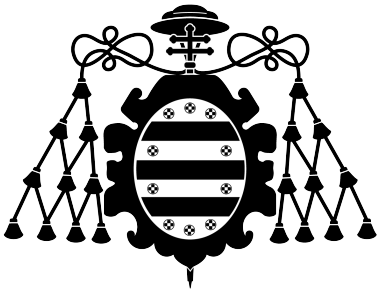

Universidad de Oviedo

Facultad de Ciencias

Trabajo Fin de Grado en Física

CÁLCULOS DE ESTRUCTURA ELECTRÓNICA CON TÉCNICAS DE MACHINE LEARNING

SUPERCONDUCTIVIDAD Y CLASIFICACIÓN BINARIA

Autora:

Olaya Folgueiras González

Tutores:

Víctor Manuel García Suárez

Juan Luis Fernández Martínez

14 de julio de 2024

Resumen

Este trabajo explora la posibilidad de predecir la superconductividad de materiales cristalinos utilizando técnicas de aprendizaje automático (machine learning), en específico, de clasificación binaria. La investigación se basa en la premisa de que la densidad de estados (DOS) de un material, una medida que describe la distribución de energías disponibles para los electrones, podría estar relacionada con su capacidad de superconducir. Para ello, se ha recopilado información de dos bases de datos: AFLOW, que contiene información de la DOS de un gran número de materiales, y 3DSC, que proporciona información sobre materiales superconductores.

El estudio se centra en analizar esta información para identificar patrones y relaciones entre la DOS y la superconductividad. Se implementan y evalúan diferentes modelos de aprendizaje automático y se exploran técnicas de remuestreo para abordar el desequilibrio en los datos. Además, se aplica el análisis de componentes principales (PCA) para reducir la dimensionalidad del conjunto de datos y mejorar el rendimiento del modelo.

Abstract

This work explores the possibility of predicting the superconductivity of crystalline materials using machine learning techniques, specifically binary classification. The research is based on the premise that the density of states (DOS) of a material, a measure that describes the distribution of energies available to electrons, could be related to its ability to superconduct. To this end, information has been collected from two databases: AFLOW, which contains DOS information on a large number of materials, and 3DSC, which provides information on superconducting materials.

The study focuses on analysing this information to identify patterns and relationships between DOS and superconductivity. Different machine learning models are implemented and evaluated and resampling techniques are explored to address the imbalance in the data. In addition, principal component analysis (PCA) is applied to reduce the dimensionality of the dataset and improve model performance.

Índice general

Capítulo 1

Fundamentos teóricos

En el mundo de la física de materiales, es crucial entender las propiedades de diferentes compuestos, tanto los que creamos en el laboratorio como los que encontramos en la naturaleza. La evolución de la tecnología y la electrónica ha creado la necesidad de encontrar materiales con propiedades como podría ser la superconductividad.

En esta sección, se expondrán los conocimientos indispensables en estructuras cristalinas, teoría de bandas y teoría de la superconductividad. Toda la información está basada en las referencias [1], [2], [3], [4] y [5], donde se encuentra una explicación más detallada de estos temas.

1.1. Estructuras Cristalinas

Entorno al siglo IV a.C, el filósofo griego Demócrito describió los átomos como partículas materiales pequeñas, inaccesibles e indivisibles, que conforman toda sustancia material. Y, estos átomos, diferenciados por su forma, posición y disposición, constituyen la base fundamental de la realidad observable. Esta idea atomista, si bien formulada en un contexto filosófico, hace eco en la cristalografía moderna, donde se estudian las estructuras cristalinas presentes en numerosos sólidos, y en las cuales la disposición de los átomos es fundamental para poder determinar sus propiedades.

Podemos definir las estructuras cristalinas como aquellas que se caracterizan por la disposición regular y periódica de átomos, iones o moléculas en un patrón tridimensional que se extiende a lo largo de todo el material. El patrón de distribución de estos elementos se denomina red de Bravais y la unidad más pequeña que, manteniendo su forma y contenido, se repite a lo largo del cristal es denominada celda unidad.

A modo de ejemplo, podríamos visualizar el cristal como una pared de ladrillos, donde la estructura cristalina sería la pared completa, la red de Bravais la disposición de las juntas entre los ladrillos y la celda unidad un solo ladrillo.

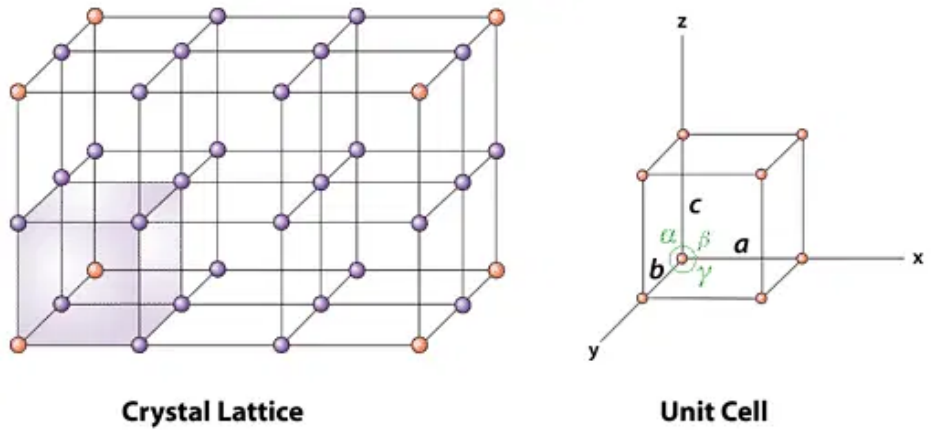

Figura 1.1: Esquema de una red cristalina y su correspondiente celda unidad. [20]

