Universidad del País Vasco Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

MÁSTER UNIVERSITARIO EN

INGENIERÍA DE MATERIALES AVANZADOS

TRABAJO FIN DE MÁSTER

DESARROLLO DE UNA SOLUCIÓN APLICADA DE INTELIGENCIA ARTIFICIAL EN LA DISTRIBUCIÓN DE TAMAÑO DE PARTÍCULAS DEL CEMENTO FABRICADO EN UNA PLANTA CEMENTERA TURCA

Estudiante *Iturrioz, Aguirre, Jon Ander*
Director *Larrañaga, Espartero, Aitor*
Departamento *INGENIERIA MINERA Y METALURGICA. CIENCIA DE LOS MATERIALES*
Curso académico *<2021-2022>*

Bilbao, 1, Septiembre, 2022

RESUMEN

Los procesos de fabricación de materiales a día de hoy cada vez están más digitalizados, intengrando nuevas tecnologías que puedan permitir controlar todos los parámetros internos de la secuencia de producción, desde la extracción de los minerales hasta el producto final, asegurando sus propiedades y calidad, para lograr un material ideal para la construcción. En este caso, el material creado es un cemento Portland donde la fabricación está completamente digitalizada a través de sensores instalados a lo largo del proceso, los cuales se encargan de recoger y almacenar datos en directo dentro de una base de datos privada. Recordemos que la digitalización se basa en el desarrollo por el cual procesos analógicos y objetos físicos se convierten al formato digital, pudiendo ser controlados de una manera más sencilla a través de un ordenador. Esta digitalización, actualmente, está directamente ligada con estudios y desarrollos de inteligencia artificial (IA), que gestionan toda esa información a consiguen predecir secuencias y/o paquetes de datos nuevos, antes de que se generen, pudiendo dar la opción de adelantarse a situaciones de: mantenimiento de maquinaria, roturas de herramientas, gestión de eficiencia de procesos, fallos en la calidad en la producción de materiales, análisis de procesos complejos, etc.
En este trabajo se desarrolla una solución aplicada de IA en una planta cementera de Turquía con el fin de lograr inferir el resultado de PSD (*Particle Size Distribution*) del cemento producido al final del proceso de fabricación, analizando todos los sensores representativos que se encuentran instalados en la planta cementera. El alcance del trabajo va desde la recogida de datos de la planta a través de una API (*Application Programming Interface*), pasando por el análisis exploratorio de los mismos, llegando a desarrollar diferentes algoritmos predictivos que gestionen esos datos en busca de la mejor predicción del PSD. Todo este proceso emplea el lenguaje de programación Python, mediante el editor de código VS Code, usando diferentes librerías de extracción y transformación de datos, y de algoritmia basada en aprendizaje supervisado.

ABSTRACT

The manufacturing processes of materials today are becoming increasingly digitized, integrating new technologies that can control all the internal parameters of the production sequence, from the extraction of minerals to the final product, ensuring its properties and quality, to achieve an ideal material for construction. In this case, the material created is a Portland cement where the manufacturing is completely digitized through sensors installed throughout the process, which are responsible for collecting and storing live data in a private database. Recall that digitization is based on the development by which analog processes and physical objects are converted to digital format, being able to be controlled in a simpler way through a computer. This digitization, nowadays, is directly linked to studies and developments of artificial intelligence (AI), which manage all this information and manage to predict sequences and/or new data packages, before they are generated, being able to give the option to anticipate situations of: machinery maintenance, tool breakage, process efficiency management, quality failures in the production of materials, analysis of complex processes, etc.
In this work, an applied AI solution is developed in a cement plant in Turkey in order to infer the PSD (Particle Size Distribution) result of the cement produced at the end of the manufacturing process by analyzing all the representative sensors installed in the cement plant. The scope of the work goes from the collection of data from the plant through an API (Application Programming Interface), through the exploratory analysis of the data, to the development of different predictive algorithms that manage these data in search of the best prediction of the PSD. This whole process uses the Python programming language, through the VS Code editor, using different libraries for data extraction and transformation, and algorithms based on supervised learning.

LABURPENA

Gaur egun materialen fabrikazio-prozesuak gero eta digitalizatuago daude, ekoizpen-sekuentziaren barne-parametro guztien kontrola ahalbidetu dezaketen teknologia berriak integratuz, mineralak erauztetik azken produkturaino, bere propietateak eta kalitatea bermatuz, eraikuntzarako material aproposa lortzeko. Kasu honetan, sortutako materiala Portland zementua da, non fabrikazioa guztiz digitalizatuta dagoen prozesu osoan instalatutako sentsoreen bidez, datu-base pribatu batean zuzeneko datuak biltzeaz eta gordetzeaz arduratzen direnak. Gogora dezagun digitalizazioa prozesu analogikoak eta objektu fisikoak formatu digitalera bihurtzen dituen garapenean oinarritzen dela, ordenagailu baten bidez modu sinpleago batean kontrolatu ahal izateko. Digitalizazio hori zuzenean lotuta dago adimen artifizialaren (AA) ikerketekin eta garapenekin, informazio hori guztia kudeatzen baitute sekuentzia eta datu-pakete berriak aurreikusteko, sortu baino lehen, honako egoerak aurreikusteko aukera emanez: makineria mantentzea, tresnen haustura saihestea, prozesuen eraginkortasuna kudeatzea, materialen ekoizpenean kalitate akatsak, prozesu konplexuen azterketa, etab.
Lan honetan, Turkiako zementu planta batean aplikatutako AA irtenbide bat garatzen da, fabrikazio-prozesuaren amaieran ekoitzitako zementuaren PSD (*Particle Size Distribution*) emaitza ondorioztatzeko, sentsore adierazgarri guztiak aztertuz. zementu plantan instalatzen dira. Lanaren esparrua API baten bidez (*Application Programming Interface*) bidez lantegiko datuak biltzetik, horien azterketa esploratzailera, datu horiek kudeatzen dituzten algoritmo prediktibo ezberdinen garapenera iritsiz. PSDren iragarpen onena. Prozesu oso honek Python programazio lengoaia erabiltzen du, VS Code kode editorearen bidez, datuen erauzketa eta eraldaketa liburutegi desberdinak eta ikaskuntza gainbegiratuan oinarritutako algoritmoak erabiliz.

Índice general

Índice de figuras

Índice de tablas

Acrónimos

Tabla 1: Tabla de acrónimos empleadas en este trabajo.

ACRÓNIMO	NOMBRE COMPLETO
IA	Inteligencia Artificial
ML	Machine Learning
PSD	Particle Size Distribution
VPN	virtual private network
HTTP	Hypertext Transfer Protocol
API	Aplication Programming Intergace
IP	Internet Protocol
HIL	Hardware In the Loop
OP	Objetivos Teóricos
PLC	Programmable logic controller
ETL	Extract Transform Load
SQL	Structured Query Language
CRM	Customer Relationship Management
EDA	Exploratory Data Analysis

Capítulo 1

Memoria

1.1. Introducción

El cemento es un material muy cotizado en todo el mundo por su gran usabilidad en muchos ámbitos de la construcción además de tener una de las mejores relaciones entre el coste del material y sus propiedades. Es realmente barato para todos los beneficios que aporta a nivel estructural comparado con otros materiales con características similares. Este material es creado a través de plantas cementeras donde el proceso comienza desde la extracción de las materias primas hasta la producción final del propio cemento. Estas plantas localizadas alrededor de todo el planeta comparten algunos de los problemas referentes a su proceso de producción. En general, todas ellas sufren carencias en los mismos puntos de la secuencia de fabricación, generando gastos monetarios excesivos por culpa de parones de mantenimiento imprevistos, calidad de producto final insuficiente, consumo excesivo de agua para la refrigeración del material, etc, todo ello por no disponer de tecnologías que ayuden a rectificar el proceso a tiempo. En este trabajo se planea desarrollar una solución digital, que sea capaz de automatizar una parte del proceso de producción del cemento, pudiendo así optimizar el resultado de calidad final del producto, mediante el uso de tecnologías digitales basadas en desarrollos de Inteligencia Artificial (IA).
Para poder desarrollar una solución de IA aplicada, el primer paso es identificar cual es el problema que nos interesa resolver y no confundir el problema con la solución, es decir, si la idea inicial se basa en "el problema es que no tenemos datos", en realidad no estamos definiendo el problema, sino que mostramos un sesgo de cual creemos que debería ser la solución. Por otro lado, sucede algo semejante cuando le damos demasiado valor a una tecnología o método "sofisticado" como puede ser en este caso el uso de la IA. Por muy privilegiado que pueda ser emplear una técnica de este calibre, nuestra perspectiva del problema no debe volverse errónea y no se debe dar por hecho que siempre será la mejor opción. Como decía Maslow (1966) "si sólo tienes un martillo, todo parece un clavo". Por ello, se desarrollan en el tiempo varias reuniones con el cliente (Planta cementera de Turquía), con el fin de poder comprender el alcance del problema en base a su experiencia, hablando desde sus encargados y comerciales hasta los propios operarios de planta. Este punto es crucial porque muchas veces la información recibida por un comercial es muy sesgada, y en ocasiones, lejana a la realidad, por eso, se decide hablar también con operarios del día a día, los cuales comprenden a la perfección las limitaciones del proceso. El fin de invertir tiempo en este punto es el de acotar perfectamente cual es el problema real, definir todas las variables que influyen en el proceso y plantear cual debería ser la mejor solución aplicable.

1.2. Contexto

El trabajo actual es un trabajo de fin de máster, el cual ha sido llevado a cabo desde las instalaciones de Tecnalia, un centro de investigación y desarrollo tecnológico de referencia en Europa, con 1.400 expertos de 30 nacionalidades diferentes, centrados en transformar la tecnología en PIB mejorar la calidad de vida de las personas, creando oportunidades de negocio para las empresas. Sus principales ámbitos de actuación son: transformación digital, fabricación avanzada, energía, transición, movilidad sostenible, ecosistema urbano y salud.

El máster que se ha cursado para llegar a este punto (Master de Ingeniería de Materiales Avanzados de la Escuela de Ingeniería de Bilbao) ofrece una sólida formación en el área de la Ingeniería de Materiales Avanzados, combinando la adquisición de conocimientos sobre materiales metálicos, poliméricos, cerámicos, compuestos y funcionales, pudiendo adquirir las habilidades necesarias para gestionar sus aplicaciones en entornos reales. El máster está orientado a formar profesionales de la ingeniería en entornos aplicados a la obtención/fabricación, estructura/propiedades, selección y diseño de los materiales ingenieriles.
Por esta razón, y fusionando ambas entidades, se ha desarrollado el trabajo enfocando la investigación en crear una solución aplicada a un problema real del cliente, en este caso, una planta cementera Turca, pudiendo optimizar la eficiencia de su proceso de fabricación de cemento, reduciendo costes en la fabricación y logrando tener una calidad superior en su producto final. El desarrollo se basa en controlar digitalmente el proceso de obtención del cemento para poder lograr mejores propiedades.

1.3. Hipótesis

Este trabajo está orientado a validar la siguiente hipótesis mediante un trabajo de investigación de recogida y análisis de datos, elaborando una propuesta práctica e innovadora como respuesta a una necesidad detectada.

Las técnicas de IA son capaces de predecir la Distribución del Tamaño de Partículas (Particle Size Distribution, PSD).

Para validar la hipótesis se procede a definir una serie de objetivos.

1.4. Objetivos y alcance del proyecto

1. Solucionar la problemática referente al PSD del proceso de producción de una planta cementera, generando mayor valor añadido al producto del cliente, aplicando tecnologías de IA.

2. Estudiar cómo extraer los datos necesarios de su servidor, además del formato de los mismos para comprender que tipo de análisis aplicar.

3. Desarrollar diferentes soluciones de IA dependiendo del enfoque que se haya decidido emplear a modo de resolución. Compararlas entre sí para ver cuál es la mejor.

1.5. Estrategias para alcanzar objetivos

La estrategias planteadas son las siguientes:
- Estudiar las necesidades del cliente con el fin de acotar lo mejor posible el problema y ejecutar la solución más ajustada.

- Evitar generar riesgos adicionales dentro del desarrollo del proyecto, ya que actualmente existen riesgos referentes a errores en la extracción de los datos, que los algoritmos predictivos no tengan la precisión suficiente en sus inferencias, no tener los recursos necesarios para el correcto desarrollo de los algoritmos (procesamiento suficiente para entrenar las redes neuronales por ejemplo).

- Establecer un plazo de resolución de 5 meses desde la recepción de los datos de la planta cementera. No es necesario planificar cada paso a través de un cronograma.

1.6. Beneficios que aporta el trabajo

Los beneficios que aporta este proyecto son variados, entre los cuales se encuentran:

- Poder llegar a automatizar una parte del proceso de producción, haciéndolo controlable digitalmente, logrando almacenar y tratar información de lo que sucede a lo largo del proceso de manera más precisa. Este punto se puede extrapolar a otras partes de la planta cementera, pudiendo desarrollar finalmente un gemelo digital que simule todas las partes de la obtención del cemento, recabando datos de cada una de ellas, y que mediante IA, puedan desarrollarse modelos predictivos que optimicen todas esas secuencias de fabricación, anteponiéndose a paradas imprevistas del proceso por mantenimiento, errores de configuración de la composición del material u otros errores.

- Actualizar el conocimiento en los actuales puestos de trabajo, pudiendo formar a la plantilla en nuevas tecnologías innovadoras, como es el conocimiento sobre aplicaciones de IA. Esto genera que la empresa pueda disponer de nuevas capacidades profesionales manteniendo su plantilla inicial, pudiendo incluso generar nuevas oportunidades laborales que incorporen este tipo de expertos.

- Poder crear controles de calidad más rigurosos con menor esfuerzo por parte de los técnicos, evitando así el error humano, aumentando el ratio de mejora del producto.

- Usar los datos recopilados se puede llegar a obtener simulaciones de IA que predigan nuevas propiedades en los materiales actuales, e incluso nuevos materiales.

Capítulo 2

Inteligencia artificial y PSD en la producción de cemento

2.1. Inteligencia artificial

La IA es la habilidad de una máquina de presentar las mismas capacidades que los seres humanos, como el razonamiento, el aprendizaje, la creatividad y la capacidad de planear. Permite que los sistemas tecnológicos perciban su entorno, se relacionen con él, resuelvan problemas y actúen con un fin específico. La máquina que tenga IA integrada recibe datos (ya preparados o recopilados a través de sus propios sensores, por ejemplo, una cámara), los procesa y responde a ellos, prediciendo comportamientos u otras acciones que lo convierten en un sistema inteligente ante sucesos que no siguen un patrón específico, adaptando su comportamiento para trabajar de manera autónoma.

El principio fundamental de la IA es replicar, y luego superar, la forma en que los humanos perciben y reaccionan ante el mundo de manera que se está convirtiendo rápidamente en la piedra angular de la innovación. La IA, impulsada por varias formas de machine learning que reconocen patrones en los datos para permitir predicciones, puede agregar valor a cualquier negocio en torno a:

- Proporcionar una comprensión más completa de la abundancia de datos disponibles.
- Confiar en las predicciones para automatizar tareas excesivamente complejas o mundanas.

En este trabajo, se implementan desarrollos de IA por su versatilidad a la hora de resolver problemas complejos, que de una manera más tradicional no podrían haberse solucionado, aplicando metodologías referentes a esta temática para predecir la PSD de una fábrica de cemento, pudiendo rectificar los parámetros de la elaboración del producto final antes de que sea creado.

2.2. Descripción breve del funcionamiento de la planta cementera

Para comenzar con la descripción se muestra en la Figura 2.1 el proceso simplificado de una planta cementera convencional:

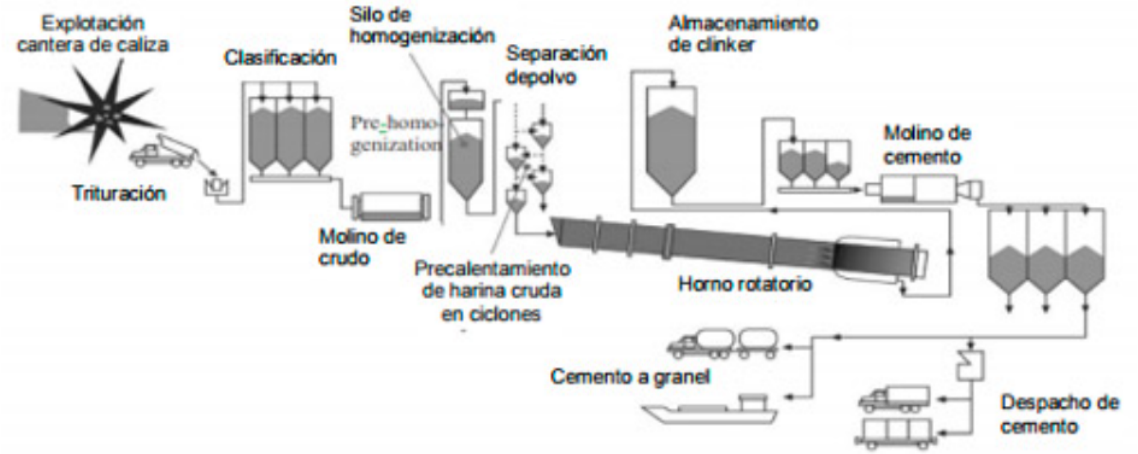


Figura 2.1: Proceso de obtención del cemento en una planta cementera industrial.

La industria cementera procesa como materia prima materiales como la piedra caliza, la arcilla y el hierro, mezclándolas entre sí a través de un proceso de trituración y molienda. La mezcla molida se homogeniza y se lleva a un proceso de pre calcinación que después ingresa a un horno rotativo, en el que se desarrollan reacciones físico-químicas, calcinándose la mezcla hasta el punto fundente, donde alcanza temperaturas superiores a 1500C, destruyendo completamente todos los compuestos orgánicos y consiguiendo así que las trazas de metales pesados se integren en la nueva estructura originada llamada Clinker.

- El horno.
En esta primera etapa del proceso, los componentes reaccionan y forman el clinker, precursor del cemento. El proceso del horno consta de dos sub-etapas: la etapa de pre-calcinación que tiene lugar en los ciclones y la etapa de clinkerización. El Clinker se somete a un proceso de enfriamiento rápido con el fin de cristalizar el material, explicado a continuación:

- El enfriador.
Segunda etapa del proceso global de producción de cemento, el enfriador contiene diferentes elementos para reducir el calor del clinker cuando sale del horno. En esta etapa, la velocidad de enfriamiento afecta a la cristalización de los componentes dentro del clinker y desempeña un papel clave en la calidad y la triturabilidad en las siguientes etapas del proceso.

Posteriormente se muele agregando una pequeña proporción de yeso (sulfato de calcio) y envasando el resultado en paquetes o sacas, dependiendo de su futura aplicación. Este producto final es denominado cemento, el cual tiene un formato pulverizado, con diferentes distribuciones de partícula (PSD) en su composición interna. Este parámetro es el que se va a predecir a lo largo de este proyecto.

- La molienda.
El proceso de molienda implica la trituración del clinker enfriado para obtener el producto final. La molienda se realiza en un molino horizontal de bolas con dos diafragmas que se colocan en el centro y en la salida del molino y que actúan como tamices y forman cámaras que contienen diferentes tamaños de clinker molido, lo que ayuda al proceso de molienda. El molino se enfría mediante la pulverización de agua. Para ello se necesitan unos 5 m3/h de agua, que dependen también de la temperatura del clinker.

2.3. Sistema actual de recogida de datos

En esta planta cementera hay muchos sensores instalados en los procesos de horno, enfriador y molino para medir diferentes parámetros de calidad. Sin embargo, en algunos lugares críticos, no se dispone de dichos dispositivos y no es posible dar retroalimentación al sistema, por lo que en esas zonas, las mediciones

se realizan manualmente por el departamento de calidad, teniendo que extraer muestras puntuales del proceso, y analizando las mismas lo más rápido posible. Las zonas donde sí que hay sensórica instalada, recogen los datos cada medio segundo, que son almacenados en directo, fuera de línea, en una base de datos relacional con un control de redundancia para garantizar que no contenga duplicados o valores NaN (*Not Available Number*).

2.4. Estimación de cal libre en el Clinker de cemento

En el proceso de producción de cemento descrito anteriormente, la cantidad de cal libre en el clinker de cemento es uno de los factores cruciales que afectan a la calidad del producto de cemento. La cantidad de cal libre en el clinker de cemento está relacionada con varios parámetros, que incluyen, entre otros, la materia prima, el funcionamiento del horno y el proceso de combustión.
La materia prima se almacena en silos de crudo después de reducir su tamaño mediante el proceso de trituración. La harina cruda se calienta hasta 700C y entra en el molino de rodillos cuando comienza la calcinación. A continuación, los materiales son calentados por el quemador y la clinkerización se completa a 1300C - 1600C en el horno de rodillos. La cantidad de combustible quemado varía entre 5-20 toneladas/h. El resultado del proceso descrito es el clinker de cemento.

La cantidad de cal libre en el clinker de cemento se mide actualmente mediante un proceso químico, trasladando las muestras al laboratorio para su análisis manual, utilizando los siguientes 2 métodos:
- Análisis de cal libre con el método del etilenglicol.
- Análisis de cal libre con el método del acetato de amonio.
Aunque es una parte importante del proceso, en este trabajo no se calculan predicciones referentes a este punto. Solamente se prevé inferir datos referentes al PSD en base a los datos recogidos de manera automática en la planta cementera.

2.5. Estimación de la distribución del tamaño de las partículas. PSD

En el caso de uso del cemento, uno de los retos es automatizar la manera en que se miden los valores de la distribución del tamaño de las partículas (PSD) del cemento producido para que sea posible actuar con mayor rapidez para mantener los estándares de calidad. La previsión es poder actuar y modificar el proceso antes de que ese material sea producido, modificando los actuadores de la planta cementera basándose en el análisis predictivo de los sensores de todo el proceso. La manera de medir el PSD de planta actualmente se basa en mediciones manuales por parte de sus técnicos de laboratorio, tomando una muestra cada cuatro horas. Esta metodología es un proceso que requiere mucho tiempo, ya que sólo cuando se obtienen los resultados del laboratorio, los operarios pueden actuar sobre los actuadores para recuperar valores óptimos del producto. La única forma de medir la PSD en tiempo real es instalando un costoso analizador de PSD en línea, que no todas las plantas de cemento pueden permitirse.
En este caso, la planta cementera ha decidido instalar dicho medidor, pudiendo implementar la recogida automática de los valores de PSD en 7 distribuciones de tamaño diferentes, totalmente en directo.

Estudios recientes, [1] demuestran las capacidades de los algoritmos de IA para crear un sensor virtual (explicado en los siguientes puntos) que sea capaz de estimar la PSD. Un sensor virtual puede hacer una estimación de la PSD esperada basándose en las medidas automáticas recopiladas por los sensores a lo largo de la línea de producción de cemento [2]. Esto puede dar al operador del molino información relevante para ajustar los parámetros de operación de una manera más rápida para asegurar la producción de cemento de calidad consistente. Por ello, en este trabajo se desarrollará un sensor virtual que estime los valores esperados de PSD en función de las variables del proceso. Este sensor blando se basará en uno o varios algoritmos de aprendizaje automático o Machine Learning (ML) y se validarán utilizando los datos del analizador de PSD, el cual devuelve de manera automática valores reales del resultado final.

2.6. Análisis del sistema de sensores actual

Dado que el modelado de un sensor blando de PSD requiere datos de diferentes partes de la línea de producción de cemento, se ha realizado un estudio de los actuales sistemas de sensores implementados dentro de la planta de cemento con el fin de detectar los sensores que proporcionarán los datos más significativos. En esta investigación, se han analizado dos secciones críticas de la línea de producción de cemento: el horno y el molino.

La razón de centrarse en estas dos partes es doble. Por un lado, el conocimiento experto de los operadores de la planta de cemento sugiere centrarse en estas dos secciones, y por otro lado, los trabajos de investigación estudiados utilizan los sensores de estas dos líneas como entradas para el sensor virtual.
Como se ha sugerido anteriormente, el sistema actual de adquisición de datos está compuesto por un conjunto de diferentes controladores lógicos programados instalados que capturan señales analógicas y digitales.
Además, existe un gestor de base de datos para extraer los datos históricos de los sensores y permitir que los procesos de minería de datos analicen los datos capturados o ejecuten diferentes protocolos de optimización cuando sea necesario.

2.7. Nuevo sistema digital de sensores. Sensor Virtual

Se han analizado los trabajos de investigación más relevantes del estado del arte para determinar los valores del proceso que pueden afectar a la PSD, con el objetivo de detectar las variables del proceso que no se miden actualmente en la planta de cemento y que deben ser monitorizadas a través de nuevos sensores. Esta finalidad puede ser no necesaria, ya que si se consigue relacionar la información recogida actualmente por los sensores para poder predecir valores certeros de PSD, no será necesario instalar nueva sensórica.
La tabla (2.1) muestra los sensores más relevantes de la línea de proceso de la fábrica. Cada sensor instalado tiene su identificación interna única, su posición dentro del molino de cemento, su tipo (analógico o digital) y su unidad de medida. A modo de resumen, en las tablas solamente se describen los datos referentes a descripción del sensor e identificador único.

Tabla 2.1: Resumen de todos los sensores de los que se extraen datos para la realización de este trabajo.

CÓDIGO NUMÉRICO	**SENSOR**
3185	EXCHANGER INLET TEMPERATURE
3237	FRESH AIR
3239	FLUE GAS FAN
3264	KLS SECONDARY AIR PRESSURE
3265	KILN AT KW
3297	3.CYCLONE GAS TEMPERATURE
3300	4.CYCLONE BOTTOM SUCTION
3301	4.CYCLONE GAS TEMPERATURE MERSİN
3303	KLS BRICK TEMPERATURE MOUNTAIN SIDE
3326	KLS BRICK TEMPERATURE SEA SIDE
3361	4.CYCLONE GAS TEMPERATURE ADANA
3374	KILN TORQUE
3392	DIVIDING GATE SET POINT
3403	KLS SECONDER VALVE
3412	KILN SPEED
3443	FLUE O2
3446	TRANSITION O2
3448	TRANSITION NITROGEN
3454	CEMENT MILL OUTLET ELEVATOR
3475	MILL OUTLET CEMENT TEMP.
3476	MILL OUTLET GAS TEMP.
3479	MILL INLET PRESSURE
3480	CEMENT MILL SOUND LEVEL
3481	SEPARATOR KW SIGNAL
3482	SEPERATOR SPEED
3483	MILL OUTLET PRESSURE
3499	SEPAX FILTER TEMPERATURE
3506	SEPAX FAN KW SIGNAL
3507	CEMENT MILL MOTOR KW
3523	TOTAL FEED
3529	SPEED CONTROL
3563	FILTER INLET TEMP. MOUNTAIN
3569	CLINKER TEMPERATURE
3573	RED.BUR.WATER FLOW
3575	REDUCTION BURNER WATER
3577	WATER VALVE Z1 POSITION
3577	WATER VALVE Z1 POSITION
3585	COOLER TEMPERATUR
3597	PRESSURE BEFORE EP
3599	CY. DUST BLC. INLET THRM. SEA
3622	FLOW F01-T73PS1Z1
3625	FLOW F03-T73PS1Z3
3848	FILTER DIFF. PRESSURE
5214	KILN FEED SET POINT
5215	CALCINER COAL
5217	KILN HEAD COAL
7211	Clinker Moisture
7400	CEMENT MILL 90M
7401	CEMENT MILL 64M
7402	CEMENT MILL 48M
7403	CEMENT MILL 45M
7404	CEMENT MILL 32M
7405	CEMENT MILL 3-32M
7406	CEMENT MILL 3M
7407	CEMENT MILL BLAINE

2.8. Riesgos y contingencias

1: Es altamente probable que algunos de los sensores no midan correctamente o devuelvan datos erróneos.
Solución 1: Comprobar el comportamiento de cada uno de los sensores temporalmente para ver dónde podría haber datos erróneos y poder así tratarlos y avisar a la planta cementera de en qué zonas de la planta debería hacer un mantenimiento puntual.

2: Habrá paradas por mantenimiento, generando temporalmente huecos entre los datos.
Solución 2: Si las ventanas sin datos son pequeñas, se procederá a completar los datos faltantes con la media de valores de la serie, usando interpolaciones por ejemplo. Si por el contrario son demasiado extensas, se procederá a eliminar los datos de todos los sensores y sus respectivos resultados en esa ventana temporal, omitiendo esa falta de información para el análisis de datos.

3: Es probable que la combinación de sensores elegida no sea la óptima.
Solución 3: Se hará un análisis exploratorio de datos de todos los sensores de planta con el fin de detectar altas correlaciones y sensores inservibles, de manera que en una primera pasada quede un paquete de sensores funcional. En una segunda pasada se analizará la disposición de cada uno en planta con intención de aislar los más representativos.

4: Suponer que las fechas de recogida de cada fila de datos se pueden tomar como una serie temporal.
Solución 4: Aplicar algoritmos de resolución de Series Temporales. Estos algoritmos se basan en, recoger datos del pasado mediante una ventana temporal especificada por el usuario, y predecir valores futuros mediante otra ventana temporal, también definida por el usuario. La resolución se basa en que a través de los datos del pasado, el algoritmo aprende tendencias y patrones, y predice datos del futuro, en un rango de tiempo especificado.

5: Ignorar el hecho de tener fechas concretas por cada medición.
Solución 5: Aunque exista un momento temporal por cada medición adquirida, este caso no es un planteamiento real de análisis temporal porque en los datos de este problema no existe una dependencia temporal.

Capítulo 3

Estado del arte

3.1. Estado del arte acerca de la estimación del PSD con sensores

Todos la información recopilada a modo de ejemplo para ejecutar un desarrollo diferente, se puede observar resumida en la siguiente tabla (3.1). Se ha analizado en cada punto cuál ha sido la mejor metodología y tecnología aplicada, para poder innovar en una nueva solución que cumpla los requisitos definidos en el apartado de objetivos del apartado 1.4.

Tabla 3.1: Resumen de información recopilada

REF	DESARROLLO
[1]	Predice PSD con datos de toda la molienda, usando red neuronal convencional
[2]	Predice PSD con datos de la molienda usando una red regresora y otra recursiva
[3]	Predice PSD en molino vertical basándose en método PCA-TSS
[4]	Desarrolla un sensor blando con los sensores del molino y aplica Regresores
[5]	Predice PSD mediante procesamiento neuronal de imágenes.
[6]	Controla parámetros del molino a través de un modelo de PCA
[7]	Monitorización online del proceso de producción usando estadística multivariable
[8]	Implementa sensor blando adaptativo en el circuito de molienda
[9]	Comprueba que para predecir PSD, es mejor un modelo mixto que uno preentrenado
[10]	Desarrolla un sensor blanco y lo instala en la planta cementera

3.2. Detección del problema

El problema inicial del cliente se basa en la necesidad de conseguir una estimación real del valor de su PSD en su producto final, además de poder controlar el directo todos los actuadores que influyan en este resultado [4], pudiendo mantener una proporción óptima del tamaño de partícula del cemento en todo momento, a través de un control automático de la planta cementera. Esto conlleva poder conseguir mejor calidad del producto final además de reducir costes de fabricación, lo que genera un mayor valor añadido.

Se analiza si la solución se puede desarrollar con programación clásica basada en reglas expertas, que son nada menos que la expresión mínima de un desarrollo de inteligencia artificial, o en cambio es necesario implementar un desarrollo basado en ML, una de las ramas de la IA con implementaciones más complejas. Actualmente se está almacenando una gran cantidad de datos reales, que seguirán haciéndolo de forma indefinida, y que a futuro serán necesarios para implementar el control automático de la planta. Se estima que el algoritmo debería poder ingerir esos datos continuamente [10], y poder tomar decisiones por su propia cuenta, basándose en el resultado que debe conseguir, y en los valores que esté analizando en todo momento en el proceso. Además, debería ser un modelo flexible, que sea capaz de adaptarse a condiciones variables del entorno, siendo capaz de reaccionar ante situaciones inesperadas.

En la Figura 3.1 se puede observar la comparativa entre el procedimiento de programación tradicional vs IA.

Figura 3.1: Comparación entre procedimiento de programación tradicional e inteligencia artificial.

Mientras que el primero se basa en procesar datos a través de unas reglas definidas y estáticas para conseguir un resultado (proceso cíclico), el segundo ingiere datos y resultados estimados para poder tomar decisiones que sigan unas normas (proceso adaptativo).

3.3. Posibles soluciones

Después de plantearle al cliente diversas soluciones dentro de cada una de las dos opciones anteriores (Programación tradicional *vs.* IA), se ha decidido conjuntamente que la mejor respuesta a este problema ha de basarse en el desarrollo de algoritmia con un sistema de IA integrado, siguiendo la metodología de un sistema HIL (*Hardware In the Loop*), también denominado Sensor Blando [8], que sirva para controlar todas las variables de proceso digitalmente, y sea capaz de adaptarse a las condiciones de entorno en todo momento. Este sistema se explica a continuación.

Para poder hacer cálculos y análisis de variables sin influir en el rendimiento de la planta cementera, se procede a seguir el procedimiento HIL. Es una técnica en la que las señales reales de un controlador o dispositivo digital, como en este caso, los sensores instalados en planta, son conectadas a un sistema de pruebas que simula la realidad, engañando al controlador para que piense que está en el producto físico real. La prueba y la iteración del diseño se realiza como si se estuviera utilizando el sistema del mundo real de manera que se puedan ejecutar fácilmente miles de escenarios posibles para poner a prueba los elementos del proceso, pudiendo estudiar el comportamiento de los mismos sin influir en el coste y el tiempo asociados con las pruebas físicas de la actualidad.

En el caso del proyecto actual, se ha creado un Sensor Virtual, también conocido como Sensor Blando, el cual es un modelo digital inferencial que copia el comportamiento del proceso en la vida real, utilizando variables fáciles de medir para estimar variables de proceso que son difíciles de medir debido a limitaciones tecnológicas, grandes retrasos en la medición o altos costes de inversión. Mediante este método, es posible eliminar variables redundantes, reducir la complejidad del modelo y mejorar la precisión del mismo mediante el uso de técnicas apropiadas de selección de variables y procesamiento de datos [1]. Tal y como se ha explicado en el planteamiento del problema, el uso de estos datos se emplea en uso del sensor blando para poder estimar la distribución del tamaño de partícula (PSD) de la salida del molino y su correspondiente automatización de proceso en planta de fabricación [2], [3], [9].

