Universitat Oberta de Catalunya

uoc.edu

Análisis del consumo de la energía de un edificio mediante técnicas predictivas de *Deep Learning*

Ángel García de la Chica Herrera
Grado de Ingeniería Informática
Inteligencia Artificial

David Isern Alarcón
Xavier Baró Solé

20 de junio de 2023

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Análisis del consumo de energía de un edificio mediante técnicas predictivas de Deep Learning.*
Nombre del autor:	*Ángel García de la Chica Herrera*
Nombre del consultor/a:	*David Isern Alarcón*
Nombre del PRA:	*Xavier Baró Solé*
Fecha de entrega (mm/aaaa):	06/2023
Titulación:	*Ingeniería Informática*
Área del Trabajo Final:	*Inteligencia Artificial*
Idioma del trabajo:	*Castellano*
Palabras clave	*Deep Learning, smarthouse, eficiencia energética*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

La reducción del consumo y el aumento de la eficiencia energética son esenciales para combatir el cambio climático.

En este proyecto se analiza el uso de tecnologías de *Deep Learning* para predecir el consumo de energía eléctrica de los edificios. De esta manera, podremos mejorar la eficiencia y reducir el consumo energético.

En primer lugar, el proyecto introduce los diferentes sistemas de *Deep Learning* y los diferentes enfoques que existen en la actualidad para predecir el consumo energético.

En segundo lugar, se realiza un análisis de los datos del consumo eléctrico de múltiples edificios con los registros meteorológicos de la estación más cercana a estos.

Para ello se aplican técnicas de normalización, discretización y tratamiento de los valores atípicos y ausentes. Por otro lado, se realiza un análisis de correlación de los atributos para reducir la dimensionalidad mediante PCA (*Principal Component Analysis)*.

A continuación, se implementan diferentes modelos utilizando dos conceptos distintos. Por un lado, el de las redes neuronales artificiales y por otro, el de los árboles de decisión *LightGBM (Light Gradient Boosting Machine*).

Finamente, se comparan entre si los resultados de los diferentes modelos

empleando el error cuadrático medio (ECM).

En este trabajo se concluye que, a partir de datos meteorológicos y de los metadatos de los edificios, es posible predecir el consumo de energía eléctrica de los edificios. Además, los modelos implementados para un único edificio no requieren mucha capacidad ni mucho tiempo de cómputo con precisiones bastante buenas.

Abstract (in English, 250 words or less):

Consumption reduction and increased energy efficiency are essential to combat climate change.

This project analyses the use of Deep Learning technologies to predict the electrical energy consumption of buildings. In this way, we will be able to improve efficiency and reduce electricity consumption.

Firstly, the project introduces the different Deep Learning systems and the different approaches that currently exist to predict energy consumption.

Secondly, an analysis of electricity consumption data from multiple buildings is carried out together with historical meteorological data from the meteorological station closest to the building.

For this purpose, normalisation techniques, discretisation and treatment of outliers and missing values are applied. On the other hand, a correlation analysis of the attributes is carried out to reduce the dimensionality by means of PCA (*Principal Component Analysis*).

Then, different models are implemented using two different concepts. On the one hand, artificial neural networks and on the other hand, LightGBM (*Light Gradient Boosting Machine*) decision trees.

Finally, the results of the different models are compared using the mean square error (MSE).

This work concludes that it is possible to predict the electrical energy consumption of buildings from meteorological data and building metadata. Moreover, the models implemented for a single building do not require much capacity and computational time with good accuracies.

Índice

Índice de figuras

Índice de Tablas

1. Introducción

1.1 Contexto y justificación del Trabajo

Los edificios en la Unión Europea son responsables del 38% del consumo de energía y del 35% de las emisiones de gases de efecto invernadero (Ciucci, Parlamento Europeo, 2020). A continuación, podemos ver un desglose del consumo y de las emisiones por sector y tipo de edificio.

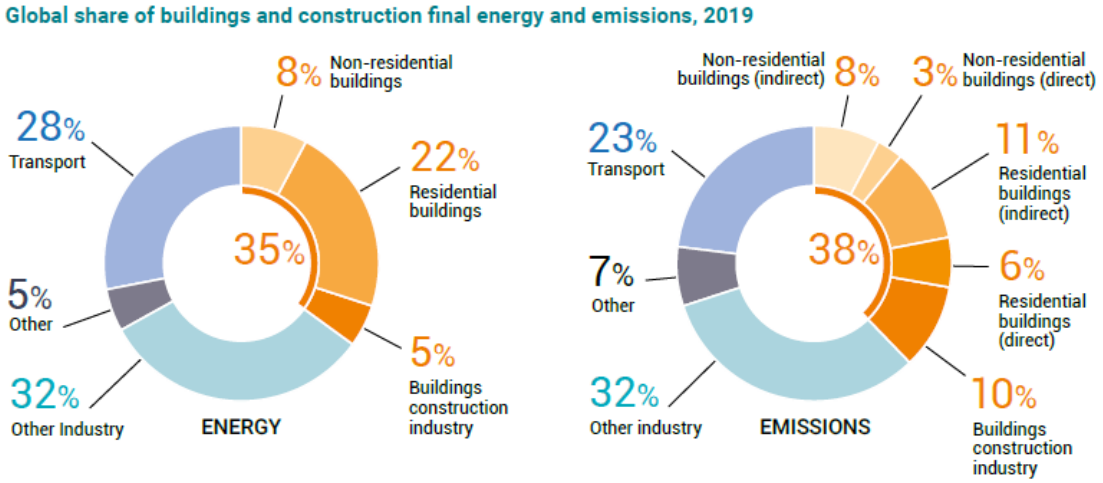

Figura 1. Consumo de energía y emisiones de gases de efecto invernadero. (Constructions, 2020)

Por esta razón la UE lleva mucho tiempo comprometida con la reducción del consumo de energía y el aumento de la eficiencia energética. En 2002, la UE fijó el objetivo de reducir el consumo de energía en un 20% para 2020. Este objetivo se cumplió, y en 2018 la UE fijó un nuevo objetivo de reducción del consumo de energía en un 32,5% para 2030. (Diario Oficial de la Unión Europea, 2002).

Concretamente la Unión Europea está adoptando una serie de medidas para alcanzar esos objetivos. Estas medidas incluyen:

- Normas de eficiencia energética para los edificios nuevos y para las renovaciones de edificios existentes. Estas normas exigen que los edificios sean más eficientes desde el punto de vista energético.

- Incentivos financieros a través de subvenciones, préstamos y desgravaciones fiscales a quienes tomen medidas que mejoren la eficiencia energética. Estos incentivos pueden ayudar a las empresas y los hogares a mejorar la eficiencia energética.

- Campañas de información y sensibilización para fomentar la eficiencia energética dando a conocer las ventajas de la eficiencia

energética y animar a la gente a tomar medidas para mejorar la eficiencia energética de sus hogares y empresas.

La UE está bien encaminada para cumplir su objetivo de 2030, y es probable que adopte nuevas medidas. Una de estas iniciativas se ha denominado La Ola Renovadora y su objetivo es renovar 30 millones de edificios en la UE de aquí a 2030. Esto ayudaría a reducir el consumo de energía en la UE en un 15%. (European Commision, 2020)

También a través de la iniciativa Ciudades y Comunidades Inteligentes se pretende ayudar a las ciudades y comunidades a ser más eficientes energéticamente y sostenibles. La iniciativa proporciona financiación y apoyo a proyectos que mejoren la eficiencia energética, reduzcan las emisiones de gases de efecto invernadero y mejoren la calidad del aire.

Desde el punto de vista nacional, el Plan de Recuperación, Transformación y Resiliencia de España también se compromete a promover un futuro sostenible en línea con las prioridades del Pacto Verde Europeo. (Plan de Recuperación, Transformación y Resiliencia, 2021)

Reducir el consumo de energía y aumentar la eficiencia energética son esenciales para combatir el cambio climático y garantizar la seguridad energética. Pero, cabe destacar que, aunque hay iniciativas en marcha, todavía queda mucho por hacer para mejorar la sostenibilidad ambiental y reducir las emisiones de CO2 (M., 2001).

Con este trabajo fin de grado se pretende analizar el consumo de energía eléctrica de un conjunto de edificios mediante técnicas de *Deep Learning* con el objetivo de mejorar la eficiencia en el consumo y, por lo tanto, disminuir las emisiones de C02.

1.2 Objetivos del Trabajo

El objetivo del proyecto es reducir el consumo de energía en edificios, mediante el análisis del consumo de un edificio mediante la aplicación de técnicas de inteligencia artificial. Por tanto, se busca maximizar la eficiencia energética y reducir el impacto ambiental.

A continuación, se distinguen los objetivos principales y los objetivos secundarios.

1.2.1 Objetivos principales

- **Obtención de los datos**: Se debe obtener datos históricos de consumo de energía de un edificio, así como otros datos que puedan influir en el consumo de energía.

- **Identificación de las variables relevantes**: Las variables relevantes pueden incluir la temperatura ambiente, el consumo de energía eléctrica, la intensidad de luz natural, entre otras.

- **Estudio de las diferentes técnicas de IA existentes para el análisis del consumo de edificios.** Se puede utilizar el aprendizaje automático supervisado para predecir el consumo de energía, o el aprendizaje reforzado para optimizar el consumo de energía en tiempo real.

1.2.2 Objetivos secundarios

- **Desarrollar el modelo.** Se desarrollará el modelo de IA utilizando los datos recopilados y las técnicas seleccionadas.

- **Evaluar los resultados.** Se evaluarán los resultados del proyecto en términos de reducción del consumo de energía y eficiencia energética, y se compararán con los resultados esperados.

1.3 Enfoque y método seguido

El proyecto se enfoca desde dos puntos de vista:

- **Punto de vista teórico** en el que se analizará el estado del arte en el uso de las técnicas de análisis de Inteligencia Artificial para la predicción del consumo de energía en edificios.

- **Punto de vista práctico** en el que se estudiarán los diferentes algoritmos para después aplicarlos en un *dataset* para evaluar y comparar los resultados.

Dadas las características del proyecto se ha escogido utilizar un ciclo de vida clásico o secuencial que está estructurado en las siguientes fases:

- **Fase de análisis de requisitos**
  En esta fase se estudiarán el estado del arte y los diferentes modelos.

- **Análisis y diseño**
  En esta fase se realizará el preprocesado del *dataset*.

- **Implementación**
  En esta fase se codificarán los algoritmos en lenguaje Python.

- **Evaluación**
  En esta fase se realizará un análisis cualitativo y cuantitativo de los resultados obtenidos y se compararan los diferentes modelos.

1.4 Planificación del Trabajo

La planificación se ha realizado en base a las diferentes entregas (PECs) de la asignatura. Además, en esta planificación se han marcado los diferentes hitos parciales y específicos de cada una de las entregas.

Durante la PEC2 se han realizado las siguientes adaptaciones del plan de trabajo:

- Redacción de la memoria: no solo se realiza durante el periodo de la PEC4 si no que se planifica desde el comienzo del semestre hasta el final de la PEC4 en la que se realiza la entrega definitiva de la memoria.

- Al desarrollar la PEC2 se decide añadir un bloque con el estado del arte. Para ello, se añade una tarea más en la PEC 2 y en la PEC3 que se realizará en paralelo con el bloque práctico.

- Se añade una tarea más de correcciones en el que se dedica tiempo a realizar los cambios y correcciones que el tutor considere oportunas.

Por lo tanto, el plan de trabajo definitivo queda como sigue:

Nombre de la tarea	Fecha de inicio	Fecha de finalización	Estado
REDACCIÓN DE LA MEMORIA	01.03.2023	20.06.2023	Abierto
PEC 0. Definición de los contenidos del traba	**01.03.2023**	**13.03.2023**	Terminado
Documetación	01.03.2023	08.03.2023	Terminado
Redacción de la propuesta	09.03.2023	13.03.2023	Terminado
PEC 1. Plan de Trabajo	**14.03.2023**	**28.03.2023**	En progreso
Documentación	14.03.2023	17.03.2023	Terminado
Definición del Contexto y justificación del Trabajo	18.03.2023	22.03.2023	Terminado
Definición de los objetivos del Trabajo	23.03.2023	25.03.2023	Terminado
Diagrama de Gantt	26.03.2023	28.03.2023	Terminado
PEC 2. Desarrollo del Trabajo Fase 1	**29.03.2023**	**03.05.2023**	Terminado
Elección de los datos con los que trabajaremos	29.03.2023	06.04.2023	Terminado
Identificación de las variables relevantes	07.04.2023	10.04.2023	Terminado
Estudio de la bibliografía	11.04.2023	13.04.2023	Terminado
Estado del arte Fase 1	14.04.2023	15.04.2023	Terminado
Correciones del tutor	15.04.2023	16.04.2023	Terminado
Elección de la técnica de análisis	17.04.2023	18.04.2023	Terminado
Análisis y parametrización de los datos	19.04.2023	30.04.2023	Terminado
Seguimiento del estado de las tareas	01.05.2023	01.05.2023	Terminado
Revisión y corrección	02.05.2023	02.05.2023	Terminado
Entregables PEC2	03.05.2023	03.05.2023	Terminado
PEC 3. Desarrollo del Trabajo Fase 2	**04.05.2023**	**29.05.2023**	Terminado
Análisis de los difrentes modelos	04.05.2023	14.05.2023	Terminado
Estado del arte Fase 2	08.05.2023	14.05.2023	Terminado
Desarrollo del modelo	10.05.2023	14.05.2023	Terminado
Correciones del tutor	10.05.2023	11.05.2023	Terminado
Entrenamiento del modelo	15.05.2023	20.05.2023	Terminado
Obtención de métricas y análisis de resultados	21.05.2023	28.05.2023	Terminado
Seguimiento del estado de las tareas	29.05.2023	29.05.2023	Terminado
Revisión y corrección	29.05.2023	29.05.2023	Terminado
Entregables PEC3	29.05.2023	29.05.2023	Terminado
PEC 4. Redacción de la memoria	**30.05.2023**	**20.06.2023**	Terminado
Redacción de los diferentes apartados	01.06.2023	12.06.2023	Terminado
Revisión y corrección	13.06.2023	19.06.2023	Terminado
Entrega de la Memoria	20.06.2023	20.06.2023	Terminado
PEC 5a. Elaboración de la presentación	**21.06.2023**	**25.06.2023**	Abierto
Preparar presentación PowerPoint	21.06.2023	22.06.2023	Abierto
Preparación de la grabación	23.06.2023	25.06.2023	Abierto
PEC 5b. Defensa pública	**28.06.2023**	**05.07.2023**	Abierto
Preparar Defensa del Trabajo	28.06.2023	02.07.2023	Abierto
Defensa	03.07.2023	05.07.2023	Abierto

Figura 2. Desglose de tareas y subtareas del trabajo final de grado.