Si nos fijamos en la figura 1.1, observamos que la celda unidad puede describirse mediante tres vectores ($\vec{a}$, $\vec{b}$ y $\vec{c}$) que coinciden con las tres aristas del paralelogramo que intersecan con ángulos α, β y γ. Por lo tanto, podremos describir cualquier punto de la red tridimensional mediante un vector $\vec{r}$ dado por:

$$\vec{r} = u\vec{a} + v\vec{b} + w\vec{c} \quad u, v, w \in \mathbb{R} \tag{1.1}$$

1.1.1. Redes de Bravais

Las redes de Bravais han de ser invariantes bajo traslaciones a la hora de construir el cristal y, mediante teoría de grupos, se ha demostrado que solo existe 14 tipos de redes, clasificadas en 7 sistemas cristalinos (triclínico, monoclínico, ortorrómbico, tetragonal, rom-

boédrico, hexagonal y cúbico). A continuación, se explicará de forma breve cada una de ellas:

- **Triclínico**: Su simetría viene dada por $a \neq b \neq c$ y $\alpha \neq \beta \neq \gamma \neq 90°$. La única red asociada es la triclínica (TRI)
- **Monoclínico**: Una longitud de ejes es diferente y dos son iguales, uno de los ángulos no es de 90 grados: $a \neq b \neq c$, $\alpha = \gamma = 90°$, $\beta \neq 90°$. Las redes asociadas son monoclínica (MCL) y monoclínica centrada (MCLC)
- **Ortorrómbico**: Las longitudes de los tres ejes son diferentes y los ángulos entre ellos son de 90 grados: $a \neq b \neq c$, $\alpha = \beta = \gamma = 90°$. Las redes asociadas son Ortorrómbica simple (ORC), Ortorrómbica centrada en el cuerpo (ORCC), Ortorrómbica centrada en las caras (ORCI), Ortorrómbica centrada en las aristas (ORCF).
- **Tetragonal**: Su simetría viene dada por: $a = b \neq c$, $\alpha = \beta = \gamma = 90°$. Las redes asociadas son Tetragonal simple (TET), Tetragonal centrada en el cuerpo (BCT).
- **Romboédrico**: Su simetría viene dada por: $a = b \neq c$, $\alpha = \beta = 90°$, $\gamma = 120°$. Su red asociada es la Rómbica Hexagonal (RHL).
- **Hexagonal**: Conformada por tres ejes de igual longitud con ángulos de 120 grados, y un eje perpendicular de diferente longitud: $a = b \neq c$, $\alpha = \beta = 90°$, $\gamma = 120°$. Su red asociada es la Hexagonal (HEX).
- **Cúbico**: Todas las longitudes de los ejes son iguales y los ángulos entre ellos son de 90 grados: $a = b = c$, $\alpha = \beta = \gamma = 90°$. Las redes asociadas son: Cúbica simple (CUB), Cúbica centrada en el cuerpo (BCC), Cúbica centrada en las caras (FCC).