Capítulo 4

Fundamentos teóricos

4.1. Proceso ETL: *Extract-Transform-Load*

El proceso ETL de datos se basa en extraer datos relevantes para un análisis específico, transformarlos a un formato concreto que depende de la tipología de dato y el problema que se quiera solucionar, y cargarlos en una base de datos que facilite el acceso y almacene toda la información de una manera más correcta.

4.1.1. Extracción

Tal y como se ha explicado en la introducción, el primer paso en el proceso es extraer los datos de todas las fuentes relevantes y compilarlos en un único directorio. Las fuentes de datos pueden incluir datos de múltiples fuentes: bases de datos de instalaciones, Sistemas CRM (*Customer Relationship Management*), plataformas de automatización de marketing, almacenes de datos en la nube, archivos estructurados (bases de datos en excel, o siendo más profesional: SQL(*Structured Query Language*) y no estructurados (emails y pdf), aplicaciones en la nube y cualquier otra fuente de la que desee obtener información a través del procesamiento analítico.
Una vez que se hayan consolidado todos los datos críticos, deben organizarse de acuerdo con la fecha, el tamaño del dato y la fuente de la que se ha extraído, para poder adaptarse al proceso de transformación con más facilidad. Es importante que se mantenga un cierto nivel de coherencia en todos los datos que se introducen en el sistema. No tiene sentido recopilar datos sin un sentido explícito respecto al problema a resolver. La complejidad de este paso puede variar significativamente, según los tipos de datos, el volumen de datos y las fuentes de datos.

4.1.2. Transformación

En esta parte del proceso, los datos extraídos de las fuentes se compilan, convierten, reformatean y limpian mediante diferentes métodos de programación, para posteriormente alimentar la base de datos de destino que los va a almacenar.
El paso de transformación implica ejecutar una serie de funciones y aplicar conjuntos de reglas a los datos extraídos para convertirlos en un formato estándar que cumpla con los requisitos del esquema de la base de datos mencionada. El nivel de manipulación requerido en Transformación ETL depende únicamente de los datos extraídos y de las necesidades del problema a resolver. Incluye la validación de datos y el rechazo si no son aceptables.

Cabe mencionar que en ciertas ocasiones no se desarrolla una transformación profunda de los datos, dándole más importancia a la recopilación de los mismos, dejando el tratamiento para más adelante. Este es el caso de este trabajo, donde los datos que se han almacenado en la base de datos de la planta cementera han sido insertados con un formato definido, pero sin una estructura ordenada ni procesada.

4.1.3. Carga de los datos

El paso final es cargar los conjuntos de datos extraídos y tratados en la base de datos escogida como almacén. Hay dos formas de hacerlo:

- La primera es una rutina de inserción de SQL que implica la inserción manual de cada registro en cada fila de la tabla de la base de datos de destino.

- La segunda es una rutina de carga masiva de datos (caso actual del trabajo).

La inserción de SQL puede ser lenta, pero realiza controles de calidad de datos con cada entrada. Si bien la carga masiva es mucho más rápida para cargar grandes cantidades de datos, no considera la integridad de los datos para cada registro. La carga masiva es ideal para conjuntos de datos de los que está seguro que no tienen errores (aunque no siempre se cumple esta condición).

En este caso, los datos hacen alusión a las medidas que dan los sensores repartidos por toda la planta cementera durante el proceso de fabricación del cemento final, los cuales son recopilados y almacenados en su base de datos de forma masiva por la propia empresa que fabrica el cemento. Para poder desarrollar una solución al problema presentado en este trabajo, esos datos se extraen de nuevo de la base de datos y se les aplica el proceso de transformación correspondiente para generar un dataset limpio y estructurado que se pueda utilizar en entrenamientos y validación de redes neuronales. Ese trabajo corresponde al autor de este trabajo.

4.2. Proceso EDA

El análisis exploratorio de datos (*Exploratory Data Analysis*) se refiere al conjunto de técnicas estadísticas cuyo objetivo es explorar, describir y resumir la naturaleza de los datos y comprender las relaciones existentes entre las variables de interés, maximizando la comprensión del conjunto de datos. El EDA es utilizado por los científicos de datos para analizar e investigar conjuntos de datos y resumir sus principales características, empleando a menudo métodos de visualización de datos, lo que permite a los científicos de datos descubrir patrones, detectar anomalías y probar una o varias hipótesis. Los científicos de datos pueden utilizar el análisis exploratorio para garantizar que los resultados que generan sean válidos y aplicables a las conclusiones y objetivos de resolución deseados. Una vez se ha completado el EDA y se ha extraído la información útil, sus características pueden utilizarse para un análisis o modelado de datos más complejo, dando paso al desarrollo de algoritmos de ML.

4.3. Machine Learning: Aprendizaje supervisado

Dentro del mundo de la inteligencia artificial, cada vez hay más vertientes que abarcan nuevos ámbitos de desarrollo inteligente. La figura 4.1 muestra los 3 principales grupos de los que parte toda la inteligencia artificial, con algún ejemplo de las tecnologías internas de cada grupo.

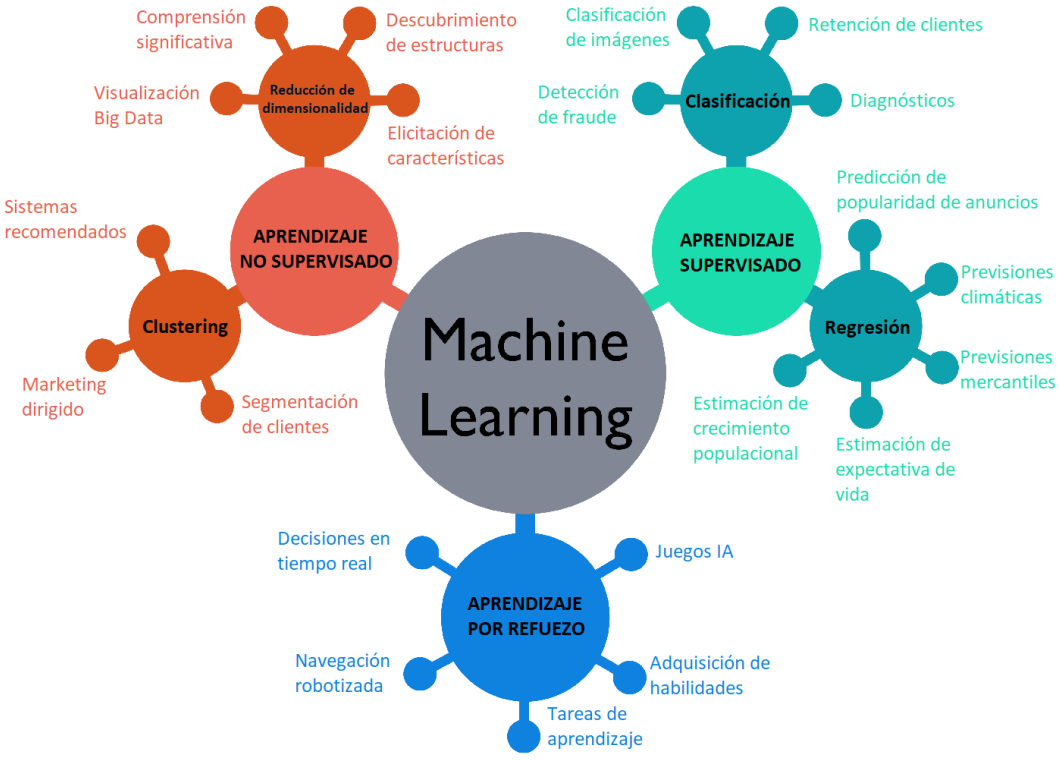


Figura 4.1: Principales grupos de los que parte la inteligencia artificial.

En este trabajo, teniendo en cuenta que los datos obtenidos tienen un resultado asignado a cada medición (una máquina de medición de PSD se encarga de medir en directo las diferentes distribuciones de tamaño de partícula, que es el parámetro que se quiere predecir en este trabajo), la rama de la inteligencia artificial que se va a usar es Aprendizaje Supervisado por el método de regresión.

En el aprendizaje supervisado, los algoritmos trabajan con datos "etiquetados" (*labeled data*), intentado encontrar una función que, dadas las variables de entrada (*input data*), les asigne la etiqueta de salida adecuada. El algoritmo se entrena con un "histórico" de datos y así "aprende" a asignar la etiqueta de salida adecuada a un nuevo valor, es decir, predice el valor de salida.

En los algoritmos de aprendizaje supervisado, se pueden solucionar dos tipos de problemas:

- **Problemas de clasificación:** identificación de dígitos, diferenciación entre diferentes clases (perro, gato, coche), etc.

- **Problemas de regresión:** predicciones meteorológicas, de expectativa de vida, el resultado de rugosidad de una muestra de fabriación aditiva, etc.

A modo visual, se representan los algoritmos de Aprendizaje Supervisado más empleados en todo tipo de problemas, cada uno de ellos con sus pros (figura 4.2) y sus contras (figura 4.3). Posteriormente se procede a seleccionar cuáles se podrán emplear de manera más eficiente en este caso.

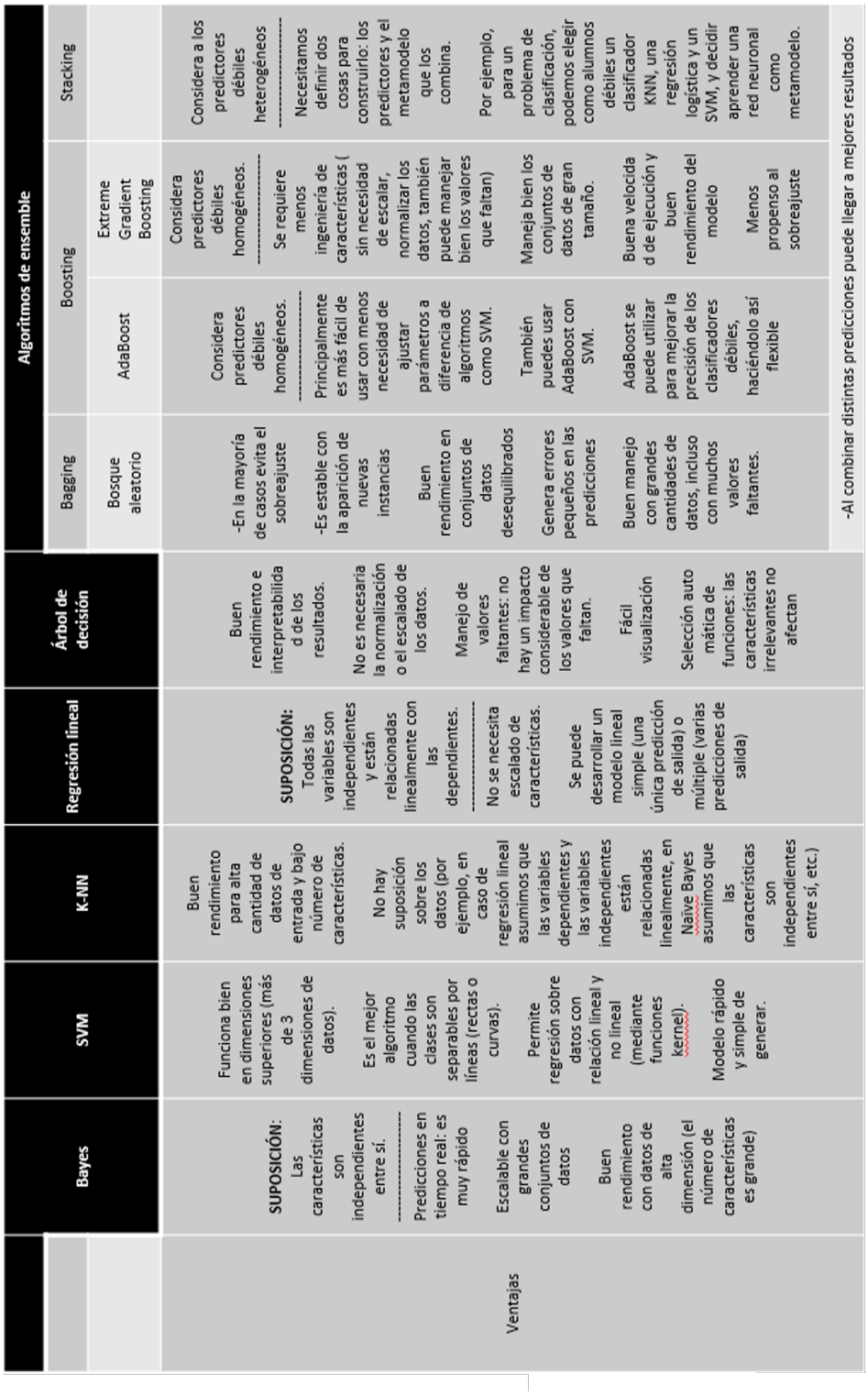
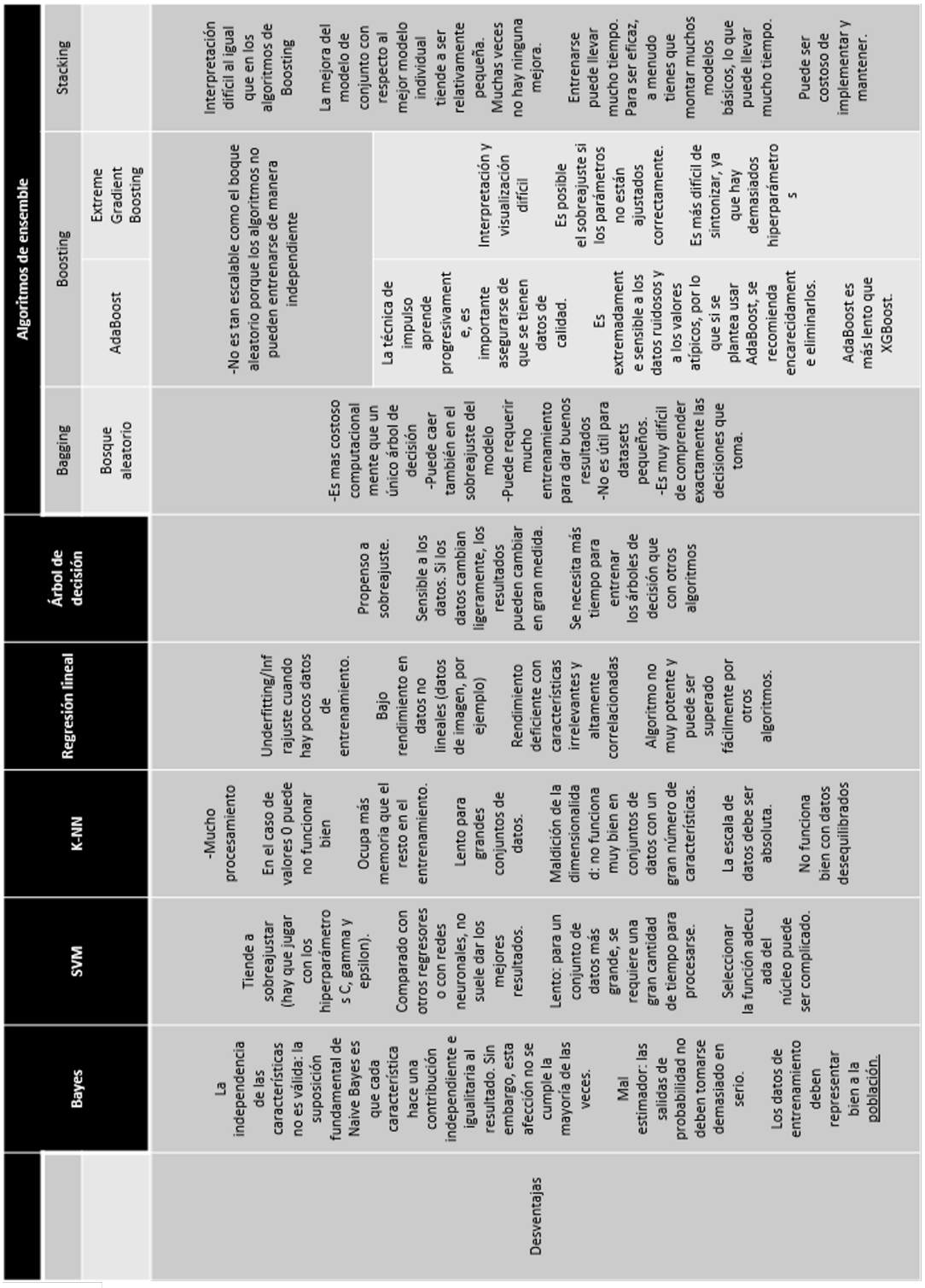


Figura 4.2: Ventajas entre modelos de ML Supervisado.



	Bayes	SVM	K-NN	Regresión lineal	Árbol de decisión	Algoritmos de ensemble			
						Bagging	Boosting		Stacking
						Bosque aleatorio	AdaBoost	Extreme Gradient Boosting	
Desventajas	La independencia de las características no es válida: la suposición fundamental de Naive Bayes es que cada característica hace una contribución independiente e igualitaria al resultado. Sin embargo, esta afección no se cumple la mayoría de las veces. Mal estimador: las salidas de probabilidad no deben tomarse demasiado en serio. Los datos de entrenamiento deben representar bien a la población.	Tiende a sobreajustar (hay que jugar con los hiperparámetros C, gamma y epsilon). Comparado con otros regresores o con redes neuronales, no suele dar los mejores resultados. Lento: para un conjunto de datos más grande, se requiere una gran cantidad de tiempo para procesarse. Seleccionar la función adecuada del núcleo puede ser complicado.	-Mucho procesamiento En el caso de valores 0 puede no funcionar bien Ocupa más memoria que el resto en el entrenamiento. Lento para grandes conjuntos de datos. Maldición de la dimensionalidad: no funciona muy bien en conjuntos de datos con un gran número de características. La escala de datos debe ser absoluta. No funciona bien con datos desequilibrados	Underfitting/Inf rajuste cuando hay pocos datos de entrenamiento. Bajo rendimiento en datos no lineales (datos de imagen, por ejemplo) Rendimiento deficiente con características irrelevantes y altamente correlacionadas Algoritmo no muy potente y puede ser superado fácilmente por otros algoritmos.	Propenso a sobreajuste. Sensible a los datos. Si los datos cambian ligeramente, los resultados pueden cambiar en gran medida. Se necesita más tiempo para entrenar los árboles de decisión que con otros algoritmos	-Es mas costoso computacionalmente que un único árbol de decisión -Puede caer también en el sobreajuste del modelo -Puede requerir mucho entrenamiento para dar buenos resultados -No es útil para datasets pequeños. -Es muy difícil de comprender exactamente las decisiones que toma.	-No es tan escalable como el boque aleatorio porque los algoritmos no pueden entrenarse de manera independiente La técnica de impulso aprende progresivamente, es importante asegurarse de que se tienen datos de calidad. Es extremadamente sensible a los datos ruidosos y a los valores atípicos, por lo que si se plantea usar AdaBoost, se recomienda encarecidamente eliminarlos. AdaBoost es más lento que XGBoost.	Interpretación y visualización difícil Es posible el sobreajuste si los parámetros no están ajustados correctamente. Es más difícil de sintonizar, ya que hay demasiados hiperparámetros	Interpretación difícil al igual que en los algoritmos de Boosting La mejora del modelo de conjunto con respecto al mejor modelo individual tiende a ser relativamente pequeña. Muchas veces no hay ninguna mejora. Entrenarse puede llevar mucho tiempo. Para ser eficaz, a menudo tienes que montar muchos modelos básicos, lo que puede llevar mucho tiempo. Puede ser costoso de implementar y mantener.

Figura 4.3: Desventajas entre modelos de ML Supervisado.

4.4. Procedimiento de ejecución y definición de variables

El primer paso es, a través de un proceso ETL, recabar los datos necesarios para poder almacenar toda la información necesaria referente al PSD. Los datos se recogen a través de una API conectada a la base de datos de la planta cementera Turca. A través de la API se hace una petición de recogida de datos en un rango temporal especificado, de manera que automáticamente se descarguen los ficheros .csv correspondientes a las medidas de los sensores instalados en planta. Una vez obtenidos todos los archivos de mediciones en las fechas seleccionadas, es necesario filtrar las variables no influyentes en el resultado final, eliminándolas de la base de datos que se va a emplear en el entrenamiento del modelo predictivo para evitar tener ruido innecesario.
Para saber cuáles de las variables no son válidas, se emplea una técnica llamada análisis exploratorio de datos (EDA), que unida a la experiencia del desarrollador de este proyecto, conduce a tomar la decisión crítica de qué sensores son representativos o no en este proceso de inferencia (como es el caso del nivel de ruido del molino de la planta cementera. No es una variable que aporte información relevante a la resolución de este problema en concreto, por lo tanto, se elimina de la base de datos de valores).
Un ejemplo simplificado de este proceso se muestra a continuación:

1. **Comprender el funcionamiento de la planta cementera.** Como primera instancia, es necesario comprender al completo qué sucede a lo largo de todo el proceso de la planta cementera, controlando todas las variables que interactúan y entendiendo todos los procedimientos que sigue el material en todo el recorrido. También es importante conocer en qué partes de la planta se localizan los sensores que se van a emplear en esta investigación, pudiendo filtrar más adelante, en base a los resultados obtenidos en la correlación de variables, si hay sensores dependientes entre sí en la misma zona de trabajo, o cada uno corresponde a una zona diferente. Por otra parte, el hecho de conocer cada sensor y su ubicación, arroja información sobre qué datos se están recogiendo en qué momento, y con qué tipo de información.

2. **Entender cuál es el problema.** Comprender el alcance del problema es indispensable ya que no se podría aplicar una solución aplicada especifica a un proceso que no está bien definido. Es indispensable comprender cada detalle dentro del proceso completo, para no tener cuellos de botella o pérdidas de tiempo rehaciendo el modelo predictivo.

3. **Buscar referencias de otros trabajos realizados (bibliografía).** Para no incurrir en trabajo que ya se ha hecho y poder brindar una solución nueva al problema actual, se ha de contrastar la información actual referente a otros procesos similares (o iguales) a este, a lo largo de todo el mundo. Es necesario conocer hasta que punto se ha llegado, con qué desarrollo, porque se ha desarrollado así... de manera que basándose en soluciones ya creadas, se pueda innovar en un nuevo planteamiento, que pueda funcionar en un entorno como este.

4. **Recolectar datos a través de la API.** En el desarrollo de un modelo predictivo, el ingrediente principal son los datos, y la manera en que estos son tratados. Para ello, a través de una API, se recolectan datos reales que han sido medidos en la planta cementera. Estos datos son recogidos continuamente, en directo, y son almacenados en una base de datos siguiendo un patrón de línea temporal. Los datos recolectados son acotados en el tiempo por decisión del usuario que los recoge, de manera que pueda elegir de qué rangos temporales quiere tener datos.

5. **Hacer iteraciones de análisis de datos hasta comprender la información que arrojan esos datos.** Una vez recolectados todos los datos, se procede a desarrollar un algoritmo inicial que sea capaz de ingerir, procesar y devolver visualizaciones de funcionamiento de cada sensor, y valores más recurrentes, de manera que se pueda estudiar si en algún momento alguno de los sensores ha dejado de medir, o ha recopilado datos erróneos o ha habido algún otro tipo de problema en la planta cementera. Para ello, se analizan todas las gráficas generadas, comprobando incongruencias y eliminando todos aquellos valores que no sean representativos para el siguiente análisis.

6. **Crear un dataset estructurado y limpio de valores.** Se procede a crear un dataset que recopile datos de una serie temporal elegida por el usuario, a modo de muestra, que sirva para poder hacer un primer análisis de relación de variables. Para ello se deben eliminar todas las variables que no estén bien definidas, o sean erróneas, siguiendo el planteamiento ideado a continuación en la figura 4.4.

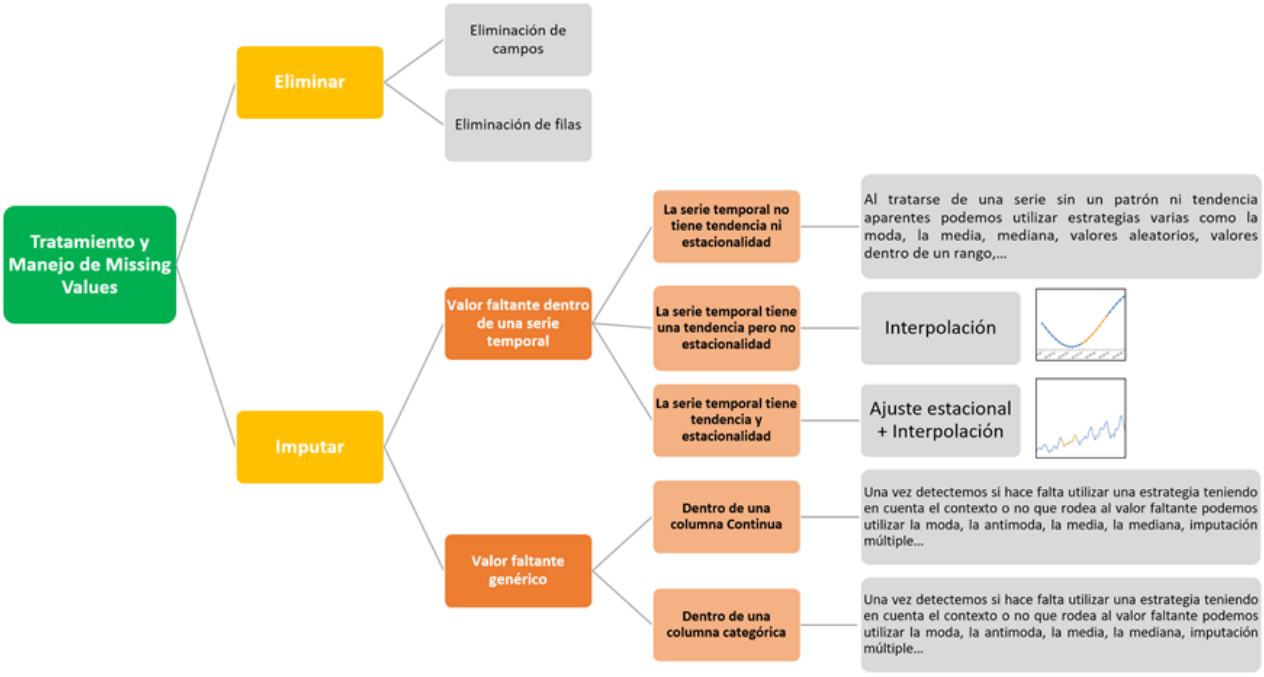


Figura 4.4: Diagrama de flujo a la hora de procesar datos faltantes en el dataset.

a) Convertir cada columna a su debido formato. Esto es completamente indispensable ya que en los lenguajes de programación, no se puede tratar una variable de tipo número entero como si fuera una cadena de texto por ejemplo, y así con todos los formatos, de manera que se ha de identificar qué formato le corresponde a cada tipo de dato, y posteriormente convertir todos y cada uno de ellos para poder gestionarlos de forma correcta. La finalidad es comprobar que el formato que cada dato es el que el editor de código es capaz de entender, como pueden ser las fechas (datetime), los datos numéricos (int o float), los datos categóricos (transformarlos a numéricos), etc.

b) Tratar inconsistencias. Muchas veces por error humano se introducen datos de forma errónea, como por ejemplo, introducir 2000 en vez de 20.000, de manera que los resultados se pueden ver afectados por este simple error. En este caso, los datos los recogen sensores de manera automática, aunque ellos tampoco son infalibles y en ocasiones también cometen errores. Por esto, todos los valores que no tengan sentido en una línea de valores completa, serán gestionados de manera que puedan ser corregidos, o, si no se está seguro de si el valor corregido puede ser bueno o malo, será eliminado. En este apartado entran todos los valores referentes a: valores atípicos (*outliers*), valores faltantes (*missing*), valores constantes y otros posibles valores erróneos.

c) Eliminación de columnas no válidas. Se ha de comprobar que las columnas tengan una estructura lógica por cada uno de los sensores, evitando valores constantes a lo largo de toda una columna, valores nulos en exceso, valores duplicados o valores altamente relacionados con los valores de otras columnas. En todos estos casos, el procedimiento a seguir es eliminar completamente la columna, que hace referencia a todos los datos de un único sensor.

d) Eliminación de filas duplicadas. Del mismo modo que las columnas son eliminadas por ciertas características, las filas también han de tratarse de la misma manera, eliminando así las que estén duplicadas.

7. **Correlación de variables.** Comprobar si en el resultado de correlación hay una clara evidencia de que hay variables dependientes entre sí, de manera que por cada grupo de variables altamente correlacionadas, se eliminen todas las que arrojen información repetida (aunque ambas variables tengan valores diferentes para cada momento temporal, realmente estarían arrojando la misma información al modelo, de manera que solamente haría falta emplear una de ellas, evitando meter ruido con variables innecesarias).

8. **Filtrado de variables.** Una vez analizadas todas las relaciones entre variables repetidas veces, y comprobando que se han hecho todos los análisis respectivos necesarios, se procede a extraer de los datos todas las variables (sensores) que no dan información suficiente en el proceso. De este modo, solamente quedan las variables realmente representativas, gracias a las cuales está previsto poder inferir un resultado óptimo del PSD del producto final de la planta cementera.

9. **Construcción del modelo.** Sabiendo cuales son las variables oficiales que van a funcionar dentro del modelo predictivo, se procede a crear un nuevo dataset que albergue datos suficientes de esos sensores. Ese dataset, de la misma manera que el anterior, será procesado de manera que sean eliminadas todas las incongruencias y quede un resultado estructurado y limpio. Con el dataset preparado, se decidirá cuál puede ser el algoritmo que mejor se adapte a este problema: clasificación, series temporales, clustering, etc. Posteriormente se decidirán cuáles serán sus hiperparámetros, y se comenzará el entrenamiento del modelo hasta comprobar que el porcentaje de acierto supera el mínimo estimado.

10. **Optimización del modelo.** Una vez logrado un modelo operativo, se contemplan otras configuraciones con el fin de comprobar si se puede mejorar su resultado o ya está lo suficientemente optimizado. Para ello, se desarrollará un algoritmo que se encargue de probar todas las métricas, funciones e hiperparámetros iterativamente, devolviendo en cada caso un resultado de probabilidad de acierto. Finalmente se elegirá la combinación de parámetros que mejor ayuden al modelo a conseguir resultados ajustados.

11. **Presentación de resultados.** Se desarrollará un documento interactivo a modo de plantilla visual donde se recogen y analizan los datos de los sensores, viendo gráficas de correlación por pares, correlaciones de Pearson, y análisis estadístico descriptivo.

12. **Conclusiones.** Se analizará si el modelo es lo suficientemente preciso como para ser aplicado en directo en la planta cementera. Además, se valorará el hecho de añadirle configuraciones nuevas modulares, que hagan el modelo escalable, pudiendo dar la opción de expandir sus habilidades a futuro. Se explorarán otros desarrollos que sean viables para conseguir resultados similares, o incluso mejores.

Capítulo 5

Metodología experimental

5.1. Datos

Las experimentaciones que se desarrollan en este trabajo son todas referentes a procesamiento de datos y aplicación de algoritmos de IA, con la finalidad de conocer en todo momento cual es el resultado del PSD del cemento final de la planta cementera, pudiendo hacer accioner correctivas antes de que ese material sea fabricado. Todos los datos utilizados en este proyecto son datos reales extraídos de la planta cementera de Turquía, pudiendo aplicar una solución de IA a un problema real.

Se ha de mencionar que dado que los datos empleados en este trabajo son mediciones recogidas a través de sensores distribuidos por toda la planta cementera. Cada uno de ellos capta información en un momento temporal diferente y está dispuesto en una localización diferente de la planta, de forma que para poder estimar una relación directa entre todos ellos, es necesario tener en cuenta que entre cada sensor existe una ventana temporal que se desconoce. Esa ventana temporal representa el tiempo que tarda el material en llegar de un sensor a otro. La única manera de conocer el offset temporal entre sensores es que en la planta cementera se decida hacer mantenimiento, parando todo el proceso de producción, de manera que al finalizar el proceso se vea cómo se van desactivando sensores uno tras otro y al arrancar de nuevo se vea cómo se van iniciando de nuevo.

En el supuesto caso que se llegara a conocer el valor temporal entre cada sensor, los datos se podrían alinear entre sí dentro del dataset, optimizando el resultado del modelo. En este caso, al desconocer esos valores, y sin previsiones de parar la planta cementera, las predicciones se basan en el momento exacto en el que el sensor ha tomado datos. Esto conlleva que a futuro, cuando se haya desarrollado un modelo funcional y usable dentro de la empresa, se deba reentrenar con los datos que tienen en cuenta el retraso (*lag*) temporal entre sensores. Esto no afecta en el resultado predictivo del modelo, ya que los sensores reflejan valores lineales en el tiempo, cada uno con su desviación típica.

5.1.1. Recogida de datos

Para poder desarrollar algoritmos de IA que solventen este problema, el primer paso a dar es la recolección de los datos. Se han de recoger todos los datos que se pueda, siempre que su formato sea el correcto, para después sacar estimaciones y relaciones entre ellos, eliminando los que no sean representativos para el aprendizaje automático y consiguiendo así un dataset (set de datos) estructurado y bien definido. Tal y como se ha mencionado en el punto anterior, los datos hacen referencia a las medidas que recogen los sensores repartidos por la planta cementera, pudiendo ser medidas que hacen referencia al consumo de potencia de un motor de cinta transportadora, o a la temperatura del material en una determinada zona del proceso, o incluso la intensidad a la que funciona un ventilador de refrigeración instalado en una chimenea.

La recogida de datos se realiza a través de una API que hace de intermediaria entre el desarrollador (cliente) y los datos almacenados en la base de datos de la planta cementera (servidor), dando acceso únicamente al sujeto con las credenciales necesarias (que son cedidas por el propietario de la base de datos), evitando así riesgos informáticos que puedan incurrir en una gestión incorrecta de los datos. La API funciona a través de solicitudes HTTP, usando una VPN (*Virtual Private Network*) que oculta la IP del usuario, encriptando sus datos, de manera que la acción de extracción de datos sea segura.

