: HACIENDO USO DE BBDD corr_filter, sin la
feels_like_activeTrue
#####           Análisis de Correlación para Evaluación las variables
haciendo solo uso de los datos entrenados
#####           y normalizados. Se seleccionan aquellas cuya
correlación tiene un valor absoluto mayor a 0.25.
#####           Se muestra en un gráfico
#####           Se crea un archivo TSV con los nombres de las
variables seleccionadas

# Se importan las bibliotecas necesarias
import pandas as pd
import matplotlib.pyplot as plt

# Leen los archivos (separados por tabulaciones '\t') y los almacenan en
DataFrames
X_train = pd.read_csv('BBDD_corr_filter/data-
X_train_normalized_corr_filter.tsv', sep='\t')
Y_train = pd.read_csv('BBDD_corr_filter/data-y_train_corr_filter.tsv',
sep='\t')

# Se calcula la correlación entre la columna X_train y "diff_tiempo" de
Y_train y se imprimen
correlation = X_train.corrwith(Y_train['diff_tiempo'])
print("Los valores de todas correlaciones son:", correlation)

# Se filtran los valores de correlación mayores a 0.25 en valor absoluto
y se imprimen
correlacion_filtrada = correlation[correlation.abs() > 0.25]
print("Nos quedamos con las siguientes variables cuya correlación es
mayor a |0.25|:", correlacion_filtrada)

# Para mostrar en un gráfico de barras las correlaciones en función de
"diff_tiempo" de Y_train
plt.figure(figsize=(10, 6))
correlation.plot(kind='bar')
plt.title('Correlación entre las variables X y Y (diff_tiempo)')
plt.xlabel('Variables X')
plt.ylabel('Correlación')
plt.xticks(rotation=80)
plt.subplots_adjust(bottom=0.32) # Ajustar los márgenes inferiores del
gráfico
```

```
plt.savefig('C1_Grafico_barras.png') # Guardar el gráfico como una
imagen PNG
plt.show()

# Crear archivo TSV con los nombres de las variables (filtradas) cuya
correlación es mayor a 0.25
variables_filtradas = correlacion_filtrada.index.tolist()
df_variables_filtradas = pd.DataFrame(variables_filtradas,
columns=['Variables'])
df_variables_filtradas.to_csv('C1_Variables.tsv', sep='\t', index=False)
```

Anexo GG. C1_Variables.tsv– python_tfg

≡ C1_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 humidity_activeTrue
7 dew_point_activeTrue
8 pop_activeTrue
```

Anexo HH. C1_Variables.tsv– python_tfg_verano

≡ C1_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 pressure_activeTrue
7 humidity_activeTrue
8 dew_point_activeTrue
9 uvi_activeTrue
10 clouds_activeTrue
```

Anexo II. C1_Variables.tsv– python_tfg_conjunto

≡ C1_Variables.tsv

```
1 Variables
2 dew_point_activeTrue
3 uvi_activeTrue
4 wind_deg_activeTrue
5 wind_gust_activeTrue
```

Anexo JJ. Código C2_Preparacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: HACIENDO USO DE BBDD corr_filter, sin la
feels_like_activeTrue
#####           Análisis de Correlación para Evaluación de todas las
variables sin separar en train y test3
#####           Nos quedamos con todas las variables filtradas
#####           Guardamos dichas variables en un archivo TSV

# Se importan las bibliotecas necesarias
import pandas as pd
import matplotlib.pyplot as plt

# Leen los archivos (separados por tabulaciones '\t') y los almacenan en
DataFrames
X_train = pd.read_csv('BBDD_corr_filter/data-
X_train_normalized_corr_filter.tsv', sep='\t')
Y_train = pd.read_csv('BBDD_corr_filter/data-y_train_corr_filter.tsv',
sep='\t')

# Se calcula la correlacion entre la columna X_train y "diff_tiempo" de
Y_train y se imprimen
correlation = X_train.corrwith(Y_train['diff_tiempo'])
print("Los valores de todas correlaciones son:",correlation)

# Para mostrar en un gráfico de barras las correlaciones en función de
"diff_tiempo" de Y_train
plt.figure(figsize=(10, 6))
correlation.plot(kind='bar')
plt.title('Correlación entre las variables X y Y (diff_tiempo)')
plt.xlabel('Variables X')
plt.ylabel('Correlación')
plt.xticks(rotation=80)
plt.subplots_adjust(bottom=0.32) # Ajustar los márgenes inferiores del
gráfico
plt.savefig('C2_Grafico_barras.png') # Guardar el gráfico como una
imagen PNG
plt.show()

# Crear archivo TSV con los nombres de todas las variables filtradas
variables = correlation.index.tolist()
df_variables = pd.DataFrame(variables, columns=['Variables'])
df_variables.to_csv('C2_Variables.tsv', sep='\t', index=False)
```

Anexo KK. C2_Variables.tsv– python_tfg

≡ C2_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 pressure_activeTrue
7 humidity_activeTrue
8 dew_point_activeTrue
9 uvi_activeTrue
10 clouds_activeTrue
11 wind_speed_activeTrue
12 wind_deg_activeTrue
13 wind_gust_activeTrue
14 pop_activeTrue
```

Anexo LL. C2_Variables.tsv– python_tfg_verano

≡ C2_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 diff_temp_int_ext
4 temp_activeTrue
5 humidity_inicial
6 pressure_activeTrue
7 humidity_activeTrue
8 dew_point_activeTrue
9 uvi_activeTrue
10 wind_speed_activeTrue
11 wind_deg_activeTrue
12 wind_gust_activeTrue
13 clouds_activeTrue
```

Anexo MM. C2_Variables.tsv– python_tfg_conjunto

≡ C2_Variables.tsv

```
1  Variables
2  diff_temp_interior
3  diff_temp_int_ext
4  temp_activeTrue
5  humidity_inicial
6  pressure_activeTrue
7  humidity_activeTrue
8  dew_point_activeTrue
9  uvi_activeTrue
10 clouds_activeTrue
11 wind_speed_activeTrue
12 wind_deg_activeTrue
13 wind_gust_activeTrue
14 pop_activeTrue
```

Anexo NN. Código C3_Preparacion.py – python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: HACIENDO USO DE BBDD corr_filter, sin la
feels_like_activeTrue
##### LASSO con datos de entrenamiento Least Absolute
Shrinkage and Selection Operator

# Importar las bibliotecas necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LassoCV
from sklearn.preprocessing import StandardScaler

# Paso 1: Preparación de datos
# Carga los datos de X en un DataFrame de Pandas
datos_X = pd.read_csv("BBDD_corr_filter/data-
X_train_normalized_corr_filter.tsv", sep="\t")

# Carga los datos de Y en un DataFrame de Pandas
datos_y = pd.read_csv("BBDD_corr_filter/data-y_train_corr_filter.tsv",
sep="\t")

# Exploración inicial de los datos
print(datos_X.head())
print(datos_X.describe())
print(datos_y.head())
print(datos_y.describe())

# Paso 2: Preprocesamiento de datos
# Elimina las columnas con valores faltantes
datos_X = datos_X.dropna()
datos_y = datos_y.dropna()

# Paso 3: Instancia un objeto de la clase LassoCV para selección
automática del parámetro de regularización
# Hacemos un CV para buscar la mejor lambda
model = LassoCV(cv=5, alphas = [2, 5])
print(datos_X)

# Ajusta el modelo utilizando los datos de entrenamiento
model.fit(datos_X, datos_y)

# Paso 4: Obtener coeficientes y características seleccionadas
# Obtén los coeficientes del modelo
```

```

coefficients = model.coef_

# Obtén las características seleccionadas
selected_features = datos_X.columns[model.coef_ != 0]

print("Coeficientes:", coefficients)
print("Características seleccionadas:", selected_features)