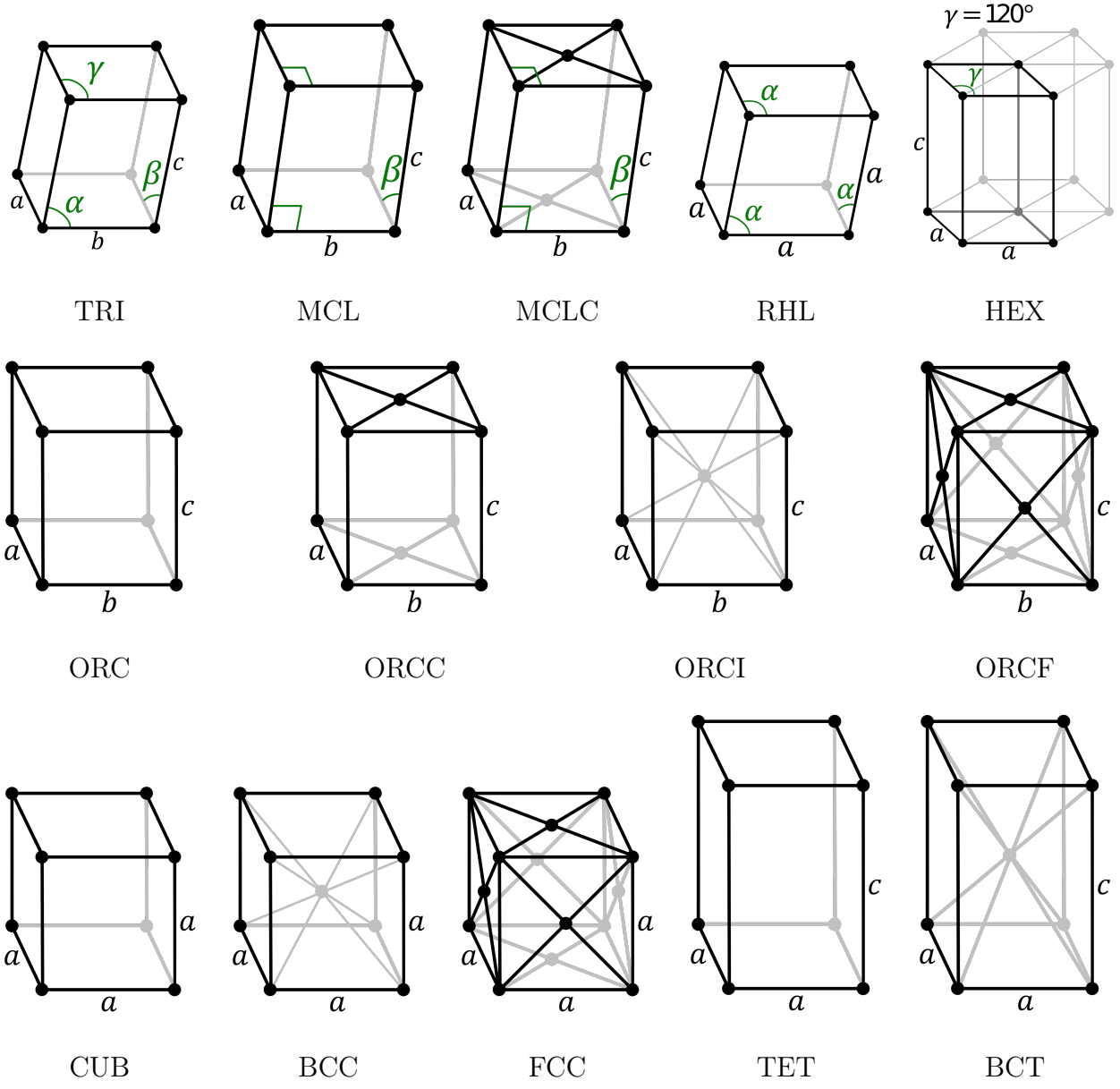

Figura 1.2: Esquemas de las catorce redes de Bravais. [18]

1.1.2. Red recíproca

En el apartado previo se ha descrito la estructura de los cristales y sus elementos en el espacio, pero, ¿cómo afecta esto a las propiedades del material?. Para poder explicarlo hemos de hacer referencia a la red recíproca.

En el contexto de la física del estado sólido, la red recíproca es un espacio matemático que

sirve como una representación alternativa del espacio real en el que se encuentra un cristal. Mientras que la red cristalina describe la disposición de los átomos en el espacio real, la red recíproca describe las longitudes de onda y direcciones de las ondas que pueden propagarse a través del cristal.

Esta se obtiene mediante la transformada de Fourier de la red cristalina. Si consideramos una función periódica $f(\vec{r})$ que describe la disposición de átomos en un cristal, su transformada de Fourier $F(\vec{k})$ nos da la amplitud de las ondas de longitud de onda $\vec{k}$ presentes en la red recíproca. La relación está dada por la fórmula:

$$F(\vec{k}) = \int f(\vec{r}) e^{-i\vec{k}\cdot\vec{r}} d\vec{r} \tag{1.2}$$

donde $\vec{r}$ es el vector de posición en el espacio real que habíamos definido en (1.1), $\vec{k}$ es el vector de onda en la red recíproca e i es la unidad imaginaria, extendiéndose la integral sobre todo el espacio.

La red recíproca se representa comúnmente utilizando la llamada primera zona de Brillouin, que es una celda unitaria en el espacio de la red recíproca que contiene toda la información relevante sobre las propiedades electrónicas de un cristal. Está definida como el conjunto de todos los vectores de onda $\vec{k}$ que cumplen con la condición:

$$-\frac{\pi}{a} < k_i \leq \frac{\pi}{a} \tag{1.3}$$

Donde a es la longitud de la celda unitaria de la red cristalina en la dirección i.

Esta zona es fundamental para comprender la dispersión de las ondas en un cristal y se utiliza ampliamente en la teoría de bandas, que veremos posteriormente, para describir las propiedades electrónicas de los sólidos cristalinos.

1.2. Teoría de Bandas y propiedades de los sólidos cristalinos

En un material cristalino, la estructura de bandas electrónicas puede describirse mediante la teoría de bandas. Según esta teoría, los electrones en un sólido ocupan niveles de energía

discretos que forman bandas continuas. Esto hace que sea una herramienta fundamental en la física del estado sólido para comprender las propiedades electrónicas de los materiales.

1.2.1. Estructura de Bandas

La teoría de bandas considera que los electrones se comportan como ondas en un potencial periódico generado por la red cristalina. Los electrones en un cristal se describen mediante la ecuación de Schrödinger:

$$\hat{H}\psi(\vec{r}) = E\psi(\vec{r}) \tag{1.4}$$

donde $\hat{H}$ es el operador Hamiltoniano, $\psi(\vec{r})$ es la función de onda del electrón y E es la energía del electrón. Para un cristal, la función de onda $\psi(\vec{r})$ se puede expresar en términos de funciones de onda periódicas $\psi_{\vec{k}}(\vec{r})$, que dependen del vector de onda $\vec{k}$ en el espacio recíproco. Esto conduce a la ecuación de Bloch:

$$\psi_{\vec{k}}(\vec{r}) = e^{i\vec{k}\cdot\vec{r}} u_{\vec{k}}(\vec{r}) \tag{1.5}$$

donde $u_{\vec{k}}(\vec{r})$ es una función periódica con el mismo periodo que la red cristalina. La solución de la ecuación de Schrödinger para un electrón en un cristal se reduce entonces a encontrar las funciones de onda permitidas $u_{\vec{k}}(\vec{r})$ y los correspondientes valores de energía $E_{\vec{k}}$.