Una vez establecida la conexión entre cliente-servidor, los datos se recogen en formato .csv de la base de datos, haciendo una solicitud (request) que englobe todos los sensores que se quieren recolectar, filtrando la cantidad de datos a través de fechas, es decir, que se pueden descargar datos de un solo día,

o de varios meses seguidos. En este caso, dado que se quiere desarrollar un modelo predictivo de ML, la cantidad y calidad de los datos es crucial, debiendo recabar la mayor cantidad que sea posible (siempre que sean datos que arrojen información relevante), ya que después del proceso de limpieza muchos de ellos no servirán, y la cantidad final de datos se verá reducida parcialmente. En total, se recogen más de 20 sensores repartidos por toda la planta, en rangos temporales de 4 meses, con lecturas de sensor de entre medio segundo y 1 minuto. En total se almacenan más de 20 millones de datos referentes a mediciones de sensores en planta, de los cuales, muchos serán erróneos o tendrán algún formato incorrecto que habrá que procesar.

Los datos extraídos se recogen en archivos individuales, uno por cada sensor, con formato csv, estando todos esos datos acotados en rangos temporales definidos por el usuario. Se ha decidido extraer datos de 4 meses seguidos, desde Enero hasta Abril de 2022. El siguiente paso de este proceso es analizar qué datos se han logrado, cómo se debe modificar su formato, y comenzar a generar un dataset que recopile toda la información recogida en un único archivo que sirva de alimento para las redes neuronales.

5.1.2. Análisis previo, procesado y creación del dataset

Para poder entrenar un modelo predictivo que gestione toda esta cantidad de datos, el primer paso que hay que dar es entender los datos, prepararlos como es debido y crear un dataset estructurado y ordenado que sirva de entrenamiento del modelo predictivo.

A continuación se explica el procedimiento empleado para dicho tratamiento:

Se hace una comprobación dentro de cada uno de los .csv para ver en qué formato han sido introducidos los datos dentro de cada celda ya que el formato es importante de cara a que los programas que se van a desarrollar sean capaces de gestionar toda esa información. Una fecha mal estructurada, no va a ser leída como fecha, sino como cadena de caracteres, de manera que sería imposible agrupar los datos por minutos, horas o días.

ts,value
2021-12-15T00:00:05+00:00,47.3077
2021-12-15T00:00:13+00:00,47.5824
2021-12-15T00:00:21+00:00,48.1136
2021-12-15T00:00:28+00:00,47.9121
2021-12-15T00:00:36+00:00,47.9304
2021-12-15T00:00:44+00:00,47.4176
2021-12-15T00:00:52+00:00,47.7656
2021-12-15T00:01:00+00:00,47.8388
2021-12-15T00:01:08+00:00,47.8571
2021-12-15T00:01:17+00:00,47.7839
2021-12-15T00:01:25+00:00,47.619
2021-12-15T00:01:33+00:00,48.2784

Figura 5.1: Formato inicial de los datos dentro de cada CSV.

En la figura 5.1 los datos no están bien estructurados. Cada una de las columnas debe ir en una columna individual y los decimales de los datos deben ser puntos (aunque se representen con comas en las imágenes, internamente el algoritmo gestiona los decimales como puntos). Por otra parte, la información de la fecha es excesiva, ya que con saber únicamente el día de recogida y la hora es suficiente. El hecho de meter el GTM+00:00 solamente mete ruido en esa variable, ya que se repite dentro de cada lectura, y no varía en ningún momento. Con tener en cuenta la diferencia horaria entre España y Turquía es suficiente (aunque no es representativo en absoluto). Para solucionar este problema, se desarrolla un algoritmo que gestione todos esos cambios y cree unos nuevos csv, además de agrupar todos los datos en un mismo rango temporal (paquetes de 1 minuto) con un formato bien definido tal y como se observa a continuación:

Time	Values
15/12/21 0:00	47,3077
15/12/21 0:00	47,5824
15/12/21 0:00	48,1136
15/12/21 0:00	47,9121
15/12/21 0:00	47,9304
15/12/21 0:00	47,4176
15/12/21 0:00	47,7656
15/12/21 0:01	47,8388
15/12/21 0:01	47,8571
15/12/21 0:01	47,7839
15/12/21 0:01	47,619
15/12/21 0:01	48,2784

Time	Values
15/12/21 0:00	47,718486
15/12/21 0:01	47,944125
15/12/21 0:02	47,5929
15/12/21 0:03	47,742675
15/12/21 0:04	48,011514
15/12/21 0:05	47,971588
15/12/21 0:06	48,063843
15/12/21 0:07	47,934975
15/12/21 0:08	47,844057
15/12/21 0:09	47,820513
15/12/21 0:10	48,079213
15/12/21 0:11	48,147557

Figura 5.2: Izq: Formato corregido de los datos dentro de cada archivo./ Der: Datos agrupados en rangos temporales de 1 minuto.

En la parte izquierda de la figura 5.2 se puede observar que las primeras 8 filas tienen la misma hora (00:00), de la misma manera que las siguientes tienen (00:01), y así sucesivamente. Es un fallo de lectura de Excel ya que esas horas corresponden a segundos que Excel no es capaz de leer, es decir, por cada fila de valores, la hora real inicial es 00:00:05 incrementándose en 8 segundos por cada fila, hasta llegar al final del archivo.

Un problema a la hora de tratar con estos datos, es que es posible que algunos sensores no midan cada 8 segundos, o que haya algún error de lectura que no recoja correctamente los datos en el momento exacto. Una forma de solucionar y simplificar este tratamiento de datos es agrupar todos los valores en rangos de 1 minuto, coger todas las medidas que haya de un mismo sensor a lo largo de ese tiempo y hacer la media de todos, asignando ese valor a ese momento temporal. De este modo, todos los sensores tienen datos en un mimo momento temporal. Este procedimiento se aplica a cada uno de los archivos de los sensores, quedando en el formato que se puede observar en la parte derecha de la figura 5.2.

Para saber en qué partes del dataset es necesario aplicar limpieza y/o corrección de datos, se hace un análisis previo para de comprobar si hay algún error en la captura. Por cada archivo csv, se crean gráficas de funcionamiento y densidad de valores, que analizan el comportamiento de cada uno de los sensores individualmente, devolviendo por una parte una línea temporal que representa el funcionamiento del sensor durante los 4 meses seleccionados (parte superior de la gráfica), y por otra una campana de Gauss que refleja el valor promedio que más veces aparece en cada uno de ellos (parte inferior de la gráfica). En la figura 5.3 se ve un ejemplo de estas gráficas sobre dos sensores elegidos aleatoriamente:

En la figura 5.3 se puede observar como muchos de los valores de la línea temporal de funcionamiento (parte superior del gráfico), tienden a cero, generando picos de ruido tanto por arriba como por abajo de la línea media. Esto quiere decir que en algún momento, esos sensores han dejado de recoger datos reales por alguna razón (sensor averiado / apagón / mala conexión, etc.), devolviendo valores nulos a la base de datos. Este hecho se confirma al fijarse en la parte inferior de los gráficos, donde se ve representada una campana de Gauss de dos picos que explica cuáles son los valores más recogido por ese sensor. En todas las gráficas analizadas se ven los 2 picos cuando solamente debería haber uno. El pico más a la izquierda está situado sobre el valor 0 del eje X, lo que confirma que muchos de los valores son literalmente nulos.

Hasta ahora se han descargado, y cambiado todas las variables a su respectivo formato, además de comprobar cuantos de los datos son realmente válidos. Antes de hacer la limpieza integral de los valores de cada sensor, se van a eliminar los sensores que no son representativos de analizar para este problema, es decir, los que no aportan información relevante al entrenamiento del modelo predictivo, como puede ser un sensor que mida la velocidad de un ventilador de chimenea, o sensores que estén altamente correlacionados entre sí y aporten el mismo tipo de información al modelo. Estos sensores se descartan a criterio del analista de datos que los esté tratando, basándose en el la localización en planta, su uso dentro del proceso, y del tipo de información que arroje.

En el caso de descartar un sensor que pueda dar información importante, más adelante se puede recular y reinsertar dicho sensor dentro del grupo de datos de entrenamiento y validación de nuevo. La siguiente tabla (6.1 muestra la decisión de una primera criba de sensores de toda la planta). Posteriormente, después del análisis exploratorio de datos, se terminarán de descartar los que se crea que no aportan información al proceso.

En este punto, se tienen datos correctamente formateados de únicamente los sensores que se cree que

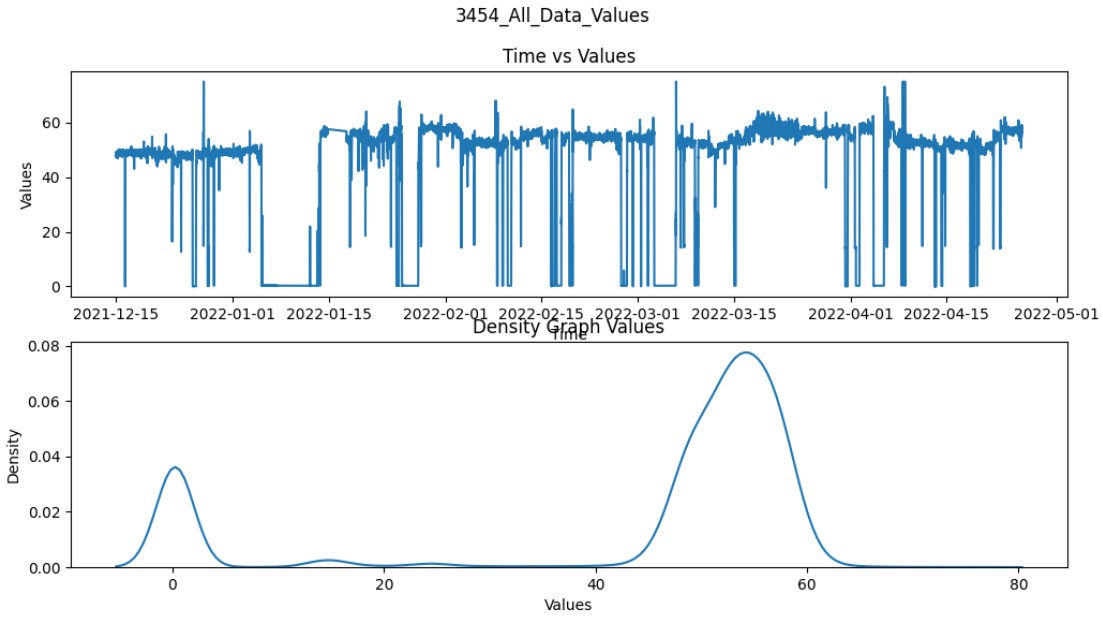


Figura 5.3: Representación gráfica del funcionamiento y valor promedio de 1 de los sensores.

pueden generar buenas predicciones, a falta de limpiarlos completamente, corregir errores que puedan tener y ensamblarlo todo en un único archivo .csv. Para facilitar la lectura de esos datos, se decide crear el dataset unificado de valores ahora y no después (figura 5.4), de manera que las modificaciones se puedan hacer directamente sobre él, sin necesidad de ir leyendo cada uno de los sensores de forma individual.

Tabla 5.1: Resultado de los sensores elegidos para el modelo después de realizar un primer cribado.

SENSOR	RAZÓN SELECCIÓN
3454	Potencia del elevador a la salida del molino
3475	Tª del cemento a la salida del molino
3476	Tª del aire a la salida del molino
3479	Presión a la entrada del molino
3480	Sonido dentro del molino
3481	Potencia del separador de material
3482	Velocidad del separador
3483	Presión a la salida del molino
3499	Temperatura del filtro
3506	Potencia del ventilador del filtro
3507	Potencia del motor del molino
3523	Velocidad total del molino
3529	Control de velocidad del molino
3848	Diferencia de presión en el filtro
7400	Mide PSD hasta 90mm
7401	Mide PSD hasta 64mm
7402	Mide PSD hasta 48mm
7403	Mide PSD hasta 45mm
7404	Mide PSD hasta 32mm
7405	Mide PSD entre 3-32mm
7406	Mide PSD hasta 3mm
7407	Mide la finura del cemento (Superficie/peso)

Fechas filtradas	3454.csv	3475.csv	3476.csv	3479.csv	3480.csv	3481.csv
15/12/21 0:00	47,718486	109,39829	97,5824	-0,177045	52,346057	186,44343
15/12/21 0:01	47,944125	109,37275	97,5824	-0,182234	52,47865	187,38725
15/12/21 0:02	47,5929	109,304	97,571943	-0,175301	52,653071	184,61543
15/12/21 0:03	47,742675	109,304	97,596138	-0,166361	52,408425	187,82663
15/12/21 0:04	48,011514	109,25686	97,561486	-0,150706	53,001929	185,84343
15/12/21 0:05	47,971588	109,2765	97,57325	-0,199328	52,719788	185,531
15/12/21 0:06	48,063843	109,31971	97,5824	-0,166754	52,680986	184,727
15/12/21 0:07	47,934975	109,2765	97,57325	-0,172161	52,7778	185,07938
15/12/21 0:08	47,844057	109,28829	97,551029	-0,190127	52,607729	185,22943
15/12/21 0:09	47,820513	109,29025	97,5092	-0,180555	52,835775	187,03288
15/12/21 0:10	48,079213	109,37275	97,5275	-0,202228	53,092163	185,824
15/12/21 0:11	48,147557	109,414	97,5092	-0,165707	53,403114	184,08486
15/12/21 0:12	48,399725	109,414	97,5092	-0,169414	53,241763	187,16738
15/12/21 0:13	48,273157	109,36686	97,5092	-0,205477	53,288	186,22

Figura 5.4: Muestra de una porción del dataset de valores estructurados.

Como se puede ver, los sensores están dispuestos en columnas, almacenando todos sus valores en vertical. Las filas corresponden a las fechas en las que se han recogido los datos, agrupándolos en paquetes de 1 minuto. De este modo, cada fila completa corresponde a todos y cada uno de los valores recogidos por cada sensor en su localización en planta. Esto genera una combinación de valores de sensores con los que se pretende estimar las predicciones de PSD.

Una vez que se ha creado la estructura del dataset, tal y como se ha descrito en el párrafo anterior, es necesario eliminar y/o modificar todos los valores faltantes y nulos y los valores erróneos (valores atípicos o constantes por ejemplo) localizando las zonas conflictivas dentro del dataset tal y como se ve en la siguiente figura 5.5.

12/01/2022 20:42	435,054875	0	77,9976	-2,1245413	141,8865
12/01/2022 20:43	435,081	0	77,9976	-1,4379897	139,026714
12/01/2022 20:44	434,934286	0	77,9976	-1,8524314	139,703429
12/01/2022 20:45	434,761625	0	77,9976	-2,0964575	139,0965
12/01/2022 20:46	434,652	0	77,9976	-1,6532345	138,773
12/01/2022 20:47	434,641571	0	77,9976	-2,8417929	138,182429
12/01/2022 20:48	434,725125	0	77,9976	-2,7533575	137,265
12/01/2022 20:49	434,688625	0	77,9976	-2,6129425	138,61425
12/01/2022 20:50	434,369571	0	77,9976	-2,46363	137,728857
12/01/2022 20:51	434,066125	0	77,9976	-1,6898663	136,489625
12/01/2022 20:52	433,951	0	77,9976	-2,36874	138,231429
12/01/2022 20:53	434,194375	0	77,9976	-2,8852263	138,712
12/01/2022 20:54	434,121125	0	77,9976	-1,9792425	137,960875
12/01/2022 20:55	434,285857	0	77,9976	-2,3059471	138,182286
12/01/2022 20:56	434,306857	0	77,9976	-1,9291814	137,010286
12/01/2022 20:57	434,194375	0	77,9976	-2,09524	137,197875
12/01/2022 20:58	434,084375	0	77,9976	-2,9059825	135,970875
12/01/2022 20:59	433,972	0	77,9976	-2,7301571	135,496143
12/01/2022 21:00					
12/01/2022 21:01					
12/01/2022 21:02					
12/01/2022 21:03					
12/01/2022 21:04					
12/01/2022 21:05					
12/01/2022 21:06					
12/01/2022 21:07					
12/01/2022 21:08					
12/01/2022 21:09					
12/01/2022 21:10					
12/01/2022 21:11					

Figura 5.5: Tipos de valores erróneos dentro del dataset

En naranja se pueden observar los valores faltantes. Son valores que no han sido capturados en las fechas que tienen asignadas, representados con un hueco en la celda donde debería haber un valor numérico. Se puede deber a que la planta cementera esté parada, o a que el sensor no esté recibiendo alimentación o pueda no estar conectado con la base de datos.

El color azul representa los valores constantes (valores que se repiten con decimales exactos en cada una de las fechas de recogida de datos). Se puede deber a tener un bug (problema en un programa de computador o sistema de software que desencadena un resultado indeseado) y no es capaz de medir correctamente los valores reales, devolviendo valores erróneos de medición.

En verde se pueden observar valores nulos, representados con un 0. Estos se pueden deber a fallos de medición al igual que en los valores constantes.

Otro tipo de datos que puede generar malos resultados son los valores atípicos. Los *outliers* o valores atípicos, son valores que se salen de la media del conjunto espontáneamente, generando picos de ruido y distorsionando el set de datos. Entre tanta cantidad de datos es prácticamente imposible detectarlos visualmente, de modo que para lograr saber si en este set de datos hay valores atípicos, y conocer su localización, se calcula la media de valores del conjunto de datos y se le asigna un umbral por arriba y por abajo. Mediante un barrido, todos los valores que sobrepasen esos rangos se consideran atípicos, y se gestionan de dos maneras, eliminándolos o convirtiéndolos a la media de valores de esa columna de datos. (a conveniencia del científico de datos. Teniendo datos de sobra, se pueden eliminar directamente. Teniendo pocos datos, en cambio, merece la pena guardarlos aunque se hayan modificado y sean datos artificiales).

Se aplica el procedimiento de limpieza del dataset para corregir o eliminar valores inconsistentes siguiendo el apartado 3.4, punto número 6 de la descripción sobre el tratamiento de valores en un dataset.

En la siguiente imagen figura 5.6 se puede observar cómo después de hacer toda la limpieza de valores faltantes y nulos, se han detectado valores atípicos (izq), y cómo queda la representación gráfica después de tratarlos (der), donde la diferencia es que al eliminar esos valores, la representación gráfica adapta la escala para ajustarse a los nuevos valores.

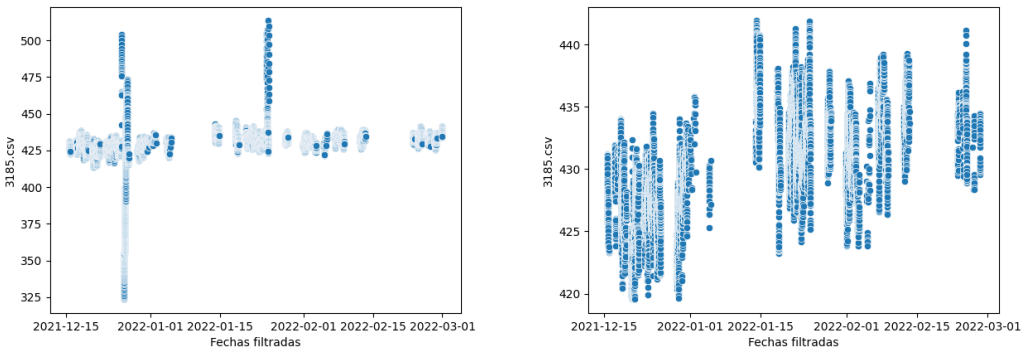


Figura 5.6: Izq: Sin filtrar valores atípicos / Der: Filtrando valores atípicos

Finalmente, una vez realizado todo el procesamiento de los datos, se da paso al análisis exploratorio de los mismos, el cual mediante resúmenes numéricos, estadística descriptiva y visualizaciones, da opción de explorar los datos en todo su contexto, además de identificar posibles relaciones entre variables.

5.1.3. Análisis exploratorio de datos

En este apartado, se explican los pasos seguidos en el análisis de las variables seleccionadas. Se aplican análisis individuales mediante histogramas, análisis por pares mediante distribución de puntos, análisis individuales estadístico-descriptivos para visualizar la distribución de valores de cada sensor, y por último, correlaciones de Pearson. Todos y cada uno de estos análisis ayudan a entender mejor el comportamiento de los sensores y ver si alguno de ellos no debería estar en esta parte del proceso.

Para comenzar con el análisis, se representa el histograma de valores individuales (figura 5.7). Cada histograma corresponde a un sensor que representa su distribución de valores de todo el set de datos recopilado.

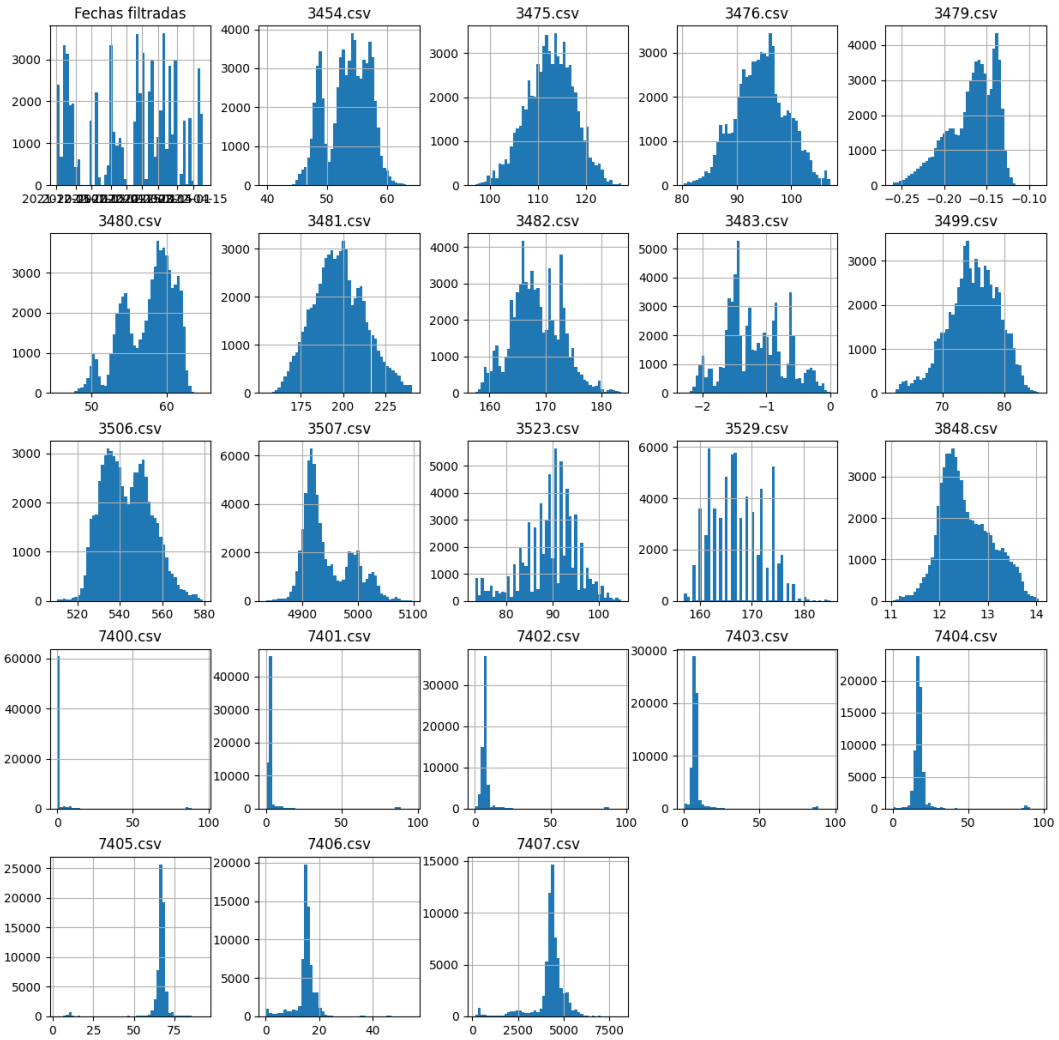


Figura 5.7: Análisis individual de variables.

Tal y como se ve en la figura 5.7, se puede ver que cada uno de los sensores tiene una tendencia diferente. Algunos de los sensores tienen datos sesgados a la derecha, otros a la izquierda y otros son completamente simétricos (histograma uniforme). También se puede observar algún caso bimodal (dos picos de datos en el histograma). Todos y cada uno de ellos se explican a continuación:

En un histograma sesgado a la izquierda (la media es siempre menor que la mediana) se dice que está sesgado negativamente. Esta distribución tiene muchos valores en el lado superior del eje x (lado derecho) y menos elementos en el valor inferior (lado izquierdo), lo que indica que tienen unos valores pequeños que impulsan la media hacia abajo, pero no afectan en el que el eje central de la gráfica (es decir, la mediana).

En un histograma sesgado a la derecha (la media es mayor que la mediana) se dice que están sesgados positivamente. Esta distribución tiene un gran número de valores en el lado inferior del eje x (lado izquierdo) y pocos en el valor superior (lado derecho). Esto sucede porque los datos asimétricos derecho tienen unos valores grandes que impulsan la media hacia arriba, pero no afectan en el eje central de la gráfica.

En el histograma uniforme, en cambio, cada contenedor contiene aproximadamente el mismo número de recuentos(frecuencia). Los datos son más o menos iguales en altura y ubicación en cada lado del centro del histograma.

La bimodalidad en el conjunto de datos podría indicar que la información provino de dos sistemas, procesos,máquinas u otras fuentes diferentes. Este no es el caso ya que todos esos datos provienen de un mismo sensor, por lo que es posible que los sensores que tengan dos picos de histograma tengan algún problema de recogida.

Por último, la multimodalidad en este tipo de gráficas no tiene un patrón discernible y puede indicar una distribución que tiene varios modos o picos, lo que puede significar que hay demasiados grupos de datos diferentes dentro de un mismo sensor, es decir, que no termina de recoger valores de un modo constante.

Una vez se han analizado por individual las variables independientes, se estudian las relaciones de las variables por pares, de forma que se pueda ver gráficamente si existe alguna dependencia entre ellas. Algunas se visualizan como puntos dispersos sin ningún tipo de relación, lo que conlleva a la disentropía. Las variables que tienen alguna de relación, en cambio, tienden a organizar los puntos generando algún tipo de forma, ya sea lineal, curva, agrupada, etc.

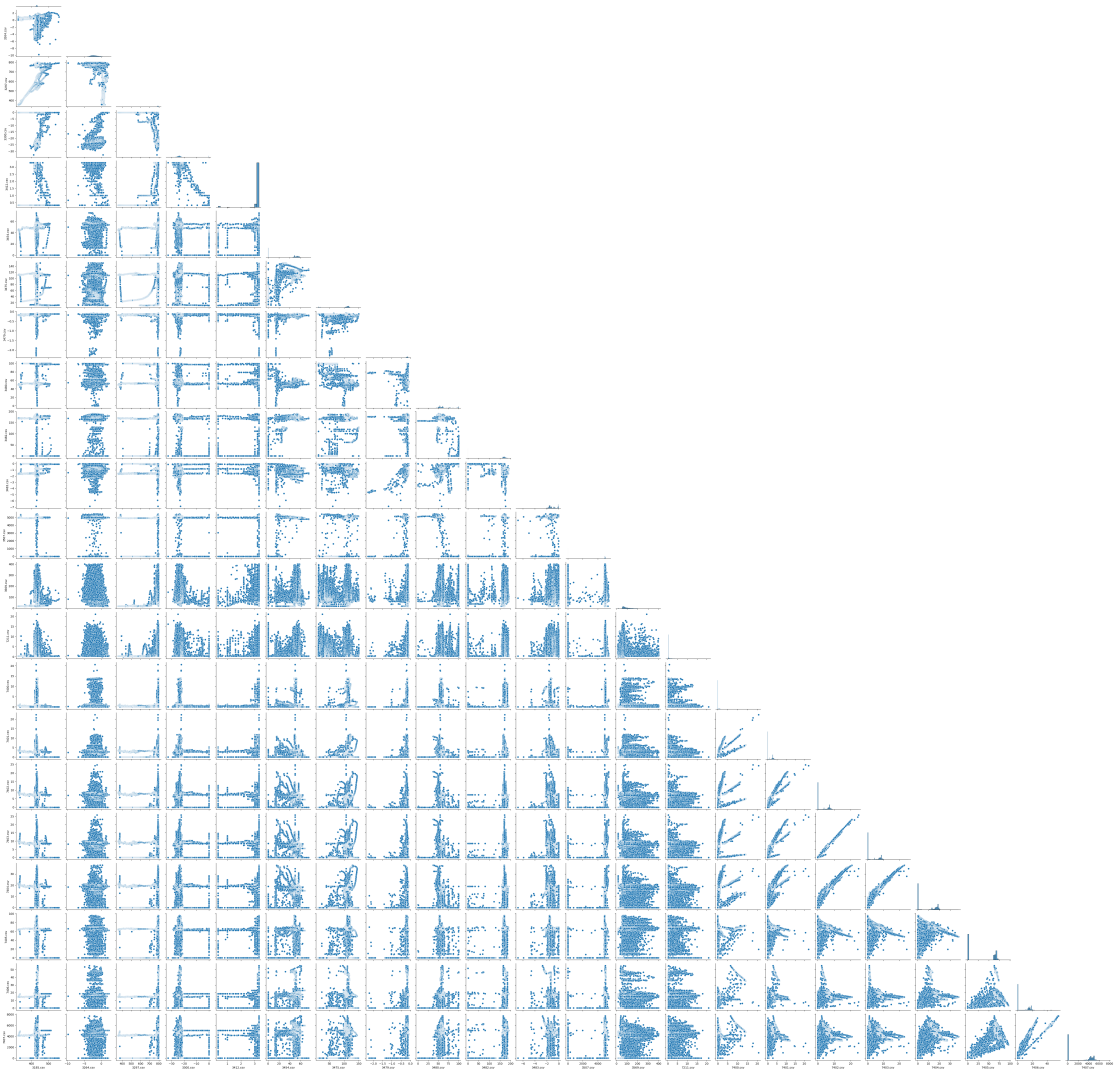


Figura 5.8: Análisis por pares de variables.

En la figura 5.8 cada uno de los recuadros corresponde a una relación por pares entre dos sensores, relacionando los valores de ambos en una distribución de puntos en una gráfica de dos ejes. Esa distribución de puntos representa la relación que existe entre ambos sensores representando la dispersión de puntos

de cada gráfica. En la figura 5.9 se observa una sección ampliada de la figura 5.8.

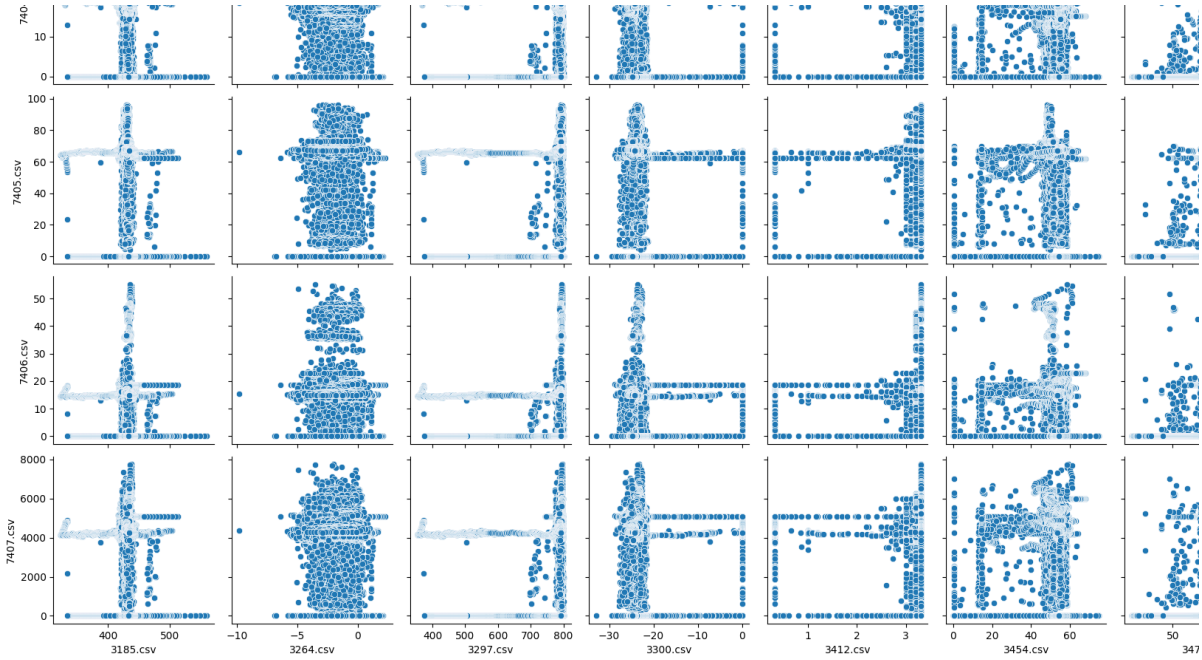


Figura 5.9: Análisis por pares de variables ampliado.

Un método más eficaz de decidir si existe algún par de variables altamente relacionadas entre sí es la matriz de correlación de Pearson, que se desarrolla en el siguiente punto, la cual devuelve un resultado numérico de correlación (de -1 a 1) sabiendo así el valor exacto de relación que hay entre ambas. Aun así, siempre viene bien discriminar variables con diferentes métodos, de ahí el uso del análisis por pares.

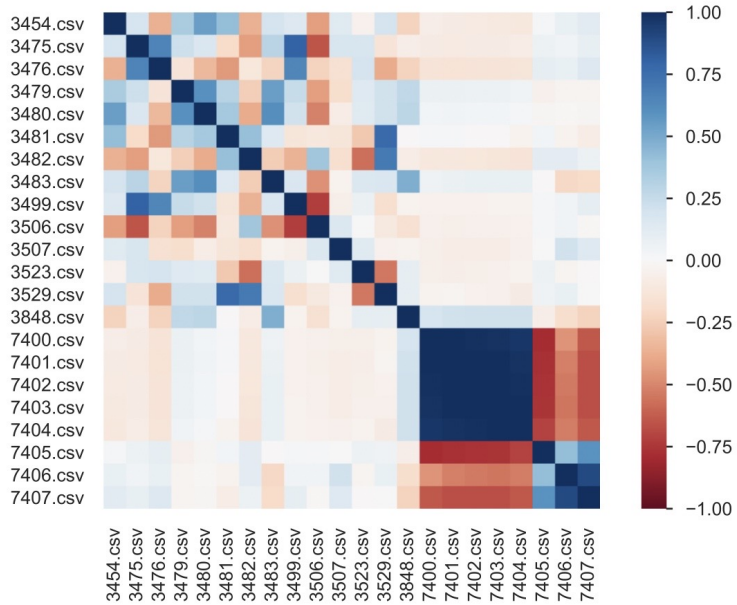


Figura 5.10: Correlaciones de Pearson entre variables

Una vez visto el conjunto de datos en global mediante los dos análisis anteriores, se procede a hacer un análisis estadístico descriptivo a cada uno de los sensores (fig: 5.11), asegurando así las confirmaciones que se han mencionado sobre la distribución y sesgo de los puntos, comprobándolo directamente sobre los valores estadísticos de media, mediana, cuartiles, etc, además de comprobar si realmente se han eliminado todos los valores nulos y/o faltantes.