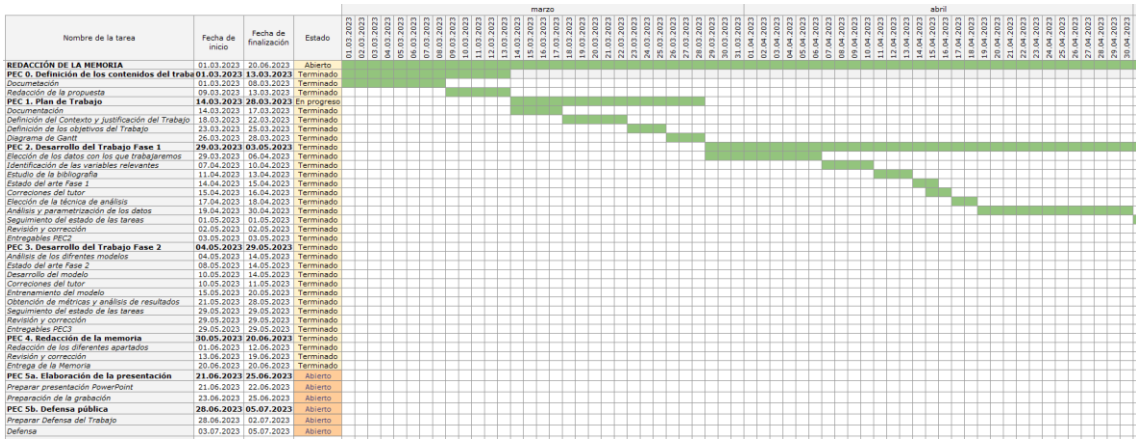

Figura 3. Diagrama de Gantt de los meses de marzo y abril

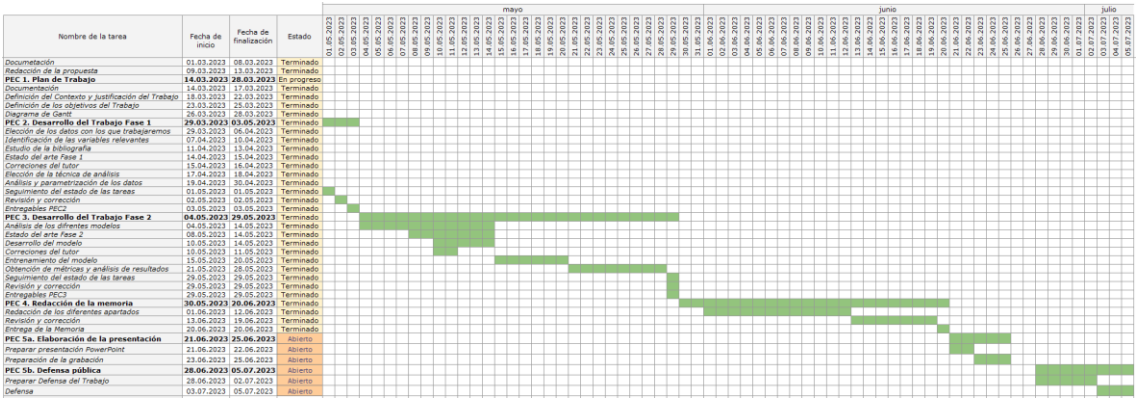

Figura 4. Diagrama de Gantt de los meses de mayo, junio y julio.

1.5 Herramientas empleadas:

Para la realización del TFG se han utilizado principalmente cuatro herramientas:

- **Microsoft Office 365** para la redacción de la Memoria, implementación del diagrama de Gantt y la presentación del TFG.

- **Python (versión 3)** para el análisis del *dataset* y la implantación de los modelos.

- **Google COLAB Pro** como entorno de trabajo. Características de la instancia:
  - Memoria RAM de 25 GB
  - Memoria en disco de 166,8 GB
  - Acelerador por Hardware GPU de 15 GB
  - Disco de 166.8 GB

- **Google Drive** para guardar los datos generados y entregables.

1.6 Breve sumario de productos obtenidos

Como resultado del proyecto se obtendrán los siguientes productos:

- **Memoria del proyecto**, donde se presenta un análisis teórico del estado del arte del análisis del consumo en edificios, así como un ejemplo práctico en el que se implementa y se evalúan diferentes modelos.

- **Cuaderno de Jupyter** con el código de análisis del dataset y la implantación y comparación de los diferentes modelos.

1.6 Breve descripción de los otros capítulos de la memoria

El resto de los capítulos de la memoria del proyecto son:

- **Capítulo 2**. *Estado del arte*
  Estado del arte en el uso de técnicas de inteligencia artificial para la predicción de la energía consumida por los edificios.

- **Capítulo 3**. *Sistema de predicción de consumo en edificios aplicando algoritmos de Deep Learning.*
  En este apartado se analizarán las diferentes técnicas de análisis y se realizará un caso práctico.

- **Capítulo 4**. *Conclusiones.*
  En este apartado se incluirán las conclusiones y reflexiones y un análisis crítico del seguimiento de la planificación. Además, se incluirán una reflexión sobre las líneas de trabajo futuro que no se han podido explorar en este trabajo y que han quedado pendientes.

2. Estado del Arte

En la actualidad existen numerosos estudios para predecir el consumo de energía eléctrica. Pero existen varios enfoques en lo que se refiere a los datos de entrada y de salida. A continuación, hacemos un resumen con ejemplos de los enfoques más importantes que existen en la actualidad.

- **Basado en reconocimiento de imágenes**. Por ejemplo, existe un estudio que proponen utilizar modelos de aprendizaje profundo basados en imágenes para predecir el consumo eléctrico a nivel de edificio a partir de imágenes aéreas (Markus Rosenfelder, Moritz Wussow, Gunther Gust, Roger Cremades, Dirk Neumann, 2021).

- Cálculo del consumo a partir de **factores de forma y materiales de los edificios**. Por ejemplo, en el año 2012 se publicó un estudio en el que los autores utilizan un conjunto de datos de 12 formas de edificios, cada uno compuesto por 18 elementos. Todos los edificios tienen el mismo volumen y utilizan los mismos materiales; solo varían las superficies acristaladas, las orientaciones, la altura, las superficies (dejados, paredes y totales), etc. El documento compara un enfoque clásico de regresión lineal con los bosques aleatorios, para estimar la carga de calefacción y la carga de refrigeración. (Athanasios Tsanasa y Angeliki Xifarab, 2012).

- Predicción de la **etiqueta energética** del edificio. Por ejemplo, en el año 2022 se publicó un artículo que propone un marco de fusión de características para la predicción de la eficiencia energética de los edificios con datos disponibles públicamente. El marco implica la recopilación de datos EPC de factores descriptivos de edificios y datos de imágenes a nivel de calle a través de StreetView. El marco se aplica a la ciudad de Glasgow (Reino Unido) para comprobar su viabilidad. La investigación demuestra el potencial del uso de datos de múltiples fuentes en la predicción de la eficiencia energética de los edificios con gran precisión y poco tiempo de inferencia. (Maoran Sun, 2022)

- Predecir el consumo eléctrico **a partir de los históricos** de consumo eléctricos en edificios. Por ejemplo, en el año 2019 se publicó un estudio titulado "*A novel deep learning approach for building energy forecasting*" (Xiaofei Zhang, 2019) que propone utilizar una memoria a corto plazo para aprender las características temporales de los datos de consumo de energía de los edificios. Los autores evaluaron el método en un conjunto de datos de consumo eléctrico de un edificio comercial obteniendo un alto grado de precisión.

En este proyecto nos centraremos en la predicción del consumo de la energía eléctrica en función del uso del edificio, de la superficie y de los datos históricos de consumo y los registros meteorológicos.

En general, existen varias técnicas de aprendizaje profundo y aprendizaje automático que pueden utilizarse para predecir el consumo energético de los edificios. Estos métodos pueden aplicarse de forma global o local. Esto dependerá de las características de los datos y pueden utilizarse para generar predicciones precisas del consumo de energía. Entre ellos se encuentran el *soft-computing*, las redes neuronales artificiales, las redes bayesianas, las máquinas de vectores soporte y las máquinas de aprendizaje extremo (Vergara Pla, 2015)

En la publicación titulada "*Machine learning for estimation of building energy consumption and performance: a review*" (Saleh Seyedzadeh, Farzad Pour Rahimian, Ivan Glesk and Marc Roper, 2018) realiza un estudio de los diferentes atículos publicados y de las tecnicas empleadas para la predicción del consumo de la energía electrica en edificios. A continuación, mostramos una tabla resumen de las técnicas empleadas

Modelo	Número	Proporción
ANN	22	57 %
GPR	7	19 %
GMM	2	5 %
SVM	7	19 %

Tabla 1. Resumen de las técnicas más empleades para predecir el consumo de energía

La conclusión del artículo es que el uso de técnicas de aprendizaje automático, como las redes neuronales artificiales, las máquinas de vectores de apoyo, las regresiones gaussianas y la agrupación de datos, puede ser eficaz para predecir y mejorar el rendimiento energético de los edificios.

Por último, cabe destacar que la investigación en este campo sigue su curso, pero los resultados obtenidos hasta ahora son prometedores.

3. Sistema de predicción de consumo en edificios aplicando algoritmos de *Deep Learning*

En este apartado se realizará un análisis de los métodos y algoritmos de *Machine Learning* y *Deep Learning* con los que se pueden implementar un modelo.

A continuación, se analizará un *dataset* y se analizarán los resultados.

3.1 Introducción

El aprendizaje automático (*Machine Learning*) y el aprendizaje profundo (*Deep Learning*) son dos campos que han experimentado un gran crecimiento y desarrollo en los últimos años. Esto ha sido posible sobre todo a la gran cantidad de datos que se generan diariamente y al aumento de la capacidad de análisis de los ordenadores.

El *Machine Learning* se enfoca en desarrollar algoritmos que permitan a los sistemas computacionales aprender de los datos de entrada sin ser programados explícitamente para realizar una tarea específica. En lugar de eso, el algoritmo analiza los datos y aprende a través de la experiencia, lo que le permite mejorar su precisión en la toma de decisiones.

Los algoritmos de *Machine Learning* se dividen en tres categorías principales: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo.

- **El aprendizaje supervisado** se utiliza para entrenar modelos de predicción, donde el algoritmo aprende a partir de datos de entrada y salida previamente etiquetados.

  A continuación, podemos ver un esquema de un modelo de aprendizaje supervisado aplicado a la predicción del consumo en edificios. (Saleh Seyedzadeh, Farzad Pour Rahimian, Ivan Glesk and Marc Roper, 2018)

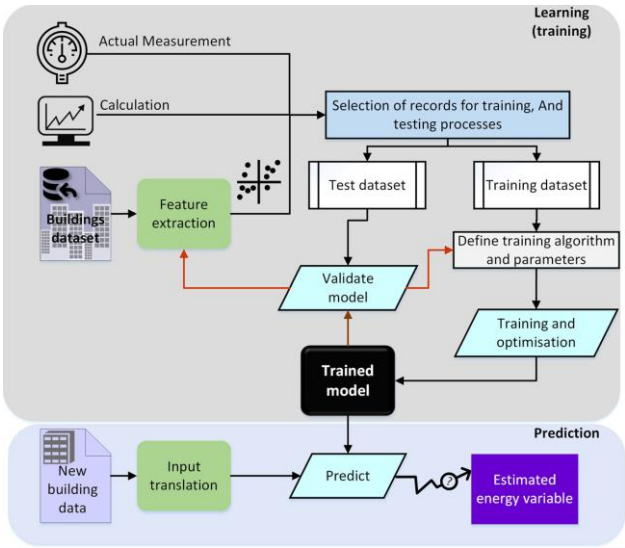

Figura 5. Diagrama esquemático general del aprendizaje supervisado

- **El aprendizaje no supervisado** se utiliza para descubrir patrones en datos sin etiquetar y agruparlos en categorías.
  A continuación, podemos ver un esquema de clasificación (Saleh Seyedzadeh, Farzad Pour Rahimian, Ivan Glesk and Marc Roper, 2018)

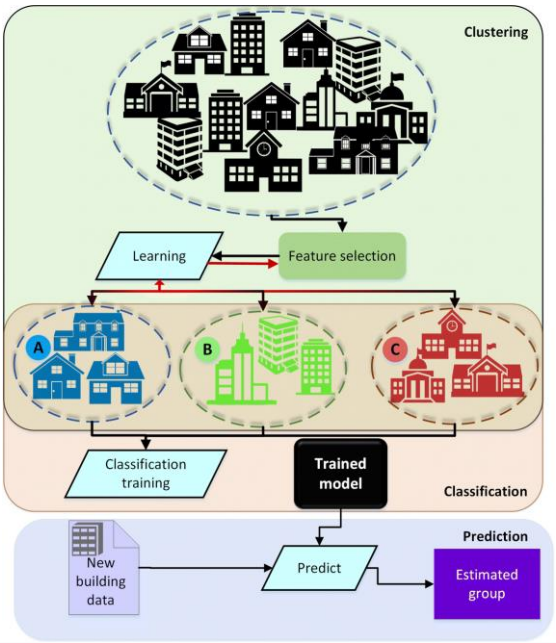

Figura 6. Diagrama de agrupamiento de edificios para evaluación comparativa de energía

- **Aprendizaje por refuerzo** se utiliza para entrenar sistemas que interactúan con su entorno y aprenden a partir de las recompensas o castigos que reciben por sus acciones.

Por otro lado, el *Deep Learning* es una subcategoría del *Machine Learning* que utiliza redes neuronales artificiales para procesar grandes cantidades de datos y extraer patrones complejos. Estas redes se inspiran en la estructura y funcionamiento del cerebro humano y se

componen de múltiples capas de neuronas artificiales que procesan los datos de entrada de forma gradual y van refinando los resultados a medida que avanzan en las capas.

Figura 7. Representación de una red neuronal humana

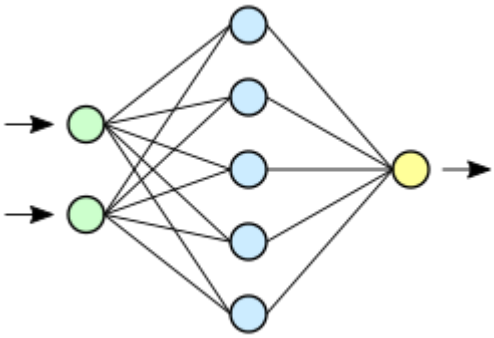

Figura 8. Representación de una Red Neuronal Artificial

Una de las características principales del *Deep Learning* es su capacidad para procesar grandes cantidades de datos no estructurados, como imágenes, audio y texto, y extraer características relevantes de ellos. Esto ha permitido avances significativos en áreas como la visión por computadora, el reconocimiento de voz y el procesamiento de lenguaje natural.

El *Deep Learning* se utiliza en una gran variedad de aplicaciones, desde la detección de fraudes en transacciones financieras (Maryam Habibpour a, Hassan Gharoun b, Mohammadreza Mehdipour c, AmirReza Tajally d, Hamzeh Asgharnezhad f, Afshar Shamsi e, Abbas Khosravi e, Saeid Nahavandi e, 2023) hasta la detección de enfermedades en imágenes médicas (Fatima Yousaf, Sajid Iqbal, Nosheen Fatima, Tanzeela Kousar, Mohd Shafry Mohd Rahim, 2023). También se utiliza en el desarrollo de asistentes virtuales como Siri de Apple, Alexa de Amazon y Google Assistant, que utilizan el procesamiento de lenguaje natural para entender y responder a las preguntas de los usuarios.

3.2 Análisis de los datos

Los datos con los que vamos a trabajar se han extraído de un concurso convocado por ASHARAE: Great Energy Predictor III finalizado en el año 2020 (ASHARAE, 2019).

ASHRAE (del inglés *American Society of Heating, Refrigerating and Air-Conditioning Engineers*), es una sociedad global que promueve el bienestar humano a través de la tecnología sostenible para el entorno construido. ASHRAE cuenta con más de 57.000 miembros en más de 130 países de todo el mundo.

El concurso convocado por ASHRAE proporciona gran cantidad de datos sobre los consumos de electricidad, agua caliente, agua fría y vapor en

edificios situados en diferentes partes del mundo. Además, proporciona los metadatos de los edificios e históricos de datos meteorológicos.

Los datos están divididos en 5 archivos CSV:

- **train.csv**

Esta tabla contiene los datos de las lecturas de consumo energético para entrenar el modelo. El *dataframe* contiene más de 20 millones de muestras y ocupa 648 MB. La tabla está compuesta por los siguientes atributos:

- ***building_id***. Identificador del edificio en el que se toma la lectura, Es clave externa de la tabla de metadatos de los edificios.