La solución de esta ecuación de Schrödinger conduce a la formación de lo que se conoce como estructura de bandas en un cristal. En esta estructura, las energías permitidas para los electrones se agrupan en regiones continuas llamadas bandas de energía, separadas por regiones prohibidas de energía. Dentro de cada banda, los electrones pueden ocupar una variedad de estados de energía, como vemos en la figura 1.3 habría una serie de bandas llenas u ocupadas y otras vacías, cuya separación viene dada por una energía E_F. En el estado fundamental del sistema a temperatura cero absoluto, se denomina energía de Fermi (E_F) a la energía más alta ocupada por los electrones.

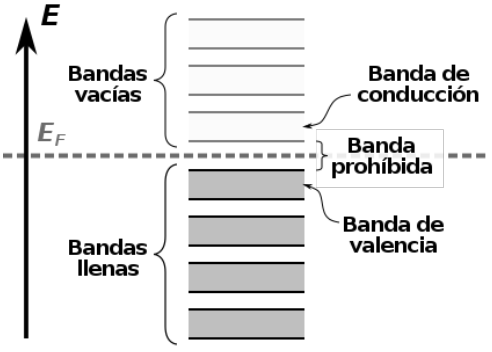

Figura 1.3: Esquema de la estructura de bandas. [24]

Dentro de esta estructura de bandas es importante destacar dos bandas específicas: la banda de valencia y la banda de conducción. La banda de valencia es la banda de energía más alta ocupada por electrones en el estado fundamental del sistema. La siguiente banda observada sería la banda de conducción, que se puede definir como la banda de energía donde los electrones pueden percibir perturbaciones por parte de campos eléctricos externos y, por tanto, se pueden generar corrientes eléctricas. La brecha de energía entre la banda de valencia y la banda de conducción se conoce como la banda prohibida.

1.2.2. Densidad de estados

Uno de los conceptos fundamentales a la hora de aplicar esta teoría es la densidad de estados (DOS), ya que proporciona información sobre la distribución de energía de los estados cuánticos disponibles para los electrones. Se puede definir como la medida de la cantidad de estados permitidos por unidad de energía en un material. En un sistema tridimensional, se define como:

$$D(E) = \frac{dN}{dE} \tag{1.6}$$

donde $D(E)$ es la densidad de estados, dN es el número de estados en un rango de energía dE.

Al representar la DOS frente a la energía, podemos identificar las regiones donde hay una alta concentración de estados, que corresponden a las bandas de energía, así como las regiones donde no hay estados disponibles. En la figura 1.4, observamos una de estas representaciones, donde la zona roja se correspondería a la banda de valencia y la zona verde a las bandas de conducción. Por otra parte, el espacio disponible entre ambas se corresponde a la banda prohibida dada por la energía de Fermi.

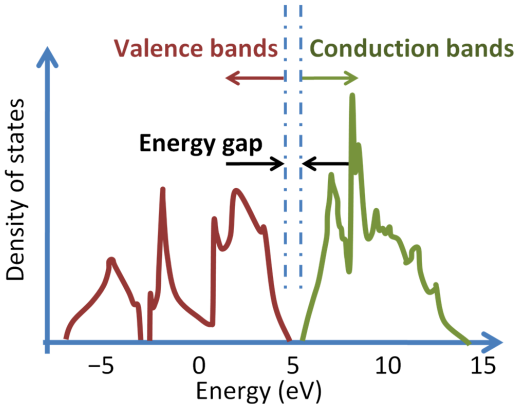

Figura 1.4: Representación de la DOS de un material frente a la energía. [26]

1.2.3. Banda prohibida y conducción

La posición de la banda de conducción (E_C) y la banda de valencia (E_V) en relación con la energía de Fermi (E_F) determina las propiedades eléctricas de un material. La diferencia de energía entre la banda de conducción y la banda de valencia es lo que determina la banda prohibida (E_g). La conductividad eléctrica de un material está estrechamente relacionada con la superposición de la banda de conducción y la banda de valencia con la energía de Fermi, permitiendo definir los materiales conductores, semiconductores y aislantes.

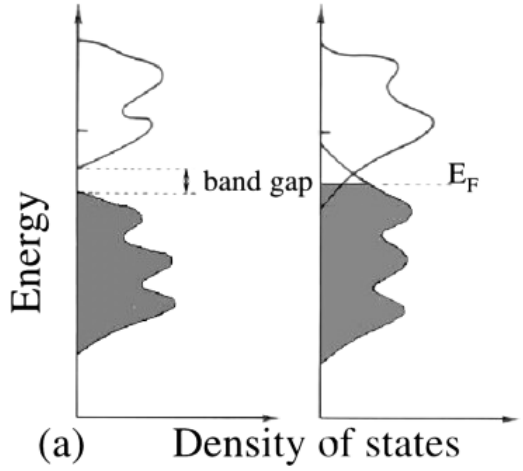

Figura 1.5: A la izquierda la DOS de un aislante, a la derecha la de un conductor.