En la figura 5.10 se representa la relación de variables con porcentajes entre -1 y 1, siendo -1 No correlacionado y 1 Altamente correlacionado. Este paso se hace a modo de profundizar el análisis de

variables por pares, ya que en el gráfico por pares, visualmente se puede ver si hay alguna correlación clara entre variables, pero en caso de que la correlación no sea tan evidente, es necesario visualizarlo de forma cuantitativa. Aquí es donde entra en juego la matriz de correlación de Pearson. Mediante la matriz de resultados y aplicándole un mapa de calor, se observa claramente qué variables tienen más relación dentro del conjunto de datos. Para poder estimar una alta correlación se define el límite de alta correlación en valores superiores al 80 por ciento.

En este caso, en el cuadrante inferior derecho se ve como todos los sensores tienen una alta correlación. Aunque esto sea así no se deben eliminar dado que son los sensores que reflejan los datos del resultado de PSD. Todos los sensores de PSD están colocados en una misma localización de la planta cementera, y teniendo en cuenta que todos miden datos sobre la distribución de partículas en diferentes tamaños de partícula, están todos ellos altamente correlacionados. Sin esos sensores no habría resultados de medición de la distribución de partículas del cemento al final del proceso.

En la figura 5.11 se visualiza el resultado final del análisis completo de cada variable, dando como resultado un diagrama interactivo que relaciona todas las variables y calcula la estadística descriptiva de cada una por individual. Se denomina interactivo porque se ha generado un archivo que se puede abrir a través del explorador de internet mediante el cual se pueden manejar las gráficas usando el ratón. En la figura 5.11 solamente se muestra uno de los ejemplos analizados.

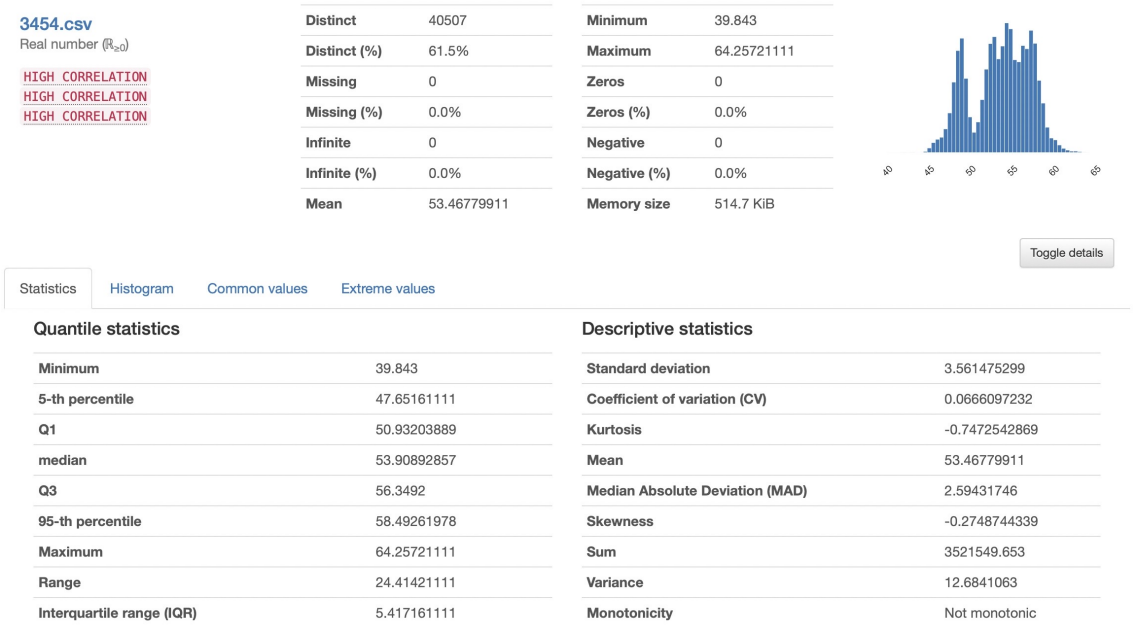


Figura 5.11: Análisis descriptivo de cada variable por individual

Como punto final se decide conservar las variables del problema que aparecen en la tabla 6.1 como variables definitivas. Todas esas variables son las que deberían poder servir para predecir los valores de PSD del cemento al final del proceso. Como se ha mencionado anteriormente, el hecho de elegir estas variables ahora no es algo definitivo. Más adelante si se comprueba que los modelos predictivos no son capaces de predecir aunque se hayan optimizado al máximo, se pueden eliminar y/o añadir nuevos sensores al set de datos.

Tabla 5.2: Resultado de los sensores elegidos para el modelo después de realizar el cribado final.

SENSOR	RAZÓN SELECCIÓN
3475	Tª del cemento a la salida del molino
3479	Presión a la entrada del molino
3480	Sonido dentro del molino
3482	Velocidad del separador
3483	Presión a la salida del molino
3507	Potencia del motor del molino
3523	Velocidad total del molino
3848	Diferencia de presión en el filtro
7400	Mide PSD hasta 90mm
7401	Mide PSD hasta 64mm
7402	Mide PSD hasta 48mm
7403	Mide PSD hasta 45mm
7404	Mide PSD hasta 32mm
7405	Mide PSD entre 3-32mm
7406	Mide PSD hasta 3mm
7407	Mide la finura del cemento (Superficie/peso)

Una vez hechas todas las comprobaciones de variables, y habiendo decidido las definitivas, se procede a desarrollar el algoritmo de inteligencia artificial que pueda solucionar este problema.

5.2. Experimentación con algoritmos de IA

A continuación se explican los diferentes algoritmos de IA propuestos como solución al problema. Todos ellos se basan en desarrollos de aprendizaje supervisado, donde se tiene un conjunto de variables independientes (todos y cada uno de los valores de los sensores), y una única variable dependiente (el resultado final de la PSD del cemento). Han sido usados en una proporción de 80-20 para entrenar-validar. Como se trata de un aprendizaje supervisado, los datos de entrenamiento consisten de pares de objetos (normalmente vectores): una componente del par son los datos de entrada y el otro, los resultados deseados.

El set de datos se parte en 2 (proporción 80-20) para poder entrenar el modelo y posteriormente comprobar cómo de preciso es, es decir, el primer 80 % de los datos se usa para entrenar la red neuronal y el otro 20 % se emplea para validar las respuestas predictivas que ha dado el modelo como resultado. Esta proporción se puede variar dependiendo de la cantidad de datos que se disponga. Teniendo datos de sobra, se puede ampliar el set de datos de validación, por ejemplo.

Por otra parte, es necesario entender cuál es el procedimiento correcto de partición del dataset en datos de entrenamiento y de validación. Este paso es necesario para poder entrenar y comprobar cómo de bien es capaz de predecir el algoritmo, comparando los valores predichos por el modelo contra los que realmente debería haber inferido.
Dependiendo del modelo que se emplee, el proceso de partición del dataset es diferente, ya que no es lo mismo aislar un paquete de datos que sigue una temporalidad entre cada fila de valores, a un paquete de datos que ha sido escogido aleatoriamente de todo el conjunto disponible. En algoritmos que mantienen una temporalidad en los datos, como pueden ser las series temporales, los datos han de ser recogidos manteniendo el orden en el que fueron captados. En otros algoritmos que no sigan esa regla, como pueden ser los algoritmos de ensemble, los datos pueden ser escogidos aleatoriamente del dataset.

En la figura 5.12 se muestra una representación gráfica que explica visualmente la elección de valores dependiendo del tipo de problema:

Figura 5.12: Partición del dataset según el tipo de modelo predictivo. Izq: Partición de datos para Series Temporales. Der: Partición de datos para otros modelos no lineales temporalmente.

Una vez repartidos los datos en entrenamiento y testeo y habiendo entrenado los modelos predictivos con esos datos, cada uno de los modelos devuelve una o varias predicciones (dependiendo el algoritmo usado) por cada paquete de datos introducido. Se puede evaluar cómo de bien son los resultados creados por esas predicciones, usando métricas de evaluación. Las métricas empleadas en este proyecto son la precisión y el error cuadrático medio.

5.2.1. Algoritmo de Series Temporales

Una serie temporal es un conjunto de datos u observaciones que hace referencia a una o varias variables y que están ordenadas cronológicamente. Son muy importantes en ecnonomía porque casi todas las variables que se recogen se analizan a lo largo del tiempo y no en un determinado instante. De ahí que siempre que se analicen las variables económicas se hable de ciclos económicos o de tendencias. Y aunque la economía sea la rama principal por excelencia que se nutre de este tipo de desarrollos para conseguir estimaciones y posibles beneficios, no es el único sector que puede ser resuelto mediante algoritmos de esta índole. Los algoritmos de inteligencia artificial cada vez son mas generalistas, abarcando todo tipo de problemáticas que pueden ser resueltas con un mismo método.

Teniendo en cuenta que en las series temporales el orden de los datos importa se puede decir que las observaciones no son independientes, esto es, el pasado puede afectar al futuro.

El procedimiento a seguir en el desarrollo de series temporales es el siguiente:
Tal y como se ha mencionado antes, el primer paso es obtener los datos en una serie temporal continua. Después, la principal adaptación que se necesita hacer es transformar la serie temporal en un matriz en la que, cada valor, está asociado a la ventana temporal (lag) que le precede. En la siguiente figura 5.13 se aporta un ejemplo de lo anteriormente comentado.

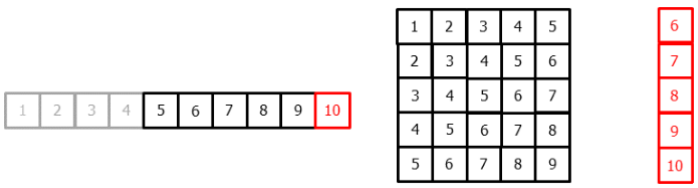


Figura 5.13: Formateado de los datos para análisis de series temporales. El vector de la izquierda, a medida que avanza, es el que va generado la matriz y columna de la derecha. Cada posición avanzada por el vector crea una fila completa de la matriz con su resultado (en rojo).

Cuando se trabaja con series temporales, raramente se quiere predecir solo el siguiente elemento de la serie (t+1), sino todo un intervalo futuro o un punto alejado en el tiempo (t+n). A cada uno de esos pasos de predicción se les conoce como *step*. Para poder generar tantas predicciones seguidas existen varias estrategias que permiten generar ese tipo de resultados múltiples. En la figura 5.14 se representa el proceso de predicción múltiple a través de *steps* consecutivos.

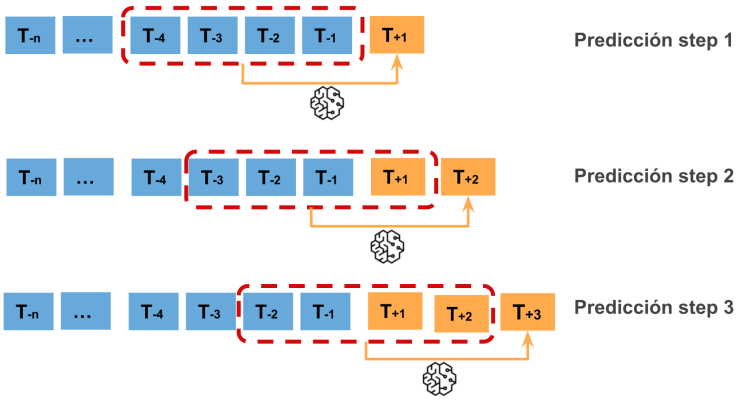


Figura 5.14: Predicción multi-step recursivo para predecir 3 steps a futuro utilizando los últimos 4 lags de la serie como predictores.

La principal complejidad de esta aproximación consiste en generar correctamente las matrices de entrenamiento para cada modelo, debiendo separar los datos tal y como se ve en la siguiente figura 5.15. Estos datos han sido preparados para que visualmente se pueda entender mejor el proceso de partición de los datos y su posterior predicción. No son datos reales de este trabajo). En caso de hacer una incorrecta repartición de los datos de *train-test*, el modelo no funcionará como es debido.

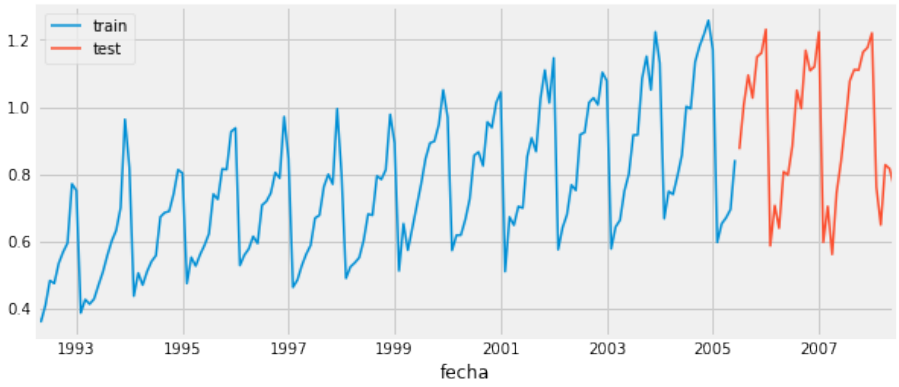


Figura 5.15: Ejemplo de partición de datos de train-test.

Además, es muy importante tener en cuenta que esta estrategia tiene un coste computacional más elevado que otros algoritmos ya que requiere entrenar múltiples modelos y es posible que con las características computacionales empleadas en este trabajo, no se tenga procesamiento suficiente.
La ventana temporal móvil empleada en la gestión de datos temporales que se observa en la figura 5.14 puede ser más amplia o más reducida, dependiendo de la cantidad de datos y la capacidad de procesamiento de la que disponga el desarrollador que esté codificando la serie temporal.

En la siguiente figura 5.16 se observa una predicción hecha (color anaranjado sobre los datos de color rojo) sobre la serie temporal de los datos de la figura 5.15.

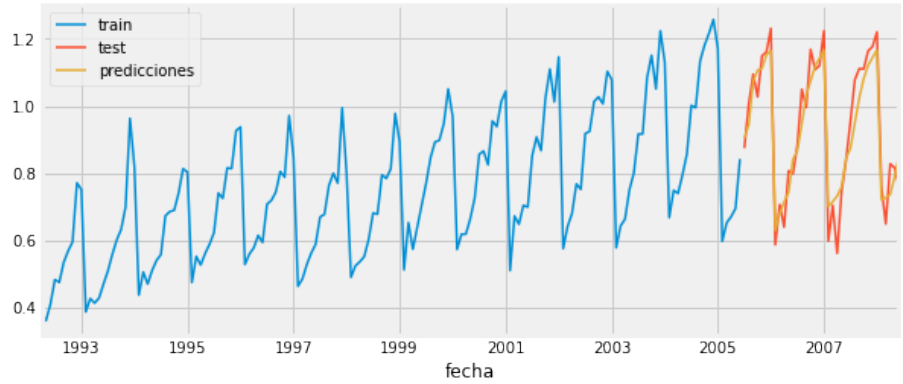


Figura 5.16: Ejemplo de predicción de la serie temporal.

5.2.2. Algoritmos de Ensamble

Los algoritmos de *boosting* o impulso, son algoritmos que utilizan varios predictores en vez de uno solo para poder concluir un resultado final, lo que les brinda una capacidad de generalización integrada que puede ser mucho más fuerte que la de un solo predictor como es el caso de otros modelos.

Ejemplo: Si se hace una misma pregunta compleja a miles de personas al azar y luego se resumen sus respuestas en una única respuesta, en la mayoría de los casos la respuesta resumida es más certera que la respuesta de un experto en la materia, lo que basa al algoritmo en la sabiduría de las masas.

Teniendo en cuenta que los algoritmos de ensamble son una versión mejorada de los algoritmos de *bagging*, ya que estos primeros usan como motor de procesamiento a los segundos usando una metodología diferente, se prevé utilizar *boosting* en la resolución de este problema. Los algoritmos de impulso planteados son los siguientes:

ADABOOST

AdaBoost fue el primer algoritmo de impulso, y sigue siendo uno de los más utilizados y estudiados, con aplicaciones en numerosos campos. Funciona de manera que cambia la distribución de la muestra modificando los puntos de datos de peso para cada iteración que hace el modelo. Podemos dividir la idea de AdaBoost en 3 grandes conceptos:

1. USO DE TOCONES o MUÑONES COMO APRENDICES DÉBILES

En un método ensamble se combinan múltiples aprendices o predictores débiles para hacer un modelo de aprendiz fuerte. En AdaBoost, se utilizan estos predictores débiles junto a un árbol de decisión de 1 nivel (Tocón). La idea principal al crear un clasificador débil es encontrar el tocón que mejor pueda separar los datos minimizando los errores generales (figura 5.17).

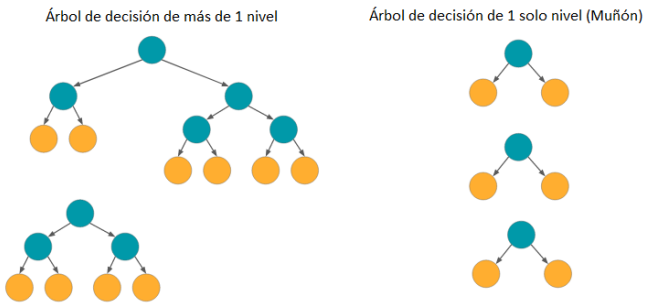


Figura 5.17: Principio de funcionamiento de Adaboost

2. INFLUENCIA DEL TOCÓN ANTERIOR EN EL SIGUIENTE
A diferencia del *bagging* (árboles de decisión completos, figura 4.2 y 4.3), que calcula datos en paralelo, *boosting* entrena secuencialmente, lo que significa que cada muñón (alumno débil) se ve afectado por el muñón anterior. La forma en que el resultado afecta al siguiente muñón es dando diferentes pesos a los datos que se utilizarán en el siguiente paso de la secuencia. Esta ponderación se basa en cálculos de error, si un dato se predice incorrectamente en el primer tocón, entonces los datos recibirán un mayor peso en el siguiente proceso de fabricación de tocones (figura 5.18).

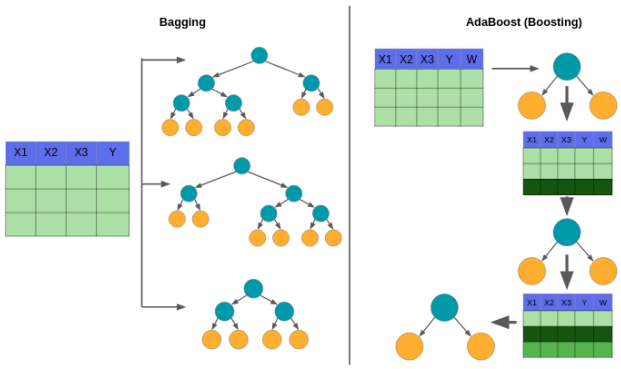


Figura 5.18: Comparativa entre bagging y boosting.

3. VOTO PONDERADO
En el algoritmo AdaBoost, cada muñón tiene un peso diferente, el peso de cada muñón se basa en la tasa de error resultante. Cuanto menores sean los errores generados por un muñón, mayor será el peso del muñón. El peso de cada uno se utiliza en el proceso de votación, dentro del cual, cuanto mayor sea el peso total obtenido por una de las clases, entonces esa clase se utilizará como la clase final (figura 5.19).

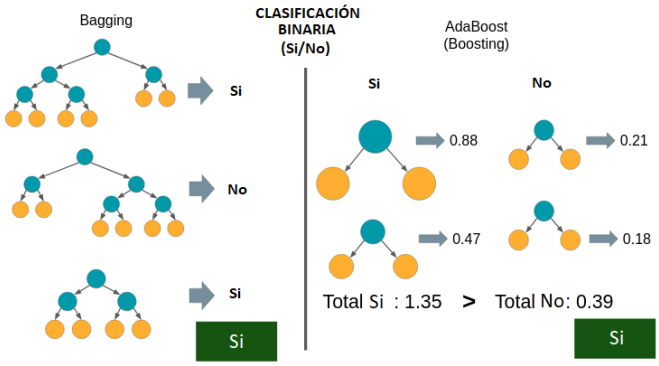


Figura 5.19: Predicción final del boosting usando voto ponderado.

Otro de los algoritmos empleados en este desarrollo es el XGBoost, que tiene un funcionamiento similar, con un mejor rendimiento.

XGBOOST

El algoritmo de XGBoost basa su funcionamiento en el mismo procedimiento que el algoritmo anterior. Emplea predictores débiles que acaban generando una predicción fuerte usando al igual que el modelo anterior, un motor interno basado en árbol de decisión.
En las siguientes líneas se explica todo el procedimiento que sigue la resolución mediante este algoritmo:

- Construcción de árboles paralelizados
  XGBoost aborda el proceso de construcción secuencial de árboles de decisión utilizando la implementación paralelizada.

- Poda de árboles
  A diferencia de otros algoritmos de *boosting*, donde la poda de árboles se detiene una vez que se encuentra una pérdida negativa, XGBoost hace crecer el árbol hasta su máxima extensión y luego poda hacia atrás hasta que la mejora en la función de pérdida esté por debajo de un umbral.

- Reconocimiento de caché y computación fuera del núcleo
  XGBoost ha sido diseñado para reducir eficientemente el tiempo de cómputo y asignar un uso óptimo de los recursos de memoria. Esto se logra mediante el reconocimiento de la caché mediante la asignación de búferes internos en cada subproceso para almacenar estadísticas de degradado. Este es un paso fundamental para procesar grandes cantidades de datos con ordenadores que no están optimizados para cálculos de redes neuronales, lo que hace este modelo atractivo comparándolo con las series temporales desarrolladas anteriormente, que limitan la precisión del resultado en base a la capacidad de procesamiento.

- Regularización
  La mayor ventaja de xgboost es la regularización. La regularización es una técnica utilizada para evitar el sobreajuste en modelos lineales y basados en árboles que limita, regula o reduce el coeficiente estimado hacia cero, evitando sobreajustar el modelo a los datos de entrenamiento y que no sea capaz de genealizar con datos reales de validación.

- Controla el valor que falta
  Este algoritmo tiene características importantes de manejo de valores faltantes por lo que funciona muy bien aunque falten datos en el dataset (Esto no es relevante en este caso ya que todos los valores faltantes han sido tratados correctamente en este trabajo).

- Validación cruzada integrada
  El algoritmo viene con un método de validación cruzada incorporado en cada iteración, eliminando la necesidad de programar explícitamente esta búsqueda y especificar el número exacto de iteraciones de impulso requeridas en una sola ejecución.

- Regularización extra y pérdida de formación
  XGBoost ofrece un término de regularización adicional que controla la complejidad del modelo, lo que nos ayuda a evitar el sobreajuste. La función objetivo es medir qué tan bien se ajusta el modelo a los datos de entrenamiento. Consta de dos partes: la pérdida de entrenamiento y el plazo de regularización.

Capítulo 6

Plataforma experimental

6.1. Hardware utilizado

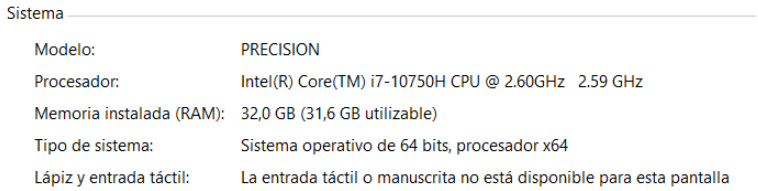


Figura 6.1: Características del ordenador utilizado en el proyecto.

6.2. Herramientas técnicas utilizadas

Se ha empleado el editor de código denominado Visual Studio Code con licencia gratuita. El lenguaje de programación utilizado para el desarrollo de los algoritmos de ML es Python3. Se ha creado un entorno virtual personalizado mediante Anaconda (editor de entornos de desarrollo) para ejecutar todos los algoritmos con sus respectivas dependencias en entornos aislados, evitando errores entre versiones.

6.3. Librerías utilizadas

Las librerías de código aportan funcionalidades desarrolladas por terceras personas, normalmente en equipos de trabajo profesionales. Pueden ser de pago o abiertas a la comunidad de desarrolladores y su uso se basa en facilitar la codificación mediante el uso de funciones predefinidas por otros usuarios. Todas las librerías empleadas en este proyecto son gratuitas y se han instalado dentro del editor de código Visual Studio Code, para ser empleadas con el lenguaje de programación Python.

Tabla 6.1: Librerías utilizadas en el desarrollo del proyecto.

FUNCIÓN DE LA LIBRERÍA	NOMBRE DE LIBRERÍA
Tratamiento de datos	pandas = 1.4.2
Procesamiento numérico	numpy = 1.22.4
Representación gráfica	matplotlib = 3.5.2
Control temporal de la ejecución del código	tqdm = 4.64.1
Generación de gráficas interactivas	pandas-profiling = 3.3.0
Desarrollo de modelos de ML	scikit-learn = 1.1.2
Exportación de parámetros de redes entrenadas	joblib = 1.1.0

Capítulo 7

Resultados

7.1. Introducción

En esta sección se explican los resultados preliminares que se han logrado implementando los modelos mencionados en el apartado 5.2. Cada uno de ellos es explicado comparando el resultado de las predicciones respecto a los valores que debería dar en la realidad (recordemos que se trata de aprendizaje supervisado, tenemos factores y sus respectivos resultados en la vida real) tanto de manera gráfica, representando los puntos de predicción como de manera cuantitativa, mostrando los porcentajes de predicción y error de cada clase (las clases son los diferentes sensores analizados en este proceso).

Después de comparar varios de los modelos desarrollados y ver cual es la estructura que mejor se adapta a este problema, se decide cuál va a ser el modelo predictivo final, con intención de poder implementar modificaciones y/o mejoras a futuro.

7.2. Resultados de Series Temporales

Siguiendo el desarrollo explicado en el capítulo anterior, los resultados obtenidos con este algoritmo son los siguientes. Por una parte, se reparten los datos en entrenamiento y testeo, teniendo en cuenta que el paquete de testeo (zona naranja de la figura 7.1) debe ser el 20 % del set final de datos de la serie temporal. Los datos se recogen de esta manera porque para entrenar un modelo de series temporales es necesario mantener el orden temporal entre mediciones, ya que de no hacerlo, no habría una línea temporal sobre la que buscar patrones, y no funcionaría el modelo predictivo.
Para poder entrenar el modelo, se emplean 2 ventanas temporales que se encargan de aprender de los valores del pasado y predecir valores del futuro. La primera de ellas recorre los datos del pasado, aprendiendo tendencias y patrones que la otra ventana aprovecha para poder predecir valores a futuro. Observar figura 7.1.

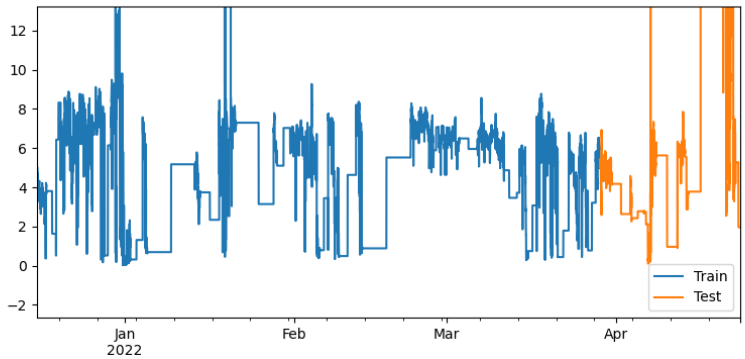


Figura 7.1: Partición real de los datos de serie temporal. Vista general de los datos de train-test.

Después de repartir los datos en una proporción 80-20, se estiman las ventanas móviles que usan los datos de entrenamiento para la predicción, generando el siguiente resultado (figuras 7.2 y 7.3):

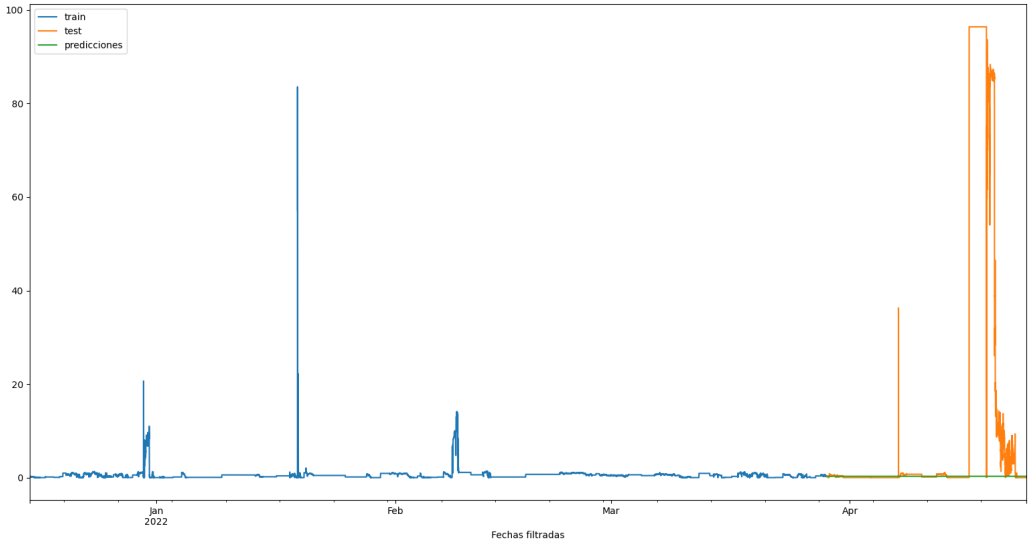


Figura 7.2: Resultado de la predicción de Series Temporales. Línea verde sobre la línea naranja.

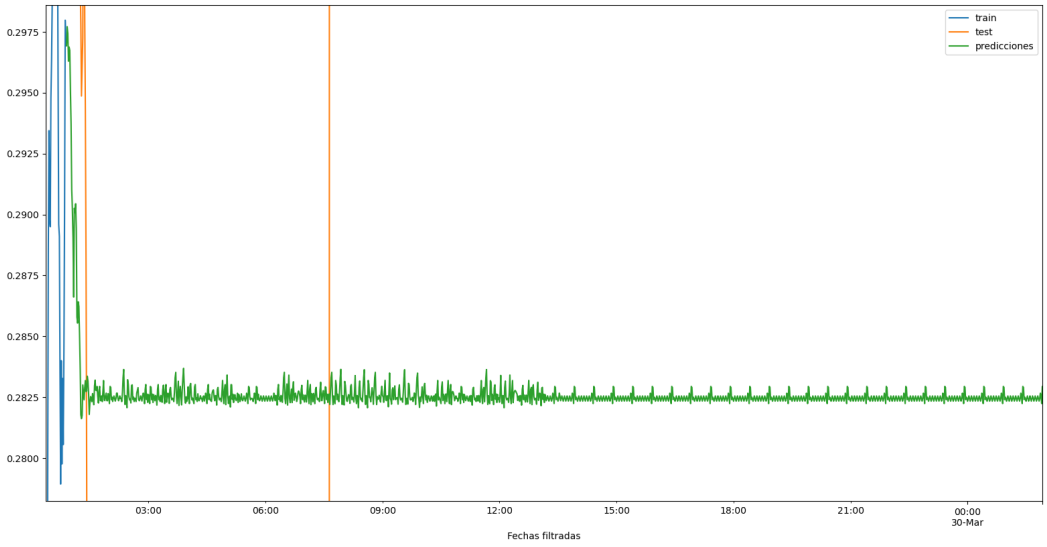


Figura 7.3: Vista aumentada de la sección de datos sobre la zona de predicción.

Teniendo en cuenta la capadidad de procesamiento de la que dispone el ordenador, se estima que la ventana temporal del pasado (*lag*) que se va a emplear para predecir datos del futuro no debe superar los 7 días. De este modo, la manera óptima de predecir datos a futuro sería obtener un vector con la misma longitud que los datos del *lag* (7 días de datos). El problema de esto radica en que a la hora de predecir valores, es necesario abarcar datos en la misma cantidad que el set de datos de testeo, de forma que se pueda hacer una comparativa entre realidad y predicción y estimar el porcentaje de acierto del modelo. En la imagen se observa el resultado de predicción de la serie temporal con un *lag* de 10080 minutos (7 días completos de datos), con un *step* de predicción a futuro de 26 días (representado con la línea verde de la figuras 7.2 y 7.3).
Se ve claramente como ni siquiera se ajusta a la realidad de los datos que debería predecir, ya que debería seguir la línea de tendencia naranja (datos de testeo, figura 7.2), y solamente predice ruido, que a partir de cierto punto en el eje X se convierte en una línea con picos intermitentes constantes.

Por otra parte, el tiempo de procesamiento de las predicciones es de 30 minutos para conseguir los resulta-

dos que se aprecian en verde. Está claro que no es un resultado certero, y que en relación procesamiento-resultados requiere de demasiada capacidad de procesamiento, y si solamente con 7 días de datos no ha sido capaz de llegar más allá en el cálculo de variables, el hecho de plantear un *lag* de entrenamiento de 4 meses es un proceso inviable en este caso. Con este método y la capacidad de este ordenador, no hay opción de conseguir datos predictivos que tengan consistencia como para conseguir una solución funcional, de manera que no se profundiza en el desarrollo y modificación de hiperparámetros de este modelo, y se da paso al desarrollo del siguiente, que requiere de otras características de entrenamiento.

7.3. Resultados de algoritmos de ensamble

7.3.1. ADABOOST

En este caso, el procedimiento para separar los datos es igual que el modelo de series temporales, repartiendo el 80 % de los datos para entrenamiento del modelo y el otro 20 % para validarlo, con la única diferencia que en este caso los datos son escogidos aleatoriamente dentro del set de datos, sin necesidad de mantener una relación temporal entre una fila de datos y la siguiente.