- ***meter***. Tipo de lectura que se realiza. Puede tomar los siguientes valores:
  - 0 - electricidad
  - 1 - agua refrigerada
  - 2 - vapor
  - 3 - agua caliente

- ***timestamp*** - Fecha en la que se realizó la medición.

- ***meter_reading*** - La variable objetivo. Consumo de energía en kWh (o equivalente). La propia ASHARAE ya nos advierte de que se tratan de datos reales con errores de medición.

A continuación, podemos ver una representación de los atributos de la tabla:

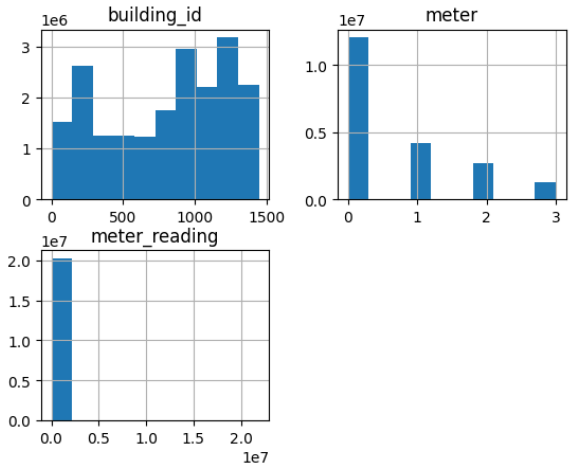

Figura 9. Representación de los atributos de la tabla train.csv

Como se puede observar en el histograma anterior la mayor parte de las mediciones son de los cosnumos de energía eléctrica (*meter* = 0).

- **building_meta.csv**

Esta tabla contiene los metadatos de los edificios donde se han tomado las muestras. El *dataframe* contiene 1449 muestras y ocupa 0.45 MB. La tabla está compuesta por los siguientes atributos:

- ***site_id***. Identificador de la ubicación. Es clave externa de la tabla con los datos de los datos meteorológicos.
- ***building_id*.** Identificador del edificio. Es clave foránea para los datos de entrenamiento.
- ***primary_use***. Indicador de la categoría principal de actividades del edificio.
- ***square_feet*.** Superficie bruta del edificio medido en pies.
- ***year_built***. Año de inauguración del edificio.
- ***floor_count***. Número de plantas del edificio.

A continuación, podemos ver una representación de los atributos de la tabla:

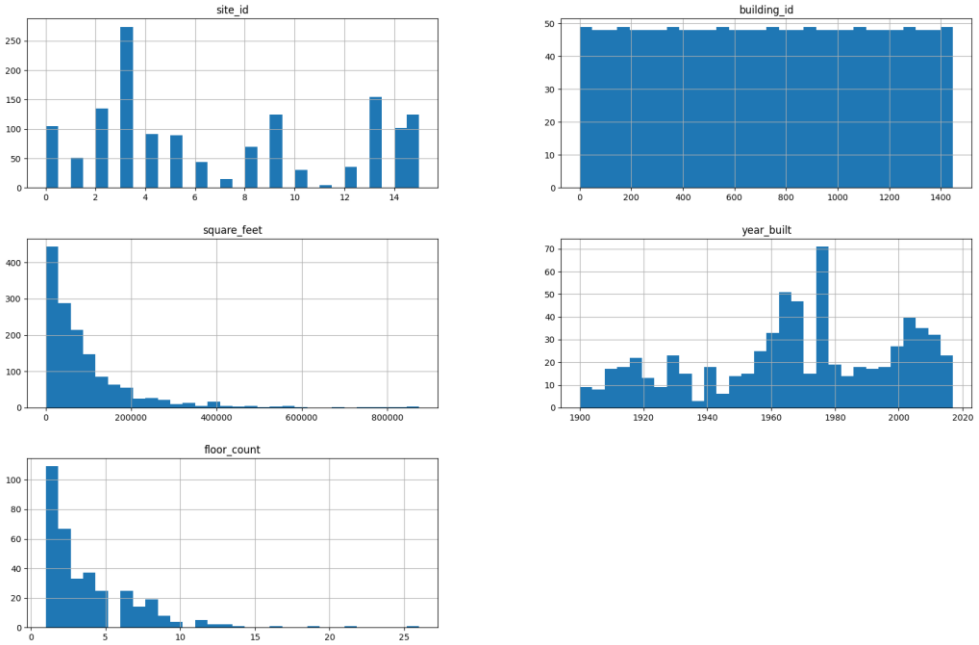

Figura 10. Representación de los atributos de la tabla building_meta.csv

- **weather_train.csv**

Datos meteorológicos de las estaciones meteorológicas del área donde se ubica el edificio. El *dataframe* contiene más de 139000 muestras y ocupa 7.2 MB. La tabla está compuesta por los siguientes atributos:

- ***site_id***. Identificador de la ubicación. Es clave foránea para los metadatos de los edificios.

- ***air_temperature***. Grados Celsius

- ***cloud_coverage***. Porción del cielo cubierta de nubes medido en octas. En meteorología, un octa es una unidad de medida utilizada para expresar la cantidad de nubes en el cielo. El término "octa" se deriva de "octavo", lo que indica que se divide el cielo en ocho partes iguales. Puede tomar los siguientes valores:
  - 0 octa: El cielo está despejado, sin nubes visibles.
  - 1 octa: Solo un octavo del cielo está cubierto de nubes.
  - 2 octas: Dos octavos del cielo están cubiertos de nubes.
  - 3 octas: Tres octavos del cielo están cubiertos de nubes.
  - 4 octas: Cuatro octavos del cielo están cubiertos de nubes.
  - 5 octas: Cinco octavos del cielo están cubiertos de nubes.
  - 6 octas: Seis octavos del cielo están cubiertos de nubes.
  - 7 octas: Siete octavos del cielo están cubiertos de nubes.
  - 8 octas: El cielo está completamente cubierto.
  - 9 octas: Cielo obstruido a la vista.

  (Wikipedia, Octa, 2023)

- ***dew_temperature***. Temperatura de rocío expresada en grados Celsius. La temperatura de rocío es la temperatura más alta a la que empieza a condensarse el vapor de agua contenido en el aire, produciendo rocío, neblina, cualquier tipo de nube o, en caso de que la temperatura sea lo suficientemente baja, escarcha. (Wikipedia, Punto de rocío, 2023)

- ***precip_depth_1_hr***. Precipitación expresada en milímetros por hora.

- ***sea_level_pressure*****.** Presión atmosférica expresada en milibares / hectopascales.

- ***wind_direction***. Dirección del viento expresada en grados (0-360).

- ***wind_speed***. Velocidad del viento expresada en metros por segundo

A continuación, podemos ver una representación de los atributos de la tabla:

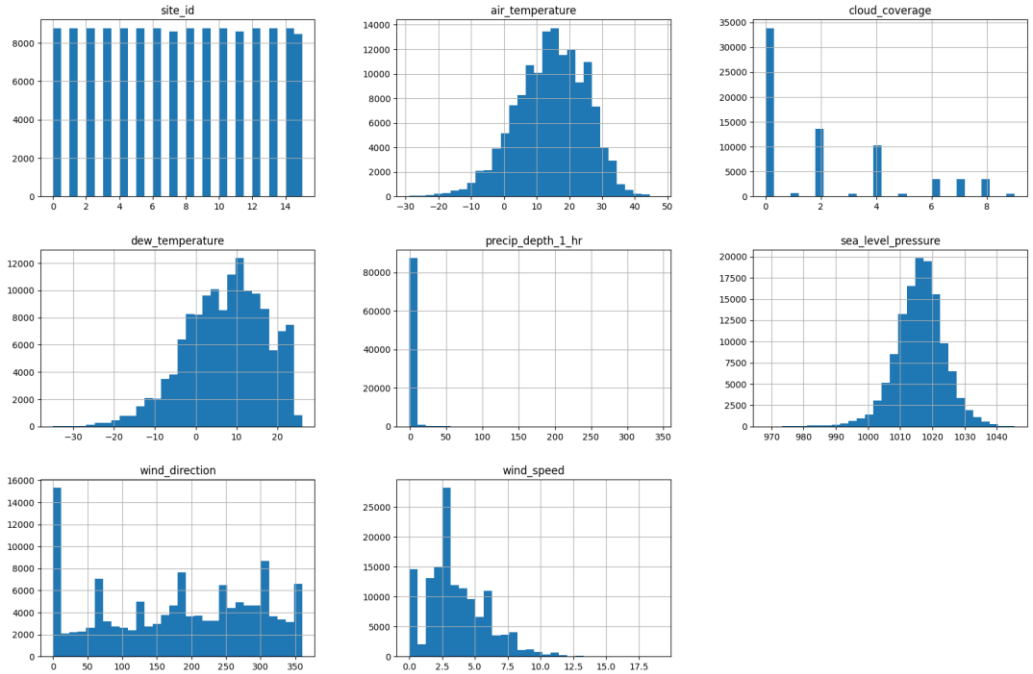

Figura 11. Representación de los atributos de la tabla weather_train.csv

Además, el concurso aportaba 2 conjuntos de datos más para evaluar nuestro modelo y enviar los resultados. Como ese no es nuestro objetivo, solo utilizaremos: *train.csv*, *building_meta.csv*, *weather_train.csv* y *weather_test.csv*. Este último lo utilizaremos porque nos aportará información muy útil para tratar las lecturas nulas o valores atípicos.

Por otro lado, el objetivo de este trabajo es predecir el consumo eléctrico por lo que, en la fase de pretratamiento de los datos, descartaremos las otras tres medidas (agua caliente, agua fría y vapor).

3.3 Pretratamiento de los datos

Antes de unir los distintos conjuntos de datos para entrenar los modelos, trataremos cada tabla por separado para que el modelo sea capaz de interpretarlos y para reducir lo máximo posible la carga de cómputo de los modelos. Este apartado estará dividido en siete fases:

- **Selección de características.** En el que seleccionaremos las características que nos pueden ser útiles para entrenar a nuestros modelos.

- **Limpieza de los datos**. En el que trataremos los valores nulos, valores repetidos y valores atípicos u *outliers*.

- **Análisis gráfico de los atributos**. Donde representaremos las variables y calcularemos la relación entre estas con el objetivo de reducir la dimensionalidad.

- **Codificación de las variables discretas** donde transformaremos las variables discretas a un formato que los modelos puedan entender.

- **División del conjunto de datos** donde dividiremos los datos para entrenar y evaluar nuestros modelos.

- **Normalización de los datos** donde se reescalarán los datos y trataremos las variables discretas.

- **Reducción de la dimensionalidad (PCA).** Donde aplicaremos uno de los métodos más comúnmente utilizados para reducir la dimensionalidad: el Análisis de Componentes Principales (PCA, por sus siglas en inglés).

3.3.1 Selección de características

En este trabajo solo nos centraremos en el estudio del consumo eléctrico. Por lo tanto, eliminaremos todas las observaciones que no sean del consumo eléctrico y el atributo *meter* del conjunto de datos de entrenamiento.

Una vez eliminados estos atributos nos quedamos con más de 12 millones de observaciones. A continuación, podemos ver la representación de los datos.

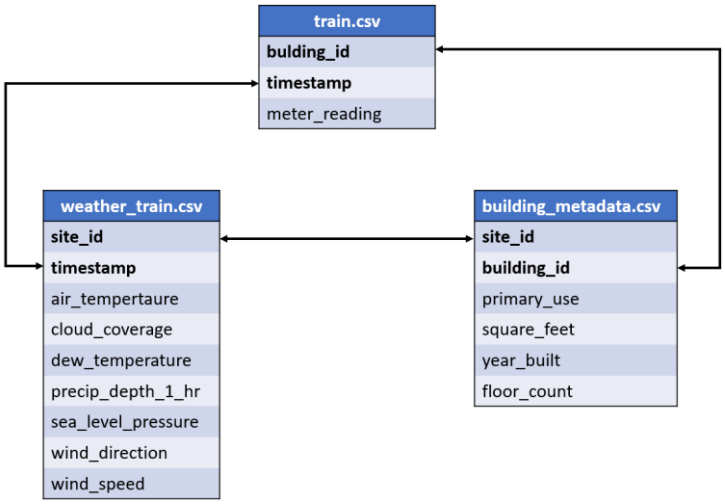

Figura 12. Representación de las tablas y de las claves foráneas

3.3.2 Limpieza de los valores nulos o en blanco

Los valores ausentes corresponden a la situación en la que el valor de un atributo para un determinado objeto no se conoce (Vicenç Torra i Reventós, 2007). Si no se tratan correctamente puede provocar que los

modelos nunca converjan. Por lo tanto, trataremos cada uno de estos valores para cada tabla.

Datos de entrenamiento (*train.csv*)

	Nulls (%)
building_id	0.000
timestamp	0.000
meter_reading	0.000

Tabla 2. Número de valores nulos en los datos de entrenamiento

	Zeros (%)
meter_reading	4.396
building_id	0.073
timestamp	0.000

Tabla 3. Número de valores igual a cero en los datos de entrenamiento

No tenemos ningún valor nulo, pero tenemos bastantes medidas igual a cero. Estas medidas son perfectamente posibles, pero se decide borrarlas ya que nuestro objetivo es predecir el consumo eléctrico en función de unas variables de entrada. Dejar estas medidas podrían provocar una distorsión en el aprendizaje de los modelos.

Datos meteorológicos (*weather_train.csv*)

	Nulls (%)
cloud_coverage	49.490
precip_depth_1_hr	35.979
sea_level_pressure	7.597
wind_direction	4.484
wind_speed	0.217
dew_temperature	0.081
air_temperature	0.039
site_id	0.000
timestamp	0.000

Tabla 4. Valores nulos en los datos meteorológicos

	Zeros (%)
precip_depth_1_hr	55.740
cloud_coverage	24.233
wind_speed	9.689
wind_direction	9.411
site_id	6.284
dew_temperature	1.440
air_temperature	0.581
timestamp	0.000
sea_level_pressure	0.000

Tabla 5. Valores igual a cero en los datos meteorológicos

	-1s (%)
precip_depth_1_hr	4.135
dew_temperature	0.176
air_temperature	0.068
site_id	0.000
timestamp	0.000
cloud_coverage	0.000
sea_level_pressure	0.000
wind_direction	0.000
wind_speed	0.000

Tabla 6. Valores igual a -1 en los datos meteorológicos

Los valores a cero parecen correctos, pero vemos que tenemos bastantes valores nulos que tenemos que tratar.

Existen varias maneras de tratar estos valores. En función del tipo de variable lo trataremos de una manera u otra.

- **Atributos continuos**

En el caso de valores nulos en atributos continuos optamos por sustituirlos por la media en función del sitio. Estos atributos son:

- *precip_depth_1_hr*
- *sea_level_pressure*
- *wind_direction*
- *wind_speed*
- *dew_temperature*
- *air_temperature*

Además, durante el procesamiento de los datos hemos detectado muchos valores igual a '-1' en el atributo *precip_depth_1_hr*. Probablemente se trate de un error en la lectura, ya que no es un valor posible. Por lo tanto, los consideraremos nulos y a continuación los sustituimos por la media de la precipitación en función de la ubicación.

También hemos detectado lugares en los que todos los valores de *precip_depth_1_hr* (*site_id* 5 y 12) y *sea_level_pressure* (*site_id* 5) son nulos. En estos casos decidimos sustituirlos por la media total del conjunto de datos de los atributos *precip_depth_1_hr* y *sea_level_pressure*.

- **Atributos discretos**

Por otro lado, el atributo *cloud_coverage* lo trataremos de manera distinta, ya que se trata de una variable discreta. En este caso la sustituiremos por la cantidad de nubes que más se repite en las observaciones en función de la ubicación. Esto es lo que se conoce como la moda.