En los conductores, la banda de conducción se superpone con E_F, lo que permite que los electrones se muevan fácilmente en respuesta a un campo eléctrico externo, resultando en una alta conductividad eléctrica.

$$E_C > E_F \tag{1.7}$$

En los aislantes, la banda de conducción está separada de E_F por una brecha de energía significativa, lo que impide que los electrones se muevan libremente y conlleva una baja conductividad eléctrica.

$$E_C - E_F > E_g \tag{1.8}$$

En los semiconductores, la brecha de energía es lo suficientemente pequeña como para que algunos electrones puedan saltar a la banda de conducción en temperaturas moderadas, permitiendo una conductividad que puede controlarse mediante la temperatura. La conductividad en estos materailes se describe por la ecuación de Arrhenius:

$$\sigma = \sigma_0 \exp\left(-\frac{E_g}{2k_B T}\right) \tag{1.9}$$

donde σ es la conductividad, σ_0 es una constante, k_B es la constante de Boltzmann, T es la temperatura absoluta y E_g es la banda prohibida. Esta ecuación muestra cómo la

conductividad en semiconductores aumenta exponencialmente con la temperatura debido a la excitación de electrones a la banda de conducción.

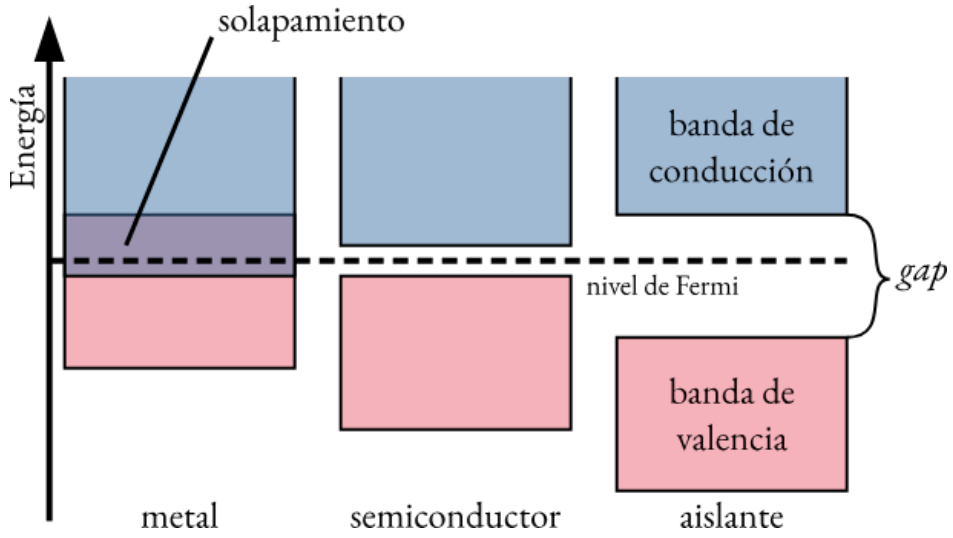

Figura 1.6: Esquema de separación de las bandas y la energía de Fermi en cada tipo de material.

1.3. Superconductividad

La superconductividad es un fenómeno que se caracteriza por la pérdida total de resistencia eléctrica en ciertos materiales a temperaturas bajas. Esta propiedad fue descubierta en 1911 por Heike Kamerlingh Onnes, abriendo el camino al desarrollo de varias teorías durante el siglo XX que tratarían de explicar este fenómeno. Pero antes de entrar en la teoría, vamos a exponer algunas de las propiedades definitorias de los superconductores. Cabe mencionar que en el desarrollo de este trabajo nos centraremos en la superconductividad a bajas temperaturas.

1.3.1. Propiedades de los materiales superconductores

La principal característica de estos materiales es la pérdida de total de resistencia eléctrica por debajo de la temperatura crítica (T_C). La resistividad eléctrica (ρ) de un material conductor está relacionada con la velocidad de deriva (v_d) de los electrones y la densidad de

carga (n) mediante la relación:

$$\rho = \frac{m_e v_d}{ne^2} \tag{1.10}$$

donde m_e es la masa del electrón y e es la carga elemental. En un conductor convencional, la colisión de los electrones con los iones de la red cristalina genera una resistencia eléctrica. En un superconductor, la formación de pares de Cooper, que se explicarán más adelante, reduce la resistividad eléctrica a cero, teniendo entonces $\rho_s = 0$.

Por otra parte, el efecto Meissner, descubierto por Walther Meissner y Robert Ochsenfeld en 1933, es una característica fundamental de los superconductores. Implica la expulsión total del campo magnético externo del interior de un superconductor cuando se enfría por debajo de T_C. En contraste con los conductores normales, donde las corrientes inducidas por un campo magnético externo se desvanecen al retirar este último, en un superconductor, la corriente inducida persiste incluso sin la presencia del campo magnético. Esta corriente persistente, que no se disipa debido a la nula resistencia eléctrica, genera un campo magnético que cancela completamente el campo magnético aplicado dentro del superconductor. Este comportamiento define la diamagnética perfecta, es decir, la expulsión total del campo magnético del interior del superconductor.