Después de entrenar el modelo y validarlo con los datos correspondientes, se representan los resultados de la relación entre los datos predichos y los de la realidad (figura 7.4). El eje X representa los valores esperados y el eje Y los valores inferidos por el algoritmo. La línea a 45º representa como de bien se ajustan esos datos, devolviendo un resultado en formato de porcentaje que refleja la precisión en cada caso.

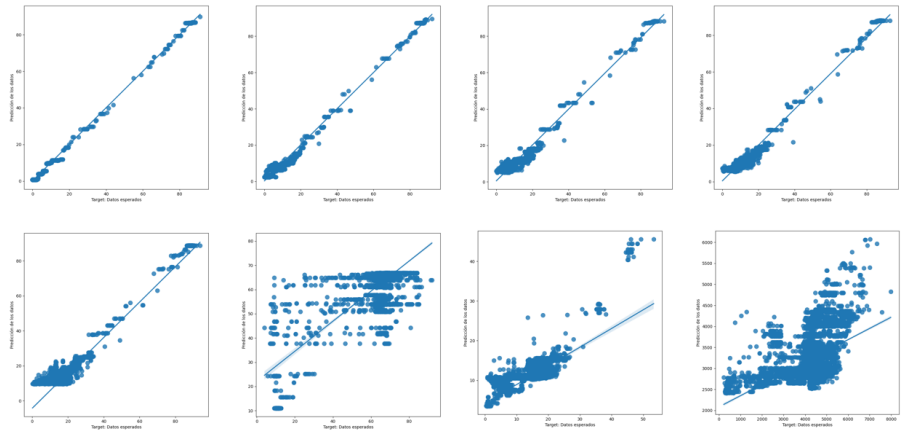


Figura 7.4: Resultados de precisión de las predicciones de cada uno de los 8 sensores de salida del PSD usando Adaboost.

En la siguiente tabla se muestran los resultados de precisión y error de cada una de las gráficas de la figura 7.1

Tabla 7.1: Resultados del algoritmo Adaboost.

ADABOOST	7400	7401	7402	7403	7404	7405	7406	7407
Precision Train	0,9980	0,9925	0,9805	0,9768	0,7283	0,4841	-0,1806	-1,1654
Precision Test	0,9981	0,9924	0,9797	0,9760	0,7240	0,4535	-0,2757	-1,2188
Error MSE	0,05548	0,0614	0,0515	0,0578	0,1165	0,0244	0,1331	0,1824

El procedimiento a la hora de validar el modelo se basa en contrastar la precisión obtenida del modelo sobre los datos de entrenamiento vs los datos de testeo, es decir, después de tener entrenado el modelo, se comprueba como de bien predice con datos de entrenamiento y como de bien con datos de testeo. Cada uno de ellos va a dar un resultado de 0-100. Si el resultado de precisión introduciendo datos de entrenamiento es muy alto, pero el resultado de precisión de los datos de testeo es claramente más bajo, quiere decir que el modelo está sobreajustado a los datos de entrenamiento y no va a ser capaz de generalizar bien con datos nuevos. En cambio, si los resultados de precisión son similares tanto en entrenamiento como en testeo, quiere decir que el modelo está bien entrenado, y va a ser capaz de predecir valores con

datos nuevos (Siempre y cuando el porcentaje de precisión del modelo sea alto, superior al 90 por ciento por ejemplo).
Como se puede apreciar, en los primeros 6 sensores de medición de PSD los resultados de predicción son muy certeros, perdiendo precisión en los 4 últimos, donde no es capaz de generalizar bien los resultados. Se puede ver que el modelo está entrenado correctamente, pero que la precisión a la hora de generar predicciones no es suficiente en los sensores 7404 (precisión de 72,40 %), 7405 (precisión de 48,41 %), 7406 y 7407 que tienen valores de precisión negativos, lo que representa que el modelo ni siquiera se acerca a poder estimar buenas predicciones.

Teniendo en cuenta estos resultados respecto a los que se han logrado mediante el algoritmo de series temporales, el resultado es bastante mejor de lo estimado, aunque no es lo suficientemente bueno como para dar por finalizado el trabajo. Viendo que este algoritmo funciona razonablemente bien, y basándose en la experiencia del autor de este TFM, se propone desarrollar otro algoritmo similar a este, con un motor y procesamiento diferentes.

7.3.2. XGBOOST

La repartición de los datos de entrenamiento y testeo se hacen de la misma manera que con Adaboost. En cuanto al entrenamiento y validación del algoritmo también se procede de la misma manera. A continuación en la figura 7.5 los resultados obtenidos:

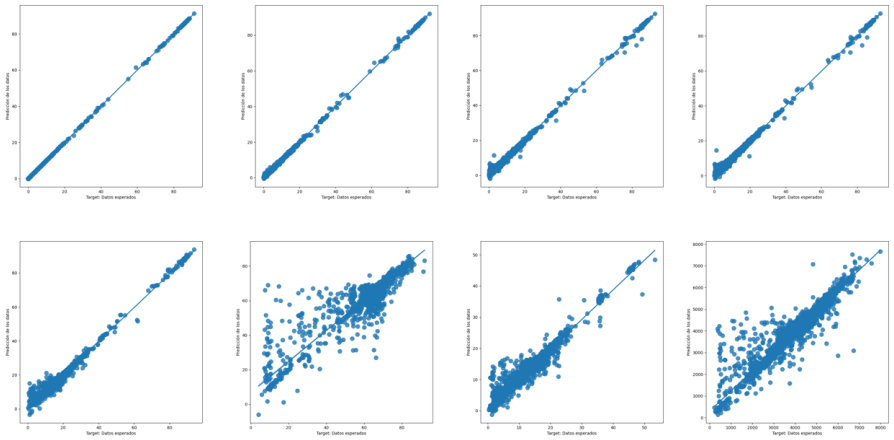


Figura 7.5: Resultados de precisión de las predicciones de cada uno de los 8 sensores de salida del PSD usando XGBoost.

En ete caso, se aprecia visualmente que los datos predichos se ajustan mejor a la realidad que con el modelo anterior, porque aunque en alguno de los sensores haya un poco de dispersión, los puntos son cercanos a la recta de validación del modelo. Los últimos 3 sensores son los que más dispersión tienen en las predicciones creadas, aunque generalizan muy bien todas ellas. Para cuantificar la precisión de cada inferencia, se muestra la siguiente tabla de resultados de precisión y error de cada sensor:

Tabla 7.2: Resultados del algoritmo XGBoost.

XGBOOST	7400	7401	7402	7403	7404	7405	7406	7407
Precision Train	0,9999	0,9999	0,9994	0,9993	0,9971	0,9724	0,9854	0,9780
Precision Test	0,9999	0,9997	0,9987	0,9984	0,9935	0,9086	0,9604	0,9399
Error MSE	5,42E-06	0,0020	0,0050	0,0055	0,0068	0,0071	0,0089	0,0099

En este caso se observa como todos los porcentajes de precisión del modelo superan el 90 %. Todas las comparaciones entre precisiones de entrenamiento-testeo son muy similares entre sí, confirmando el hecho de que el modelo no está sobreajustado. Los únicos dos casos en los que la comparativa es ligeramente diferente entre ambos es en los sensores 7405 y 7407, donde los resultados de precisión de entrenamiento son del 97,24 % y los de testeo 90,86 % y 93,99 % respectivamente, lo que marca un error de precisión del 5 % o más en ambos casos. Esto sugiere que el modelo está ligeramente sobreajustado en esos dos sensores. Teniendo en cuenta que el resto de ellos predice con porcentajes cercanos al 100 %, se estima que

la mejor maniobra para mejorar los resultados de esos dos sensores es modificando los hiperparámetros del modelo en ambos casos, buscando un mejor ajuste posible para cada uno, sin modificar el modelo para el resto de los casos.

7.4. Modelo final

El modelo final escogido para este problema es el algoritmo XGBoost,ya que es el que mejor adapta sus predicciones al set de datos, no requiere de demasiado procesamiento, y aunque en 2 de los sensores de PSD no ajuste su resultado más que al 90 %, tiene margen para poder modificar sus hiperparámetros, pudiendo conseguir ajustarlo en esos sensores a porcentajes superiores a los actuales.

Los resultados de precisión y error finales usando ese algoritmo son los de la tabla 7.3:

Tabla 7.3: Resultado final de las diferentes precisiones obtenidas en la predicción de cada sensor de PSD.

XGBOOST	**7400**	**7401**	**7402**	**7403**	**7404**	**7405**	**7406**	**7407**
Precisión	0,9999	0,9997	0,9987	0,9984	0,9935	0,9086	0,9604	0,9399
Error MSE	5,42E-06	0,0020	0,0050	0,0055	0,0068	0,0071	0,0089	0,0099

Como breve resumen, se muestran los hiperparámetros escogidos intermanente en el algoritmo, a fin de poder modificar sobre esto una mejor opción en los sensores que peor ajuste han dado. Observar tabla 7.4.

Tabla 7.4: Hiperparámetros empleados en el algoritmo XGBoost.

XGBOOST	Descripción	Hiperparam.
subsample	Ratio para entrenamiento de instancias	1
colsample_bytree	Fraccion de columnas elegidas para el arbol	1
solcample_bylevel	Fraccion de columnas elegidas para entrenar ese nivel	1
gamma	Mínima reducción de pérdida para generar un split	0
booster	Tipo de booster escogido	”gbtree”
learning_rate	Paso de aprendizaje	0,3
max_depth	Profundidad maxima	6
monotone_constraints	Incremento de restriccion de los predictores	”( )”
n_stimators	Número de arboles de impulso	100
reg_alpha	Regularización L1	0
reg_lambda	Regularización L2	1
scale_por_weight	Parámetro contra data desvalanceada	1

Capítulo 8

Conclusiones

8.1. Resumen del trabajo

El trabajo abarca desde la extracción de datos de los sensores instalados por toda la fábrica de cemento, cuya información es almacenada de forma masiva en su base de datos privada, hasta la resolución de algoritmos de ML que sean capaces de resolver el problema de la estimación de PSD del producto, pasando por la preparación y el análisis de los datos extraídos.

Todos y cada uno de esos procesos se han cumplido secuencialmente siguiendo la hoja de ruta especificada en este trabajo, donde se explica cómo se ha desarrollado en detalle cada uno de los puntos y cuáles han sido los resultados.

La extracción de datos se ha hecho mediante una API, almacenando toda la información de cada sensor por individual, en formato .csv. Se han modificado todos los formatos de los datos y se han ensamblado todos los registros de sensores en un único set de datos, de manera que pueda ser transformado con mayor facilidad. Se han aplicado técnicas de procesamiento y análisis exploratorio de datos, para poder entender qué tipo de información se estaba manejando, pudiendo así decidir cuál podría ser el mejor método de aprendizaje automático para este problema.

Se han desarrollado diferentes modelos predictivos, que han sido entrenados con el set de datos creado, teniendo en cuenta cual es la mejor repartición de datos para cada modelo.

Como punto final, se han comparado los modelos predictivos entre sí eligiendo como modelo final el que mejores resultados ha logrado. Ese modelo es el que se instalará en la planta para poder predecir valores de PSD en directo, y poder lograr así una calidad óptima del producto final.

Usando los modelos predictivos definidos en este proyecto y entrenándolos con la información recogida a lo largo de todo el proceso de fabricación del cemento, se ha logrado automatizar una parte del proceso productivo, consiguiendo desarrollar una tecnología digital capaz de controlar automáticamente la calidad del cemento producido en la fábrica.

8.2. Repaso de objetivos cumplidos y conclusiones

El primer punto a mencionar es el hecho de haber logrado finalizar este trabajo de fin de máster, consiguiendo resolver el problema de la planta cementera, cumpliendo el primer objetivo. Tal y como se explicaba en el punto 1.4, donde uno de los puntos a conseguir era finalizar el trabajo con buenos resultados, hecho que se ha comprobado tras la visualización de los resultados finales.

A lo largo de todo el proceso desde la extracción de datos hasta la resolución misma del problema, han ido surgiendo problemas que complicaban el desarrollo. Esos problemas abarcaban fallos a la hora de la extracción de los datos, falta de comunicación por parte de los técnicos de la planta cementera complicando la comprensión de su proceso de fabricación, parones de mantenimiento imprevistos, etc. Todos y cada uno de esos problemas se han ido resolviendo paso a paso, procurando mantener una estructura limpia y ordenada, evitando así que la resolución de un problema generara otro en otra parte del desarrollo.

Las soluciones desarrolladas de IA han cumplido los objetivos, pudiendo ser viable predecir valores de PSD dentro de la planta cementera. El siguiente paso de este trabajo es poder implementar el algoritmo final dentro de sus instalaciones, por lo que será necesario contar con la ayuda de técnicos de la propia planta que ayuden en la integración de esta solución en su proceso de producción, pudiendo así controlar en todo momento la calidad final del cemento, que es en lo que se basa el desarrollo de este trabajo.

El desarrollo de este trabajo ha conseguido evitar el uso de una máquina de medición de PSD automática, con un importe superior a 200.000€ la unidad. El hecho de poder controlar la calidad del cemento sin necesidad de instalar maquinaria adicional, y más aún con ese importe tan elevado, genera una reducción de gastos enorme dentro de la planta cementera, además de colocar a la empresa en una posición más privilegiada dentro del mercado por el hecho de implementar nuevas tecnologías innovadoras dentro de sus instalaciones.

Además, el hecho de automatizar la recogida de datos y gestionarlos a través de IA también ha logrado reducir consumos en el agua de refrigeración empleada en el proceso posterior a la calcinación del clinker ya que el hecho de conocer qué sensores pueden dar información suficiente en la predicción del PSD da la opción de poder regular la calidad del material a través de todos los factores que influyen en el proceso, pudiendo aislar diferentes secciones y optimizarlas mediante IA. Esto conlleva de nuevo la reducción de gastos por parte de la planta cementera ya que se empleará menos agua en el proceso de refrigeración.

La conclusión final a la que se llega es que el hecho de poder implementar nuevas tecnologías como la IA en procesos de fabricación industriales, permite además de mejorar el control y la calidad del producto final (generando mayor valor añadido), sustenta el apoyo al ecosistema del planeta a través de la reducción de costes energéticos y emisiones de CO2.

Capítulo 9

Anexos

9.1. Gráficas

9.1.1. Gráficas de análisis univariable: Funcionamiento y Densidad

En este apartado se muestran todas las figuras generadas en el análisis de variables de este proyecto. Todas ellas son representaciones de los datos tal como han sido recogidos, sin filtrar ni limpiar los datos erróneos.

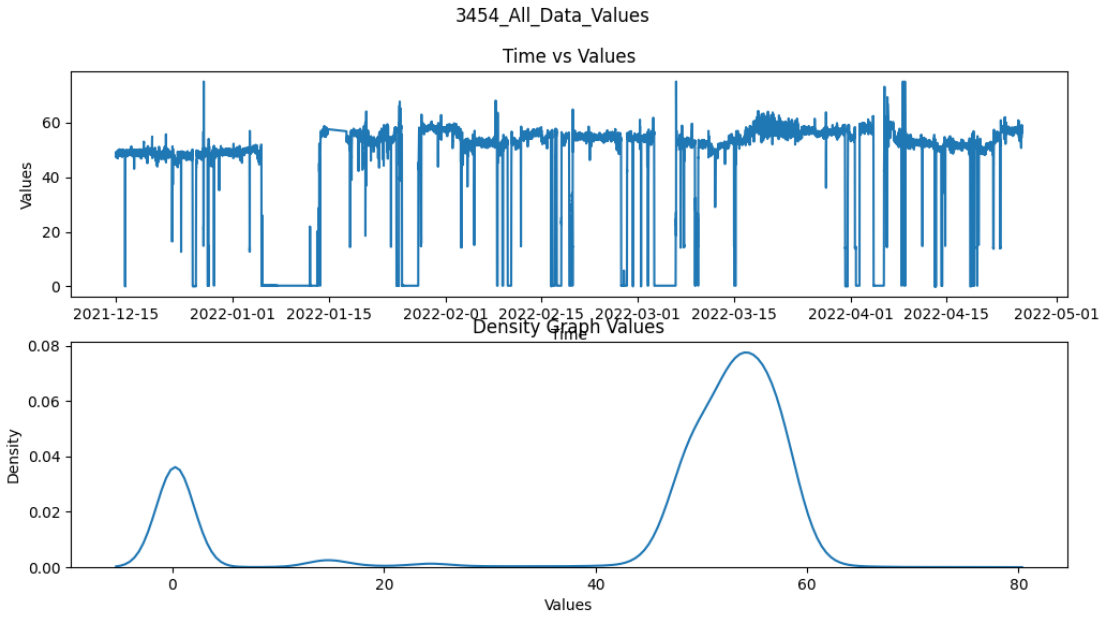


Figura 9.1: Análisis sensor 3454.

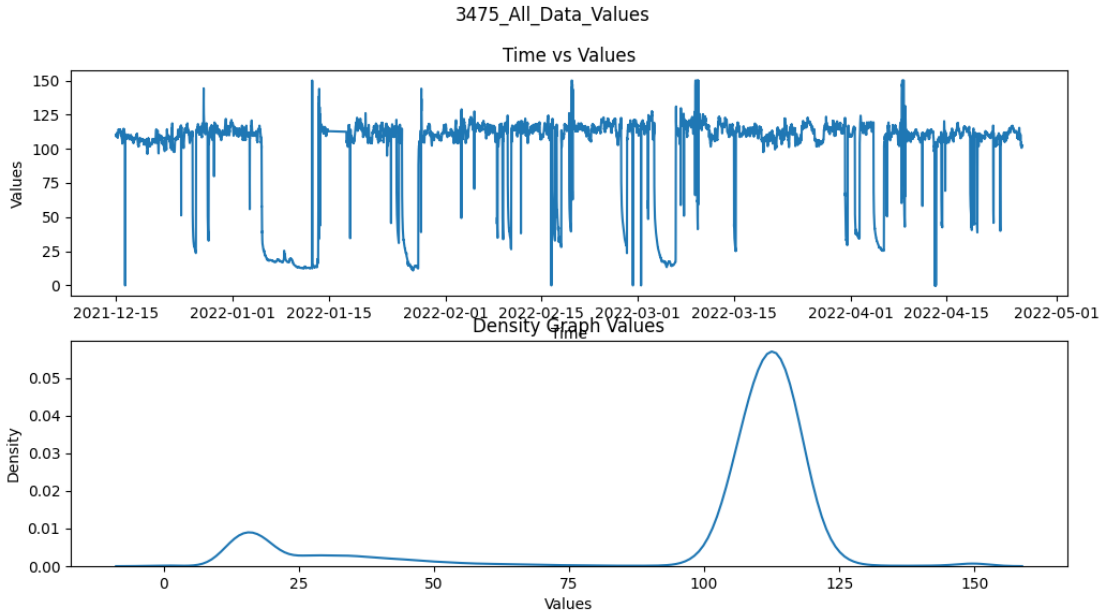


Figura 9.2: Análisis sensor 3475.

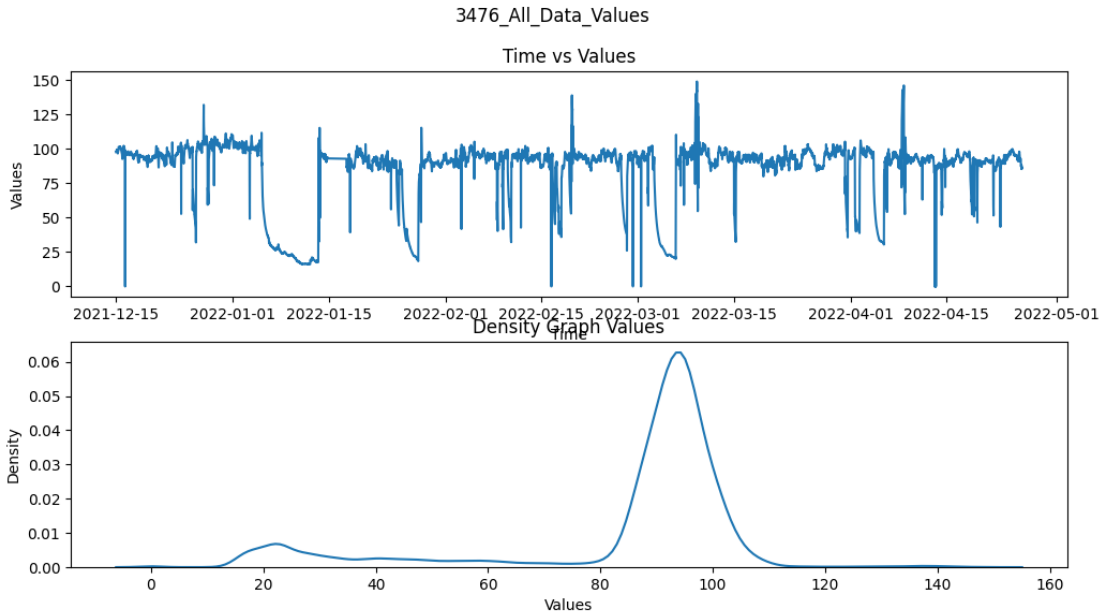


Figura 9.3: Análisis sensor 3476.

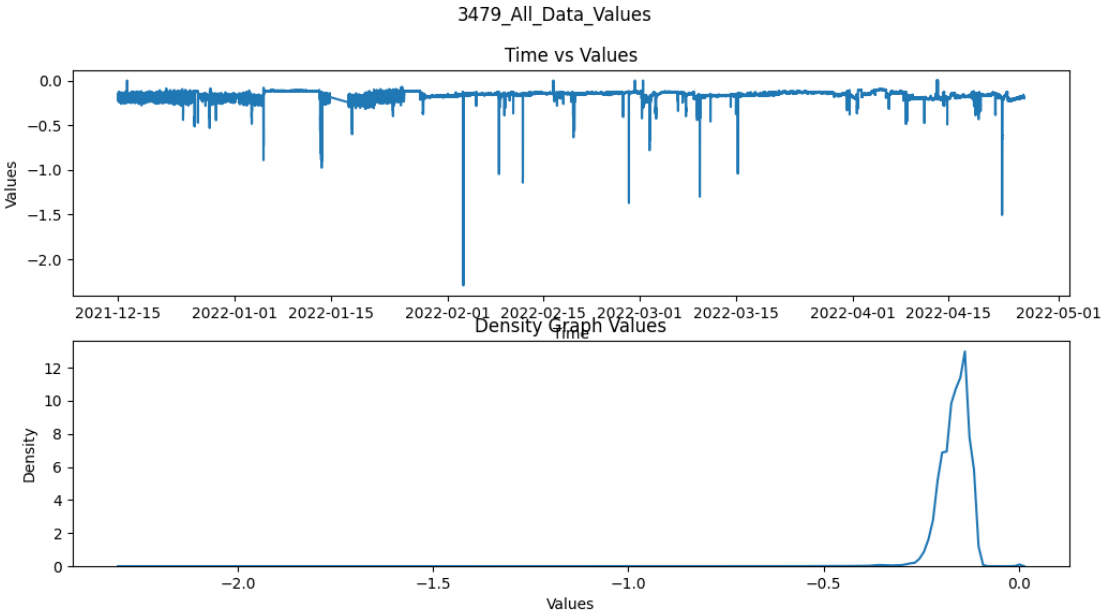


Figura 9.4: Análisis sensor 3479.

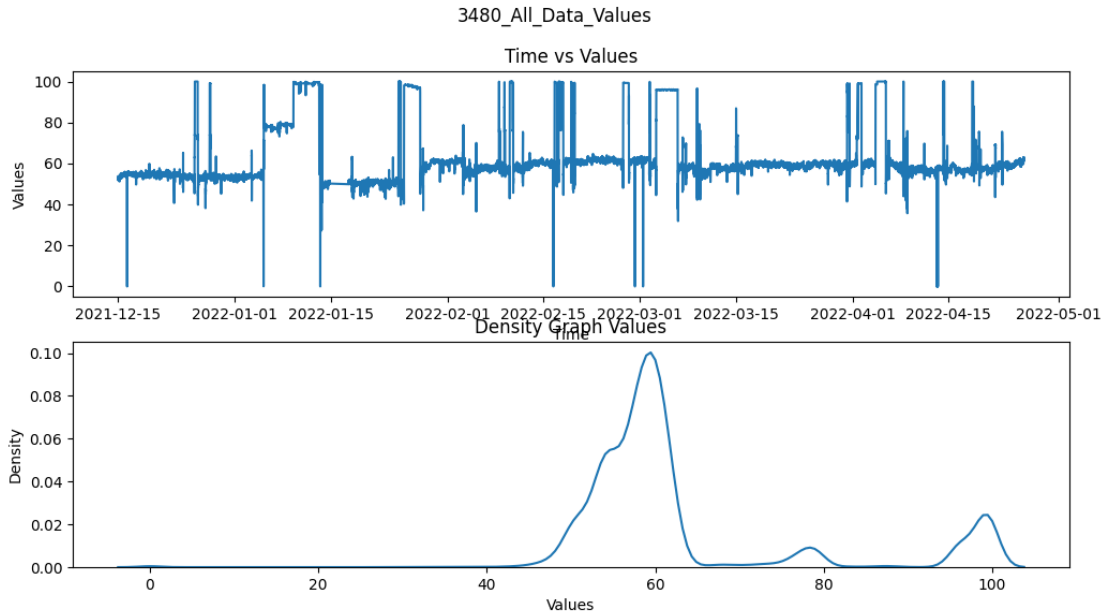


Figura 9.5: Análisis sensor 3480.

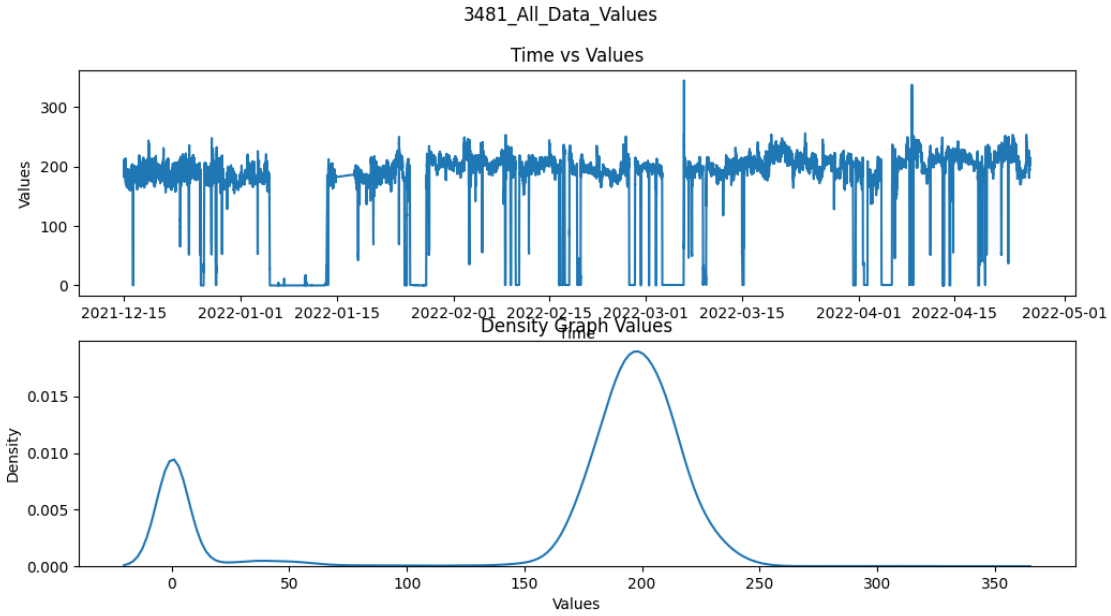


Figura 9.6: Análisis sensor 3481.

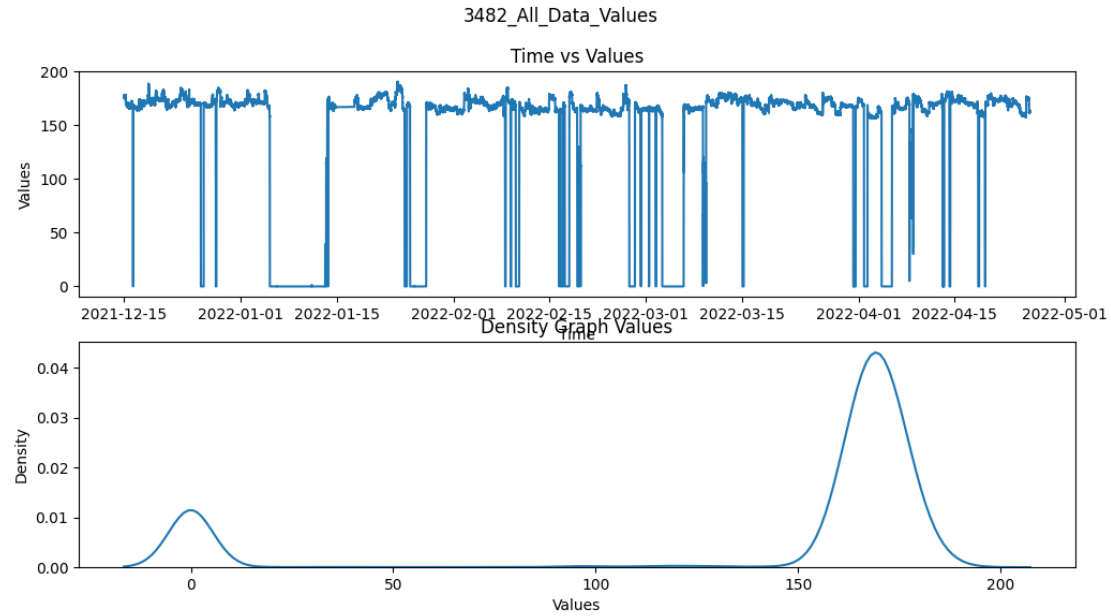


Figura 9.7: Análisis sensor 3482.

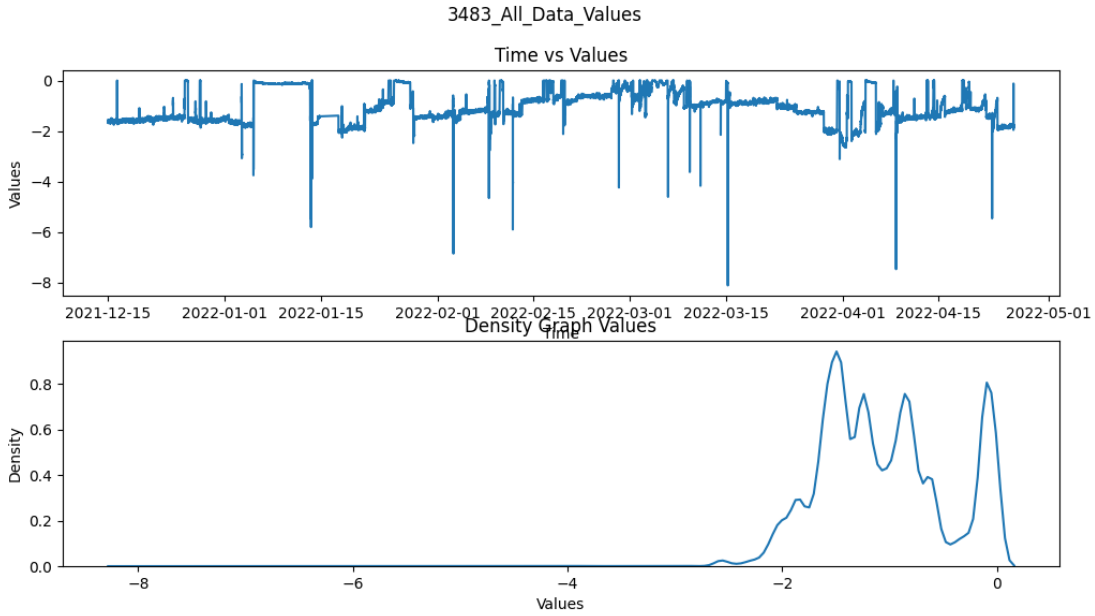


Figura 9.8: Análisis sensor 3483.

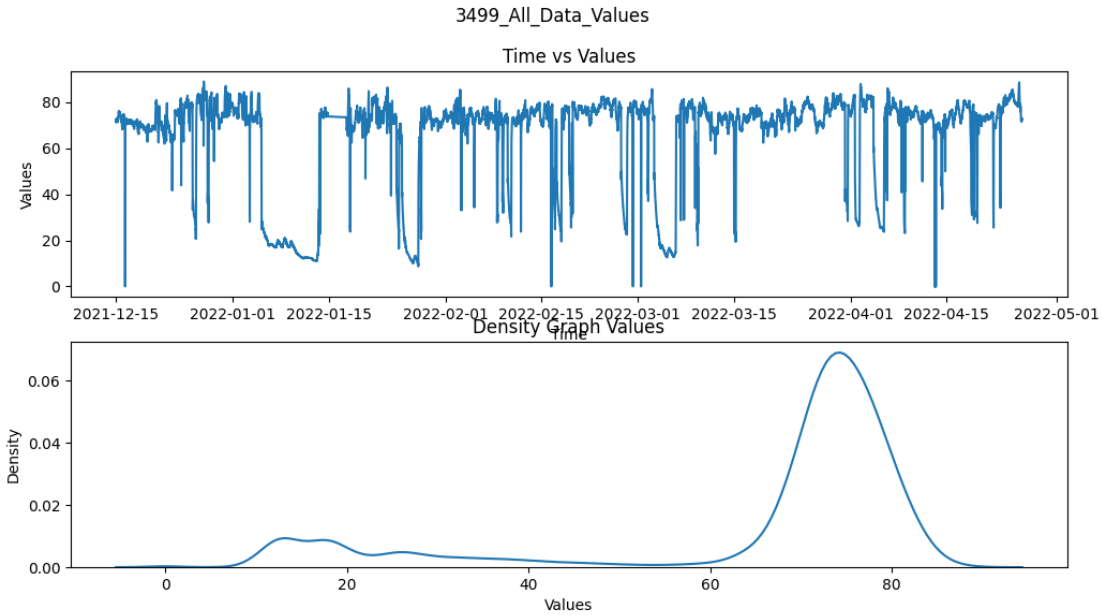


Figura 9.9: Análisis sensor 3499.