En estadística, la moda se refiere a la medida de tendencia central que representa el valor o valores que ocurren con mayor frecuencia en un conjunto de datos. Dicho de otra manera, es el valor más común o popular en un conjunto de observaciones.

También nos volvemos a encontrar con sitios para los cuales la medida de *cloud_coverage* es siempre nula (*site_id* 7 y 11). En estos dos casos los sustituiremos por la moda del conjunto de los datos.

Metadatos de los edificios (*building_meta.csv*)

	Null (%)
floor_count	75.500
year_built	53.416
site_id	0.000
building_id	0.000
primary_use	0.000
square_feet	0.000

Tabla 7. Valores nulos en los metadatos de los edificios

	Zero (%)
site_id	7.246
building_id	0.069
primary_use	0.000
square_feet	0.000
year_built	0.000
floor_count	0.000

Tabla 8. Valores igual a cero en los metadatos de los edificios

Observamos que tenemos más de un 75 % de observaciones sin el número de pisos de los edificios. Por otro lado, también encontramos más de un 50 % sin el dato del año de construcción con el mismo problema.

Por la gran cantidad de valores nulos, se decide eliminar estos dos atributos. Podríamos sustituir estos valores nulos por la media como hemos hecho en los casos anteriores, pero la gran cantidad de valores nulos podría desvirtuar el aprendizaje de los modelos.

3.3.3 Limpieza de los valores duplicados

No se observa ningún registro duplicado en ninguna de los tres conjuntos de datos.

3.3.4 Limpieza de los valores atípicos o *outliers*

Los valores atípicos, también conocidos como *outliers* en inglés, son observaciones que difieren significativamente del patrón general de un conjunto de datos. Estos valores son inusuales o extremos en comparación con el resto de los datos y pueden afectar negativamente al análisis estadístico y la interpretación de los resultados.

Los *outliers* pueden surgir debido a diversos factores, como errores de medición, entrada incorrecta de datos, variabilidad natural en los datos o eventos inusuales. Pueden manifestarse como valores muy altos o bajos en relación con el resto de los datos, o bien como valores que se desvían significativamente de la tendencia o patrón observado en la mayoría de las observaciones.

Los modelos de aprendizaje profundo son particularmente sensibles a los valores atípicos debido a su capacidad para aprender patrones complejos en grandes conjuntos de datos (Asmaa F. Hassan, 2022).

Si un modelo se entrena con un conjunto de datos que contiene valores atípicos, es posible que el modelo no pueda generalizar bien a nuevos datos y tenga un rendimiento deficiente (Asmaa F. Hassan S. B., 2022). Por lo tanto, es muy importante detectar y eliminar los valores atípicos antes de entrenar un modelo de aprendizaje profundo para garantizar que el modelo sea preciso.

A continuación, buscaremos los valores atípicos de cada conjunto de datos con la ayuda de valores estadísticos y de la representación de los puntos mediante diagramas de cajas.

Un diagrama de caja es una representación estadística de la distribución de una variable a través de sus cuartiles. Los extremos de la caja representan los cuartiles inferior y superior, mientras que la mediana (segundo cuartil) está marcada por una línea dentro de la caja (Plotly Open Source Graphing Libraries, 2023).

- **Datos de entrenamiento (*train.csv*)**

Del conjunto de datos de entrenamiento solo analizaremos *meter_reading* ya que es la única variable que puede contener valores atípicos.

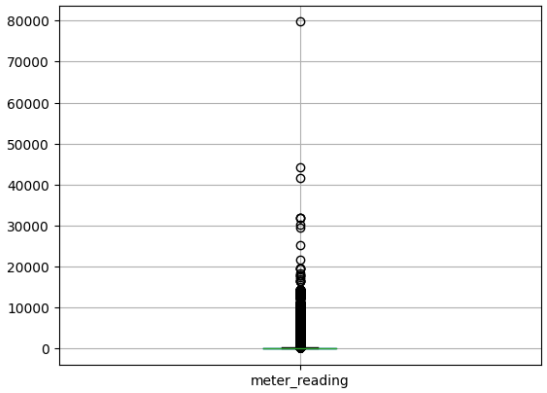

Figura 13. Diagrama de caja de la variable objetivo *meter_reading*

La medida está expresada en Unidades Térmicas Británicas (kBtu) que se corresponde a 0.2931 kWh. Por lo tanto, 80000 kBtu son 24 kWh que es un consumo muy elevado pero posible. De todas formas, como está muy alejado del resto de mediciones, se decide eliminar esta lectura.

- **Datos meteorológicos**

La variable *wind_direction* no es discreta, pero al estar acotada entre 0 y 360 no tiene sentido buscar valores atípicos, ya que hemos comprobado que todos los valores están comprendidos entre 0 y 360 y cualquier valor en este rango es posible.

Por otro lado, la variable *cloud_coverage* tampoco la representaremos ya que es una variable discreta que puede tomar cualquier valor entero entre 0 y 9 incluidos.

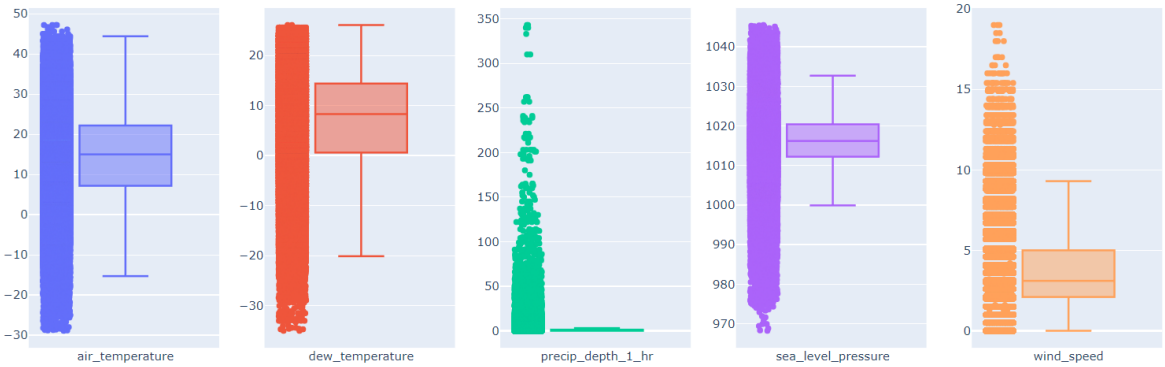

Figura 14. Diagramas de cajas de las variables: *air_temperature*, *dew_temperature*, *precip_depth_1_hr*, *sea_level_preasure* y *wind_speed*

En los gráficos anteriores podemos observar lo siguiente:

- **Temperatura del aire.** La mayoría de las lecturas se concentran entre 22.2º C y 7.2º C. La mínima es de -28.2º C y la máxima es de 47.2º C. Son bastante extremas pero posibles. Por lo tanto, no encontramos ningún *outlier*.

- **Temperatura de rocío.** La mayoría de las lecturas se concentran entre 14.4ºC y 0.6º C. La mínima es de -35º C y la máxima de 26.1º C. Como antes, son bastante extremas pero posibles. Por lo tanto, no encontramos ningún *outlier*.

- **Precipitación.** La mayoría de las lecturas se concentran entre 1.05 y 0. La mínima es de 0 y la máxima de 343. La máxima es muy alta, teniendo en cuenta que según la *World Meteorological Organization* la máxima registrada ha sido de 305 mm en una hora (World Meteorological Organization, 2023). Por lo tanto, se tratan claramente de *outliers* que tenemos que tratar.

- **Presión.** La mayoría de las lecturas se concentran entre 1020.4 hPa y 1012.2 hPa. La mínima es de 968.2 hPa y la máxima de 1045.5 de hPa. Estas medidas son normales, por lo que no encontramos ningún *outlier*.

- **Velocidad del viento.** La mayoría de las lecturas se concentran entre 2.10 mm/s y 5 mm/s. La mínima es de 0 mm/s y la máxima es de 19 mm/s. La velocidad máxima es alta, pero es posible (tengamos en cuenta que un aviso amarillo por viento en el Litoral de Barcelona

se da para rachas de 70 km/h (19,4 m/s) (AEMET, 2023). Por lo tanto, no encontramos ningún *outlier*.

A continuación, analizaremos los valores atípicos de la precipitación:

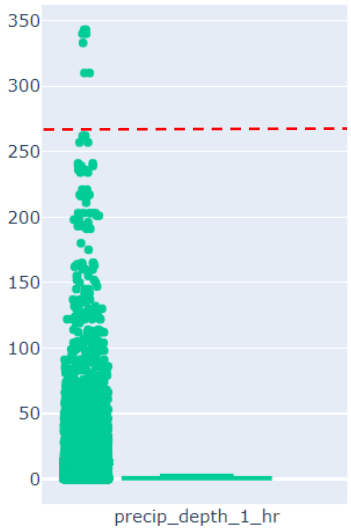

Figura 15. Diagrama de caja de la variable *precip_depth_1_hr*

Del gráfico anterior observamos un salto entre 262 mm/h y el siguiente valor de 310 mm/h. Más concretamente existen 7 registros mayores a 262. Se decide marcar como máximo 262 que, aunque sea muy elevado, es un valor posible y sustituir los valores que lo rebasan por este máximo.

- **Metadatos de los edificios**

La variable *primary_use* es discreta por lo que lo único que vamos a ver es que las etiquetas de los usos tengan sentido y se puedan identificar.

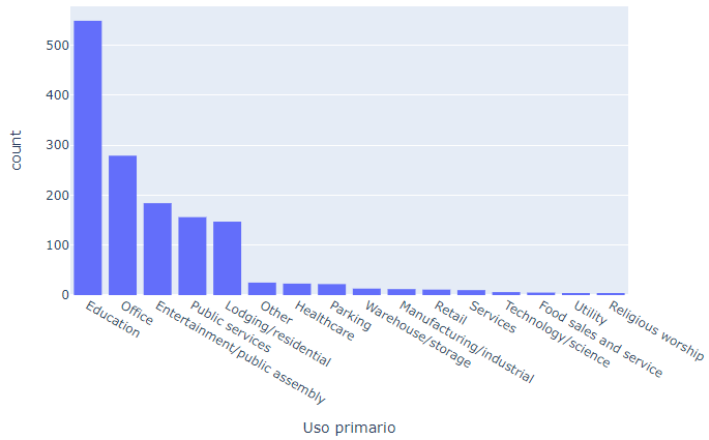

Figura 16. Gráfico de barras con los usos primarios de los edificios a estudiar

En el gráfico de barras anterior podemos observar cómo la mayoría de los edificios evaluados están destinados a la educación y que todas las

etiquetas de los usos son correctas, pero muy desiguales. Por lo que se decide contar el número de usos y agrupar las etiquetas.

Education	549
Office	279
Entertainment/public assembly	184
Public services	156
Lodging/residential	147
Other	25
Healthcare	23
Parking	22
Warehouse/storage	13
Manufacturing/industrial	12
Retail	11
Services	10
Technology/science	6
Food sales and service	5
Utility	4
Religious worship	3

Tabla 9. Número de edificios en función del uso primario

Como se puede observar en la tabla anterior, a partir de los usos de “Otros” el número de edificios registrados son mucho menores con respecto a los edificios destinados a la Educación, a Oficinas, Entretenimiento, Servicios públicos y Residencial. Por lo tanto, para simplificar los datos y reducir su tamaño, se decide unir los usos por debajo de “Otros” en el mismo grupo. Los datos quedan de la siguiente forma:

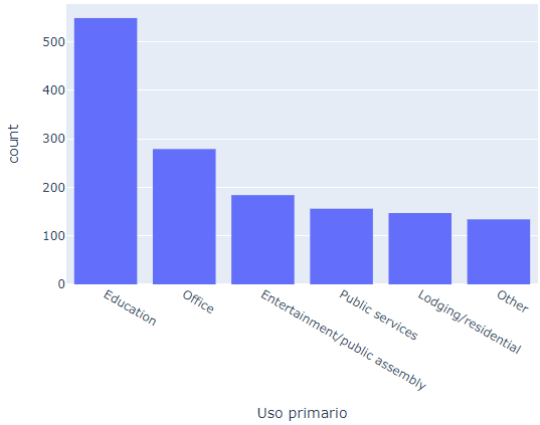

Figura 17. Gráfico de barras con los usos primarios después de la agrupación

Education	549
Office	279
Entertainment/public assembly	184
Public services	156
Lodging/residential	147
Other	134

Tabla 10. Número de usos primarios después de la agrupación

Por otro lado, revisaremos los *outliers* de la variable *square_feet.*

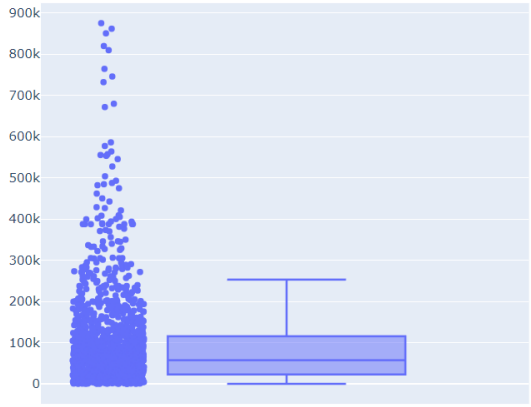

Figura 18. Diagrama de caja de la variable *square_feet*

Podemos observar diferentes valores de superficie. Estos valores se encuentran sobre todo entre 115.704k y 22.99k. Por otro lado, vemos como el máximo es 875k y el mínimo es 283. Todos los valores son normales, por lo tanto, no detectamos ningún *outlier*.

3.3.5 Análisis Gráfico de los atributos

Después de haber limpiado los datos unimos los tres conjuntos de datos de datos vamos a representar los atributos de todos los atributos.

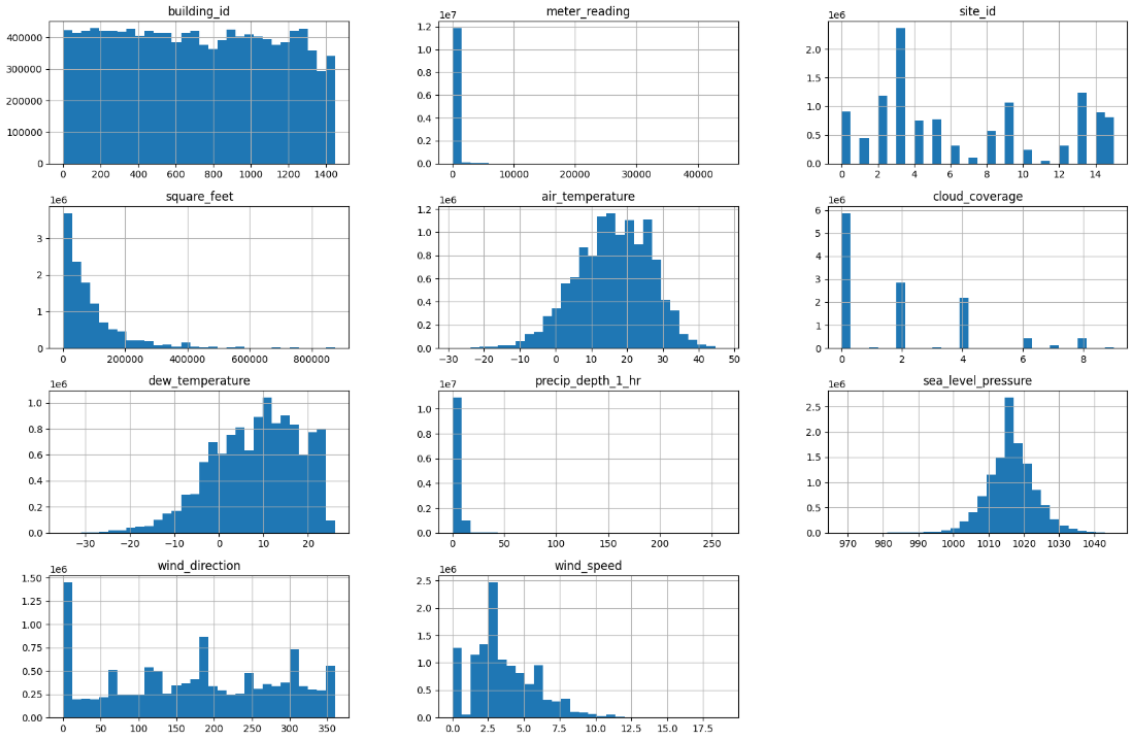

Figura 19. Histogramas de los atributos después de limpiar los datos y de unir las tablas

Por otro lado, con el objetivo de encontrar atributos que estén estrechamente relacionados y ver si podemos descartar alguno para reducir el número de atributos vamos a ver que relación existe entre los

atributos. Para ello vamos a utilizar el coeficiente de correlación de Pearson.