Para comprender el efecto Meissner, podemos considerar la ecuación de Maxwell para el campo magnético:

$$\nabla \times B = \mu_0 J \tag{1.11}$$

donde B es el campo magnético, J es la densidad de corriente y μ_0 es la permeabilidad magnética del vacío. En términos de esta ecuación, el efecto Meissner se traduce en que la densidad de corriente J en un superconductor se mantiene incluso cuando el campo magnético aplicado B se elimina.

Por último, el efecto túnel, descubierto por Leo Esaki y Robert Schrieffer en 1957, permite el paso de corriente eléctrica a través de una barrera aislante entre dos superconductores sin necesidad de que los electrones tengan mayor energía cinética que la diferencia de potencial de la barrera. Esto es debido a que los pares de Cooper no tienen masa efectiva, por lo que pueden moverse libremente a través del material sin disipar energía.

1.3.2. Superconductores tipo I y II

Los superconductores pueden ser clasificados en función de respuesta a un campo magnético externo en dos tipos: I y II. Para comprender esta diferencia, es útil analizar la magnetización (M) de un superconductor en función del campo magnético aplicado (H). En la figura 1.7, se muestra de forma gráfica la magnetización de los dos tipos de superconductores en función del campo magnético.

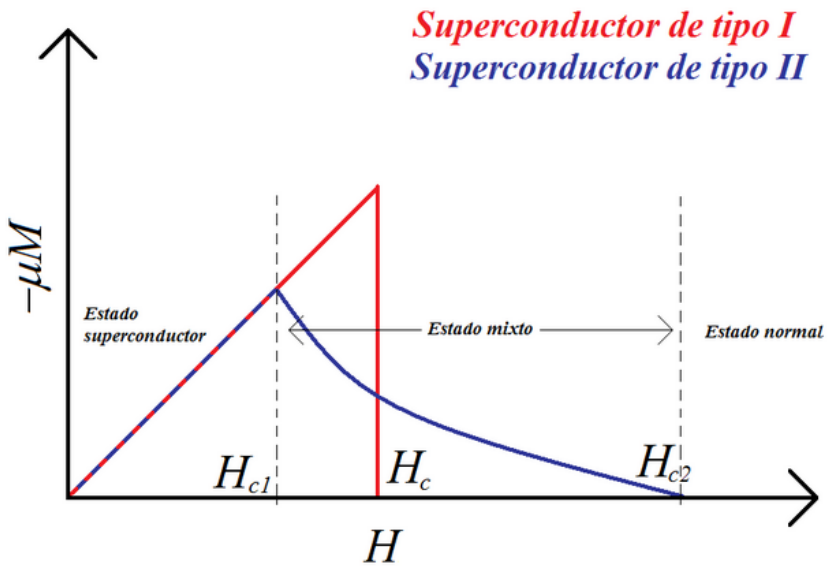

Figura 1.7: Magnetización de un superconductor tipo I y de un superconductor tipo II como función del campo aplicado. [19]

En un superconductor tipo I, el material permanece en estado superconductor y expulsa completamente el campo magnético (efecto Meissner) hasta que se alcanza un valor crítico de campo (H_c). En este punto, el material experimenta una transición abrupta a un estado normal, donde permite la penetración completa del campo magnético. Esto se representa en la figura por la línea roja, donde la magnetización permanece constante hasta H_c y luego cae bruscamente a cero.

En un superconductor tipo II, el comportamiento es diferente. Existe un primer valor crítico de campo (H_{c1}) por debajo del cual el material expulsa completamente el campo

magnético y se comporta como un superconductor perfecto. Sin embargo, al aumentar el campo magnético aplicado por encima de H_{c1}, el material entra en un estado mixto donde el campo magnético comienza a penetrar el material. Este comportamiento se representa en la figura por la línea azul, la cual presenta una gradual disminución de la magnetización a partir de H_{c1}.

En este estado mixto, el campo magnético se canaliza a través de "vórtices"que atraviesan el material. Estos vórtices son regiones donde los electrones se emparejan débilmente. A medida que aumenta el campo magnético aplicado, la densidad de vórtices aumenta. Finalmente, se alcanza un segundo valor crítico de campo (H_{c2}) por encima del cual el material se encuentra en estado normal y permite la penetración completa del campo magnético.

1.3.3. Teoría BCS

Como se mencionaba anteriormente, las propiedades de los superconductores se pueden explicar a través de la formación de pares de Cooper. La teoría BCS, desarrollada por John Bardeen, Leon Cooper y Robert Schrieffer en 1957 [3], se basa en ellos para explicar la superconductividad y, aunque presenta algunas limitaciones, como la dificultad para explicar este fenómeno a altas temperaturas, sigue sirviendo como marco teórico.

Un par de Cooper es un estado ligado de dos electrones que surge a temperaturas muy bajas y donde los electrones se comportan como si se atrayeran a pesar de tener cargas de mismo signo. Están compuestos por dos electrones con momentos iguales y opuestos y espín total del par nulo. Cuando un electrón se desplaza a través del cristal, perturba la red cristalina, causando vibraciones que se propagan a través de ésta. Estas vibraciones se denominan fonones y, en la descripción en el espacio de momento dada por la red recíproca, se visualizan como modos normales de vibración. Estos fonones interactúan con otros electrones, llevando a la formación de pares de Cooper que son capaces de moverse a través del material sin resistencia y causando las propiedades descritas en el apartado anterior.