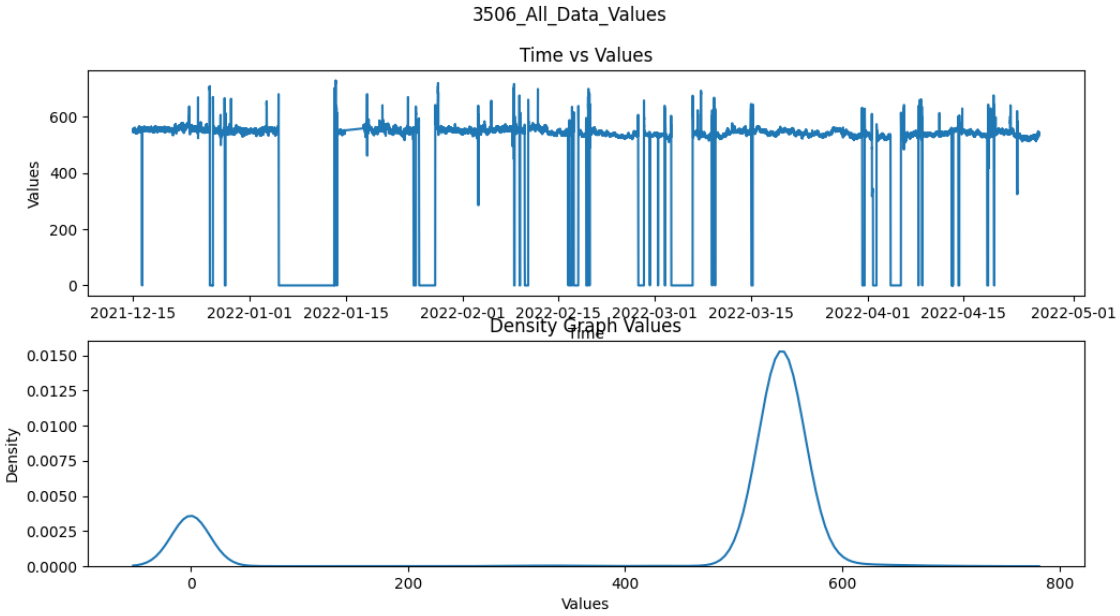


Figura 9.10: Análisis sensor 3506.

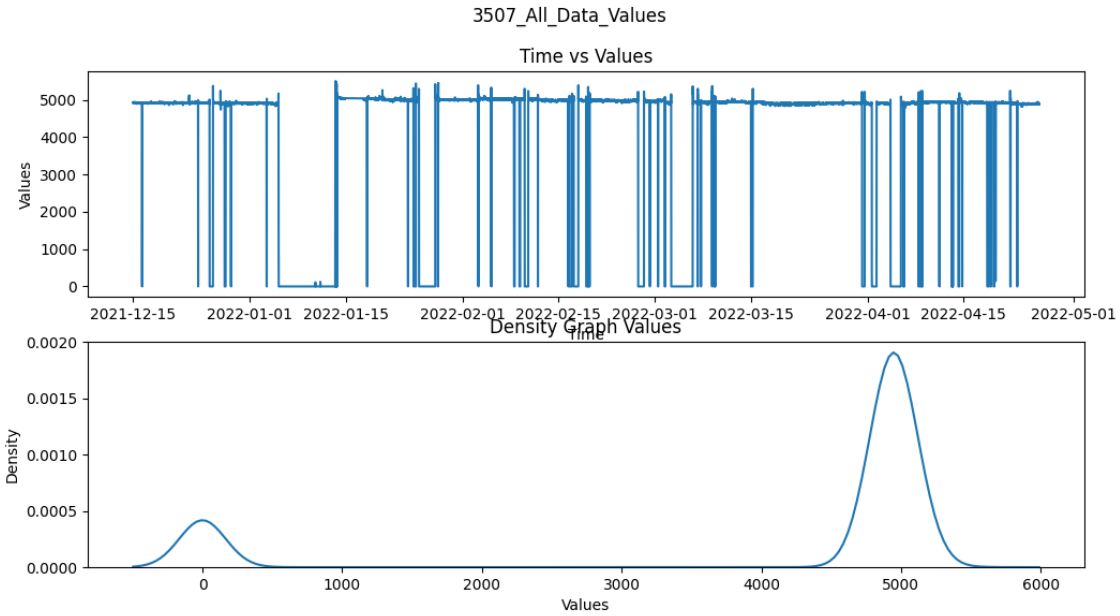


Figura 9.11: Análisis sensor 3507.

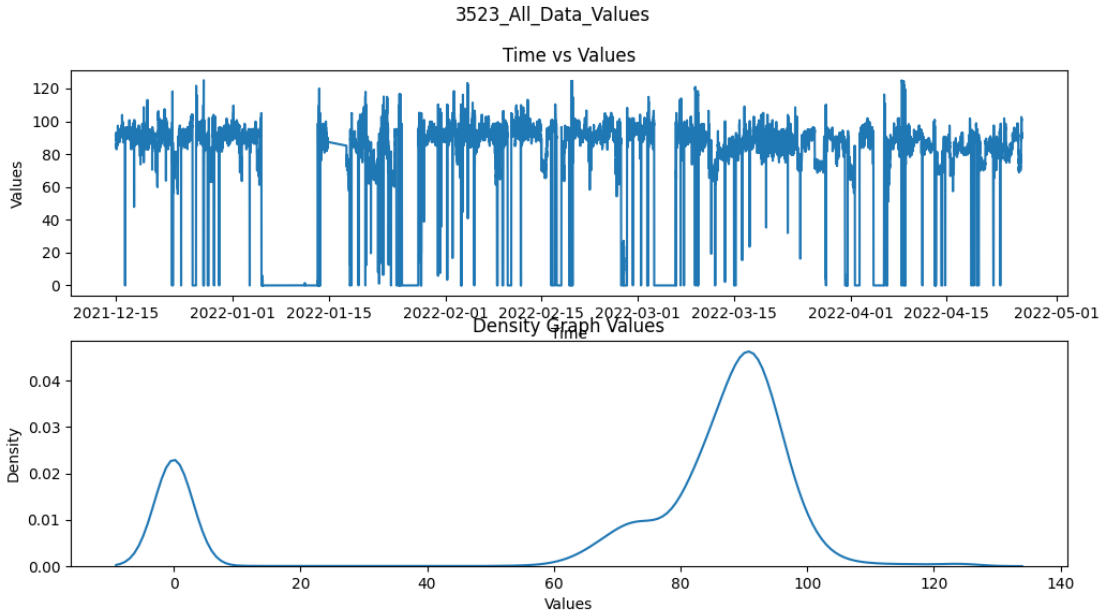


Figura 9.12: Análisis sensor 3523.

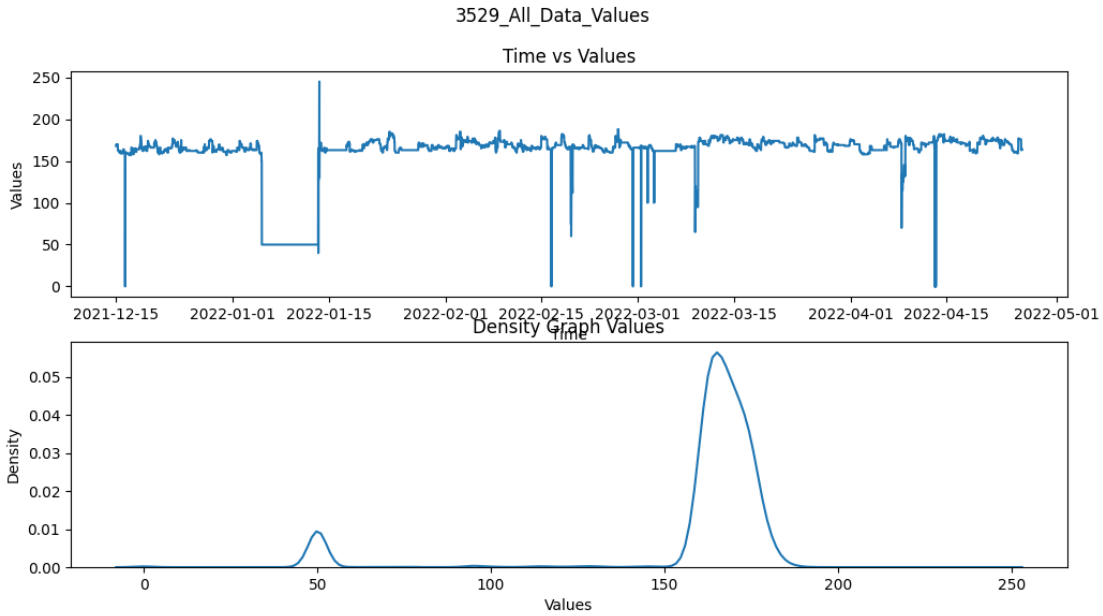


Figura 9.13: Análisis sensor 3529.

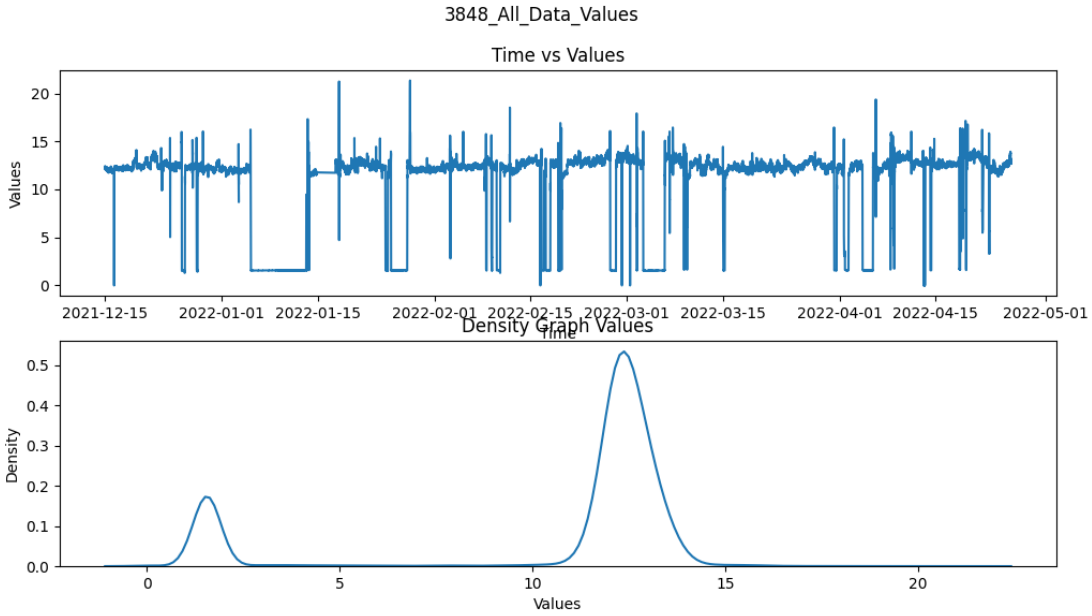


Figura 9.14: Análisis sensor 3848.

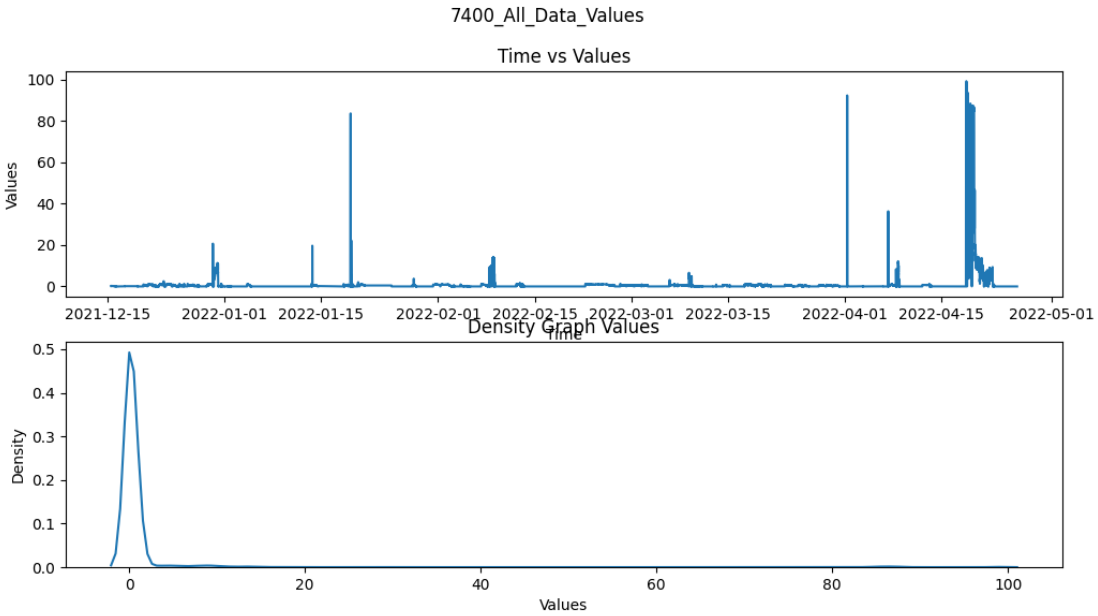


Figura 9.15: Análisis sensor 7400.

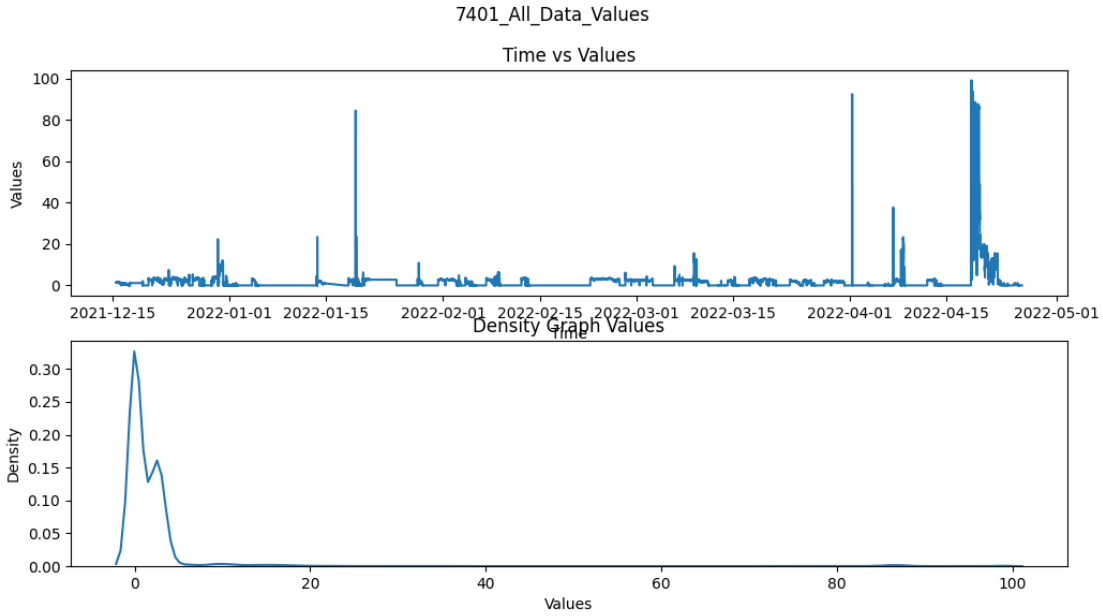


Figura 9.16: Análisis sensor 7401.

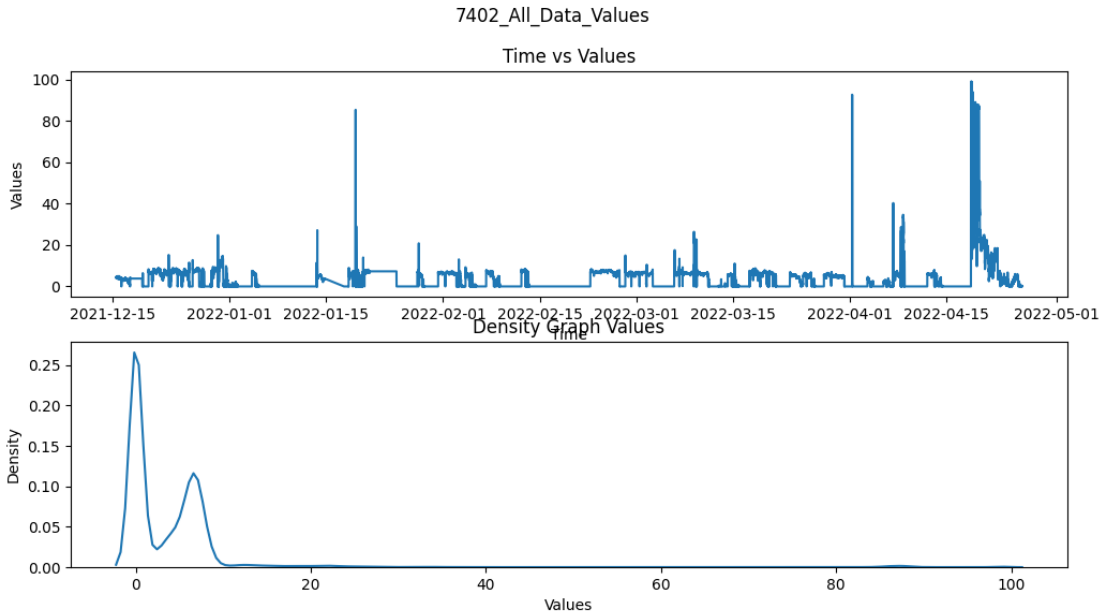


Figura 9.17: Análisis sensor 7402.

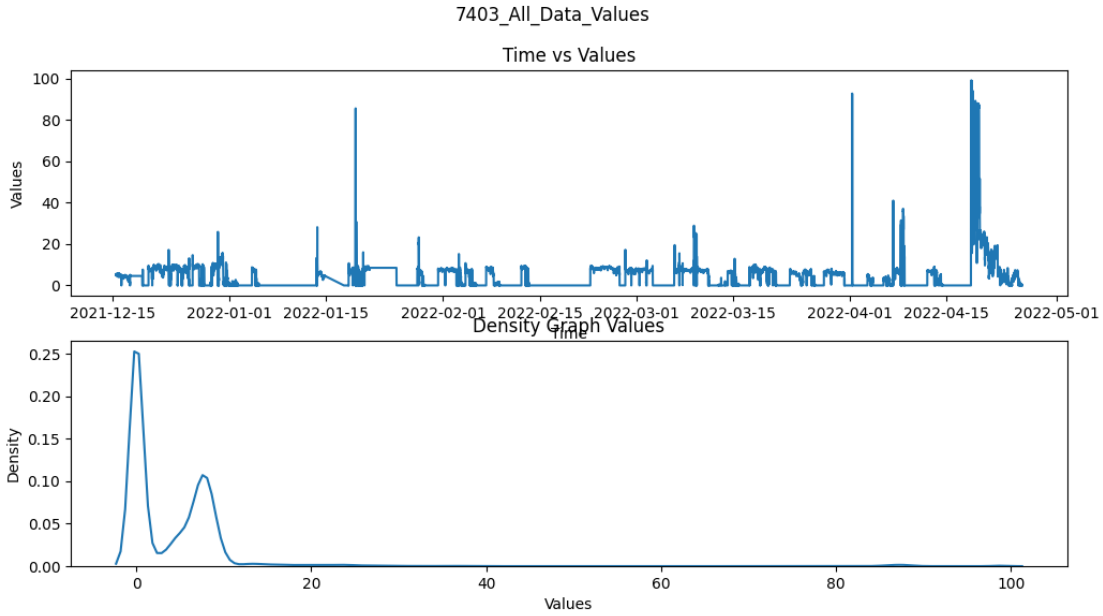


Figura 9.18: Análisis sensor 7403.

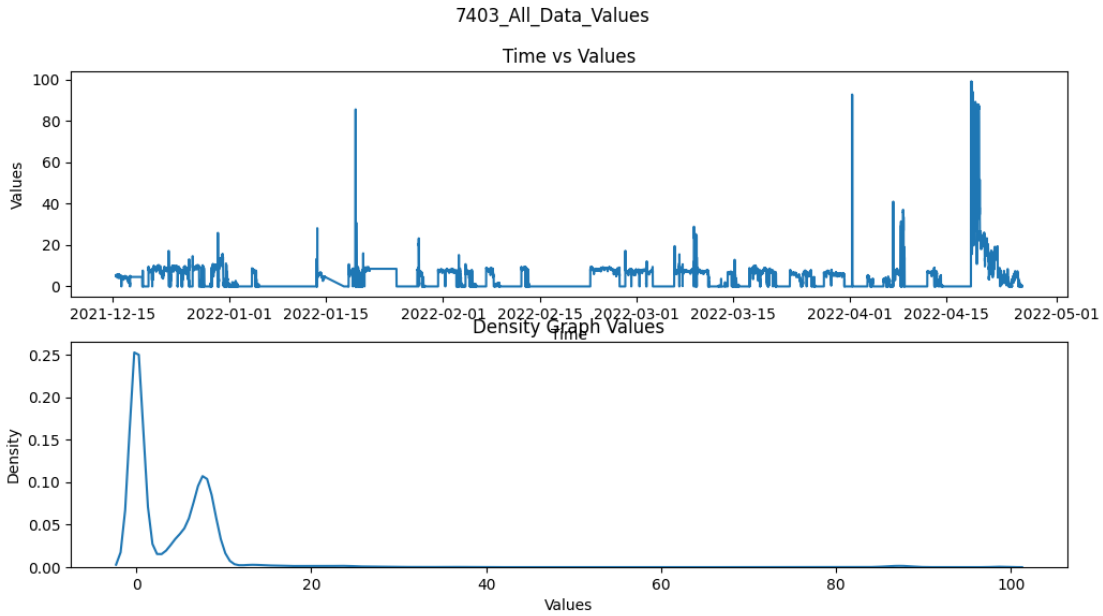


Figura 9.19: Análisis sensor 7403.

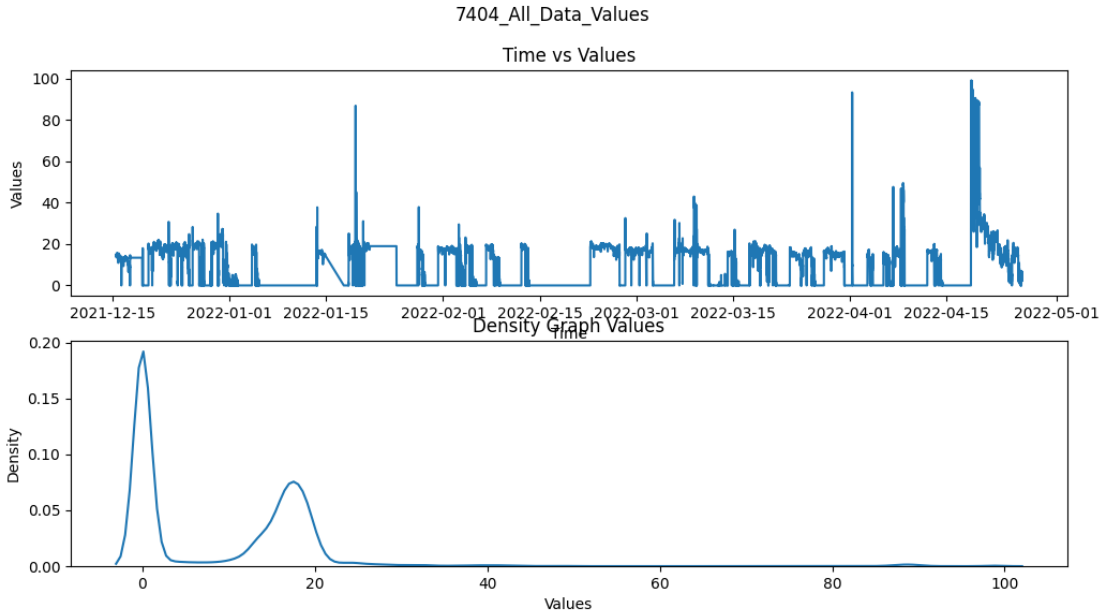


Figura 9.20: Análisis sensor 7404.

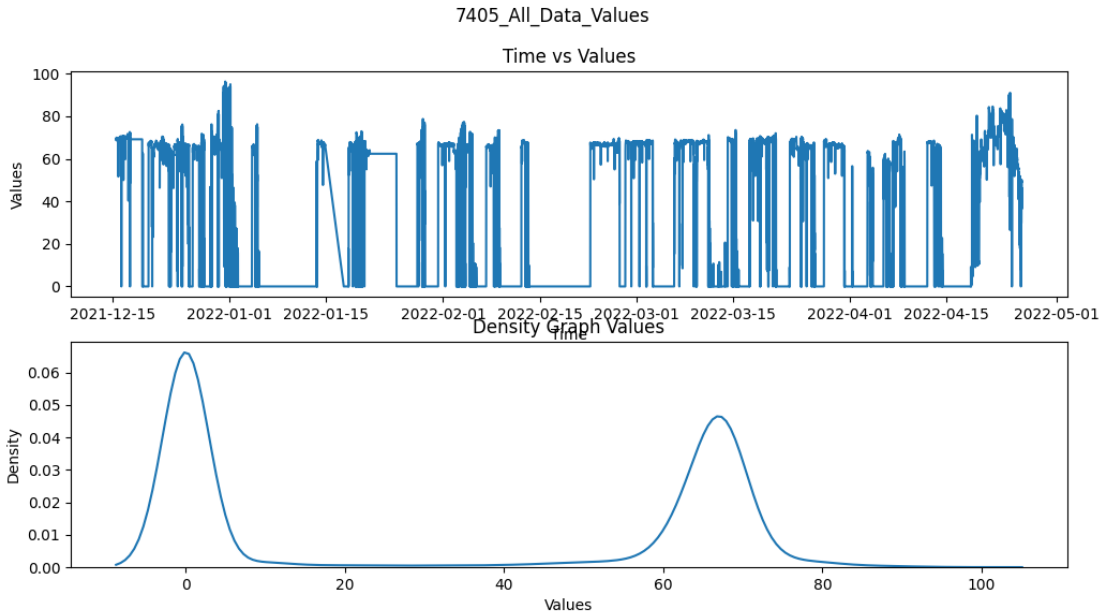


Figura 9.21: Análisis sensor 7405.

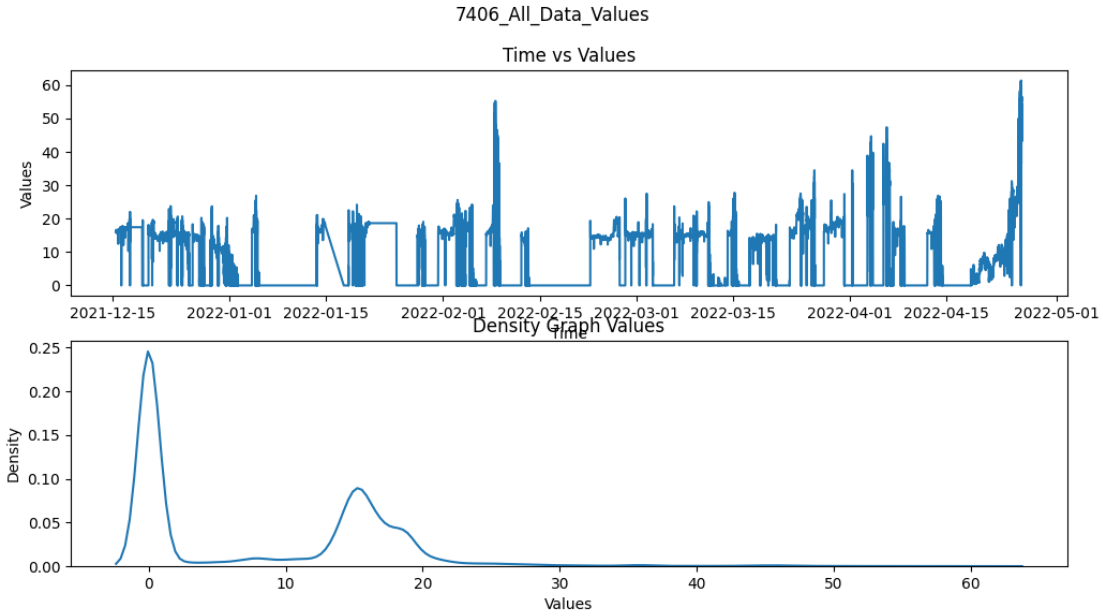


Figura 9.22: Análisis sensor 7406.

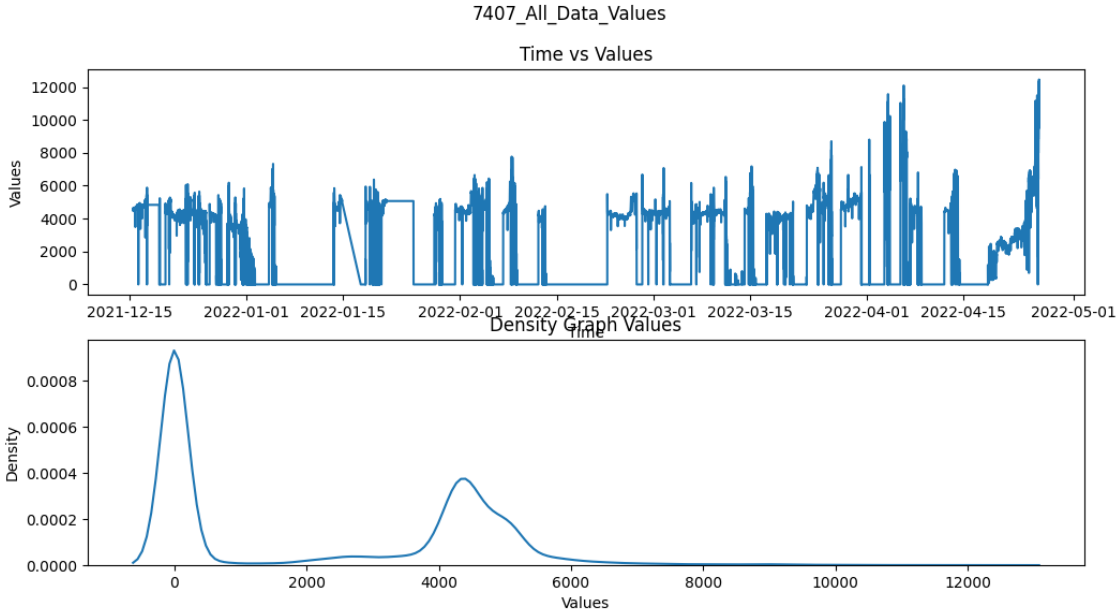


Figura 9.23: Análisis sensor 7407.

9.1.2. Gráficas de datos procesados

En este apartado se muestran todas las figuras generadas en el análisis final de variables de este proyecto, mostrando los resultados habiendo sidos filtradas todas las variables del set de datos (a través del procedo EDA).

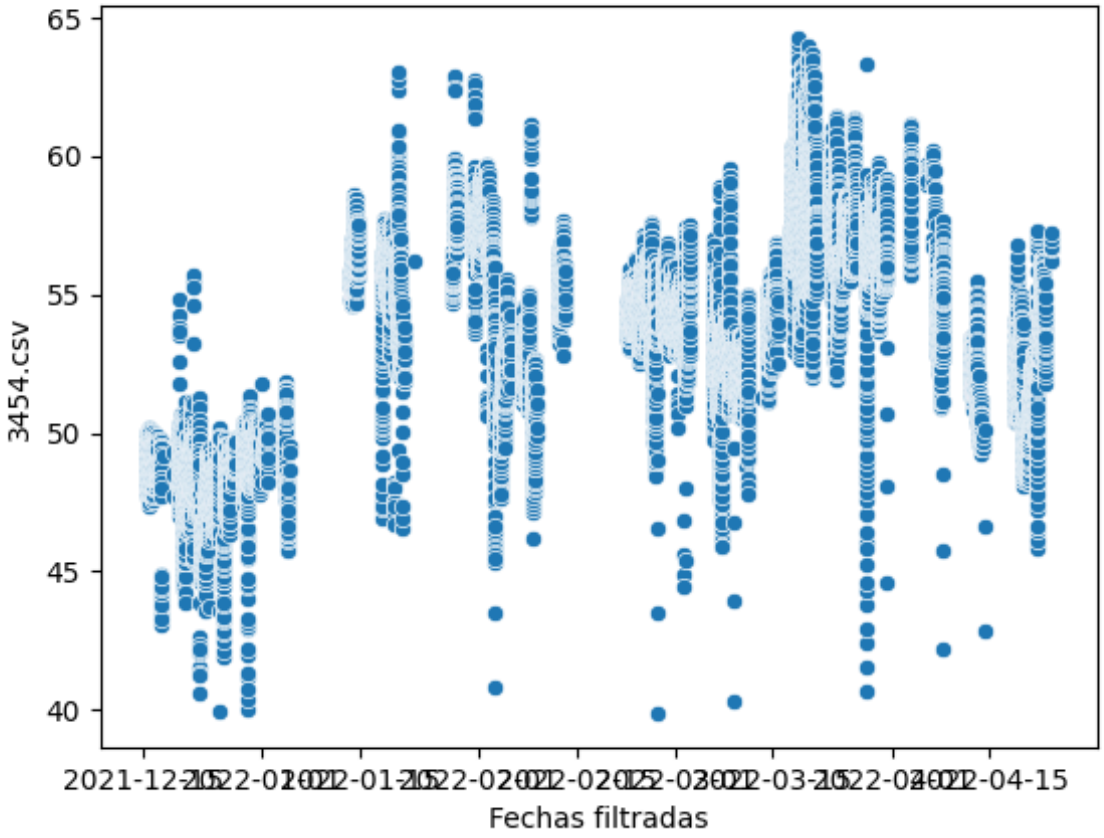


Figura 9.24: Análisis filtrado sensor 3454.