Este coeficiente de correlación es una medida estadística que evalúa la fuerza y la dirección de una relación lineal entre dos variables. Se calcula mediante la siguiente fórmula:

$$\rho_{X,Y} = \frac{cov(X,Y)}{std(x) \cdot std(Y)}$$

Donde:

$\rho_{x,y}$ es el coeficiente de correlación de Pearson
cov(X,Y) es la covarianza de X e Y
std(X) es la desviación típica de X
std(Y) es la desviación típica de Y

El resultado es un número entre -1 y +1, donde un valor de -1 indica una correlación negativa perfecta, un valor de +1 indica una correlación positiva perfecta y un valor de 0 indica que no existe ninguna relación.

	building_id	meter_reading	site_id	square_feet	air_temperature	cloud_coverage	dew_temperature	precip_depth_1_hr	sea_level_pressure	wind_direction	wind_speed
building_id	1.000000	0.004287	0.975536	0.086285	-0.282372	-0.280994	-0.125827	0.219620	0.037480	0.029676	0.000178
meter_reading	0.004287	1.000000	0.013694	0.564593	-0.002890	-0.044330	0.004957	0.050551	-0.009118	-0.002302	-0.022572
site_id	0.975536	0.013694	1.000000	0.094763	-0.266708	-0.266895	-0.145479	0.221214	0.013730	0.024033	-0.002046
square_feet	0.086285	0.564593	0.094763	1.000000	-0.005480	-0.061900	-0.025234	0.016122	-0.020227	-0.020058	-0.039734
air_temperature	-0.282372	-0.002890	-0.266708	-0.005480	1.000000	0.125875	0.757989	-0.077815	-0.279472	-0.104261	-0.081060
cloud_coverage	-0.280994	-0.044330	-0.266895	-0.061900	0.125875	1.000000	0.127123	-0.069448	-0.033959	0.006982	0.115901
dew_temperature	-0.125827	0.004957	-0.145479	-0.025234	0.757989	0.127123	1.000000	0.014468	-0.197973	-0.169826	-0.130879
precip_depth_1_hr	0.219620	0.050551	0.221214	0.016122	-0.077815	-0.069448	0.014468	1.000000	-0.033398	0.005704	0.031155
sea_level_pressure	0.037480	-0.009118	0.013730	-0.020227	-0.279472	-0.033959	-0.197973	-0.033398	1.000000	-0.095215	-0.197313
wind_direction	0.029676	-0.002302	0.024033	-0.020058	-0.104261	0.006982	-0.169826	0.005704	-0.095215	1.000000	0.411141
wind_speed	0.000178	-0.022572	-0.002046	-0.039734	-0.081060	0.115901	-0.130879	0.031155	-0.197313	0.411141	1.000000

Figura 20. Representación de correlaciones de los atributos

En la tabla de correlaciones podemos observar una fuerte relación entre *site_id* y *building_id*. Esto probablemente se deba a que el identificador del edificio se haya establecido en función de su ubicación.

Por otro lado, como cabría esperar, observamos una fuerte relación entre los metros cuadrados y el consumo de energía eléctrica.

Además, observamos que la temperatura de rocío está fuertemente relacionada con la temperatura del aire y que la velocidad del viento está muy relacionada con la dirección.

Por lo tanto, comprobaremos si podemos reducir el número de variables del *dataframe* manteniendo la información.

3.3.6 Codificación de las variables discretas

Antes de comprobar si podemos reducir los datos o de entrenar los modelos con estos codificaremos las variables discretas, ya que para su interpretación necesitamos que estos valores sean numéricos.

En el conjunto de datos tenemos las siguientes variables discretas:

- *site_id*
- *cloud_coverage*
- *primary_use*

Para la codificación de estas variables utilizaremos el método de codificación *one-hot*.

Con este método de representación creamos una columna para cada valor distinto que exista en la característica que estamos codificando y, para cada registro, marcaremos con un 1 la columna a la que pertenezca dicho registro y dejaremos a 0 el resto.

En el caso de *primary_use*, por ejemplo, tendremos la siguiente codificación de los primeros tres usos:

PRIMARY_USE	PRIMARY_ USE_ EDUCATION	PRIMARY_ USE_ OFFICE	PRIMARY_ USE_ ENTERTAINMENT/ PUBLIC_ASSEMBLY
EDUCATION	**1**	0	0
OFFICE	0	**1**	0
ENTERTAINMENT/PUBLIC ASSEMBLY	0	0	**1**

Tabla 11. Ejemplo de codificación *one-hot* sobre los usos primarios de los edificios

Continuaremos aplicando el mismo sistema hasta codificar cada uno de los usos.

Cabe destacar que empleamos la codificación *one-hot* porque es útil cuando no hay un orden intrínseco en las categorías y se desea evitar una interpretación numérica. Además, esta codificación ayuda a evitar posibles suposiciones de orden o magnitud entre las categorías.

Por otro lado, también tenemos la variable *timestamp* que se trata de un objeto de tipo *datetime*. Para poder procesarlo transformaremos esta variable a un número entero.

3.3.7 División del conjunto de los datos

Cuando se tiene solo un conjunto de datos lo que se suele hacer es dividirlo en cuatro conjuntos de datos. Estos cuatro grupos son:

- **Conjunto de características de entrenamiento:** es el conjunto de características que usaremos para entrenar a los modelos. En nuestro caso utilizaremos un 80 % de las observaciones.

- **Variable objetivo de entrenamiento:** este conjunto solo contendrá la variable objetivo *meter_reading* del conjunto de entrenamiento.

- **Conjunto de características de test:** es el conjunto de características que usaremos para evaluar nuestros modelos. En nuestro caso utilizaremos un 20 % de las observaciones.

- **Variable objetivo de test**: este conjunto solo contendrá la variable objetivo *meter_reading* del conjunto de test.

Introduciendo a nuestros modelos el conjunto de características de test y comparando los resultados de las predicciones con el conjunto de la variable objetivo de test, obtendremos el error entre la magnitud real y la predicha.

3.3.8 Normalización de los datos

Otro problema con el que nos encontramos cuando procesamos los datos son los provocados por los diferentes dominios de los atributos. Este sesgo puede provocar:

- **Sesgos en el aprendizaje**: Los modelos de aprendizaje pueden verse afectados por atributos con rangos de valores muy diferentes. Algunos algoritmos de aprendizaje automático, como las redes neuronales, son sensibles a la escala de los atributos y pueden dar más importancia a aquellos con valores más altos. La normalización ayuda a evitar este sesgo al ajustar y reescalar los atributos a un rango común.

- **Convergencia lenta o nula de los modelos**: Al normalizar los datos, se reduce la escala de los atributos a un rango más manejable, lo que puede ayudar a que los algoritmos de aprendizaje automático converjan más rápidamente durante el entrenamiento.

Existen varías técnicas para normalizar los datos, pero en este caso se decide aplicar **estandarización** por las siguientes razones:

- La mayoría de los atributos son numéricos o categóricos (Vicenç Torra i Reventós, 2007).

- En métodos como las redes neuronales puede ayudar a que los modelos converjan más deprisa (Sons, 1994).

Por lo tanto, vamos a ver brevemente como se aplica la estandarización al conjunto de los datos.

La estandarización consiste en normalizar los valores de manera que el valor medio sea cero y la desviación 1. Cada valor v asociado al atributo A para el objeto O se transforma en v' aplicando la siguiente fórmula:

$$v' = \frac{\left(v - media(A(X))\right)}{desviación(A(X))}$$

Donde la media es:

$$media(A(X)) = \frac{\sum_{i=1}^{N} x_i}{N}$$

Y la desviación es:

$$desviacion(A(X)) = \sqrt{\frac{\sum_{i=1}^{N} \left(x_i - media(A(X))\right)^2}{N}}$$

Técnicamente podríamos normalizar todo el conjunto de datos antes de realizar la división de los conjuntos, pero lo más correcto es que primero dividamos los datos en un conjunto de entrenamiento y otro de test para, a continuación, aplicar estandarización al conjunto de entrenamiento. Por último, aplicaremos estandarización sobre el conjunto de test aplicando la media y la desviación típica de los datos de entrenamiento.

Si aplicáramos estandarización antes de dividir los datos estaríamos prefiltrado los datos de test y entrenamiento por lo que, al evaluar nuestros modelos, podríamos obtener resultados mejores de los reales.

Cabe señalar que una vez tengamos nuestro modelo entrenado, los datos de entrada de nuestro modelo deberán tratarse utilizando exactamente los mismos criterios utilizados hasta ahora. Es decir, codificación de las variables discretas aplicando *one-hot* y estandarización utilizando la media y la desviación típica del conjunto de entrenamiento.

3.3.9 Reducción de la dimensionalidad PCA

Como hemos visto en el apartado de análisis gráfico de las variables había una fuerte relación entre varios atributos. Por lo tanto, es muy posible que podamos utilizar la descomposición PCA (del inglés *Principal Component Analysis*) para reducir la dimensionalidad de los datos manteniendo la mayor parte de la información.

La reducción de la dimensionalidad a través de PCA puede ser beneficiosa en varios aspectos. Por un lado, ayuda a eliminar la redundancia y el ruido en los datos, lo que puede mejorar el rendimiento de los algoritmos de aprendizaje automático. Además, puede acelerar los cálculos y reducir los requisitos de almacenamiento al trabajar con grandes conjuntos de datos.

El objetivo principal de la reducción de la dimensionalidad mediante PCA es encontrar una representación más compacta y significativa de los datos, manteniendo la mayor cantidad posible de información relevante.

La idea básica detrás de PCA es transformar un conjunto de variables correlacionadas en un nuevo conjunto de variables no correlacionadas llamadas componentes principales. Estos componentes principales están ordenados en función de su varianza, lo que significa que los primeros componentes principales capturan la mayor parte de la variabilidad de los datos originales. A continuación, vamos a ver como se calcula.

En primer lugar, habría que centrar los datos en cero restándoles su media, pero como hemos aplicado estandarización en el apartado anterior, la media es cero:

$$\hat{X} = X - M = X$$

Donde X es la matriz de datos y M la media de las columnas de X.

Cabe destacar que hemos utilizado los datos que hemos normalizado, pero para aplicar PCA hubiéramos podido utilizar también los datos originales (sin normalizar).

En segundo lugar, calculamos la matriz de covarianza.

$$C = \frac{1}{N} \hat{X}^T \times \hat{X}$$

Donde N es el número de ejemplos de los datos.

A continuación, calcularemos los valores y vectores propios de la matriz de covarianzas.

Recordemos que los valores propios de una matriz son aquellos que cumplen la siguiente desigualdad

$$|C - \lambda I| = 0$$

Donde C es nuestra matriz de covarianzas, I la matriz identidad y λ los valores propios que cumplen esta igualdad.

```
[8.07690561e-02 5.68222970e-02 4.85419154e-02 4.56492858e-02
 4.23716990e-02 3.97723919e-02 3.70271110e-02 3.54762779e-02
 3.51709582e-02 3.29717523e-02 2.93872378e-02 2.83766082e-02
 2.78087242e-02 2.71480195e-02 2.67547339e-02 2.58395086e-02
 2.53246127e-02 2.51162884e-02 2.46807456e-02 2.44881836e-02
 2.44553400e-02 2.44249624e-02 2.43692618e-02 2.41360874e-02
 2.27097295e-02 2.21766393e-02 1.90204375e-02 1.88258244e-02
 1.73966332e-02 1.57888050e-02 1.37825389e-02 1.29401098e-02
 1.27699855e-02 1.14225166e-02 8.69464352e-03 4.79556847e-03
 2.63257638e-03 1.60933267e-04 6.23100716e-32 4.50771381e-32
 2.17865361e-32]
```

Una vez calculados los autovalores, calculamos los vectores propios. Para ello tendremos que resolver el siguiente sistema de ecuaciones para cada autovalor:

$$(C_{n\times n} - \lambda_i I) \cdot X = 0$$

Donde X es el autovector con sus componentes x_1, x_2, …, x_n que serán las incógnitas del sistema de ecuaciones. A cada autovalor le corresponde un autovector. Los autovectores representan las direcciones en las que los datos tienen la mayor variabilidad, mientras que los autovalores indican la cantidad de varianza explicada por cada autovector.

	building_id	timestamp	square_feet	air_temperature	dew_temperature	precip_depth_1_hr	sea_level_pressure	wind_direction	wind_speed	site_id_0	...
0	0.417	-0.074	0.070	-0.331	-0.262	0.132	0.057	0.047	0.015	-0.185	...
1	-0.028	-0.155	-0.098	-0.331	-0.324	-0.041	0.075	0.247	0.301	-0.162	...
2	-0.052	-0.031	0.070	0.020	-0.039	-0.055	-0.202	0.144	0.189	0.042	...
3	0.327	0.125	-0.053	0.183	0.275	0.102	-0.059	-0.061	0.011	0.075	...
4	0.059	0.087	0.042	0.064	0.206	0.177	0.067	-0.133	-0.104	-0.073	...
5	0.185	0.013	-0.035	-0.074	-0.033	0.397	0.071	0.153	0.100	-0.023	...
6	-0.095	0.011	-0.198	0.217	0.183	0.223	-0.431	0.311	0.394	0.039	...
7	-0.127	0.125	0.209	-0.057	-0.089	0.253	0.030	-0.034	-0.030	0.314	...

Tabla 12. Representación de los primeros 7 auto vectores

La interpretación de cada componente es la combinación lineal de las variables, por ejemplo, para la componente 0 tendremos:

0.417·building_id - 0.074·timestamp + 0.070·square_feet + […]

Y así, hasta las 41 componentes principales.

A continuación, normalizaremos los valores propios anteriores realizando la división de cada uno entre la suma de todos ellos para que su suma sea 1. Por último, los ordenamos en orden decreciente para, posteriormente, calcular la información original que preserva cada dimensión calculando la suma acumulada de los autovalores.

```
[0.08076906 0.13759135 0.18613327 0.23178255 0.27415425 0.31392665
 0.35095376 0.38643003 0.42160099 0.45457274 0.48395998 0.51233659
 0.54014531 0.56729333 0.59404807 0.61988758 0.64521219 0.67032848
 0.69500922 0.71949741 0.74395275 0.76837771 0.79274697 0.81688306
 0.83959279 0.86176943 0.88078987 0.89961569 0.91701232 0.93280113
 0.94658367 0.95952378 0.97229376 0.98371628 0.99241092 0.99720649
 0.99983907 1.         1.         1.         1.         ]
```

Gráficamente:

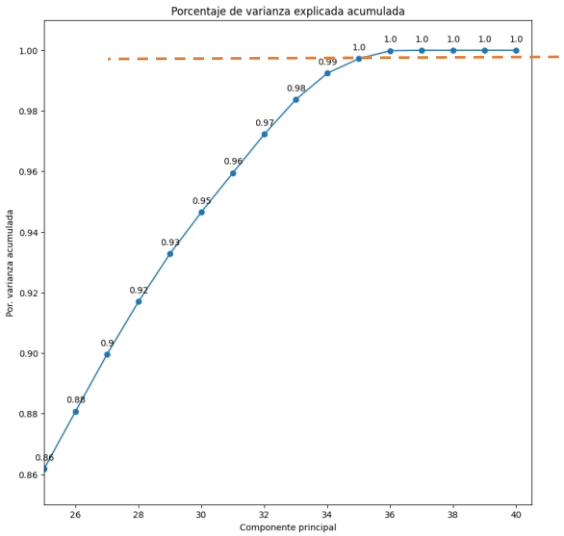

Figura 21. Representación de la suma acumulada de los autovalores

Esto quiere decir que, podemos preservar casi el 100 % de la información descartando los últimos 5 componentes.