# Paso 5: Visualización de resultados
# Visualiza los coeficientes
plt.figure(figsize=(10, 6))
plt.bar(datos_X.columns, coefficients)
plt.xticks(rotation=70)
plt.xlabel('Características')
plt.ylabel('Coeficientes')
plt.title('Coeficientes del modelo LASSO')
plt.subplots_adjust(bottom=0.31) # Ajustar los márgenes inferiores del
gráfico
plt.savefig('C3_Grafico_barras.png') # Guardar el gráfico como una
imagen PNG
plt.show()

# Se filtran los valores de correlación
print("Nos quedamos con las siguientes variables tras aplicar LASSO:",
selected_features)

# Crear archivo TSV con los nombres de las variables (filtradas)
variables_filtradas = selected_features.tolist()
df_variables_filtradas = pd.DataFrame(variables_filtradas,
columns=['Variables'])
df_variables_filtradas.to_csv('C3_Variables.tsv', sep='\t', index=False)

```

Anexo ÑÑ. C3_Variables.tsv– python_tfg

≡ C3_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 temp_activeTrue
4 humidity_inicial
5 uvi_activeTrue
6 wind_deg_activeTrue
7 wind_gust_activeTrue
8 pop_activeTrue
```

Anexo OO. C3_Variables.tsv– python_tfg_verano

≡ C3_Variables.tsv

```
1 Variables
2 diff_temp_interior
3 humidity_inicial
4 pressure_activeTrue
5 dew_point_activeTrue
6 uvi_activeTrue
7 wind_speed_activeTrue
```

Anexo PP. C3_Variables.tsv– python_tfg_conjunto

≡ C3_Variables.tsv

```
1 Variables
2 temp_activeTrue
3 dew_point_activeTrue
4 uvi_activeTrue
5 clouds_activeTrue
6 pop_activeTrue
```

Anexo QQ. Código fn_modelos.py- python_tfg

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Función auxiliar para calcular los resultados de los 4
modelos.
```

```
# Se importan las bibliotecas necesarias
```

```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error, r2_score,
mean_absolute_error
import numpy as np
from sklearn.model_selection import GridSearchCV
```

```
def r2_custom(y, pred):
    corr_matrix = np.corrcoef(y, pred)
    corr = corr_matrix[0,1]
    R_sq = corr**2
    return R_sq
```

```
# Los datos de entrada de la función serán los datos X e Y de
entrenamiento y test. Además de un archivo que proporcione el orden de
las variables
```

```
def fn_model(X_train, X_test, y_train, y_test, variable_order):

    # Regresión Logística
    linear_model = LinearRegression()
    linear_model.fit(X_train, y_train)
    linear_predictions_train = linear_model.predict(X_train)
    linear_predictions_test = linear_model.predict(X_test)
    linear_mse_train = mean_squared_error(y_train,
linear_predictions_train)
    linear_mse_test = mean_squared_error(y_test,
linear_predictions_test)
    linear_r2_train = r2_custom(y_train, linear_predictions_train)
    linear_r2_test = r2_custom(y_test, linear_predictions_test)
    linear_coefs = linear_model.coef_
    linear_mae_train = mean_absolute_error(y_train,
linear_predictions_train)
```

```

        linear_mae_test = mean_absolute_error(y_test,
linear_predictions_test)
        linear_rmse_train = np.sqrt(linear_mse_train)
        linear_rmse_test = np.sqrt(linear_mse_test)

# Árbol de decisión
    # Se definen los hiperparámetros a explorar para cada modelo.
Para árbol de decisión se tiene max_depth y min_samples_split
        tree_param_grid = {
            'max_depth': [None, 5, 10],
            'min_samples_split': [2, 5, 6] # Como mucho el número total
de filas dividido entre 2
        }
        # Se crea un modelo de Árbol de decisión
        tree_model = DecisionTreeRegressor()
        # Se Crea un objeto GridSearchCV para encontrar los mejores
hiperparámetros
        tree_grid_search = GridSearchCV(estimator=tree_model,
param_grid=tree_param_grid, cv=5, scoring='neg_mean_squared_error')
        tree_grid_search.fit(X_train, y_train)
        tree_best_params = tree_grid_search.best_params_

        # Crear el mejor modelo de Árbol de Decisión con los
hiperparámetros encontrados
        best_tree_model = DecisionTreeRegressor(**tree_best_params)
        best_tree_model.fit(X_train, y_train)
        # Realizar predicciones y calcular métricas para el conjunto de
entrenamiento y prueba
        tree_predictions_train = best_tree_model.predict(X_train)
        tree_predictions_test = best_tree_model.predict(X_test)
        tree_mse_train = mean_squared_error(y_train,
tree_predictions_train)
        tree_mse_test = mean_squared_error(y_test, tree_predictions_test)
        tree_r2_train = r2_custom(y_train, tree_predictions_train)
        tree_r2_test = r2_custom(y_test, tree_predictions_test)
        tree_importances = best_tree_model.feature_importances_
        tree_mae_train = mean_absolute_error(y_train,
tree_predictions_train)
        tree_mae_test = mean_absolute_error(y_test,
tree_predictions_test)
        tree_rmse_train = np.sqrt(tree_mse_train)
        tree_rmse_test = np.sqrt(tree_mse_test)

# Random Forest
        forest_param_grid = {
            'n_estimators': [100, 200, 300, 500],

```

```

        'max_depth': [None, 5, 10],
        'min_samples_split': [2, 5]
    }
    forest_model = RandomForestRegressor()
    forest_grid_search = GridSearchCV(estimator=forest_model,
param_grid=forest_param_grid, cv=5, scoring='neg_mean_squared_error')
    forest_grid_search.fit(X_train, y_train)
    forest_best_params = forest_grid_search.best_params_
    forest_best_score = -forest_grid_search.best_score_

    best_forest_model = RandomForestRegressor(**forest_best_params)
    best_forest_model.fit(X_train, y_train)
    forest_predictions_train = best_forest_model.predict(X_train)
    forest_predictions_test = best_forest_model.predict(X_test)
    forest_mse_train = mean_squared_error(y_train,
forest_predictions_train)
    forest_mse_test = mean_squared_error(y_test,
forest_predictions_test)
    forest_r2_train = r2_custom(y_train, forest_predictions_train)
    forest_r2_test = r2_custom(y_test, forest_predictions_test)
    forest_importances = best_forest_model.feature_importances_
    forest_mae_train = mean_absolute_error(y_train,
forest_predictions_train)
    forest_mae_test = mean_absolute_error(y_test,
forest_predictions_test)
    forest_rmse_train = np.sqrt(forest_mse_train)
    forest_rmse_test = np.sqrt(forest_mse_test)

# XGBoost
xgb_param_grid = {
    'n_estimators': [100, 200, 300, 500],
    'max_depth': [None, 5, 10],
    'learning_rate': [0.01, 0.1, 0.3]
}
xgb_model = XGBRegressor()
xgb_grid_search = GridSearchCV(estimator= xgb_model,
param_grid=xgb_param_grid, cv=5, scoring='neg_mean_squared_error')
xgb_grid_search.fit(X_train, y_train)
xgb_best_params = xgb_grid_search.best_params_
xgb_best_score = -xgb_grid_search.best_score_

    best_xgb_model = XGBRegressor(**xgb_best_params)
    best_xgb_model.fit(X_train, y_train)
    xgb_predictions_train = best_xgb_model.predict(X_train)
    xgb_predictions_test = best_xgb_model.predict(X_test)
    xgb_mse_train = mean_squared_error(y_train,
xgb_predictions_train)
    xgb_mse_test = mean_squared_error(y_test, xgb_predictions_test)

```

```

xgb_r2_train = r2_custom(y_train, xgb_predictions_train)
xgb_r2_test = r2_custom(y_test, xgb_predictions_test)
xgb_importances = best_xgb_model.feature_importances_
xgb_mae_train = mean_absolute_error(y_train,
xgb_predictions_train)
xgb_mae_test = mean_absolute_error(y_test, xgb_predictions_test)
xgb_rmse_train = np.sqrt(xgb_mse_train)
xgb_rmse_test = np.sqrt(xgb_mse_test)

result = {
    'linear_predictions_train':linear_predictions_train,
    'linear_predictions_test':linear_predictions_test,
    'linear_mse_train':linear_mse_train,
    'linear_mse_test':linear_mse_test,
    'linear_r2_train':linear_r2_train,
    'linear_r2_test':linear_r2_test,
    'linear_coefs': dict(zip(variable_order, linear_coefs)),
    'linear_mae_train': linear_mae_train,
    'linear_mae_test': linear_mae_test,
    'linear_rmse_train': linear_rmse_train,
    'linear_rmse_test': linear_rmse_test,