Al considerar un sistema con múltiples parejas de electrones, se observa que el emparejamiento da lugar a una banda prohibida de energía. Si observamos la figura 1.8 se muestra

como en el cero absoluto existe una zona prohibida alrededor de la energía de Fermi, similar al comportamiento de los semiconductores. Sin embargo, la existencia de los pares de Cooper en los superconductores y la no existencia de resistividad eléctrica permite a los electrones moverse libremente, al contrario de lo que ocurre en los semiconductores.

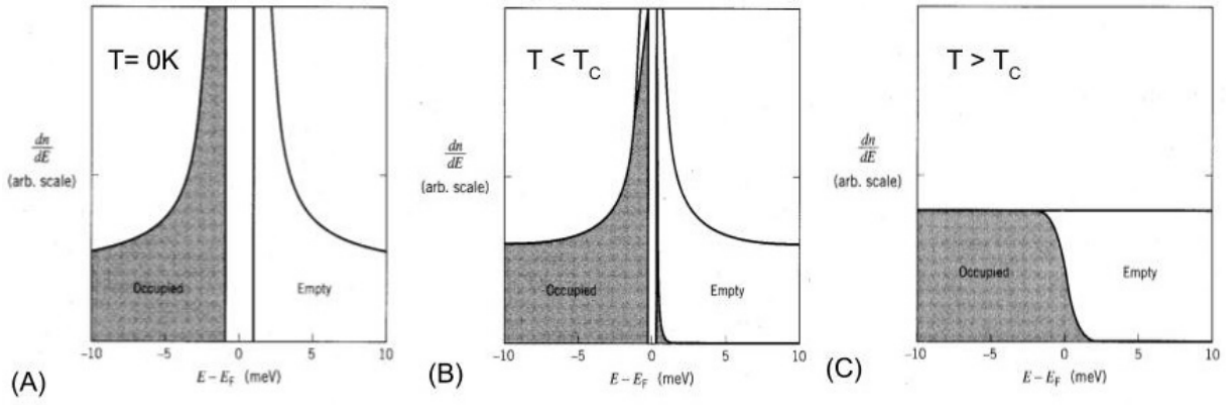

Figura 1.8: Evolución de la densidad de estados de un superconductor con la temperatura. [17]

Mediante el desarrollo matemático de la teoría, es posible llegar a una expresión que describe la banda prohibida de los superconductores a temperatura 0K. Esta expresión es:

$$\Delta = 2\hbar\omega_D \exp(-1/V_0 N(0)) \tag{1.12}$$

donde donde Δ es la banda prohibida, ω_D es la frecuencia de Debye y $V_0 N(0)$ es constante y depende del material.

1.3.4. Superconducción a altas temperatura

La teoría BCS, aunque explica de forma satisfactoria la superconductividad a bajas temperaturas, no logra explicar la superconductividad a altas temperaturas, es decir, por encima de la temperatura de ebullición del nitrógeno líquido (77K). Los primeros superconductores de alta temperatura, descubiertos en 1986, fueron los compuestos de óxidos de cobre, como el *LaBaCuO*. Estos materiales exhiben temperaturas críticas superiores a 100K, lo que les permite operar en condiciones más accesibles. Sin embargo, la interacción fonónica no es suficientemente fuerte a esas temperaturas.

Actualmente, no existe una teoría definitiva que explique la superconductividad a altas temperaturas. Aunque se han propuesto diversas, ninguna ha logrado una aceptación universal. Entre las teorías más relevantes se encuentran la teoría de la resonancia magnética (RVB) y el Modelo de Hubbard.

La teoría RVB, desarrollada por Philip Anderson, propone que la superconductividad a altas temperaturas está relacionada con la resonancia entre los estados de enlace de valencia de los electrones. En este modelo, los electrones no forman pares de Cooper convencionales, sino que se comportan como una colección de espines que interactúan entre sí en una red. Los espines forman pares singletes de corta vida que resonan entre diferentes configuraciones de enlaces, lo que da lugar a la superconductividad.

El Modelo de Hubbard, por otro lado, se centra en la interacción entre electrones en materiales con fuertes correlaciones electrónicas. Este modelo describe la interacción entre los electrones en la red cristalina, considerando principalmente la repulsión de Coulomb entre electrones en el mismo sitio y el movimiento de electrones entre sitios adyacentes. Aunque el modelo de Hubbard no incluye explícitamente la interacción electrón-fonón, se puede extender para considerar otros efectos como las fluctuaciones de espín y las correlaciones electrónicas complejas, que son cruciales para entender la superconductividad a altas temperaturas.

Capítulo 2

Machine Learning

En los últimos años, la tecnología ha provocado una avalancha de datos que recibimos y recogemos a diario en todos los ámbitos. De este incremento surge la necesidad de desarrollar nuevas herramientas que nos permitan tratar y obtener información de esas fuentes de forma precisa y eficiente, dando de esta forma lugar a la ciencia de datos. Aunque pueda llegar a parecernos algo externo a nosotros, la inteligencia artificial inunda nuestro día a día, la utilizamos cuando queremos llegar a algún sitio y buscamos cuál es la ruta más óptima o cuando entramos en una plataforma de *streaming* para ver la serie que nos recomienda.