Por último, los datos originales se proyectan sobre el espacio definido por las componentes principales seleccionadas. Esto se realiza realizando la multiplicación escalar entre la matriz de datos estandarizados por la matriz de autovectores correspondientes a los componentes principales seleccionados. El resultado es un nuevo conjunto de datos con una dimensionalidad reducida.

El impacto de reducir la dimensionalidad de los datos sobre la memoria se muestra en el siguiente gráfico:

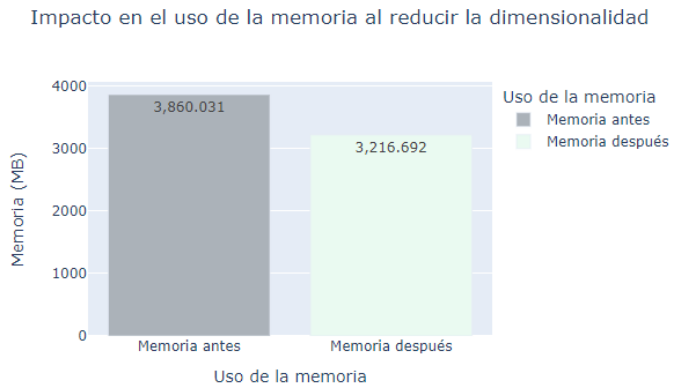

Figura 22. Representación de la memoria antes y después de reducir la dimensionalidad

Como podemos observar en el gráfico anterior, reduciendo la dimensionalidad y conservando casi el 100% de la información contenida en los atributos, podemos ahorrar un 16.67 % de memoria. Lo que repercutirá directamente en el espacio utilizado en el disco en la memoria RAM y en la velocidad de entrenamiento de los modelos.

3.4 Modelos de aprendizaje

El objetivo de este problema es predecir el consumo eléctrico de los edificios. Por lo tanto, la variable objetivo es una variable continua y el problema se trata de un problema de regresión.

Para realizar las predicciones hemos implementado los siguientes modelos que resuelven el problema de regresión:

- Modelo ANN
- Modelo *LightGBM*

En primer lugar, hablaremos del error que hemos utilizado para comprobar la validez de los resultados y para comparar los resultados entre los diferentes modelos.

En segundo lugar, se explicará cada modelo, los parámetros de ajuste más importantes de cada uno y, por último, se aplicará cada modelo con todos los datos y con los datos reducidos mediante PCA para comparar los resultados.

A continuación, volveremos a entrenar los modelos con los datos de un único edificio para comprobar si podemos mejorar la calidad de las predicciones con los datos de un único edificio.

Por último, evaluaremos y compararemos los resultados de cada modelo.

3.4.1 Evaluación del error de los modelos

El error en el contexto de un modelo de regresión es la diferencia entre el valor real de la variable dependiente y el valor predicho por el modelo. Esta diferencia se representa como un número positivo o negativo, dependiendo de si la predicción es mayor o menor que el valor real.

Concretamente, para evaluar la calidad de los modelos vamos a utilizar el error cuadrático medio (ECM) que se define de la siguiente manera:

$$\mathrm{ECM} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Donde:

- **ECM**: Error cuadrático medio.
- **n**: Número total de observaciones o datos.
- **y**: Valor real de la variable dependiente en la i-ésima observación.
- **ȳ**: vector de n predicciones.

Por tanto, el ECM es una medida del promedio de los errores al cuadrado. Esto tiene varias implicaciones importantes. La primera es que, elevando los errores al cuadrado el ECM penaliza los errores más grandes frente a los más pequeños. Además, debido a que los errores se elevan al cuadrado, el ECM siempre será un número positivo.

3.4.2 Red Neuronal Artificial ANN

Una red neuronal artificial o ANN (del inglés *Artificial Neural Network*) es un sistema informático inspirado en las redes neuronales biológicas que constituyen los cerebros de los humanos.

Las ANN están compuestas por capas de entrada, capas ocultas y capas de salida con neuronas conectadas (nodos) para simular el cerebro humano. Los nodos están interconectados y contienen una función de activación.

La primera capa recibe la entrada de los atributos y la última produce la salida. En este caso, la capa de entrada recibe los parámetros de todos los atributos que hemos visto en el apartado de análisis, y la capa de salida estará compuesta por una única variable continua que representa el consumo de energía eléctrica.

Cabe destacar que las ANN se caracterizan por ser adaptativas, lo que significa que se modifican a sí mismas a medida que aprenden del entrenamiento inicial y las ejecuciones posteriores proporcionan más información sobre el mundo.

El modelo de aprendizaje más básico se centra en la ponderación de los flujos de entrada, que es la forma en que cada nodo pondera la

información que recibe. Nosotros utilizaremos las redes neuronales *feed-forward*. Estas redes pasan la información en una única dirección a través de varios nodos de entrada hasta llegar a la capa de salida.

Para construir el modelo utilizaremos el tipo *Sequential*. Este tipo de modelo nos permite construir el modelo capa por capa. Además, utilizaremos el tipo de capa *Dense* que conecta todos los nodos de una capa con los nodos de la capa siguiente.

Los parámetros más importantes para construir y entrenar una red neuronal artificial son:

- **Número de capas**: es el número de capas que va a tener nuestra red. En todos los casos vamos a tener una capa de entrada, que recibirá los valores de los atributos y una capa de salida con la predicción del consumo de energía eléctrica del edificio. Entre la capa de entrada y la de salida tendremos las capas ocultas. En todos los modelos que hemos entrenado hemos creado 3 capas ocultas.

- **Tasa de aprendizaje**: Establece la tasa de aprendizaje, que determina el tamaño del paso del optimizador. Controla cuánto se actualizan los pesos en cada iteración de entrenamiento. En todos los modelos hemos utilizado una tasa de aprendizaje de 0.01.

- **Tamaño de lote**: Determina el número de muestras procesadas antes de actualizar los pesos. Un tamaño de lote menor puede dar lugar a un proceso de entrenamiento más estocástico (no determinista), mientras que un tamaño de lote mayor puede proporcionar una estimación del gradiente más suave. Este parámetro lo hemos ido ajustando en función del modelo.

- **Épocas**: Define el número de veces que el conjunto completo de datos pasa por la red durante el entrenamiento. Este parámetro afecta a la duración del entrenamiento y a la convergencia del modelo. Este parámetro lo hemos ido ajustando en función del modelo.

- **Función de pérdida**: La función de pérdida cuantifica el rendimiento del modelo durante el entrenamiento. En todos los modelos hemos utilizado la función de perdida el error cuadrático medio o *mse* (del inglés *mean squared error*).

- **Optimizador**: El optimizador se encarga de actualizar los pesos del modelo durante el entrenamiento. Entre los optimizadores más populares podemos encontrar: *Stochastic Gradient Descent* (SGD), *adam* y *RMSprop*. En nuestro caso, hemos decidido utilizar el optimizador adam (del inglés *Adaptive Moment Estimation*) para todos los modelos.

Escogemos este optimizador porque utiliza un ritmo de aprendizaje adaptativo. Esto significa que la tasa de aprendizaje no es fija, sino que se actualiza en función del historial de gradientes. Esto ayuda a que el modelo converja más rápidamente y evita que se atasque en mínimos locales (P. Kingma & Lei Ba, 2017).

- **callbacks**: este parámetro se utiliza para realizar acciones en puntos específicos durante el entrenamiento de una ANN. Nosotros lo vamos a aprovechar para definir un objeto *EarlyStopping* que detendrá el entrenamiento si no se produce una mejora en la métrica de pérdida de validación después de cierta cantidad de épocas (*patience*).

Para cada capa los parámetros más importantes son:

- **Número de neuronas**: Este parámetro determina la capacidad y complejidad del modelo. El número de neuronas en las capas de entrada y salida depende de los datos, mientras que el número de neuronas en las capas ocultas es un parámetro que hemos ido variando en los diferentes modelos.

- **Función de activación**: La función de activación determina cómo interactúan entre sí las neuronas de una capa. Existen muchas funciones de activación diferentes. Algunas funciones de activación comunes para problemas de regresión son relu, sigmoide y tanh. Nosotros vamos a escoger la función de activación ReLu para las capas ocultas, ya que es una de las que mejores resultados está dando (Lug, 2023).
  La función de activación Relu aplica la función de activación no saturada:
  $$f(x) = max(0, x)$$
  Es decir, devuelve 0 si su valor es negativo o el mismo valor, si es positivo.
  Por otro lado, aplicaremos la función de activación lineal para la capa de salida.

3.4.2.1 Implementación

Para implementar los modelos, en primer lugar, hemos realizado numerosas pruebas variando cada parámetro para mejorar los resultados con todos los datos (sin reducir la dimensionalidad).

A continuación, se muestra los 6 ajustes que han dado mejores resultados. Cabe señalar, que se han realizado muchas más pruebas para llegar a estos valores.

n1	n2	n3	*batch_size*	*epochs*	*learning_rate*	*patience*	*ECM*
64	32	16	256	100	0.05	∞	0.512
64	64	64	600	25	0.01	2	0.426
64	64	64	600	20	0.02	2	0.564
64	64	64	600	25	0.01	5	0.232
128	128	128	1024	25	0.01	5	0.360
128	128	128	1024	100	0.01	10	0.202

Tabla 13. Ajustes para optimizar el modelo ANN

En la siguiente figura, podemos ver una representación simplificada de la red neuronal artificial generada:

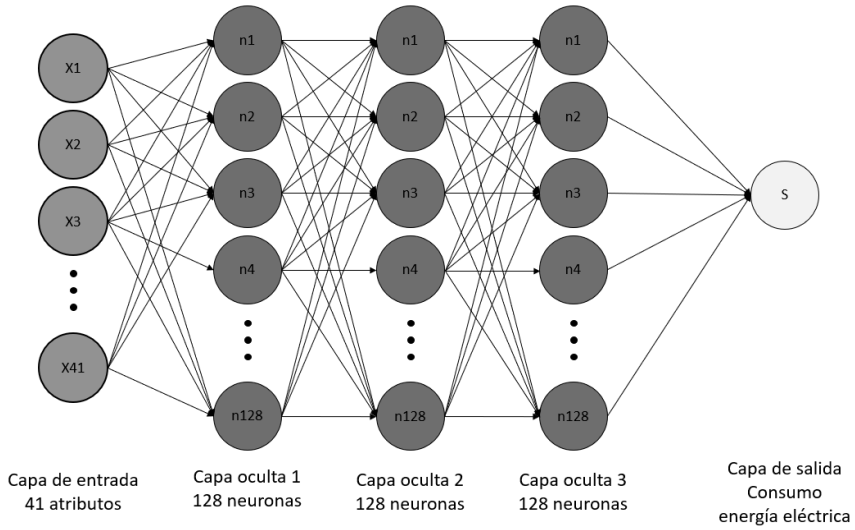

Figura 23. Representación de la ANN con todos los atributos

Cabe destacar que, aparte de lo dicho anteriormente en el análisis general de los parámetros hemos observado lo siguiente en el ajuste:

- Al tratarse de un conjunto de datos muy grande hemos podido mejorar los resultados aumentando los tamaños de lote.

- También hemos ido variando el número de neuronas en las capas ocultas, obteniendo los mejores resultados para n=128 en las tres capas.

- Por otro lado, el número de *epochs* lo hemos ido aumentando para que sea el parámetro *patience* el que termine con el entrenamiento en el caso de no mejorar los resultados en las 5 últimas épocas.

Por último, predecimos el consumo eléctrico con los datos de test y calculamos el ECM con los valores reales. A continuación, mostramos los resultados finales con los datos de entrada reducidos y sin reducir con los parámetros que mejor resultados nos han dado.

Modelo	ECM
Sin reducir la dimensionalidad	0.202

Reduciendo la dimensionalidad	0.242

Tabla 14. Resultados de los modelos ANN con todos los edificios

Como podemos observar reduciendo la dimensionalidad disminuimos la precisión de nuestro modelo en 0.04. Este valor se puede considerar aceptable ya que hemos reducido la memoria más de un 16%.

3.4.3 *LightGBM*

LightGBM (del inglés *Light Gradient Boosting Machine*) es un modelo de aprendizaje automático basado en árboles de decisión, que se utiliza principalmente para problemas de clasificación y regresión. Fue desarrollado por Microsoft y se destaca por su eficiencia y velocidad de entrenamiento.

LightGBM se basa en la construcción de múltiples árboles de decisión para realizar las predicciones. Los árboles de decisión son estructuras jerárquicas que dividen el conjunto de datos en diferentes ramas basadas en condiciones específicas. Cada rama representa una regla que se utiliza para tomar decisiones y asignar etiquetas a las instancias.

Este modelo utiliza el algoritmo de *Gradient Boosting*, que es una técnica de ensamblaje de modelos. En lugar de construir un solo árbol de decisión grande, *LightGBM* crea múltiples árboles de decisión en forma de secuencia. Cada árbol se construye para corregir los errores del árbol anterior, mejorando gradualmente la precisión del modelo.

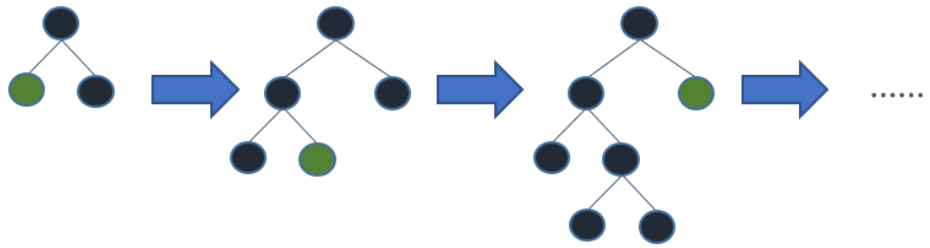

Figura 24. Representación del crecimiento de los árboles de decisión del modelo *LightGBM* (Microsoft, 2022)

Microsoft desarrollo una nueva técnica denominada Gradient-based One-Side Sampling (GOSS). Esta técnica selecciona un subconjunto de datos y se centra en las instancias con grandes gradientes (errores) durante el proceso de construcción del árbol. Esto permite reducir significativamente la cantidad de datos utilizados sin sacrificar la precisión del modelo.

En lugar de dividir los nodos en cada nivel del árbol de manera exhaustiva, *LightGBM* utiliza un enfoque "*leaf-wise*" (hoja-sabia). Este método selecciona el nodo que resulta en la mayor reducción de la función de pérdida (*loss*) en lugar de dividir todos los nodos de manera secuencial. Esto mejora la precisión y reduce la profundidad del árbol, lo que a su vez acelera el proceso de entrenamiento.

Por último, cabe señalar que *LightGBM* utiliza histogramas para representar los datos y acelerar el proceso de búsqueda de la mejor división en cada nodo. Los histogramas agrupan los valores de las características en diferentes intervalos y utilizan estas estadísticas agregadas para encontrar la mejor división. Esta técnica reduce la complejidad computacional y hace que el entrenamiento sea más rápido.