    'tree_predictions_train':tree_predictions_train,
    'tree_predictions_test':tree_predictions_test,
    'tree_mse_train':tree_mse_train,
    'tree_mse_test':tree_mse_test,
    'tree_r2_train':tree_r2_train,
    'tree_r2_test':tree_r2_test,
    'tree_importances': dict(zip(variable_order,
tree_importances))),
    'tree_mae_train': tree_mae_train,
    'tree_mae_test': tree_mae_test,
    'tree_rmse_train': tree_rmse_train,
    'tree_rmse_test': tree_rmse_test,

    'forest_predictions_train':forest_predictions_train,
    'forest_predictions_test':forest_predictions_test,
    'forest_mse_train':forest_mse_train,
    'forest_mse_test':forest_mse_test,
    'forest_r2_train':forest_r2_train,
    'forest_r2_test':forest_r2_test,
    'forest_importances': dict(zip(variable_order,
forest_importances))),
    'forest_mae_train': forest_mae_train,
    'forest_mae_test': forest_mae_test,
    'forest_rmse_train': forest_rmse_train,
    'forest_rmse_test': forest_rmse_test,

```

```

        'xgb_predictions_train':xgb_predictions_train,
        'xgb_predictions_test':xgb_predictions_test,
        'xgb_mse_train':xgb_mse_train,
        'xgb_mse_test':xgb_mse_test,
        'xgb_r2_train':xgb_r2_train,
        'xgb_r2_test':xgb_r2_test,
        'xgb_importances': dict(zip(variable_order,
xgb_importances)),
        'xgb_mae_train': xgb_mae_train,
        'xgb_mae_test': xgb_mae_test,
        'xgb_rmse_train': xgb_rmse_train,
        'xgb_rmse_test': xgb_rmse_test

    }

    return result

```

Anexo RR. Código D_Entrenamiento_y_evaluacion.py

```
##### Proyecto: TFG - Estudio y algoritmos para la activación anticipada
de elementos de climatización
##### Autor: Marina Vázquez
##### Descripción: Resultados de todos los modelos
##### Se crea Excel donde se evalúa para cada modelo R2,
MAE, MSE y RMSE de entrenamiento y test
##### Se crean 2 Excel: B y C.
##### B para BBDD. Con 3 hojas B1, B2 y B3. Se reflejan los
coeficientes e importancias de las variables según el modelo (regresion,
dt, rf, xgb)
##### C para BBDD_corr_filter. Con 3 hojas C1, C2 y C3. Se
repite (regresion, dt, rf, xgb)

# Se importan las librerías necesarias.
import pandas as pd
from openpyxl import Workbook
import os
from fn_modelos import r2_custom
from fn_modelos import fn_model

#####
##### B #####
#####

# Cargar el dataset en un DataFrame de Pandas de la ruta B
X_train_complete_B = pd.read_csv('BBDD\data-X_train_normalized.tsv',
sep="\t")
X_test_complete_B = pd.read_csv('BBDD\data-X_test_normalized.tsv',
sep="\t")

# Crear las matrices test
y_train_B = pd.read_csv('BBDD\data-y_train.tsv', sep="\t")
y_train_B = y_train_B['diff_tiempo'].values
y_test_B = pd.read_csv('BBDD\data-y_test.tsv', sep="\t")
y_test_B = y_test_B['diff_tiempo'].values

#####
##### B1 corr 0.25
```

```
#####

# Carga los datos para establecer orden
X_train_orden_B1 =
pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg_conjunto\\B1_Variable
s.tsv', delimiter='\\t')
variable_order_B1 = X_train_orden_B1.values.flatten().tolist()

b1_variables = pd.read_csv('B1_Variables.tsv', sep="\\t")

# Obtener los nombres de las variables independientes
independent_vars_B1 = b1_variables['Variables'].tolist()

# Crear las matrices X para B1
X_train_b1 = X_train_complete_B[independent_vars_B1].values
X_test_b1 = X_test_complete_B[independent_vars_B1].values

# Hacemos uso de la función auxiliar fn_model e imprimimos los resultados
resultados_b1 =
fn_model(X_train_b1,X_test_b1,y_train_B,y_test_B,variable_order_B1)

#####
##### B2 Todas
#####

# Carga los datos para establecer orden
X_train_orden_B2 =
pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg_conjunto\\B2_Variable
s.tsv', delimiter='\\t')
variable_order_B2 = X_train_orden_B2.values.flatten().tolist()

b2_variables = pd.read_csv('B2_Variables.tsv', sep="\\t")

# Obtener los nombres de las variables independientes
independent_vars_B2 = b2_variables['Variables'].tolist()

# Crear las matrices X para B2
X_train_b2 = X_train_complete_B[independent_vars_B2].values
X_test_b2 = X_test_complete_B[independent_vars_B2].values

# Hacemos uso de la función auxiliar fn_model e imprimimos los resultados
resultados_b2 =
fn_model(X_train_b2,X_test_b2,y_train_B,y_test_B,variable_order_B2)

print("Finalizado resultados B")

#####
```

```

##### B3 LASSO
#####

# Carga los datos para establecer orden
X_train_orden_B3 =
pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg_conjunto\\B3_Variable
s.tsv', delimiter='\\t')
variable_order_B3 = X_train_orden_B3.values.flatten().tolist()

b3_variables = pd.read_csv('B3_Variables.tsv', sep="\\t")

# Obtener los nombres de las variables independientes
independent_vars_B3 = b3_variables['Variables'].tolist()

# Crear las matrices X para B3
X_train_b3 = X_train_complete_B[independent_vars_B3].values
X_test_b3 = X_test_complete_B[independent_vars_B3].values

# Hacemos uso de la función auxiliar fn_model e imprimimos los resultados
resultados_b3 =
fn_model(X_train_b3,X_test_b3,y_train_B,y_test_B,variable_order_B3)

#####
##### C #####
#####

# Cargar el dataset en un DataFrame de Pandas de la ruta B
X_train_complete_C = pd.read_csv('BBDD_corr_filter\\data-
X_train_normalized_corr_filter.tsv', sep="\\t")
X_test_complete_C = pd.read_csv('BBDD_corr_filter\\data-
X_test_normalized_corr_filter.tsv', sep="\\t")

y_train_C = pd.read_csv('BBDD_corr_filter\\data-y_train_corr_filter.tsv',
sep="\\t")
y_train_C = y_train_C['diff_tiempo'].values
y_test_C = pd.read_csv('BBDD_corr_filter\\data-y_test_corr_filter.tsv',
sep="\\t")
y_test_C = y_test_C['diff_tiempo'].values

#####
##### C1 corr 0.25
#####

# Carga los datos para establecer orden

```

```

X_train_orden_C1 =
pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg_conjunto\\C1_Variable
s.tsv', delimiter='\\t')
variable_order_C1 = X_train_orden_C1.values.flatten().tolist()

c1_variables = pd.read_csv('C1_Variables.tsv', sep="\\t")

# Obtener los nombres de las variables independientes
independent_vars_c1 = c1_variables['Variables'].tolist()

# Crear las matrices X para C1
X_train_c1 = X_train_complete_C[independent_vars_c1].values
X_test_c1 = X_test_complete_C[independent_vars_c1].values

# Hacemos uso de la función auxiliar fn_model e imprimimos los resultados
resultados_c1 =
fn_model(X_train_c1,X_test_c1,y_train_C,y_test_C,variable_order_C1)