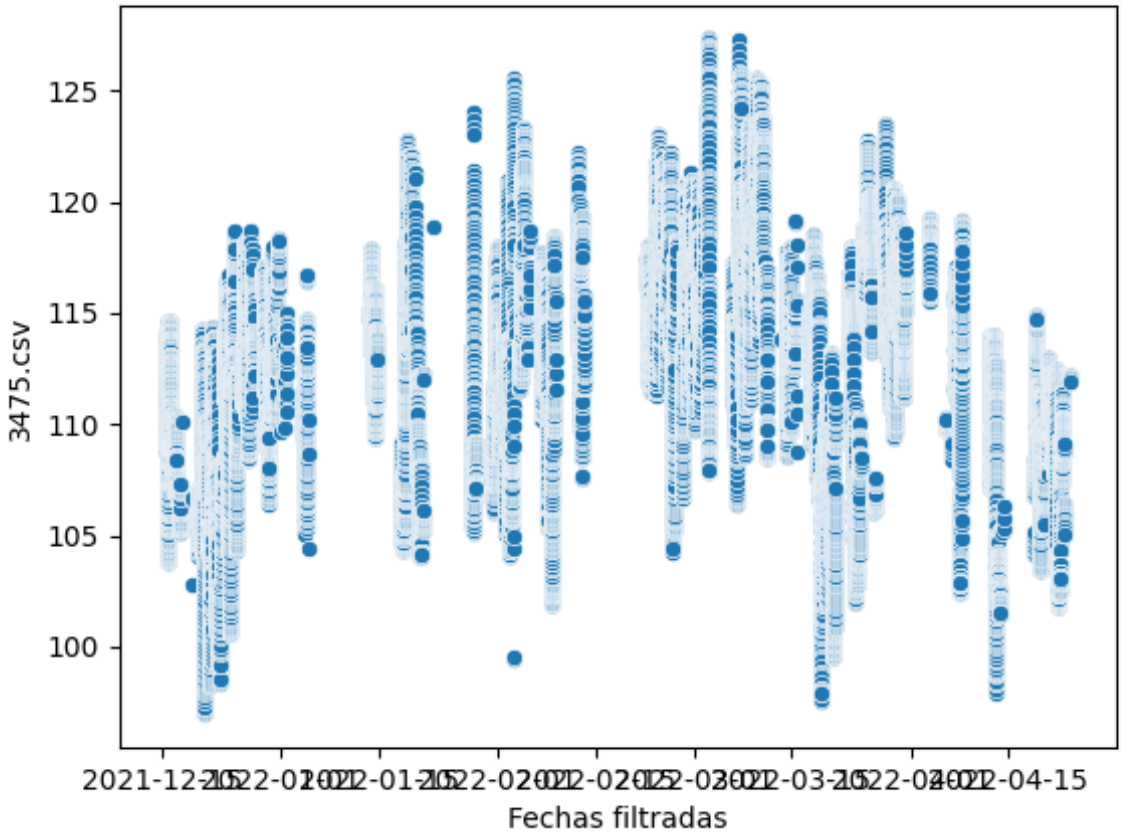


Figura 9.25: Análisis filtrado sensor 3475.

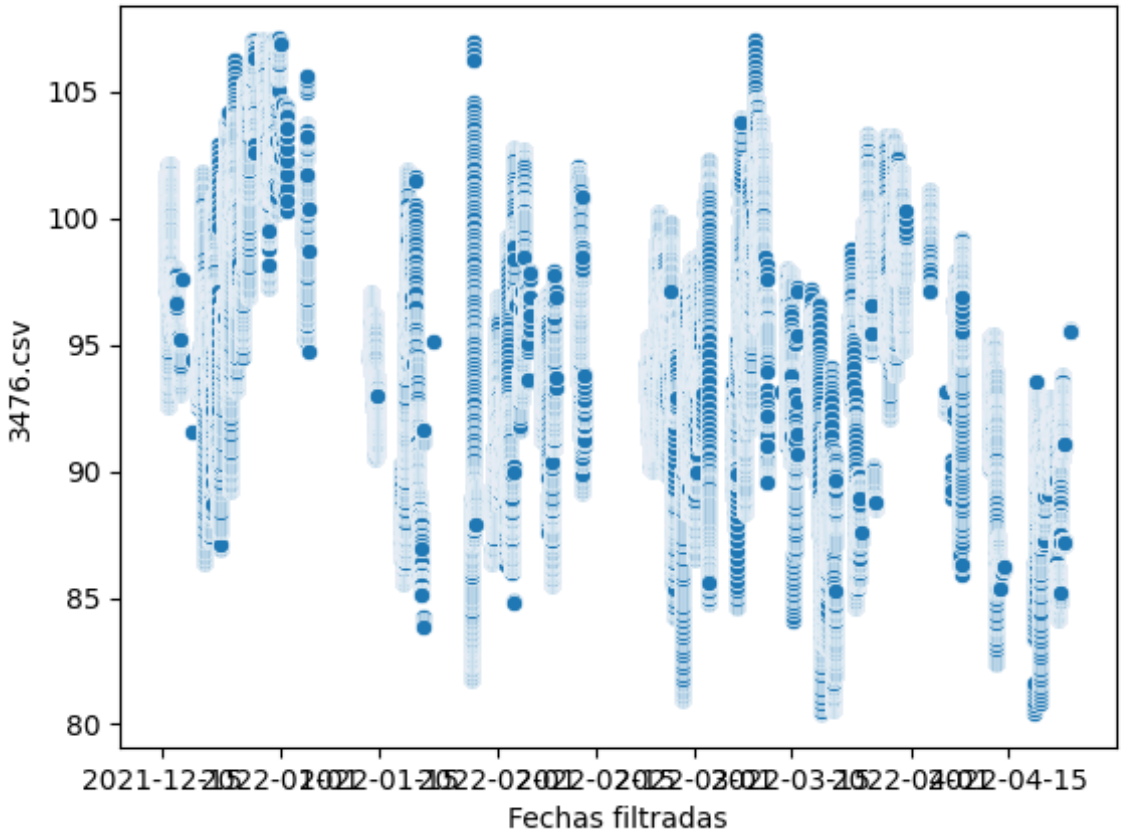


Figura 9.26: Análisis filtrado sensor 3476.

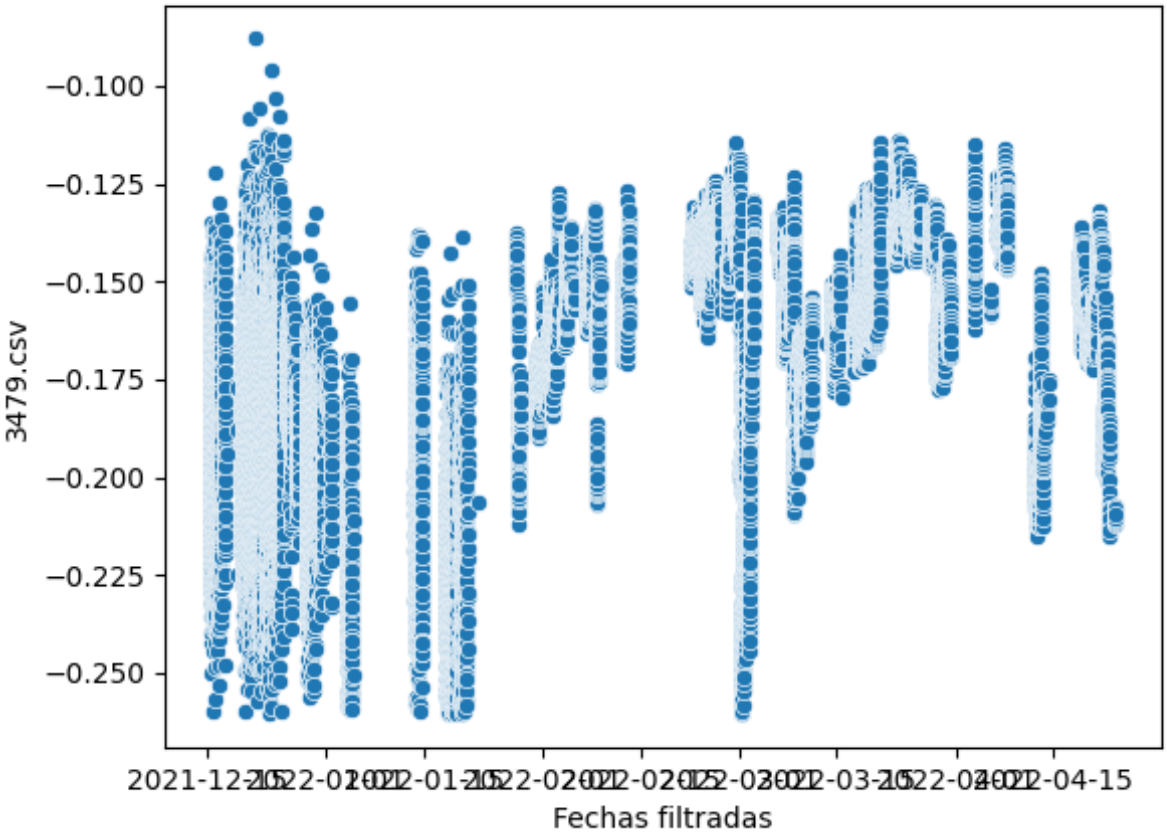


Figura 9.27: Análisis filtrado sensor 3479.

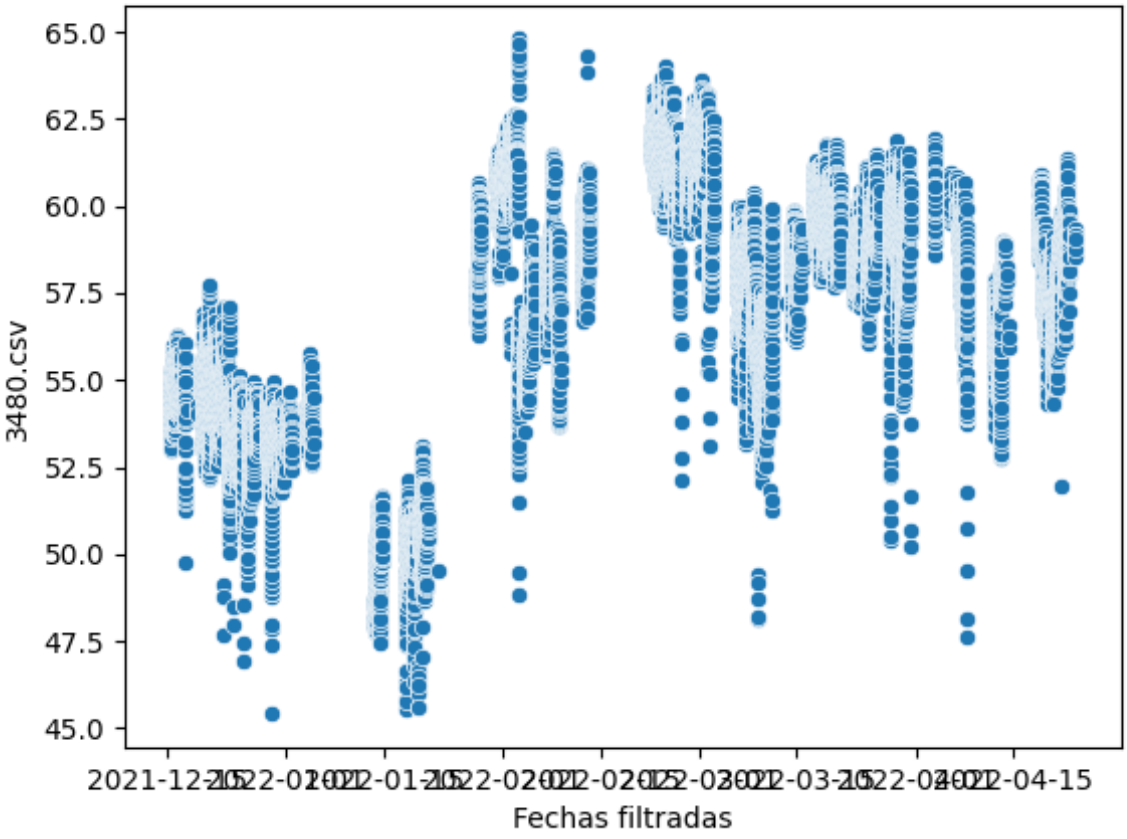


Figura 9.28: Análisis filtrado sensor 3480.

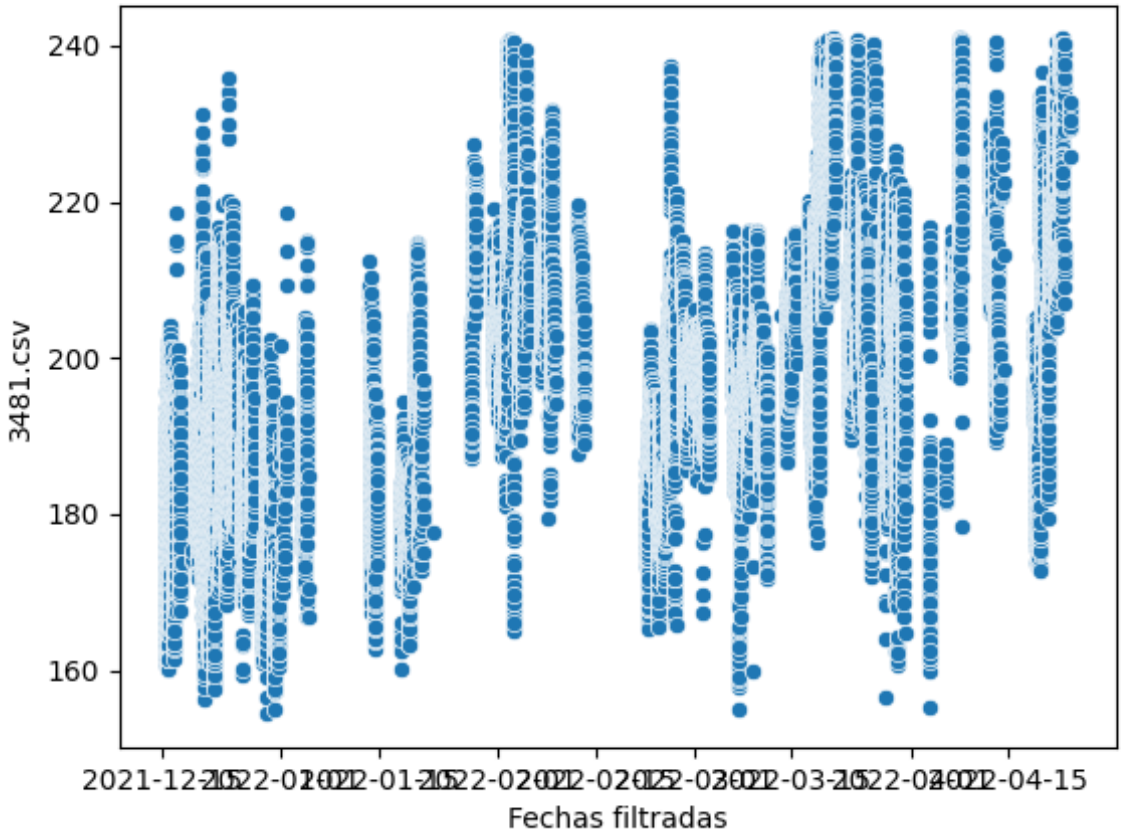


Figura 9.29: Análisis filtrado sensor 3481.

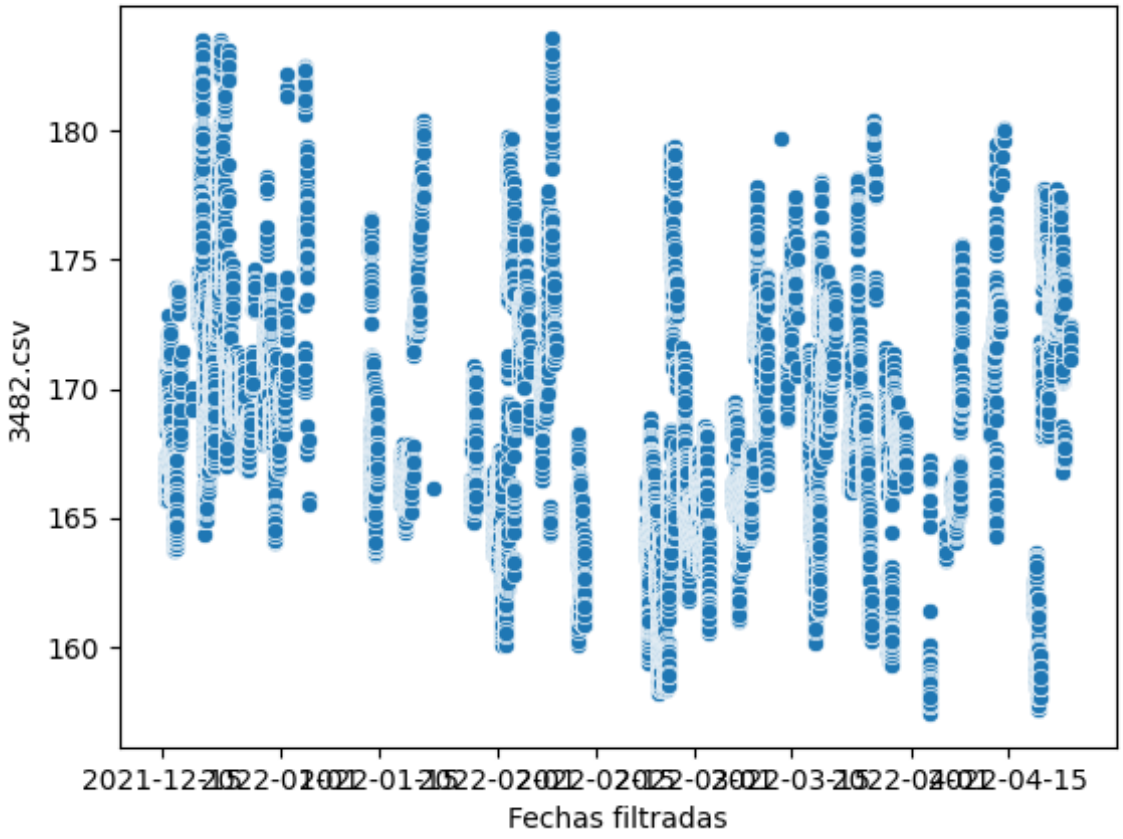


Figura 9.30: Análisis filtrado sensor 3482.

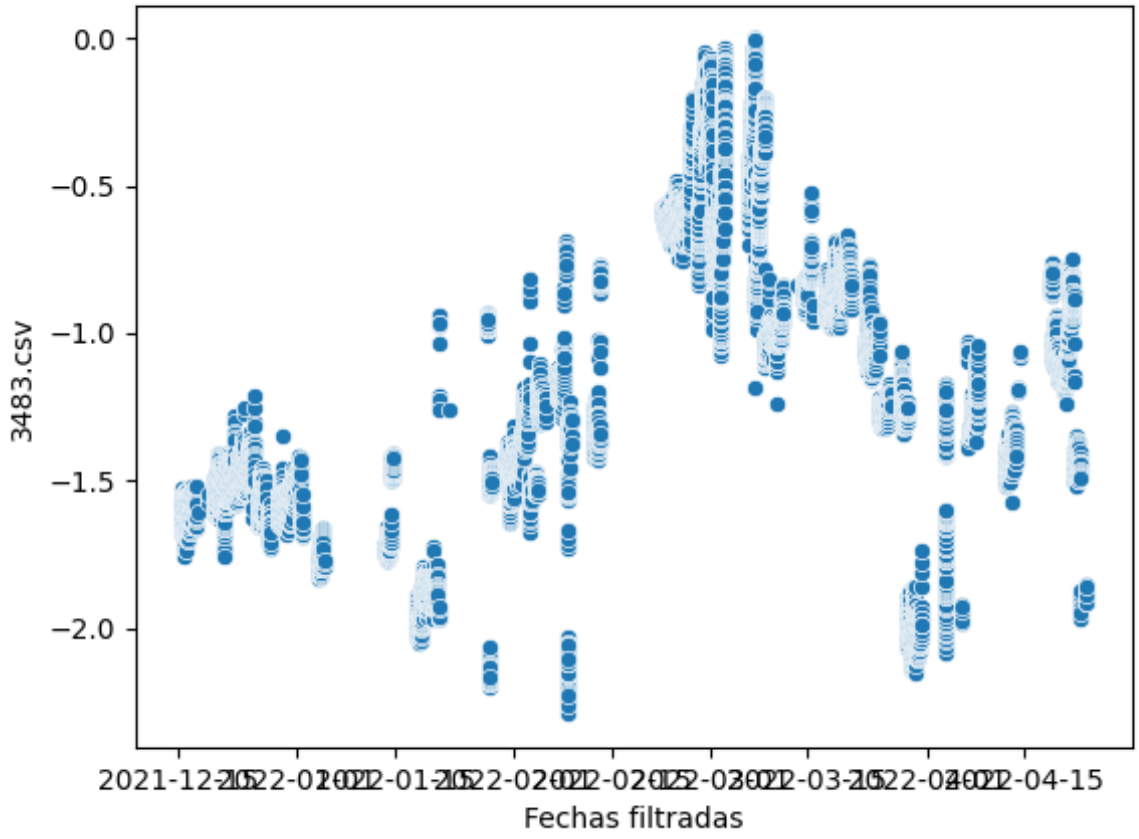


Figura 9.31: Análisis filtrado sensor 3483.

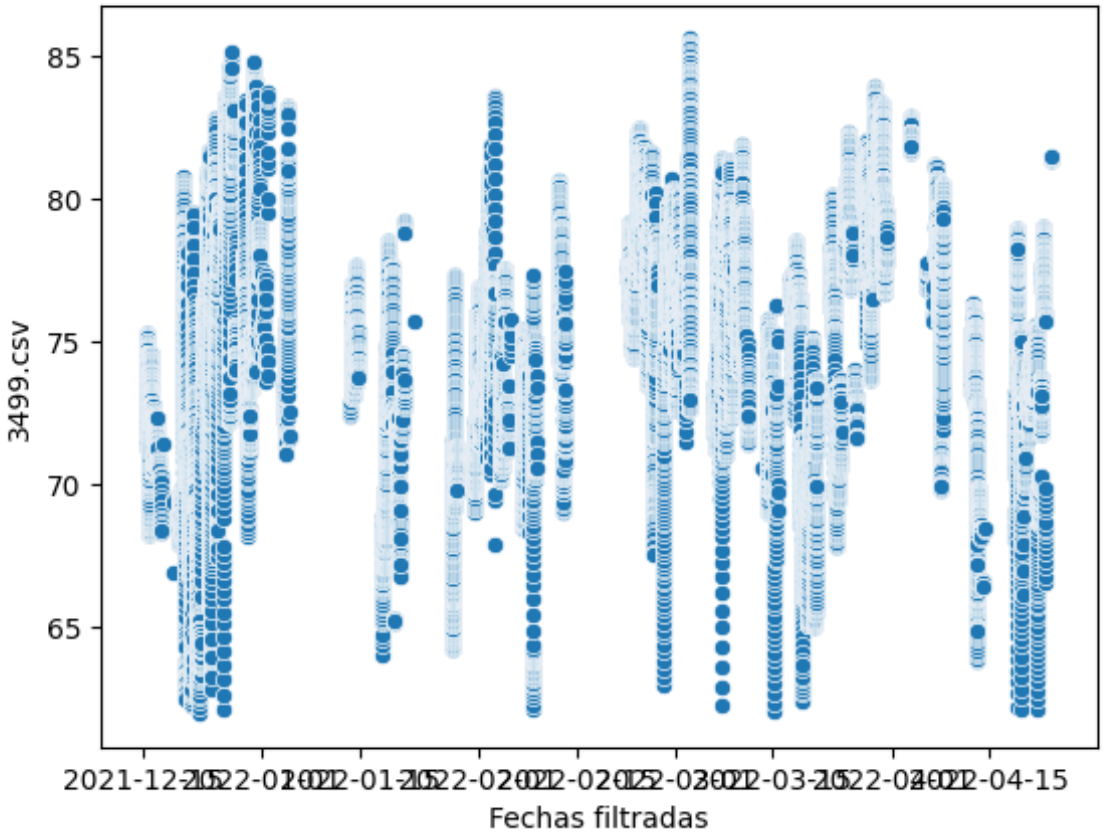


Figura 9.32: Análisis filtrado sensor 3499.

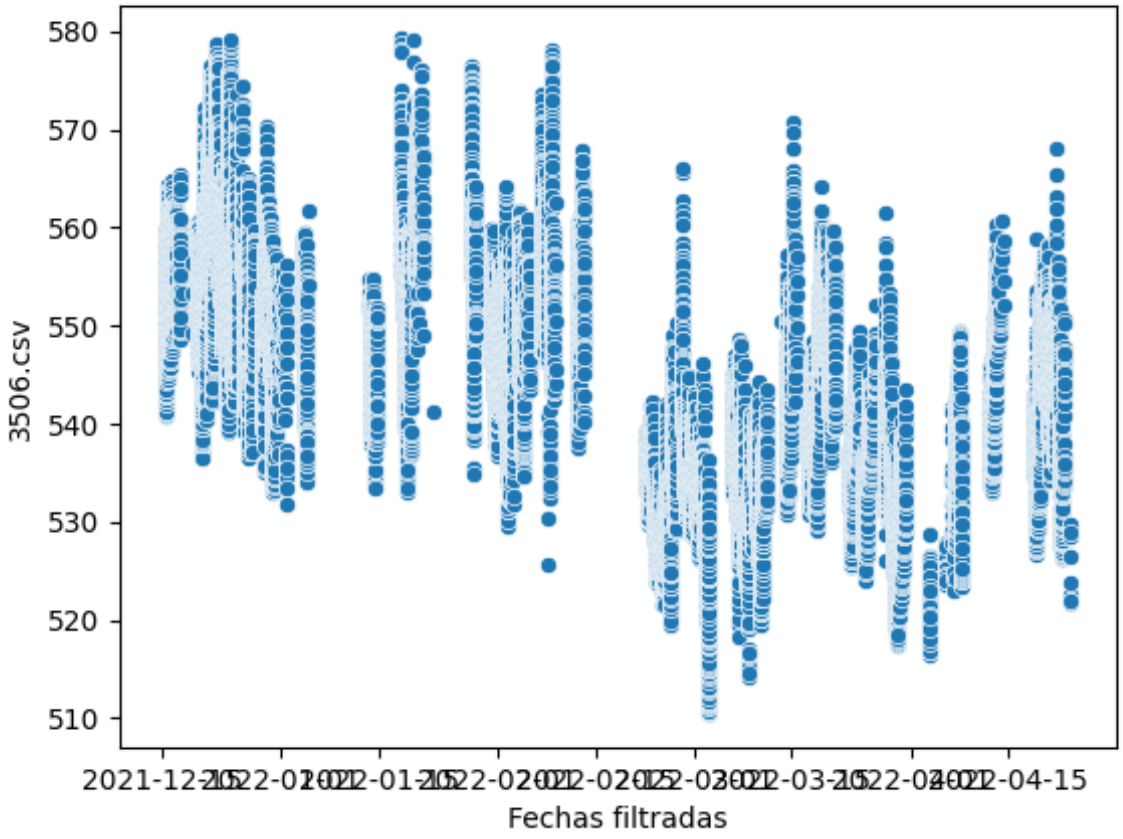


Figura 9.33: Análisis filtrado sensor 3506.

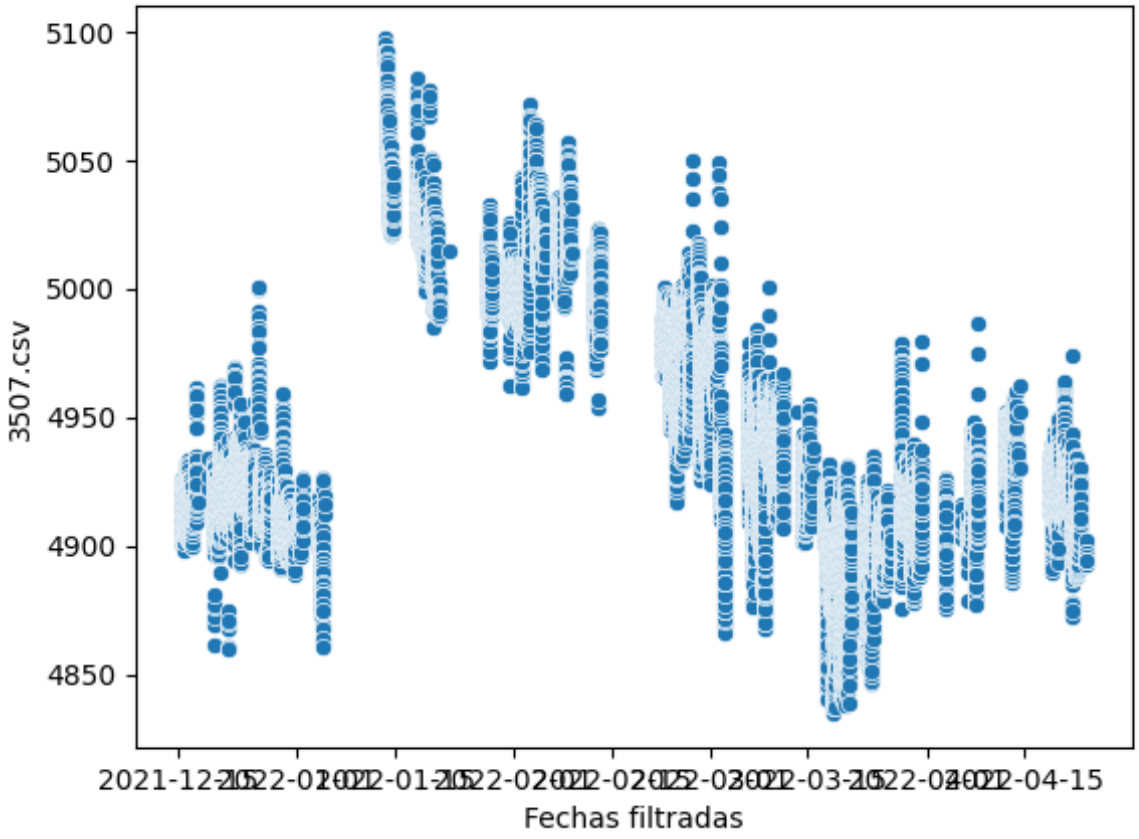


Figura 9.34: Análisis filtrado sensor 3507.

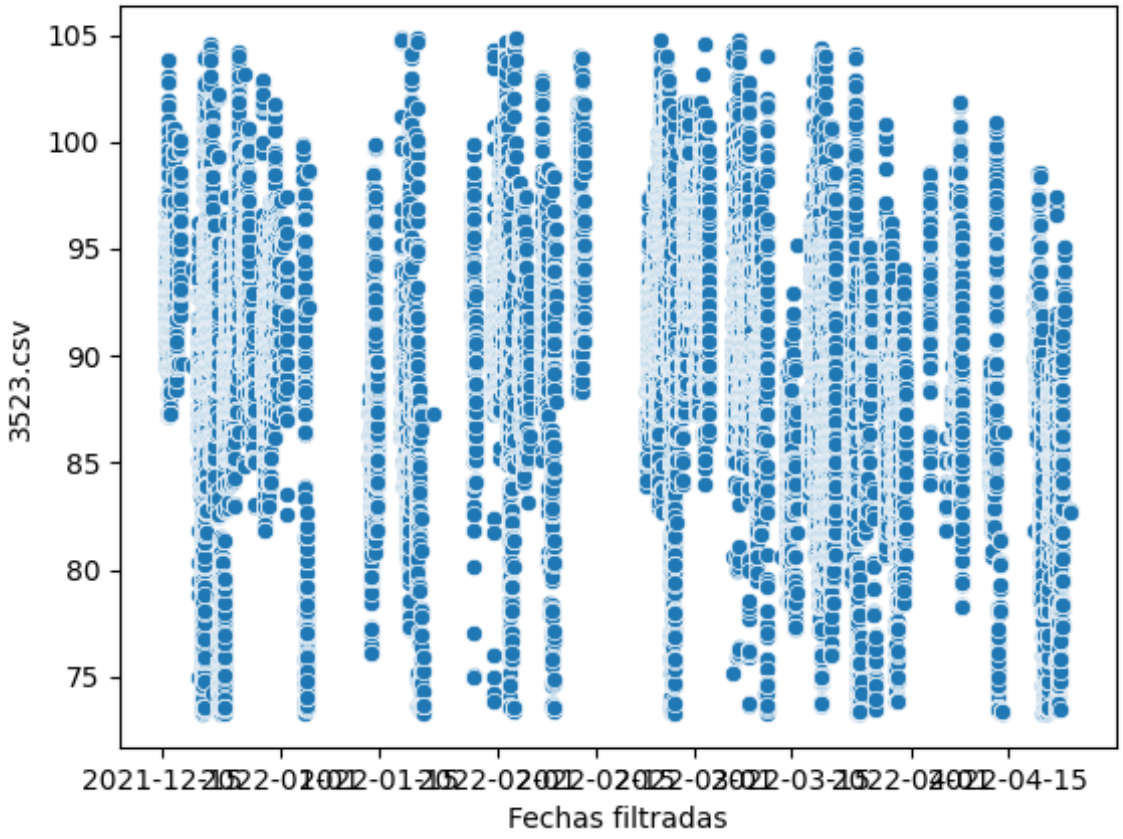


Figura 9.35: Análisis filtrado sensor 3523.

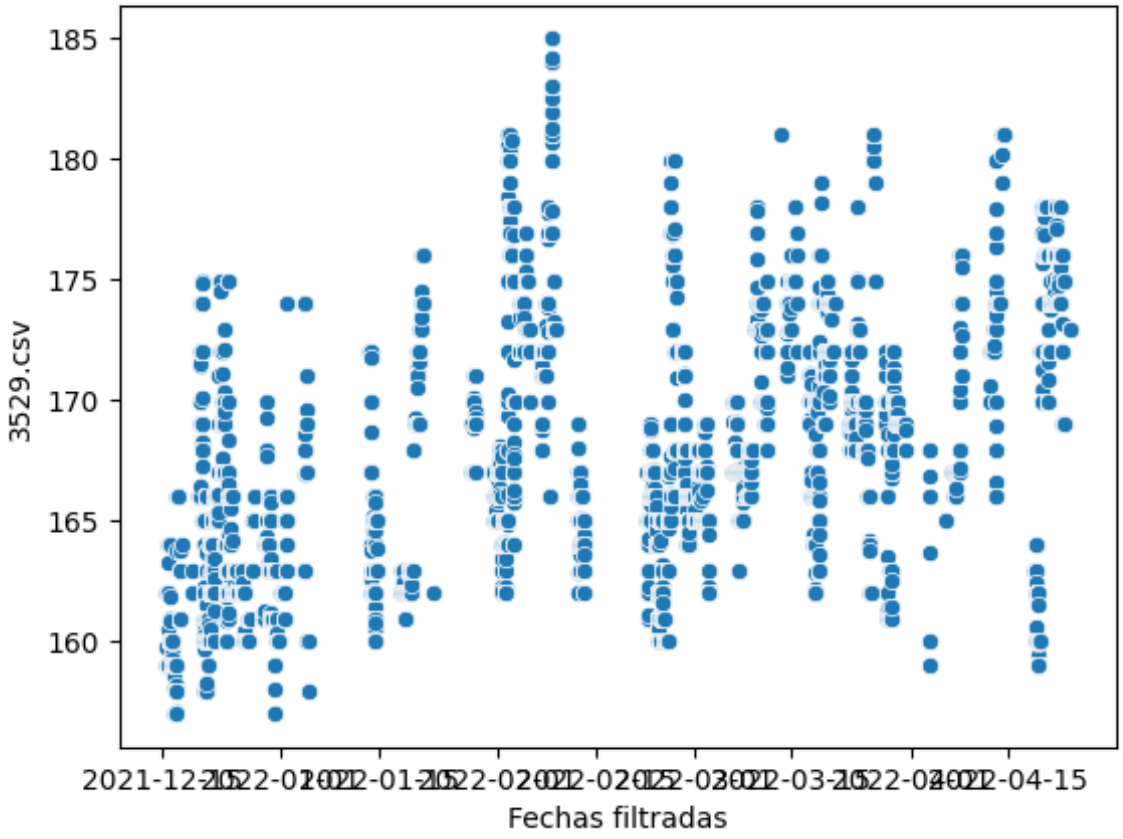


Figura 9.36: Análisis filtrado sensor 3529.