A continuación, vamos a ver los parámetros más importantes de entrenamiento del modelo:

- **objective**: Con este parámetro podemos definir el tipo de problema que se quiere resolver. En nuestro caso, al tratarse de un problema de regresión, siempre usaremos el parámetro “regression”.

- ***boosting***: Este parámetro especifica el tipo de algoritmo de *boosting* a utilizar. Las opciones más comunes son "gbdt" (del inglés *gradient boosting decision trees*) y "goss" (del inglés *gradient-based one-side sampling*). En líneas generales la diferencia entre un modelo y otro es su escalabilidad (más escalable GOSS frente a GBDT), su precisión (más preciso GBDT frente a GOSS) y su velocidad y coste computacional (más costoso y lento GBDT frente a GOSS).
  En nuestro caso vamos a utilizar siempre GBDT, ya que es más preciso.

- ***metric***: indica la métrica que se utiliza para la evaluar el conjunto de test. Como ya hemos comentado, en nuestro caso utilizaremos el error cuadrático medio o *mse* (del inglés *mean squared error*)

- ***num_leaves***: Este parámetro controla el número de hojas de cada árbol de decisión. Un mayor número de hojas dará lugar a un modelo más complejo, pero también puede llevar a un sobreajuste. Este parámetro lo hemos ido ajustando en función del modelo.

- ***learning_rate***: Este parámetro controla la tasa de aprendizaje del algoritmo boosting. Una tasa de aprendizaje mayor acelerará el aprendizaje, pero también puede llevar a un sobreajuste. Este parámetro lo hemos ido ajustando en función del modelo.

- ***feature_fraction***: Este parámetro controla la fracción de características a utilizar para cada árbol. Una fracción más pequeña dará lugar a un modelo más robusto, pero también puede conducir a un modelo menos preciso. Este parámetro lo hemos ido ajustando en función del modelo.

- ***bagging_fraction***: Este parámetro controla la fracción de puntos de datos a utilizar para cada árbol. Una fracción más pequeña dará lugar a un modelo más robusto, pero también puede

conducir a un modelo menos preciso. Por defecto el valor es 1. Este parámetro lo hemos ido ajustando en función del modelo.

- ***max_depth***: Este parámetro controla la profundidad máxima de cada árbol de decisión. Un árbol más profundo será más complejo, pero también puede dar lugar a sobreajustes. Este parámetro lo hemos ido ajustando en función del modelo.

- ***num_iterations***: Este parámetro controla el número de árboles a construir. Un mayor número de árboles dará lugar a un modelo más preciso, pero también tardará más en entrenarse. En nuestro caso usaremos en todos nuestros modelos **1500 iteraciones**.

- ***early_stopping_round***: Este parámetro nos permite controlar cuando debe parar el entrenamiento. El modelo parará de entrenar si una métrica de un dato de validación no mejora en las últimas n iteraciones. En nuestro caso hemos ajustado este parámetro a **50 iteraciones**.

- ***reg_alpha***: Este parámetro controla la intensidad de la regularización L1. Un valor mayor dará lugar a un modelo más regularizado, lo que puede ser útil para evitar el sobreajuste. Este parámetro lo hemos ido ajustando en función del modelo.

- ***reg_lambda***: Este parámetro controla la intensidad de la regularización L2. Un valor mayor dará lugar a un modelo más regularizado, lo que puede ser útil para evitar el sobreajuste. Este parámetro lo hemos ido ajustando en función del modelo.

3.4.3.1 Implementación

Como en el caso anterior, para implementar los modelos hemos realizado numerosas pruebas de ajuste variando cada parámetro para mejorar los resultados con todos los atributos (sin reducir la dimensionalidad). A continuación, mostramos 6 variaciones de los parámetros:

num_leaves	*learning_rate*	*feature_fraction*	*bagging_fraction*	*max_depth*	*reg_lambda*	*ECM*
1500	0.05	0.85	1	-1	2	0.422
1500	0.1	0.8	0.8	-1	4	0.056
1500	0.15	0.9	0.9	-1	4	0.049
1500	0.2	0.8	0.8	-1	10	0.047
2500	0.25	0.9	0.9	12	10	0.041
4000	0.25	0.9	0.9	-1	10	0.042

Tabla 15. Tabla con los diferentes ajustes realizados

Cabe destacar que, aparte de lo dicho anteriormente en el análisis general de los parámetros hemos observado lo siguiente:

- Teniendo en cuenta la gran cantidad de datos de los que disponemos, nos ha ayudado mucho aumentar el número de hojas mientras que hemos corregido el sobreajuste con el parámetro *reg_lambda.*

- También hemos ido viendo como con rangos de aprendizaje más altos obtenemos mejores resultados.

Por otro lado, hemos utilizado una herramienta muy interesante que disponible la librería *LightGBM* que nos permite representar la importancia de los atributos del modelo.

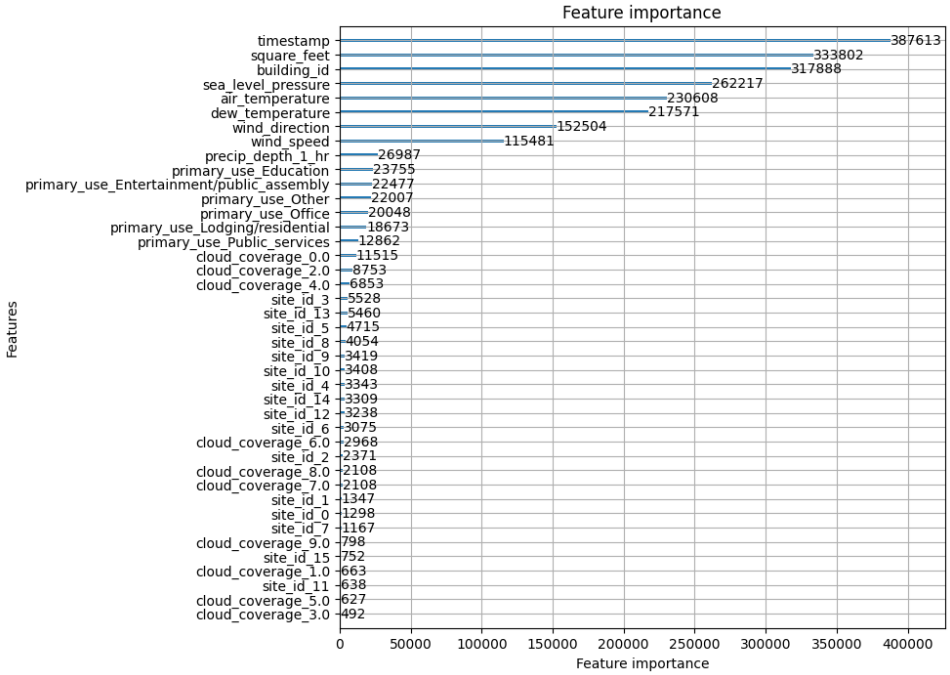

Figura 25. Representación de la importancia de los atributos

Como podemos ver en la gráfica anterior, lo que más influye en el consumo de energía eléctrica es la hora, el tamaño del edificio, el edificio, la presión, la temperatura, y el viento. Esto nos puede servir de indicativo de la calidad de los datos.

A continuación, hemos entrenado los modelos con los datos de entrada reducidos y sin reducir con los parámetros que mejor resultados nos han dado.

Modelo	ECM
Sin reducir la dimensionalidad	0.042
Reduciendo la dimensionalidad	0.456

Tabla 16. Resultados de los modelos *LightGBM* con todos los edificios

Como podemos observar, de nuevo hemos visto deteriorada la precisión de nuestro modelo al reducir la dimensionalidad. En este caso perdemos

0.414 de precisión. Hemos intentado ajustar el modelo para obtener mejores resultados sin mucho éxito. En el apartado de comparación de los resultados se reflexionará sobre este hecho.

3.4.4 Un único edificio

A continuación, vamos a repetir el mismo proceso, pero para un único edificio. De esta manera podremos comprobar si las diferencias entre los consumos de un edificio y otro y el ruido generado por el conjunto de los datos puede estar afectando a la precisión de las predicciones.

En primer lugar, buscamos el edificio que dispone de más datos.

	id	Número de registros
0	1217	8783
1	1118	8783
2	1147	8783
3	1146	8783
4	1143	8783

Tabla 17. Número de registros en función del edificio

Como podemos ver en la tabla anterior, disponemos de al menos 5 edificios con la misma cantidad de registros. Escogemos el primero y eliminamos todos aquellos atributos que son constantes para cada edificio:

- *building_id*: 1217
- *square_feet*: 73044
- *site_id*: 13
- *primary_use*: Educación

3.4.4.1 Reducción de la dimensionalidad mediante PCA.

De la misma manera que hicimos con el conjunto de datos de todos los edificios, realizamos un estudio de correlaciones para comprobar la probabilidad de poder aplicar la reducción de la dimensionalidad.

	meter_reading	air_temperature	cloud_coverage	dew_temperature	precip_depth_1_hr	sea_level_pressure	wind_direction	wind_speed
meter_reading	1.000000	-0.071805	-0.044673	-0.032569	-0.001478	0.007506	-0.031447	0.019450
air_temperature	-0.071805	1.000000	0.061317	0.935363	0.037243	-0.322047	-0.125768	-0.017560
cloud_coverage	-0.044673	0.061317	1.000000	0.048967	0.005863	-0.039452	-0.014251	0.020390
dew_temperature	-0.032569	0.935363	0.048967	1.000000	0.089999	-0.372184	-0.158668	-0.091804
precip_depth_1_hr	-0.001478	0.037243	0.005863	0.089999	1.000000	-0.082056	-0.011260	0.063431
sea_level_pressure	0.007506	-0.322047	-0.039452	-0.372184	-0.082056	1.000000	-0.059402	-0.237317
wind_direction	-0.031447	-0.125768	-0.014251	-0.158668	-0.011260	-0.059402	1.000000	0.327835
wind_speed	0.019450	-0.017560	0.020390	-0.091804	0.063431	-0.237317	0.327835	1.000000

Figura 26. Representación de correlaciones de los atributos (un edificio)

Como podemos observar en la tabla anterior volvemos a tener una relación muy fuerte entre la temperatura de rocío y la temperatura del aire.

A continuación, podemos ver una representación de la información que contiene cada atributo:

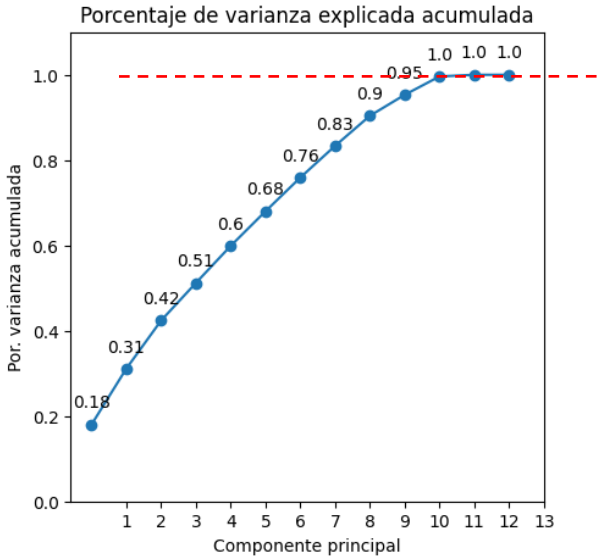

Figura 27. Representación de la suma acumulada de los autovalores (un edificio)

Como podemos ver, podemos preservar el 100 % de la información únicamente con los primeros 11 componentes.

El impacto de reducir la dimensionalidad de los datos sobre la memoria se muestra en el siguiente gráfico:

Figura 28. Representación de la memoria antes y después de reducir la dimensionalidad (un edificio)

Como podemos observar en el gráfico anterior, reduciendo la dimensionalidad conservando casi el 100 % de la información contenida en los atributos, podemos ahorrar un 21.43 % de memoria.

3.4.4.2 Red Neuronal Artificial ANN

Como en el caso anterior, hemos ido ajustando los parámetros para mejorar las predicciones. Los parámetros que mejores resultados nos han dado son:

n1	n2	n3	batch_size	epochs	learning_rate	patience
128	64	32	28	**1000**	0.001	**10**

Tabla 18. Parámetros de la ANN para un edificio

Como se puede ver, al tratarse de menos registros que en el caso anterior, hemos aumentado el número de épocas y reducido el tamaño de los bloques.

En la siguiente figura, podemos ver una representación simplificada de la red neuronal artificial generada de tres capas ocultas de 128 neuronas la primera, 64 la segunda y 32 la tercera:

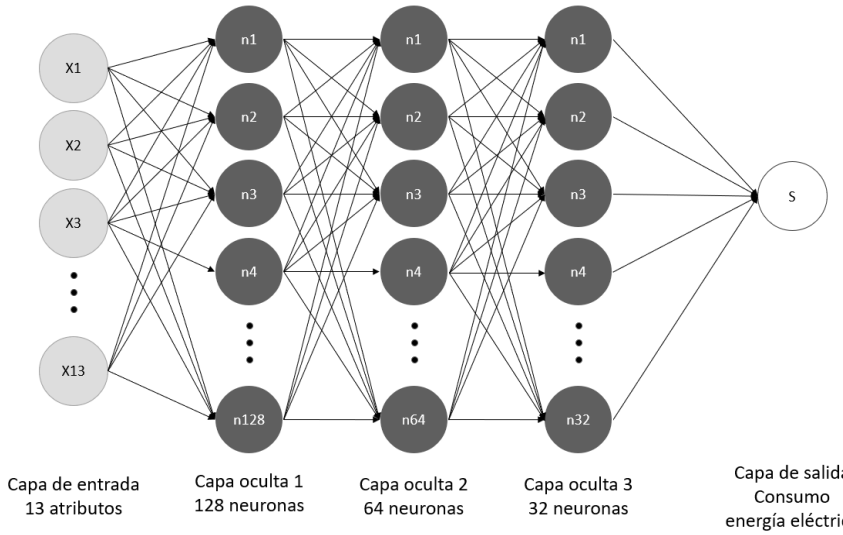

Figura 29. Representación de la ANN con todos los atributos para un edificio

Por último, predecimos el consumo eléctrico con los datos de test y calculamos el ECM con los valores reales. A continuación, mostramos los resultados finales con los parámetros que mejor resultados nos han dado.

Modelo	ECM
Sin reducir la dimensionalidad	0.080
Reduciendo la dimensionalidad	0.078

Tabla 19. Resultados de los modelos ANN para un edificio

Como podemos observar, en este caso, reduciendo la dimensionalidad hemos aumentado la precisión 0.02. Lo que es un muy buen resultado ya que hemos reducido el espacio de la memoria en más de un 20% preservando casi el 100% de la información de los atributos.

3.4.4.3 *LightGBM*

Como en el caso anterior, hemos ido ajustando los parámetros para mejorar las predicciones. Los parámetros que mejores resultados nos han dado son:

num_ leaves	learning_ rate	feature_ fraction	bagging_ fraction	max_ depth	reg_ lambda	num_ iterations
128	0.05	0.9	0.9	-1	2	2000

Tabla 20. Parámetros del modelo *LightGBM* para un edificio

Como podemos observar, en este caso en el que tenemos menos datos hemos obtenido mejores resultados reduciendo el número de hojas y el rango de aprendizaje. Por otro lado, hemos aumentado el número de iteraciones manteniendo el parámetro *early_stopping_round* en 50 épocas. Por último, hemos reducido el factor de *reg_lambda* a 2, ya que tenemos menos hojas y se corre menor riesgo de sobreajuste.