#####
##### C2 Todas
#####

# Carga los datos para establecer orden
X_train_orden_C2 =
pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg_conjunto\\C2_Variable
s.tsv', delimiter='\\t')
variable_order_C2 = X_train_orden_C2.values.flatten().tolist()

c2_variables = pd.read_csv('C2_Variables.tsv', sep="\\t")

# Obtener los nombres de las variables independientes
independent_vars_c2 = c2_variables['Variables'].tolist()

# Crear las matrices X para C2
X_train_c2 = X_train_complete_C[independent_vars_c2].values
X_test_c2 = X_test_complete_C[independent_vars_c2].values

# Hacemos uso de la función auxiliar fn_model e imprimimos los resultados
resultados_c2 =
fn_model(X_train_c2,X_test_c2,y_train_C,y_test_C,variable_order_C2)

print("Finalizado resultados C")

#####
##### C3 LASSO
#####

```

```

# Carga los datos para establecer orden
X_train_orden_C3 =
pd.read_csv('C:\\Users\\Marina\\Desktop\\python_tfg_conjunto\\C3_Variable
s.tsv', delimiter='\\t')
variable_order_C3 = X_train_orden_C3.values.flatten().tolist()

c3_variables = pd.read_csv('C3_Variables.tsv', sep="\\t")

# Obtener los nombres de las variables independientes
independent_vars_c3 = c3_variables['Variables'].tolist()

# Crear las matrices X para C3
X_train_c3 = X_train_complete_C[independent_vars_c3].values
X_test_c3 = X_test_complete_C[independent_vars_c3].values

# Hacemos uso de la función auxiliar fn_model e imprimimos los resultados
resultados_c3 =
fn_model(X_train_c3,X_test_c3,y_train_C,y_test_C,variable_order_C3)

#####
##### EVALUACIÓN #####
#####

wb = Workbook()
ws = wb.active

resultados_completos = [resultados_b1, resultados_b2, resultados_b3,
resultados_c1, resultados_c2, resultados_c3]
modelos = ["B1", "B2", "B3", "C1", "C2", "C3"]
tipo = ['Regresion', 'DecisionTree', 'RandomForest', 'XGBoost']

ws.cell(row=1, column=1, value="Modelo")
ws.cell(row=1, column=2, value="R2 de prueba")
ws.cell(row=1, column=3, value="R2 de entrenamiento")
ws.cell(row=1, column=4, value="MSE de prueba")
ws.cell(row=1, column=5, value="MSE de entrenamiento")
ws.cell(row=1, column=6, value="MAE de prueba")
ws.cell(row=1, column=7, value="MAE de entrenamiento")
ws.cell(row=1, column=8, value="RMSE de prueba")
ws.cell(row=1, column=9, value="RMSE de entrenamiento")

for i, modelo in enumerate(modelos):
    ws.cell(row=i*4 + 2, column=1, value=modelo+"-"+tipo[0])
    ws.cell(row=i*4 + 2, column=2,
value=resultados_completos[i]['linear_r2_test'])

```

```

ws.cell(row=i*4 + 2, column=3,
value=resultados_completos[i]['linear_r2_train'])
ws.cell(row=i*4 + 2, column=4,
value=resultados_completos[i]['linear_mse_test'])
ws.cell(row=i*4 + 2, column=5,
value=resultados_completos[i]['linear_mse_train'])
ws.cell(row=i*4 + 2, column=6,
value=resultados_completos[i]['linear_mae_test'])
ws.cell(row=i*4 + 2, column=7,
value=resultados_completos[i]['linear_mae_train'])
ws.cell(row=i*4 + 2, column=8,
value=resultados_completos[i]['linear_rmse_test'])
ws.cell(row=i*4 + 2, column=9,
value=resultados_completos[i]['linear_rmse_train'])

ws.cell(row=i*4 + 3, column=1, value=modelo+"-"+tipo[1])
ws.cell(row=i*4 + 3, column=2,
value=resultados_completos[i]['tree_r2_test'])
ws.cell(row=i*4 + 3, column=3,
value=resultados_completos[i]['tree_r2_train'])
ws.cell(row=i*4 + 3, column=4,
value=resultados_completos[i]['tree_mse_test'])
ws.cell(row=i*4 + 3, column=5,
value=resultados_completos[i]['tree_mse_train'])
ws.cell(row=i*4 + 3, column=6,
value=resultados_completos[i]['tree_mae_test'])
ws.cell(row=i*4 + 3, column=7,
value=resultados_completos[i]['tree_mae_train'])
ws.cell(row=i*4 + 3, column=8,
value=resultados_completos[i]['tree_rmse_test'])
ws.cell(row=i*4 + 3, column=9,
value=resultados_completos[i]['tree_rmse_train'])

ws.cell(row=i*4 + 4, column=1, value=modelo+"-"+tipo[2])
ws.cell(row=i*4 + 4, column=2,
value=resultados_completos[i]['forest_r2_test'])
ws.cell(row=i*4 + 4, column=3,
value=resultados_completos[i]['forest_r2_train'])
ws.cell(row=i*4 + 4, column=4,
value=resultados_completos[i]['forest_mse_test'])
ws.cell(row=i*4 + 4, column=5,
value=resultados_completos[i]['forest_mse_train'])
ws.cell(row=i*4 + 4, column=6,
value=resultados_completos[i]['forest_mae_test'])
ws.cell(row=i*4 + 4, column=7,
value=resultados_completos[i]['forest_mae_train'])
ws.cell(row=i*4 + 4, column=8,
value=resultados_completos[i]['forest_rmse_test'])

```

```

        ws.cell(row=i*4 + 4, column=9,
value=resultados_completos[i]['forest_rmse_train'])

        ws.cell(row=i*4 + 5, column=1, value=modelo+"-"+tipo[3])
        ws.cell(row=i*4 + 5, column=2,
value=resultados_completos[i]['xgb_r2_test'])
        ws.cell(row=i*4 + 5, column=3,
value=resultados_completos[i]['xgb_r2_train'])
        ws.cell(row=i*4 + 5, column=4,
value=resultados_completos[i]['xgb_mse_test'])
        ws.cell(row=i*4 + 5, column=5,
value=resultados_completos[i]['xgb_mse_train'])
        ws.cell(row=i*4 + 5, column=6,
value=resultados_completos[i]['xgb_mae_test'])
        ws.cell(row=i*4 + 5, column=7,
value=resultados_completos[i]['xgb_mae_train'])
        ws.cell(row=i*4 + 5, column=8,
value=resultados_completos[i]['xgb_rmse_test'])
        ws.cell(row=i*4 + 5, column=9,
value=resultados_completos[i]['xgb_rmse_train'])

wb.save('D_Evaluacion.xlsx')

print("Finalizado Excel Evaluacion")

#####
##### COEFICIENTES #####
#####

#####
##### EXCEL B #####
#####

# Se crea el Excel B1 donde se hace uso de las bases de datos BBDD
wb1 = Workbook()

#####
##### B1 corr 0.25
#####

wsb1 = wb1.active
wsb1.title = "Hoja B1"

wsb1.cell(row=1, column=1, value="Variables B1")

```

```

wsb1.cell(row=1, column=2, value="Regression coefficients")
wsb1.cell(row=1, column=3, value="Importance Decision Tree")
wsb1.cell(row=1, column=4, value="Importance Random Forest")
wsb1.cell(row=1, column=5, value="Importance XGBoost")

# En el bucle uso la función .get(key, default)
# Esta se utiliza para obtener el valor asociado a una clave en un
diccionario.
# key: La clave cuyo valor se desea obtener.
# default (opcional): El valor predeterminado que se devuelve si la clave
no existe en el diccionario.

for i, var in enumerate(independent_vars_B1):

    # Escribir las variables en la primera columna debajo del encabezado
"Variables"
    wsb1.cell(row=i+2, column=1, value=var)

    # Escribir los coeficientes en la segunda columna debajo de
"Regression coefficients"
    coef = resultados_b1['linear_coefs'].get(var, 0.00)
    wsb1.cell(row=i+2, column=2, value=coef)