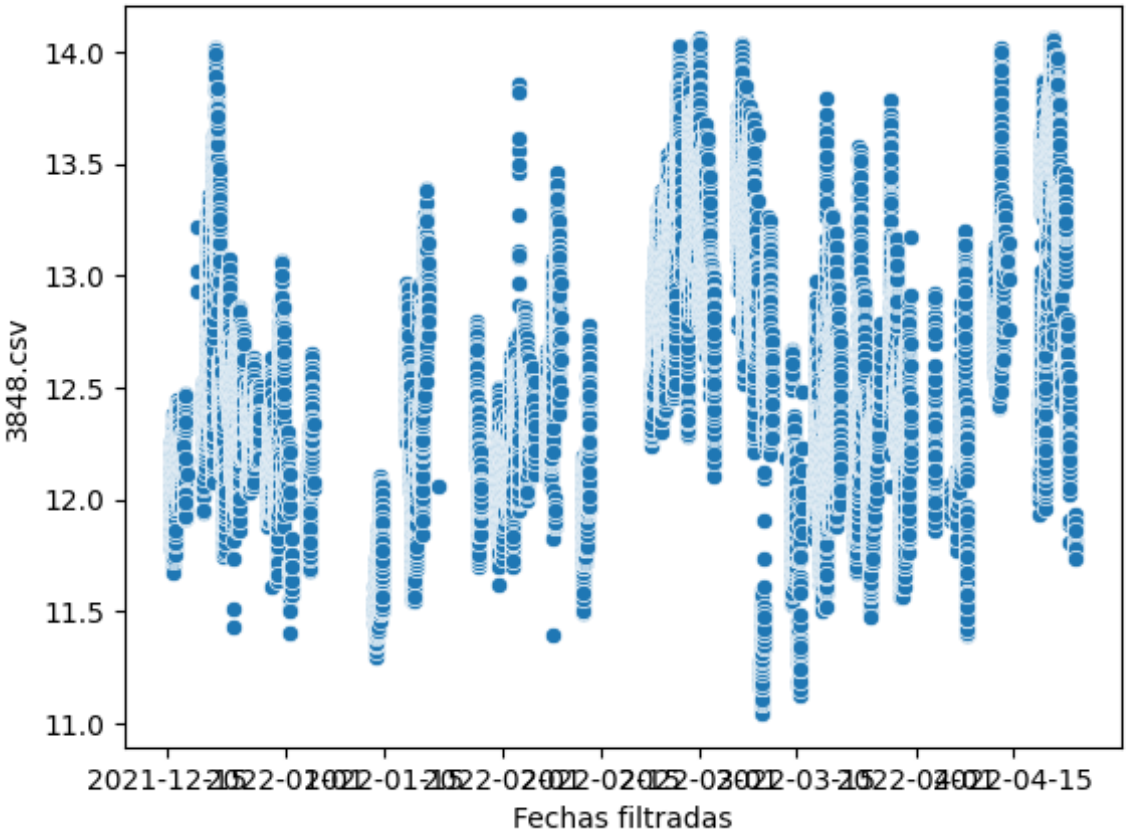


Figura 9.37: Análisis filtrado sensor 3848.

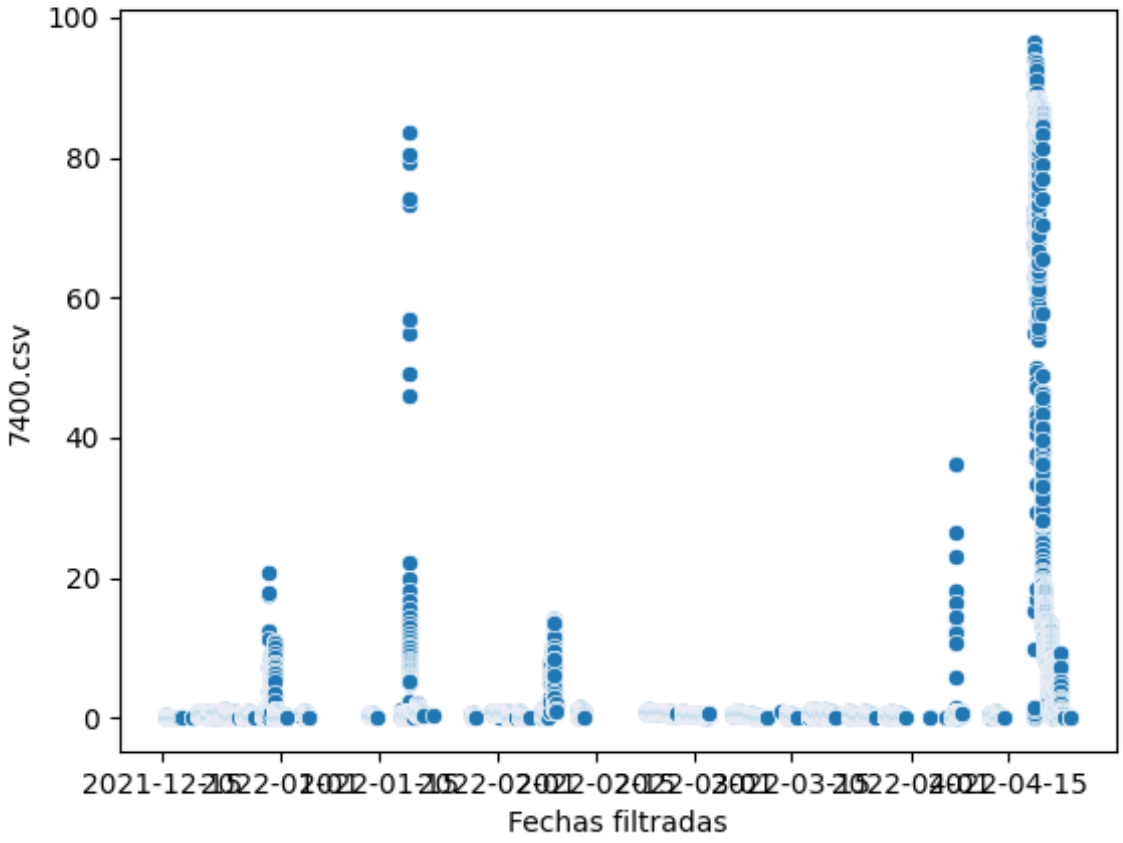


Figura 9.38: Análisis filtrado sensor 7400.

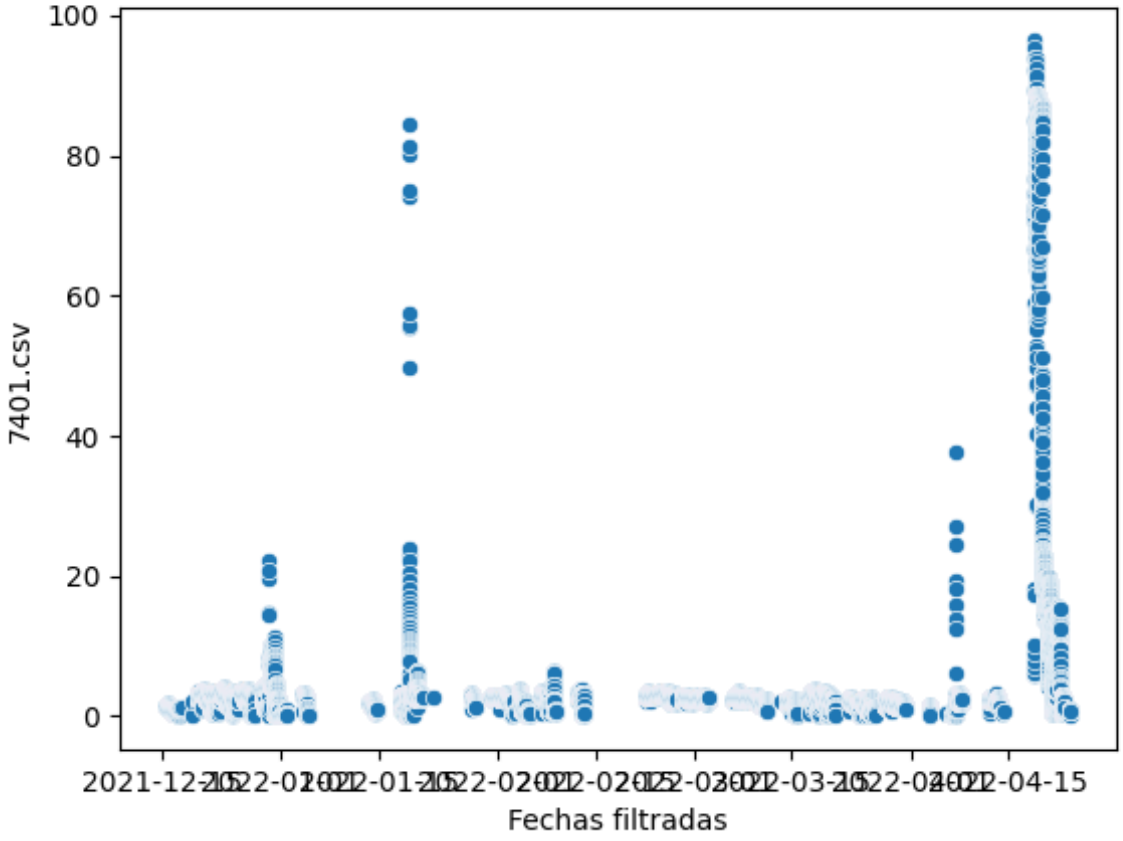


Figura 9.39: Análisis filtrado sensor 7401.

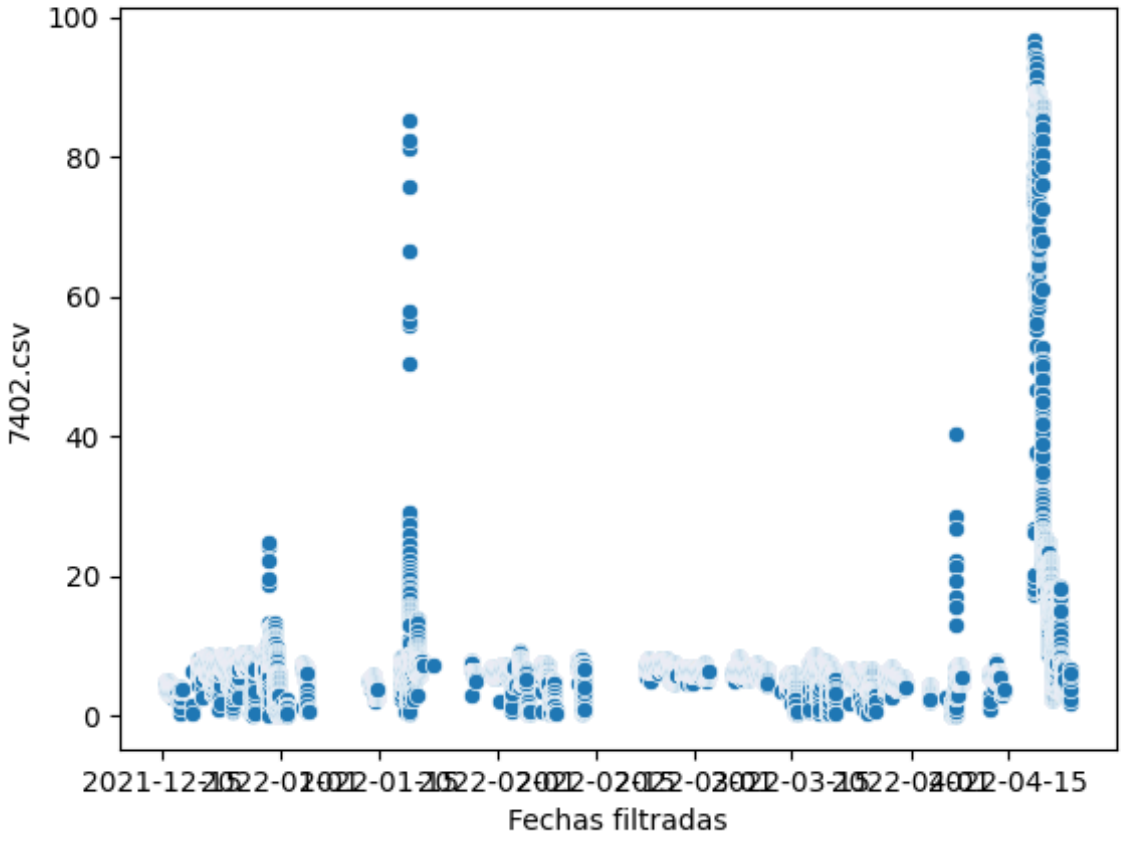


Figura 9.40: Análisis filtrado sensor 7402.

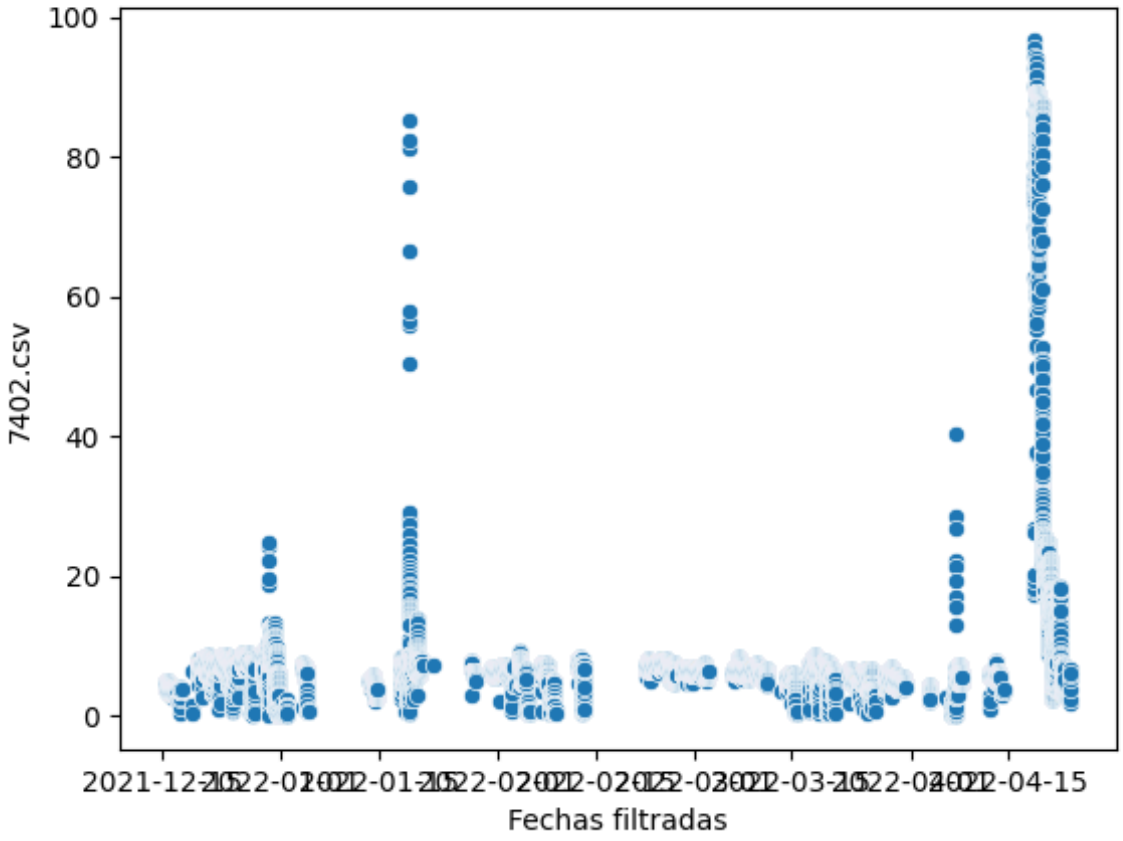


Figura 9.41: Análisis filtrado sensor 7403.

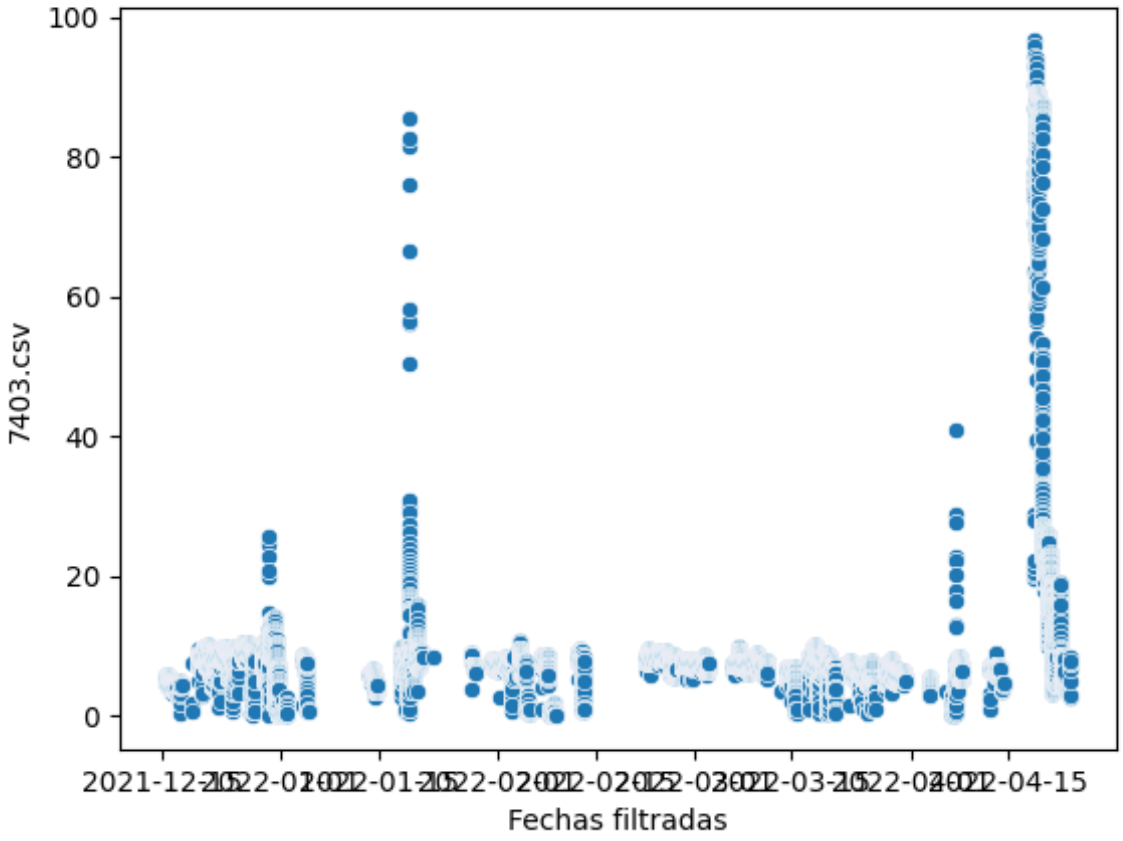


Figura 9.42: Análisis filtrado sensor 7403.

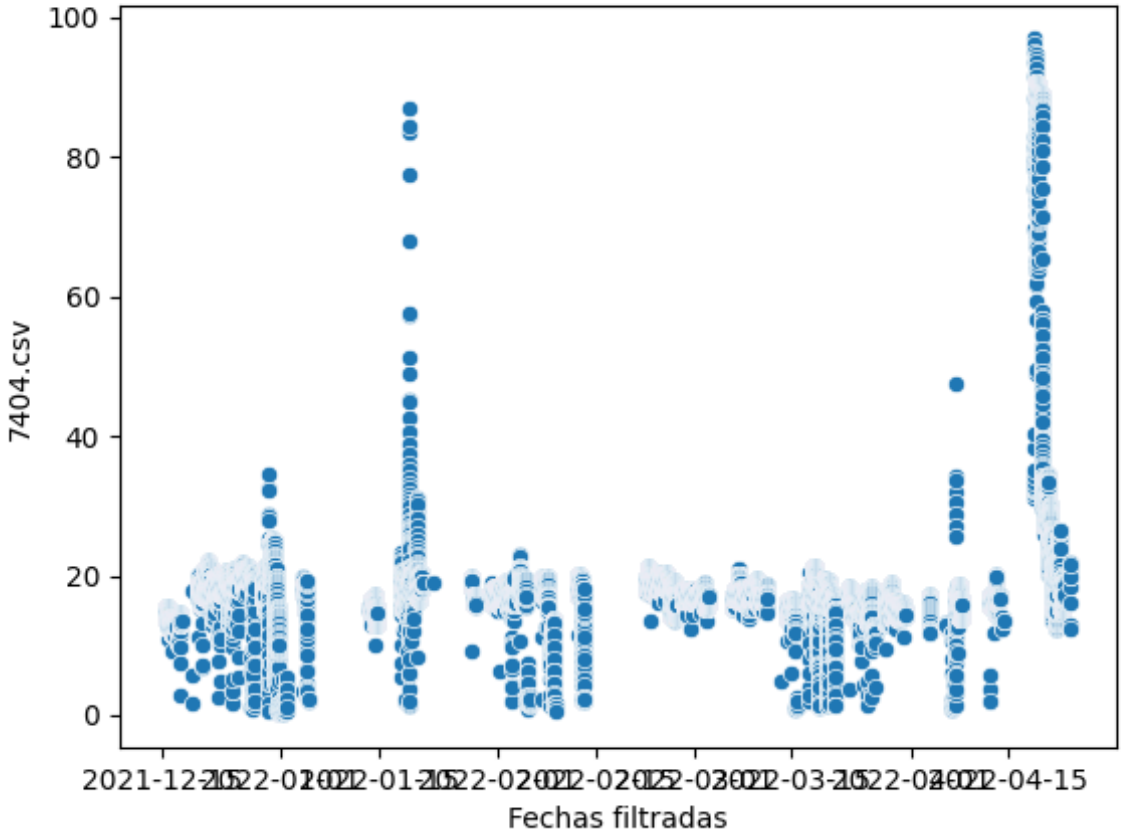


Figura 9.43: Análisis filtrado sensor 7404.

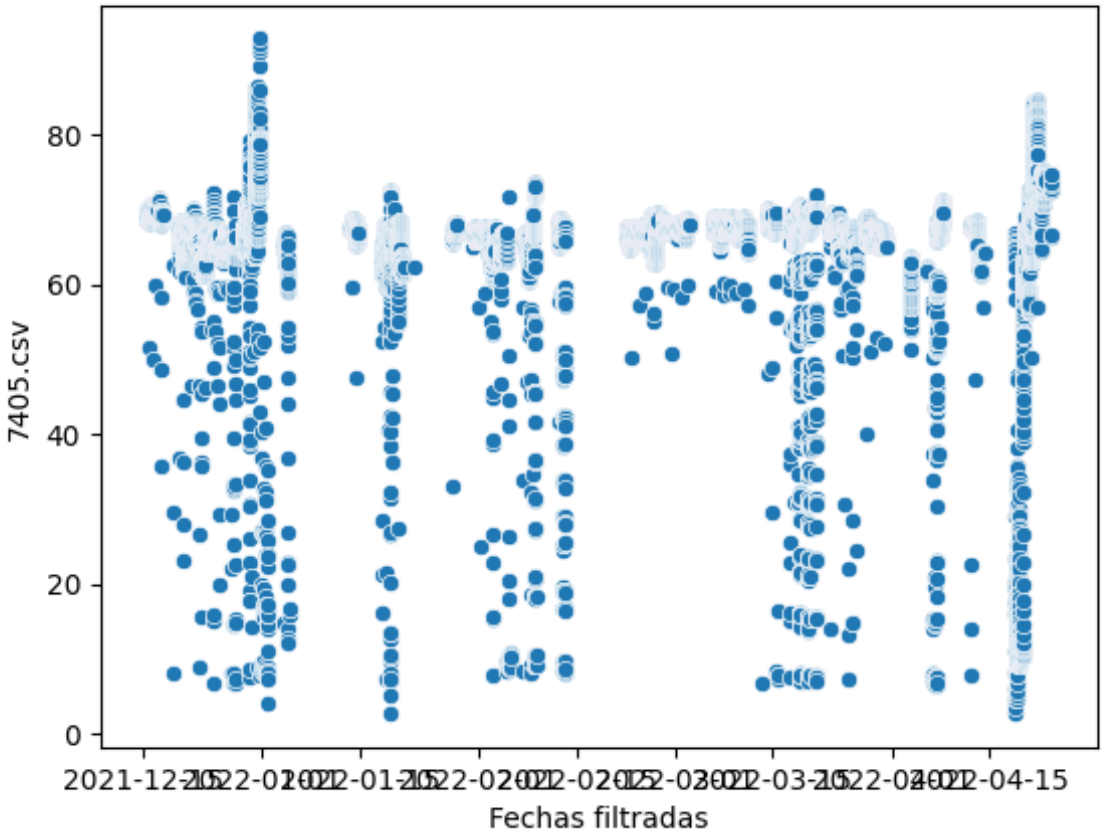


Figura 9.44: Análisis filtrado sensor 7405.

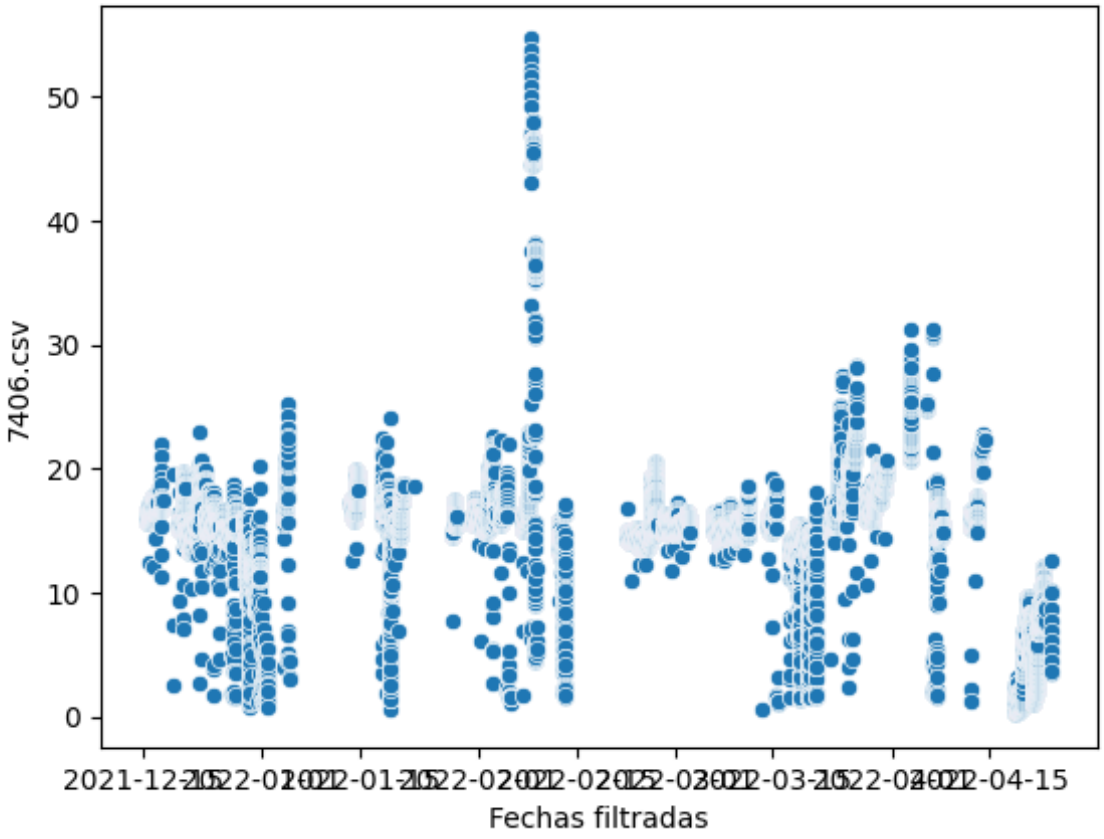


Figura 9.45: Análisis filtrado sensor 7406.

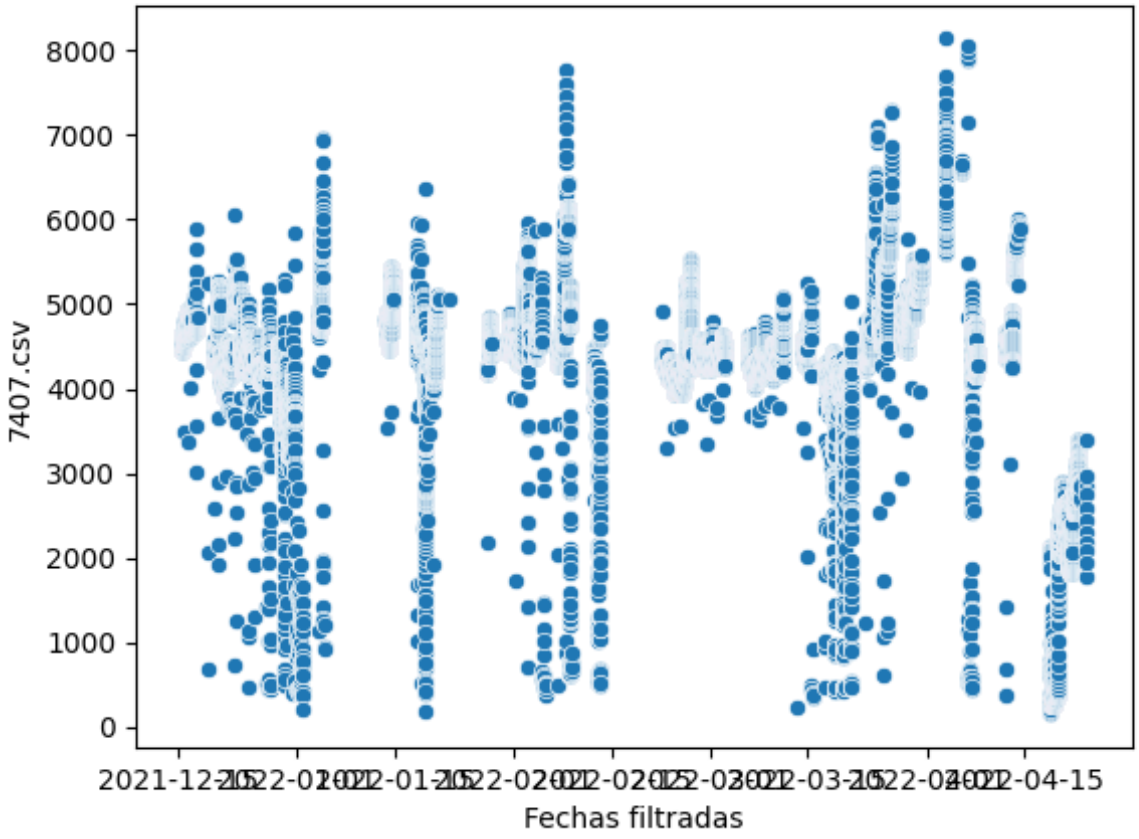


Figura 9.46: Análisis filtrado sensor 7407.

Bibliografía

[1] Karina Andreatta, Filipe Apóstolo, and Reginaldo Nunes. Soft sensor for online cement fineness predicting in ball mills. In *International Seminar of Science and Applied Technology (ISSAT 2020)*, pages 422–428. Atlantis Press, 2020.

[2] Ajaya Kumar Pani and Hare Krishna Mohanta. Online monitoring and control of particle size in the grinding process using least square support vector regression and resilient back propagation neural network. *ISA transactions*, 56:206–221, 2015.

[3] Yaoyao Bao, Yuanming Zhu, Wenli Du, Weimin Zhong, and Feng Qian. A distributed pca-tss based soft sensor for raw meal fineness in vrm system. *Control Engineering Practice*, 90:38–49, 2019.

[4] A Casali, G Gonzalez, F Torres, G Vallebuona, L Castelli, and P Gimenez. Particle size distribution soft-sensor for a grinding circuit. *Powder Technology*, 99(1):15–21, 1998.

[5] Young-Don Ko and Helen Shang. A neural network-based soft sensor for particle size distribution using image analysis. *Powder Technology*, 212(2):359–366, 2011.

[6] Ajaya Kumar Pani and Hare Krishna Mohanta. A hybrid soft sensing approach of a cement mill using principal component analysis and artificial neural networks. In *2013 3rd IEEE International Advance Computing Conference (IACC)*, pages 713–718. IEEE, 2013.

[7] Ajaya Kumar Pani and Hare Krishna Mohanta. Online monitoring of cement clinker quality using multivariate statistics and takagi-sugeno fuzzy-inference technique. *Control Engineering Practice*, 57:1–17, 2016.

[8] Daniel Sbarbaro, Pedro Ascencio, Pablo Espinoza, Felipe Mujica, and Guillermo Cortes. Adaptive soft-sensors for on-line particle size estimation in wet grinding circuits. *Control Engineering Practice*, 16(2):171–178, 2008.

[9] Darko Stanišić, Nikola Jorgovanović, Nikola Popov, and Velimir Čongradac. Soft sensor for real-time cement fineness estimation. *Isa Transactions*, 55:250–259, 2015.

[10] Xinggang Wu and Mingzhe Yuan. Soft-sensor modeling of product particle size in ball milling circuits based on fuzzy neural networks with particle swarm optimization. In *2009 International Conference on Information and Automation*, pages 1458–1461. IEEE, 2009.