A continuación, hemos entrenado los modelos con los datos de entrada reducidos y sin reducir con los parámetros que mejor resultados nos han dado.

Modelo	ECM
Sin reducir la dimensionalidad	0.030
Reduciendo la dimensionalidad	0.068

Tabla 21. Resultados de los modelos *LightGBM* para un edificio

Como podemos observar, en este caso perdemos 0.038 de precisión.

3.4.5 Comparación de los resultados

A continuación, podemos ver una tabla resumen con todos los resultados:

Modelo	Edificios	PCA	ECM	Memoria (MB)
ANN	Todos	No	0.202	3860
ANN	Todos	Sí	0.242	3216
ANN	Uno	No	0.080	0.983
ANN	Uno	Sí	0.078	0.772
LightGBM	Todos	No	0.042	3860
LightGBM	Todos	Sí	0.456	3216
LightGBM	Uno	No	0.030	0.983

LightGBM	Uno	Sí	0.068	0.772

Tabla 22. Resumen de resultados

Como podemos ver en la tabla resumen, el modelo que mejores predicciones da es el modelo *LightGBM* para un único edificio sin aplicar PCA.

Por otro lado, observamos una gran bajada de la precisión utilizando la reducción de la dimensionalidad mediante PCA. Esto se puede deber a:

- PCA no tiene en cuenta la variable objetivo, si no que tiene en cuenta la variabilidad de los atributos, por lo tanto, en función del conjunto de datos puede estar eliminado un atributo que esté disminuyendo la precisión del modelo.

- Por otro lado, hemos comprobado que, aunque proyectemos nuestros datos mediante PCA sin eliminar ninguna columna, también se disminuye drásticamente la precisión en el modelo de *LightGBM*. Esto puede deberse a que los árboles de decisión son más sensibles a la rotación de los datos.

En cuanto a si merece la pena aplicar PCA o no depende del modelo que escojamos:
LightGBM no parece muy recomendable, ya que su rendimiento con todos los datos es muy alto en cuanto a precisión, velocidad y carga de cómputo. Creo que no merece la pena sacrificar la precisión que te dan todos los datos por el ahorro en la memoria.

Por otro lado, con la ANN habría que evaluar más detenidamente.

En general, creo que en este caso aplicar PCA no ayuda tanto como debería. Por lo tanto, habría que barajar otras opciones para reducir la dimensionalidad como, por ejemplo:

- Análisis factorial (AF): que es una técnica similar al PCA, pero que permite la posibilidad de utilizar factores correlacionados.

- *Sparse regression:* que es un tipo de regresión que penaliza los coeficientes de las características que no son importantes para predecir la variable de respuesta. Esto puede ayudar a reducir la dimensionalidad de los datos eliminando las características que no son importantes para el modelo de regresión.

- Análisis de Componentes Independientes (ICA): Similar a PCA, pero busca componentes que sean estadísticamente independientes en lugar de maximizar la varianza. ICA es útil cuando se supone que las variables tienen fuentes subyacentes independientes.

Si tuviéramos que quedarnos con un modelo nos quedaríamos con el modelo de *LightGBM* por velocidad, facilidad de uso y precisión de los resultados.

Por otro lado, si tuviéramos que escoger con que datos entrenar al modelo, escogeríamos los datos de un solo edificio sin aplicar reducción de la dimensionalidad. Esto no quiere decir que la mejor opción sea hacer un estudio individual edificio por edificio. Que hayamos obtenido mejores resultados con un único edificio probablemente se deba al ruido en el conjunto de los datos.

4. Conclusiones

En este proyecto se ha realizado, por un lado, un estudio sobre las tecnologías existentes para predecir el consumo de la energía eléctrica y por otro, se han analizado y procesado los datos de consumo de múltiples edificios para implementar varios modelos predictivos.

A medida que hemos ido avanzando con el proyecto hemos ido reduciendo la cantidad de memoria mediante técnicas de preprocesado de los datos y hemos ido mejorando los resultados de los modelos variando los parámetros de ajuste de estos.

Cumplimiento de los objetivos

En lo que a los objetivos se refiere, se puede decir que este TFG ha cumplido tanto con los objetivos principales como con los objetivos secundarios, ya que:

- Se han obtenido los datos para ser analizados.

- Se han identificado las variables relevantes para este TFG mediante técnicas de preprocesamiento (apartado 3.3 Análisis de los datos).

- Se han estudiado las diferentes técnicas de IA existentes para el análisis del consumo de energía eléctrica de los edificios. Concretamente en el apartado 2 Estado del arte, en el apartado 3.1 Introducción y en el apartado 3.4 Modelos de aprendizaje.

- Se han desarrollado varios modelos de aprendizaje (apartado 3.4 Modelos de aprendizaje).

- Se han evaluado y comparado los resultados (apartado 3.4.5 Comparación de los resultados.

Desviaciones en la planificación

Por el contrario, cabe destacar que cumplir con los objetivos y respetar los plazos de las entregas no ha sido fácil. A parte de la falta de tiempo al compatibilizar la vida profesional y personal con la universidad, algunos de los contratiempos y cambios en la planificación han sido:

- **Búsqueda de los datos con los que realizar los modelos**.
  Ha sido muy difícil encontrar unos datos con los que realizar este Trabajo Final de Grado. Los datos con los que comencé a realizar los trabajos de análisis en la PEC2 se tuvieron que descartar durante la PEC3 por la poca cantidad de observaciones y por la simplicidad de estos. Después de realizar una búsqueda durante varios días encontré estos datos que creo que han cumplido bastante bien con lo que inicialmente tenía pensado.

A este respecto, me gustaría decir que me ha sorprendido lo difícil que es encontrar datos abiertos sobre el consumo de energía eléctrica en edificios. Por un lado, encontré muy pocas publicaciones científicas que publicaran los datos de sus estudios (solo los resultados). Por otro lado, también me sorprendió que ningún edificio público publique estos datos. Creo que es muy importante que se compartan (aunque sea de forma anónima), ya que puede ayudar a acelerar el desarrollo, mejorar las predicciones y poner a prueba los modelos predictivos por parte de la comunidad.

- **Capacidad de memoria y de cómputo de Google Colab**
  Al tratarse de una cantidad muy elevada de datos la licencia gratuita de Google Colab se quedó corta y era bastante frecuente que la conexión se reiniciar en mitad de un entrenamiento. Por lo tanto, la solución fue contratar una licencia de Google Colab Pro.

  Por otro lado, cabe señalar que los modelos que más carga de cómputo requerirán fueron las ANN con todo el conjunto de datos. Con los datos de un único edificio hubiera bastado con la licencia gratuita de Google Colab.

Aplicaciones

Predecir el consumo de energía pude tener las siguientes aplicaciones:

- Estudios para la colocación de paneles solares. Mediante la predicción del consumo de energía eléctrica se puede dimensionar tanto el número de paneles como la capacidad de las baterías de una instalación fotovoltaica.

- Ahorrar en la factura de la luz mediante modelos con los datos de consumo y los precios de la luz.

- Evaluación de las mejoras del aislamiento, de las ventanas etc. Comparando lo que debería consumir con lo que se predice que va a consumir.

Trabajos Futuros

Por la gran cantidad de modelos de IA que existen en la actualidad y lo rápido que evoluciona la tecnología limita mucho el trabajo que se puede realizar en un semestre a nivel de estudio los diferentes algoritmos, el entrenamiento y del ajuste de los modelos. Por lo tanto, como trabajos futuros sería interesante:

- Probar otros algoritmos para reducir la dimensionalidad que se ajustara mejor a este caso.

- Probar a entrenar otros modelos como, por ejemplo: Redes Convolucionales y bosques aleatorios y/o combinaciones de diferentes modelos.

- Por último, analizar si podemos mejorar la precisión de los modelos realizando un modelo distinto para cada zona climática

Reflexión final

Como reflexión final, me gustaría decir que predecir y comprender con precisión la eficiencia energética de los edificios es fundamental para reducir el uso total de energía, alcanzar los objetivos globales de emisiones de carbono, y para la toma de decisiones de los propietarios y/o comunidades de vecinos para la modernización y la mejora de los edificios.

Con este trabajo se demuestra que no es complejo crear un modelo que nos ayude a predecir el consumo de energía eléctrica. Es más, con el modelo LightGBM hemos visto que se pueden obtener buenos resultados muy rápidamente.

5. Glosario

IA: Inteligencia Artificial.

ML: *Machine Learning* (Aprendizaje Automático).

DL: *Deep Learning* (Aprendizaje Profundo).

ANN: *Artificial Neural Network* (Red Neuronal Artificial).

LigthGBM: *Light Gradient Boosting Machine.*

RAM: *Random Access Memory.*

PCA: *Principal Component Analysis (Ánalisis de Componentes Principales).*

ECM: Error Cuadrático Medio.

MSE: *Mean Squared Error* (Error Cuadrático Medio).

GPU: Graphical Processing Unit.

RAM: Random Access Memory.

GP: *Gaussian Process*

GMM: *Gaussian Mixture Model.*

SVM: *Support Vector Machine (*Máquinas de vectores de soporte)

Dataframe: Conjunto de datos.

Dataset: conjunto de datos.

Outlier: valor atípico en un conjunto de datos.

one-hot: sistema de codificación de variables discretas.

ICA: Análisis de Componentes Independientes.

AF: Análisis Factorial.

CO2: Dióxido de Carbono.

6. Referencias

AEMET. (2023). *AEMET*. Obtenido de https://www.aemet.es/documentos/es/eltiempo/prediccion/avisos/plan_meteoalerta/METEOALERTA_ANX1_Umbrales_y_niveles_de_aviso.pdf

Angeliki Xifara, A. T. (2012). *Accurate quantitative estimation of energy performance of residential buildings.*

ASHARAE. (2019). *Kaggle.* Obtenido de https://www.kaggle.com/competitions/ashrae-energy-prediction/data

Asmaa F. Hassan, S. B. (2022). *Towards a deep learning-based outlier*. Obtenido de https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00670-8

Asmaa F. Hassan, S. B. (2022). *Towards a deep learning-based outlier detection approach in the context of streaming data*. Obtenido de https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00670-8

AthanasiosTsanas, Angeliki Xifara. (2011). Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. 8.

Ciucci, M. (2020). *Parlamento Europeo*. Obtenido de https://www.europarl.europa.eu/factsheets/es/sheet/69/la-eficiencia-energetica (Consultado el 28/03/2023)

Ciucci, M. (2022). *Parlamento Europeo*. Obtenido de https://www.europarl.europa.eu/news/es/headlines/society/20221128STO58002/ahorro-de-energia-medidas-de-la-ue-para-reducir-el-consumo-energetico (Consultado el 28/03/2023)

Constructions, G. A. (2020). *2020 GLOBAL STATUS REPORT FOR BUILDINGS AND CONSTRUCTION*. Obtenido de https://wedocs.unep.org/bitstream/handle/20.500.11822/34572/GSR_ES.pdf

Diario Oficial de la Unión Europea. (19 de 05 de 2010). Obtenido de https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2010:153:0013:0035:es:PDF (Consultada el 28/03/2023)

Emad Elbeltagi, Hossam Wefki. (2021). *Predicting energy consumption for residential buildings using ANN through parametric modeling*. Obtenido de https://www.sciencedirect.com/science/article/pii/S2352484721002705

Fatima Yousaf, Sajid Iqbal, Nosheen Fatima, Tanzeela Kousar, Mohd Shafry Mohd Rahim. (2023). *Multi-class disease detection using deep learning and human brain medical imaging*. Obtenido de https://www.sciencedirect.com/science/article/pii/S1746809423003087

Lug, A. (2023). Obtenido de https://andrelug.com/es/acelera-tu-red-neuronal-con-relu/#:~:text=La%20Unidad%20Lineal%20Rectificada%20(ReLU)%20es%20una%20funci%C3%B3n%20de%20activaci%C3%B3n,aprendizaje%20de%20caracter%C3%ADsticas%20no%20supervisado

M., R. Q. (2001). *Indicadores de sostenibilidad ambiental y de desarrolllo sostenible:estado del arte u perspectivas* . Obtenido de https://repositorio.cepal.org/bitstream/handle/11362/5570/S0110817_es.pdf?sequence=1 (Consultado el 28/03/2023)

Maoran Sun, C. H. (2022). *Understanding building energy efficiency with administrative and emerging urban big data by deep learning in Glasgow*. Obtenido de https://www.sciencedirect.com/science/article/pii/S0378778822005023#f0015

Markus Rosenfelder, Moritz Wussow, Gunther Gust, Roger Cremades, Dirk Neumann. (2021). *Predicting residential electricity consumption using aerial and street view images*. Obtenido de https://www.sciencedirect.com/science/article/pii/S0306261921008047

Maryam Habibpour a, Hassan Gharoun b, Mohammadreza Mehdipour c, AmirReza Tajally d, Hamzeh Asgharnezhad f, Afshar Shamsi e, Abbas Khosravi e, Saeid Nahavandi e. (2023). *Uncertainty-aware credit card fraud detection using deep learning*. Obtenido de https://www.sciencedirect.com/science/article/pii/S0952197623004323

P. Kingma, D., & Lei Ba, J. (2017). *ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION*. Obtenido de https://arxiv.org/pdf/1412.6980.pdf

Plan de Recuperación, Transformación y Resiliencia. (2021). Obtenido de https://www.lamoncloa.gob.es/temas/fondos-recuperacion/Documents/30042021-Plan_Recuperacion_%20Transformacion_%20Resiliencia.pdf (Consultado el 28/03/2023)

Plotly Open Source Graphing Libraries. (2023). Obtenido de https://plotly.com/python/box-plots/#styling-outliers

Saleh Seyedzadeh, Farzad Pour Rahimian, Ivan Glesk and Marc Roper. (2018). *Machine learning for estimation of building energy consumption and performance: a review*. Obtenido de https://viejournal.springeropen.com/articles/10.1186/s40327-018-0064-7

Sons, J. W. (1994). *Signal and Image Processing with Neural Networks.*

Vergara Pla, G. (2015). *Métodos de aprendizaje máquina aplicados a la predicción del consumo eléctrico de edificios.* Obtenido de https://ruidera.uclm.es/xmlui/handle/10578/7163

Vicenç Torra i Reventós, D. M. (2007). *Apuntes de Aprendizaje Computacional.*

Wikipedia. (2023). Obtenido de https://es.wikipedia.org/wiki/Punto_de_roc%C3%ADo

World Meteorological Organization. (2023). *World Meteorological Organization*. Obtenido de https://wmo.asu.edu/content/world-meteorological-organization-global-weather-climate-extremes-archive

Xiaofei Zhang, Z. W. (2019). *A novel deep learning approach for building energy forecasting.*

Xifara, A. (s.f.). *Dataset*. Obtenido de https://archive.ics.uci.edu/ml/datasets/energy+efficiency#

7. Anexos

A continuación, se adjunta un link a una carpeta pública de Google Drive con todo el contenido de todo el proyecto.

https://drive.google.com/drive/folders/1IFC_EYfa6YAglPLDXstLDFdiEzpsUC5q?usp=sharing

Esta carpeta contiene el cuaderno Jupyter llamado agarciadelachica_TFG.ipynb con el código del pretratamiento de los datos y el entrenamiento de los diferentes modelos. Además, contiene las siguientes subcarpetas:

- **Data**: que contiene los datos originales y los datos después de realizar los procesos de pretratamiento.
- **Memoria**: que contiene una copia de esta memoria.
- **Presentación**: Con la presentación en formato PowerPoint y PDF junto al video de la presentación.

La carpeta se ha compartido de forma pública con permisos de solo lectura. Si se desea editar el cuaderno de Jupyter, la opción más sencilla es guardar una copia de la carpeta agarciadelachica_TFG en la carpeta raíz de una cuenta de Google Drive.