    # Escribir las importancias de Decision Tree en la tercera columna
debajo de "Importance Decision Tree"
    imp_dt = resultados_b1['tree_importances'].get(var, 0.00)
    wsb1.cell(row=i+2, column=3, value=imp_dt)

    # Escribir las importancias de Random Forest en la cuarta columna
debajo de "Importance Random Forest"
    imp_rf = resultados_b1['forest_importances'].get(var, 0.00)
    wsb1.cell(row=i+2, column=4, value=imp_rf)

    # Escribir las importancias de Random Forest en la cuarta columna
debajo de "Importance Random Forest"
    imp_xgb = resultados_b1['xgb_importances'].get(var, 0.00)
    wsb1.cell(row=i+2, column=5, value=imp_xgb)

#####
##### B2 Todas
#####

# La nueva hoja se insertará en la posición 1 (después de la Hoja B1)
wsb2 = wb1.create_sheet(title="Hoja B2", index=1)

wsb2.cell(row=1, column=1, value="Variables B2")

```

```

wsb2.cell(row=1, column=2, value="Regression coefficients")
wsb2.cell(row=1, column=3, value="Importance Decision Tree")
wsb2.cell(row=1, column=4, value="Importance Random Forest")
wsb2.cell(row=1, column=5, value="Importance XGBoost")

for i, var in enumerate(independent_vars_B2):

    # Escribir las variables en la primera columna debajo del encabezado
    "Variables"
    wsb2.cell(row=i+2, column=1, value=var)

    # Escribir los coeficientes en la segunda columna debajo de
    "Regression coefficients"
    coef = resultados_b2['linear_coefs'].get(var, 0.00)
    wsb2.cell(row=i+2, column=2, value=coef)

    # Escribir las importancias de Decision Tree en la tercera columna
    debajo de "Importance Decision Tree"
    imp_dt = resultados_b2['tree_importances'].get(var, 0.00)
    wsb2.cell(row=i+2, column=3, value=imp_dt)

    # Escribir las importancias de Random Forest en la cuarta columna
    debajo de "Importance Random Forest"
    imp_rf = resultados_b2['forest_importances'].get(var, 0.00)
    wsb2.cell(row=i+2, column=4, value=imp_rf)

    # Escribir las importancias de Random Forest en la cuarta columna
    debajo de "Importance Random Forest"
    imp_xgb = resultados_b2['xgb_importances'].get(var, 0.00)
    wsb2.cell(row=i+2, column=5, value=imp_xgb)

#####
##### B3 LASSO
#####

# La nueva hoja se insertará en la posición 1 (después de la Hoja B1)
wsb3 = wb1.create_sheet(title="Hoja B3", index=2)

wsb3.cell(row=1, column=1, value="Variables B3")
wsb3.cell(row=1, column=2, value="Regression coefficients")
wsb3.cell(row=1, column=3, value="Importance Decision Tree")
wsb3.cell(row=1, column=4, value="Importance Random Forest")
wsb3.cell(row=1, column=5, value="Importance XGBoost")

for i, var in enumerate(independent_vars_B3):

    # Escribir las variables en la primera columna debajo del encabezado
    "Variables"

```

```

wsb3.cell(row=i+2, column=1, value=var)

# Escribir los coeficientes en la segunda columna debajo de
"Regression coefficients"
coef = resultados_b3['linear_coefs'].get(var, 0.00)
wsb3.cell(row=i+2, column=2, value=coef)

# Escribir las importancias de Decision Tree en la tercera columna
debajo de "Importance Decision Tree"
imp_dt = resultados_b3['tree_importances'].get(var, 0.00)
wsb3.cell(row=i+2, column=3, value=imp_dt)

# Escribir las importancias de Random Forest en la cuarta columna
debajo de "Importance Random Forest"
imp_rf = resultados_b3['forest_importances'].get(var, 0.00)
wsb3.cell(row=i+2, column=4, value=imp_rf)

# Escribir las importancias de Random Forest en la cuarta columna
debajo de "Importance Random Forest"
imp_xgb = resultados_b3['xgb_importances'].get(var, 0.00)
wsb3.cell(row=i+2, column=5, value=imp_xgb)

# Guardar el archivo Excel 1 que hace uso de BBDD
wb1.save('D_Result_coef_imp_B.xlsx')

print("Finalizado Excel coef_imp B")

#####
##### EXCEL C #####
#####

# Se crea el Excel C1 donde se hace uso de las bases de datos
BBDD_corr_filter
wb2 = Workbook()

#####
##### C1 corr 0.25
#####

wsc1 = wb2.active
wsc1.title = "Hoja C1"

wsc1.cell(row=1, column=1, value="Variables C1")
wsc1.cell(row=1, column=2, value="Regression coefficients")
wsc1.cell(row=1, column=3, value="Importance Decision Tree")

```

```

wsc1.cell(row=1, column=4, value="Importance Random Forest")
wsc1.cell(row=1, column=5, value="Importance XGBoost")

for j, varc in enumerate(independent_vars_c1):

    # Escribir las variables en la primera columna debajo del encabezado
    "Variables"
    wsc1.cell(row=j+2, column=1, value=varc)

    # Escribir los coeficientes en la segunda columna debajo de
    "Regression coefficients"
    coef = resultados_c1['linear_coefs'].get(varc, 0.00)
    wsc1.cell(row=j+2, column=2, value=coef)

    # Escribir las importancias de Decision Tree en la tercera columna
    debajo de "Importance Decision Tree"
    imp_dt = resultados_c1['tree_importances'].get(varc, 0.00)
    wsc1.cell(row=j+2, column=3, value=imp_dt)

    # Escribir las importancias de Random Forest en la cuarta columna
    debajo de "Importance Random Forest"
    imp_rf = resultados_c1['forest_importances'].get(varc, 0.00)
    wsc1.cell(row=j+2, column=4, value=imp_rf)

    # Escribir las importancias de Random Forest en la cuarta columna
    debajo de "Importance Random Forest"
    imp_xgb = resultados_c1['xgb_importances'].get(varc, 0.00)
    wsc1.cell(row=j+2, column=5, value=imp_xgb)

#####
##### C2 Todas
#####

# La nueva hoja se insertará en la posición 1 (después de la Hoja B1)
wsc2 = wb2.create_sheet(title="Hoja C2", index=1)

wsc2.cell(row=1, column=1, value="Variables C2")
wsc2.cell(row=1, column=2, value="Regression coefficients")
wsc2.cell(row=1, column=3, value="Importance Decision Tree")
wsc2.cell(row=1, column=4, value="Importance Random Forest")
wsc2.cell(row=1, column=5, value="Importance XGBoost")

for j, varc in enumerate(independent_vars_c2):

    # Escribir las variables en la primera columna debajo del encabezado
    "Variables"
    wsc2.cell(row=j+2, column=1, value=varc)

```

```

        # Escribir los coeficientes en la segunda columna debajo de
"Regression coefficients"
        coef = resultados_c2['linear_coefs'].get(varc, 0.00)
        wsc2.cell(row=j+2, column=2, value=coef)

        # Escribir las importancias de Decision Tree en la tercera columna
debajo de "Importance Decision Tree"
        imp_dt = resultados_c2['tree_importances'].get(varc, 0.00)
        wsc2.cell(row=j+2, column=3, value=imp_dt)

        # Escribir las importancias de Random Forest en la cuarta columna
debajo de "Importance Random Forest"
        imp_rf = resultados_c2['forest_importances'].get(varc, 0.00)
        wsc2.cell(row=j+2, column=4, value=imp_rf)

        # Escribir las importancias de Random Forest en la cuarta columna
debajo de "Importance Random Forest"
        imp_xgb = resultados_c2['xgb_importances'].get(varc, 0.00)
        wsc2.cell(row=j+2, column=5, value=imp_xgb)

#####
##### C3 LASSO
#####

# La nueva hoja se insertará en la posición 1 (después de la Hoja B1)
wsc3 = wb2.create_sheet(title="Hoja C3", index=2)

wsc3.cell(row=1, column=1, value="Variables C3")
wsc3.cell(row=1, column=2, value="Regression coefficients")
wsc3.cell(row=1, column=3, value="Importance Decision Tree")
wsc3.cell(row=1, column=4, value="Importance Random Forest")
wsc3.cell(row=1, column=5, value="Importance XGBoost")

for j, varc in enumerate(independent_vars_c3):

    # Escribir las variables en la primera columna debajo del encabezado
"Variables"
    wsc3.cell(row=j+2, column=1, value=varc)

    # Escribir los coeficientes en la segunda columna debajo de
"Regression coefficients"
    coef = resultados_c3['linear_coefs'].get(varc, 0.00)
    wsc3.cell(row=j+2, column=2, value=coef)

    # Escribir las importancias de Decision Tree en la tercera columna
debajo de "Importance Decision Tree"
    imp_dt = resultados_c3['tree_importances'].get(varc, 0.00)
    wsc3.cell(row=j+2, column=3, value=imp_dt)

```

```

    # Escribir las importancias de Random Forest en la cuarta columna
    debajo de "Importance Random Forest"
    imp_rf = resultados_c3['forest_importances'].get(varc, 0.00)
    wsc3.cell(row=j+2, column=4, value=imp_rf)

    # Escribir las importancias de Random Forest en la cuarta columna
    debajo de "Importance Random Forest"
    imp_xgb = resultados_c3['xgb_importances'].get(varc, 0.00)
    wsc3.cell(row=j+2, column=5, value=imp_xgb)

# Guardar el archivo Excel 1 que hace uso de BBDD_corr_filter
wb2.save('D_Result_coef_imp_C.xlsx')

print("Finalizado Excel coef_imp C")

```

Anexo SS. Excel D_Evaluacion.xlsx – python_tfg

Modelo	R2 de prueba	R2 de entrenamiento	MSE de prueba	MSE de entrenamiento	MAE de prueba	MAE de entrenamiento	RMSE de prueba	RMSE de entrenamiento
B2-RandomForest	0,52654	0,91373	1433,24979	2033,16428	31,25530	34,77068	37,85829	45,09062
C3-RandomForest	0,00376	0,96426	1808,70282	1009,31862	34,33673	25,55802	42,52885	31,76978
B1-XGBoost	0,54112	0,99999	1803,50697	0,63273	36,29044	0,43031	42,46772	0,79544
C1-XGBoost	0,54112	0,99999	1803,50697	0,63273	36,29044	0,43031	42,46772	0,79544
B2-XGBoost	0,81182	0,99999	1894,23219	0,45350	36,39899	0,32730	43,52278	0,67342
C2-XGBoost	0,81182	0,99999	1894,23219	0,45350	36,39899	0,32730	43,52278	0,67342
B3-RandomForest	0,00001	0,96776	1987,96371	1024,63612	36,60242	25,29046	44,58659	32,00994
C3-XGBoost	0,14398	0,99999	2125,12596	0,45600	37,81627	0,33076	46,09909	0,67528
C2-RandomForest	0,24184	0,87642	2213,71330	1625,59621	39,65154	32,55035	47,05011	40,31868
B1-RandomForest	0,03052	0,95279	2326,16436	1087,68432	39,66953	23,98001	48,23033	32,98006
B3-XGBoost	0,03198	0,99999	2646,38372	0,45600	41,54241	0,33076	51,44301	0,67528
C1-RandomForest	0,00528	0,94266	2579,40196	984,19020	42,40228	23,29933	50,78781	31,37181
C1-DecisionTree	0,15735	1,00000	3978,64288	0,00000	51,53815	0,00000	63,07648	0,00000
B1-DecisionTree	0,00000	0,71063	3529,65761	2021,79255	52,55689	31,90602	59,41092	44,96435
B2-DecisionTree	0,00000	0,71063	3529,65761	2021,79255	52,55689	31,90602	59,41092	44,96435
C2-DecisionTree	0,00000	0,71063	3529,65761	2021,79255	52,55689	31,90602	59,41092	44,96435
B2-Regresion	0,00008	1,00000	5276,52981	0,00000	66,94808	0,00000	72,63973	0,00000
C2-Regresion	0,00001	1,00000	5285,39637	0,00000	67,10040	0,00000	72,70073	0,00000
B3-Regresion	0,00017	0,99999	5846,98989	0,09543	72,54450	0,21675	76,46561	0,30892
C3-Regresion	0,00013	1,00000	6189,69791	0,03064	74,17763	0,12254	78,67463	0,17503
C3-DecisionTree	0,35598	1,00000	6614,45187	0,00000	77,65944	0,00000	81,32928	0,00000
B3-DecisionTree	0,35598	0,71063	9974,94623	2021,79255	96,29801	31,90602	99,87465	44,96435
C1-Regresion	0,94292	0,99909	24816,82963	6,37627	146,23612	1,67469	157,53358	2,52513
B1-Regresion	0,02999	1,00000	15779937,11972	0,00000	3369,46631	0,00000	3972,39690	0,00000

Anexo TT. Excel D_Evaluacion.xlsx – python_tfg_verano

Modelo	R2 de prueba	R2 de entrenamiento	MSE de prueba	MSE de entrenamiento	MAE de prueba	MAE de entrenamiento	RMSE de prueba	RMSE de entrenamiento
C2-DecisionTree	0,64495	0,98549	15616,81536	110,50323	78,79729	7,77453	124,96726	10,51205
C1-DecisionTree	0,19316	0,98549	22194,48998	110,50323	90,96259	7,77453	148,97815	10,51205
B2-DecisionTree	0,03346	1,00000	24940,63447	0,00000	97,69100	0,00000	157,92604	0,00000
C1-XGBoost	0,27454	0,98271	25303,07294	718,87249	97,97543	15,66154	159,06940	26,81180
B1-XGBoost	0,26493	0,98277	25448,52761	718,18680	98,01494	15,65072	159,52595	26,79901
B2-XGBoost	0,63019	0,98275	24239,30176	709,06033	100,70684	15,51204	155,68976	26,62819
C2-XGBoost	0,63019	0,98275	24239,30176	709,06033	100,70684	15,51204	155,68976	26,62819
C1-RandomForest	0,62031	0,92021	19096,86988	1175,74621	109,11689	27,70098	138,19142	34,28916
B1-RandomForest	0,71633	0,91231	20735,77368	1230,87026	110,42563	28,44104	143,99921	35,08376
B2-RandomForest	0,59896	0,92287	21103,40125	1278,42388	113,05736	29,25118	145,27010	35,75505
C2-RandomForest	0,52031	0,94916	22910,58549	842,68505	113,38310	22,62330	151,36243	29,02904
B3-RandomForest	0,32846	0,96508	24512,19116	820,19280	116,97054	20,63915	156,56370	28,63901
C3-RandomForest	0,08007	0,95387	25942,50964	1155,56601	117,86943	23,12687	161,06679	33,99362
B1-DecisionTree	0,08885	1,00000	34584,52654	0,00000	118,01683	0,00000	185,96915	0,00000
B3-Regresion	0,27840	0,93590	16919,51469	488,15982	118,05998	18,65312	130,07503	22,09434
C3-Regresion	0,27840	0,93590	16919,51469	488,15982	118,05998	18,65312	130,07503	22,09434
B3-XGBoost	0,15058	1,00000	31667,21293	0,00000	124,49495	0,00044	177,95284	0,00056
C3-XGBoost	0,15058	1,00000	31667,21293	0,00000	124,49495	0,00044	177,95284	0,00056
B1-Regresion	0,27680	0,98170	34123,21749	139,40254	141,41463	8,15956	184,72471	11,80689
C1-Regresion	0,02736	0,98162	34895,69226	139,97084	144,19045	8,30411	186,80389	11,83093
B3-DecisionTree	0,35642	1,00000	40019,99374	0,00000	145,63787	0,00000	200,04998	0,00000
C3-DecisionTree	0,18019	0,98549	40552,09751	110,50323	150,28640	7,77453	201,37551	10,51205
B2-Regresion	0,05305	1,00000	116718,33633	0,00000	297,25550	0,00000	341,64065	0,00000
C2-Regresion	0,34325	1,00000	188714,39886	0,00000	402,70299	0,00000	434,41271	0,00000

Anexo UU. Excel D_Evaluacion.xlsx – python_tfg_conjunto

Modelo	R2 de prueba	R2 de entrenamiento	MSE de prueba	MSE de entrenamiento	MAE de prueba	MAE de entrenamiento	RMSE de prueba	RMSE de entrenamiento
C2-DecisionTree	0,01629	0,83808	4399,37340	1525,51966	49,38066	21,64393	66,32777	39,05790
B2-XGBoost	0,14729	1,00000	5304,93788	0,00000	62,75589	0,00052	72,83500	0,00069
B2-DecisionTree	0,23047	0,79995	10197,44475	1884,71186	64,95281	28,09680	100,98240	43,41327
C2-XGBoost	0,09985	1,00000	6103,89982	0,00000	66,88527	0,00046	78,12746	0,00061
C2-RandomForest	0,04403	0,93689	6697,42217	1128,55923	73,58038	27,84890	81,83778	33,59404
B2-RandomForest	0,00027	0,89698	7503,37925	2125,20292	77,76883	37,27362	86,62205	46,09992
B3-DecisionTree	0,14238	0,72274	12476,25321	2612,16897	83,56782	35,09433	111,69715	51,10938
C3-DecisionTree	0,13352	0,74965	13692,04217	2358,64212	88,76357	31,73585	117,01300	48,56585
B1-XGBoost	0,08639	0,99782	10083,25392	98,44380	89,41307	5,58940	100,41541	9,92188
C1-XGBoost	0,08639	0,99782	10083,25392	98,44380	89,41307	5,58940	100,41541	9,92188
B1-DecisionTree	0,04738	0,76354	14989,38815	2227,71220	89,52370	24,89339	122,43116	47,19865
C1-DecisionTree	0,04738	0,76354	14989,38815	2227,71220	89,52370	24,89339	122,43116	47,19865
B3-RandomForest	0,09912	0,78045	10456,94838	2674,33969	92,36423	43,02020	102,25922	51,71402
B3-XGBoost	0,12574	0,99735	14132,09950	103,33046	100,14193	5,90936	118,87851	10,16516
C3-XGBoost	0,12574	0,99735	14132,09950	103,33046	100,14193	5,90936	118,87851	10,16516
B3-Regresion	0,10614	0,42317	13377,84267	5434,43881	100,85959	52,63157	115,66262	73,71865
C3-Regresion	0,10614	0,42317	13377,84267	5434,43881	100,85959	52,63157	115,66262	73,71865
C3-RandomForest	0,16234	0,90039	12719,59147	1632,46177	100,86266	33,06288	112,78117	40,40373
B1-Regresion	0,00029	0,26408	12845,04394	6933,27313	102,73837	63,20187	113,33598	83,26628
C1-Regresion	0,00029	0,26408	12845,04394	6933,27313	102,73837	63,20187	113,33598	83,26628
C1-RandomForest	0,01863	0,92447	13303,68255	1589,55271	104,66660	32,47988	115,34159	39,86920
B1-RandomForest	0,01581	0,88527	13708,82388	1610,35804	106,07430	33,05900	117,08469	40,12927
C2-Regresion	0,00875	0,66034	44622,42301	3200,00455	158,98296	36,99293	211,24020	56,56858
B2-Regresion	0,00957	0,66837	53302,58979	3124,36940	166,37289	36,91073	230,87354	55,89606

Anexo VV. Excel D_Result_coef_imp de B y C – python_tfg

Del Excel B:

Variables B1	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-9085,75575	0,00000	0,21264	0,09540
diff_temp_int_ext	9745,56489	0,00000	0,01550	0,00457
temp_activeTrue	-205278,57166	0,00000	0,05329	0,01175
humidity_inicial	-6400,97597	0,87155	0,31914	0,46976
feels_like_activeTrue	-146344,01426	0,00000	0,05868	0,00000
humidity_activeTrue	-154715,85737	0,12845	0,20967	0,00000
dew_point_activeTrue	258420,34720	0,00000	0,03753	0,34647
pop_activeTrue	1760,88689	0,00000	0,09355	0,07205

Variables B2	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	169,20084	0,12845	0,13628	0,05002
diff_temp_int_ext	31,83557	0,00000	0,00898	0,00051
temp_activeTrue	-25,96348	0,00000	0,04709	0,00028
humidity_inicial	-91,11199	0,87155	0,29598	0,92807
feels_like_activeTrue	-34,23964	0,00000	0,03771	0,00000
pressure_activeTrue	-24,10976	0,00000	0,04688	0,02039
humidity_activeTrue	26,63893	0,00000	0,11823	0,00000
dew_point_activeTrue	-21,77787	0,00000	0,00864	0,00000
uvi_activeTrue	-16,55075	0,00000	0,00541	0,00000
clouds_activeTrue	-16,84752	0,00000	0,00088	0,00000
wind_speed_activeTrue	14,71499	0,00000	0,08943	0,00067
wind_deg_activeTrue	-36,54690	0,00000	0,12430	0,00007
wind_gust_activeTrue	76,31704	0,00000	0,03766	0,00000
pop_activeTrue	60,05439	0,00000	0,04254	0,00000

Variables B3	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	180,42263	0,00000	0,21551	0,06827
humidity_inicial	-89,62765	0,87155	0,31244	0,86179
feels_like_activeTrue	-53,94827	0,00000	0,09016	0,00000
uvi_activeTrue	-45,15694	0,00000	0,03482	0,06858
wind_deg_activeTrue	-34,56808	0,12845	0,17164	0,00000
wind_gust_activeTrue	76,16361	0,00000	0,08100	0,00000
pop_activeTrue	81,22773	0,00000	0,09444	0,00136

Del Excel C:

Variables C1	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	597,18782	0,01161	0,23256	0,09540
diff_temp_int_ext	-796,75003	0,01571	0,03168	0,00457
temp_activeTrue	1736,54572	0,08401	0,04106	0,01175
humidity_inicial	128,77541	0,79739	0,32556	0,46976
humidity_activeTrue	191,92884	0,09128	0,25141	0,00000
dew_point_activeTrue	-1084,44387	0,00000	0,02967	0,34647
pop_activeTrue	91,14006	0,00000	0,08806	0,07205

Variables C2	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	173,70540	0,12845	0,16546	0,05002
diff_temp_int_ext	25,12535	0,00000	0,00000	0,00051
temp_activeTrue	-35,99477	0,00000	0,02086	0,00028
humidity_inicial	-90,78835	0,87155	0,41593	0,92807
pressure_activeTrue	-21,78422	0,00000	0,06804	0,02039
humidity_activeTrue	34,95330	0,00000	0,09762	0,00000
dew_point_activeTrue	-30,06684	0,00000	0,01982	0,00000
uvi_activeTrue	-19,74608	0,00000	0,00000	0,00000
clouds_activeTrue	-18,53972	0,00000	0,02129	0,00000
wind_speed_activeTrue	13,78984	0,00000	0,04478	0,00067
wind_deg_activeTrue	-37,11694	0,00000	0,10003	0,00007
wind_gust_activeTrue	73,14860	0,00000	0,03748	0,00000
pop_activeTrue	63,38638	0,00000	0,00868	0,00000

Variables C3	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	181,76305	0,01315	0,23162	0,04246
temp_activeTrue	-52,63431	0,00000	0,05399	0,00121
humidity_inicial	-89,59031	0,66732	0,32979	0,91283
uvi_activeTrue	-46,63274	0,22825	0,04032	0,04265
wind_deg_activeTrue	-37,46428	0,09128	0,16646	0,00000
wind_gust_activeTrue	76,43496	0,00000	0,05526	0,00000
pop_activeTrue	81,92844	0,00000	0,12257	0,00085

Anexo WW. Excel D_Result_coef_imp de B y C – python_tfg_verano

Del Excel B:

Variables B1	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-156,12302	0,22719	0,28568	0,53492
diff_temp_int_ext	155,82566	0,00605	0,03680	0,06766
temp_activeTrue	-390,16257	0,00000	0,01518	0,00000
humidity_inicial	-204,26255	0,71434	0,22773	0,24889
pressure_activeTrue	65,60574	0,00052	0,10382	0,00247
humidity_activeTrue	-347,23603	0,00000	0,13465	0,05586
dew_point_activeTrue	161,39111	0,00000	0,07302	0,00000
uvi_activeTrue	93,20308	0,02870	0,08816	0,04398
wind_gust_activeTrue	17,30858	0,02320	0,03497	0,04622
clouds_activeTrue	33,70221	0,00000	0,00000	0,00000

Variables B2	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-302,44810	0,22677	0,28677	0,50620
diff_temp_int_ext	841,46684	0,08281	0,01238	0,03667
temp_activeTrue	419,01775	0,02870	0,01773	0,00000
humidity_inicial	-271,77562	0,64784	0,25194	0,19613
feels_like_activeTrue	-1074,32049	0,00000	0,03808	0,00000
pressure_activeTrue	5,20638	0,00605	0,07731	0,00186
humidity_activeTrue	497,09664	0,00072	0,10642	0,01952
dew_point_activeTrue	184,85980	0,00610	0,04658	0,00000
uvi_activeTrue	79,99685	0,00000	0,08707	0,04248
wind_speed_activeTrue	279,41828	0,00000	0,02198	0,08789
wind_deg_activeTrue	-131,22051	0,00000	0,01644	0,10925
wind_gust_activeTrue	-84,70283	0,00102	0,03716	0,00000
clouds_activeTrue	-37,50622	0,00000	0,00013	0,00000

Variables B3	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-220,24495	0,22770	0,28886	0,43598
humidity_inicial	-218,63897	0,64793	0,26087	0,42394
pressure_activeTrue	82,15836	0,01214	0,14561	0,01899
dew_point_activeTrue	35,63003	0,08280	0,12840	0,00734
uvi_activeTrue	33,00626	0,02943	0,12490	0,06481
wind_speed_activeTrue	88,78326	0,00000	0,05137	0,04894

Del Excel C:

Variables C1	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-138,32318	0,22949	0,33045	0,55872
diff_temp_int_ext	52,13525	0,00000	0,02778	0,07242
temp_activeTrue	-384,20139	0,02912	0,01404	0,00000
humidity_inicial	-175,91650	0,64002	0,20603	0,26122
pressure_activeTrue	45,62724	0,00000	0,09495	0,00233
humidity_activeTrue	-463,27464	0,10138	0,11776	0,05982
dew_point_activeTrue	209,14249	0,00000	0,11564	0,00000
uvi_activeTrue	120,94623	0,00000	0,09336	0,04549
clouds_activeTrue	10,79196	0,00000	0,00000	0,00000

Variables C2	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-439,96794	0,22949	0,25379	0,50620
diff_temp_int_ext	2195,62739	0,00000	0,01654	0,03667
temp_activeTrue	-607,72582	0,00000	0,03029	0,00000
humidity_inicial	-724,95693	0,64002	0,26376	0,19613
pressure_activeTrue	560,15990	0,00000	0,07739	0,00186
humidity_activeTrue	1518,77800	0,08402	0,07885	0,01952
dew_point_activeTrue	-579,57366	0,00000	0,09939	0,00000
uvi_activeTrue	-527,35969	0,00000	0,06741	0,04248
wind_speed_activeTrue	342,22923	0,01736	0,02940	0,08789
wind_deg_activeTrue	57,61211	0,02912	0,03498	0,10925
wind_gust_activeTrue	31,09945	0,00000	0,04780	0,00000
clouds_activeTrue	593,75514	0,00000	0,00039	0,00000

Variables C3	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-220,24495	0,22949	0,37003	0,43598
humidity_inicial	-218,63897	0,65738	0,22170	0,42394
pressure_activeTrue	82,15836	0,08402	0,12260	0,01899
dew_point_activeTrue	35,63003	0,00000	0,13695	0,00734
uvi_activeTrue	33,00626	0,02912	0,10766	0,06481
wind_speed_activeTrue	88,78326	0,00000	0,04106	0,04894

Anexo XX. Excel D_Result_coef_imp de B y C – python_tfg_conjunto

Del Excel B:

Variables B1	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
dew_point_activeTrue	-170,82367	0,24815	0,24231	0,10582
uvi_activeTrue	129,85092	0,49092	0,42628	0,55171
wind_deg_activeTrue	-35,70301	0,06066	0,17158	0,25236
wind_gust_activeTrue	-22,34274	0,20027	0,15984	0,09011

Variables B2	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-733,90892	0,68691	0,28135	0,04962
diff_temp_int_ext	337,76221	0,00000	0,03297	0,01759
temp_activeTrue	-860,51300	0,00000	0,02751	0,44133
humidity_inicial	-12,33192	0,00000	0,08419	0,04797
feels_like_activeTrue	666,42353	0,00000	0,03658	0,01244
pressure_activeTrue	-5,52022	0,00000	0,03410	0,00598
humidity_activeTrue	740,58224	0,00000	0,07715	0,00016
dew_point_activeTrue	-686,85269	0,18624	0,04670	0,04616
uvi_activeTrue	209,63719	0,00000	0,15559	0,24433
clouds_activeTrue	-104,90515	0,00000	0,00211	0,06712
wind_speed_activeTrue	255,54842	0,12685	0,08971	0,03099
wind_deg_activeTrue	68,27564	0,00000	0,08972	0,00727
wind_gust_activeTrue	-234,57322	0,00000	0,03765	0,02896
pop_activeTrue	109,40107	0,00000	0,00466	0,00008

Variables B3	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
temp_activeTrue	-319,77017	0,12851	0,31386	0,08932
dew_point_activeTrue	-12,11154	0,20166	0,18480	0,08535
uvi_activeTrue	277,25441	0,58809	0,45649	0,73438
clouds_activeTrue	-98,65464	0,00000	0,00682	0,06728
pop_activeTrue	90,71878	0,08175	0,03805	0,02367

Del Excel C:

Variables C1	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
dew_point_activeTrue	-170,82367	0,24815	0,24393	0,10582
uvi_activeTrue	129,85092	0,49092	0,40696	0,55171
wind_deg_activeTrue	-35,70301	0,06066	0,15692	0,25236
wind_gust_activeTrue	-22,34274	0,20027	0,19218	0,09011

Variables C2	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
diff_temp_interior	-805,16538	0,65566	0,34591	0,04558
diff_temp_int_ext	632,52794	0,00000	0,05549	0,01728
temp_activeTrue	-645,45625	0,00000	0,05907	0,36240
humidity_inicial	-52,86949	0,00000	0,07826	0,06581
pressure_activeTrue	-6,99936	0,00000	0,02030	0,00672
humidity_activeTrue	811,08500	0,17776	0,06479	0,00004
dew_point_activeTrue	-531,66373	0,00361	0,05283	0,02260
uvi_activeTrue	204,86232	0,00000	0,14940	0,31192
clouds_activeTrue	-104,83075	0,00000	0,00292	0,05514
wind_speed_activeTrue	314,89786	0,12108	0,05931	0,09100
wind_deg_activeTrue	65,78572	0,04189	0,07330	0,00726
wind_gust_activeTrue	-285,13331	0,00000	0,03274	0,01427
pop_activeTrue	96,99822	0,00000	0,00570	0,00000

Variables C3	Regression coefficients	Importance Decision Tree	Importance Random Forest	Importance XGBoost
temp_activeTrue	-319,77017	0,31831	0,30295	0,08932
dew_point_activeTrue	-12,11154	0,00000	0,25964	0,08535
uvi_activeTrue	277,25441	0,60288	0,37454	0,73438
clouds_activeTrue	-98,65464	0,00000	0,01286	0,06728
pop_activeTrue	90,71878	0,07881	0,05001	0,02367