Las ventajas de la inteligencia artificial no se limitan únicamente a la tecnología, si no que también juegan un papel fundamental en la ciencia. En el estudio de estructuras cristalinas en la Física del Estado Sólido, por ejemplo, puede servirnos como guía para predecir las características de un material antes de probarlo en el laboratorio, que es lo que trataremos de desarrollar en secciones posteriores. A lo largo de esta sección se desarrollarán y expondrán los principios fundamentales del Machine Learning, desde su definición hasta sus aplicaciones más inmediatas.

Las principales fuentes empleadas en el desarrollo de este capítulo son [6], [7] y [8].

2.1. Inteligencia Artificial, Machine Learning y Ciencia de Datos

En muchas ocasiones observamos como los términos de Inteligencia Artificial (IA), Machine Learning (ML) y Ciencia de Datos se entremezclan, generando confusión sobre sus significados y aplicaciones. Para entender su significado y diferenciarlas es importante tener clara la definición de cada una.

En primer lugar, la Inteligencia Artificial es un campo dentro de la informática que busca replicar la capacidad humana en máquinas, permitiendo a los sistema informáticos razonar, aprender, percibir y tomar decisiones como lo haría un humano. A su vez se puede diferenciar en dos categorías, la IA estrecha y la IA general. La IA estrecha, débil o específica se centra en desarrollar tareas específicas con limitaciones, como podía ser un sistema de recomendación. Por otra parte, la IA general trata de igualar la inteligencia humana con todas sus capacidades.

Por su parte, el Machine Learning, o aprendizaje Automático, es una disciplina dentro de la IA que se centra en analizar datos mediante algoritmos y métodos estadísticos, de tal forma que son capaces de reconocer patrones y buscar relaciones que permitan realizar predicciones, estimar valores o tomar decisiones.

Por último, la Ciencia de Datos es un campo interdisciplinario que se centra en combinar la estadística y la informática para extraer información de los conjuntos de datos. Este campo implica el conocimiento del área de los datos y la limpieza, visualización y procesamiento de estos para poder aplicar ML y extraer la información.

En resumen, la Inteligencia Artificial es el campo general que abarca la creación de sistemas inteligentes, el Machine Learning es una técnica específica dentro de la IA que se centra en el aprendizaje a partir de datos, y la Ciencia de Datos es una disciplina que se ocupa del análisis y la interpretación de datos para obtener información.

2.2. Fundamentos del Machine Learning

Es importante destacar que el Machine Learning se basa en el principio de aprendizaje a partir de ejemplos y experiencias pasadas. En lugar de ser programados explícitamente para llevar a cabo una tarea específica, los algoritmos de ML son diseñados para aprender a partir de los datos disponibles y ajustar su rendimiento a medida que se exponen a más información.

Uno de los conceptos fundamentales en el ML es el de la *generalización*, que se refiere a la capacidad de un modelo entrenado para realizar predicciones precisas sobre datos no vistos previamente. En otras palabras, un modelo de ML exitoso debe ser capaz de capturar patrones subyacentes en los datos de entrenamiento y aplicarlos de manera efectiva a nuevos datos para hacer predicciones precisas.

Dentro del campo del ML, se distinguen varios tipos de problemas y tareas principales, entre los cuales destacan la regresión, la clasificación, el agrupamiento y el aprendizaje por refuerzo. Cada uno de estos tipos de problemas requiere enfoques y algoritmos específicos para su resolución que trataremos en la sección posterior.

2.2.1. Desarrollo de un modelo de Machine Learning

El desarrollo de un modelo de Machine Learning sigue un proceso general que implica varias etapas clave:

1. **Recopilación de Datos:** El primer paso consiste en recopilar y preparar los datos necesarios para entrenar el modelo. Esto puede implicar la recopilación de datos históricos relevantes y la limpieza de datos para eliminar valores atípicos o faltantes.
2. **Visualización de Datos y Análisis Exploratorio (EDA):** Es fundamental comprender la estructura y las características de los datos mediante técnicas de visualización y análisis exploratorio. Esto puede incluir gráficos de dispersión, histogramas,

matrices de correlación y otros métodos para identificar patrones y relaciones en los datos.

3. **División de los Datos:** Una vez que se recopilan los datos, es importante dividirlos en conjuntos de entrenamiento y test. El conjunto de entrenamiento se utiliza para entrenar el modelo y el conjunto de test se utiliza para evaluar el rendimiento del modelo.

4. **Selección del Modelo:** En esta etapa, se selecciona el tipo de modelo de Machine Learning que se utilizará para resolver el problema en cuestión. Esto puede incluir modelos de regresión, clasificación, agrupamiento u otros, dependiendo de la naturaleza de los datos y el tipo de tarea que se esté abordando.

5. **Feature Engineering:** El proceso de ingeniería de características implica la creación y selección de nuevas características a partir de las existentes en los datos. Esto puede incluir la transformación de variables, la normalización de datos y otras técnicas para mejorar el rendimiento del modelo.

6. **Entrenamiento del Modelo:** Una vez que se selecciona el modelo y se lleva acabo el *Feature Engineering*, se procede a entrenarlo utilizando el conjunto de datos de entrenamiento. Durante el entrenamiento, el modelo ajusta sus parámetros para minimizar una función de pérdida o error, que mide la discrepancia entre las predicciones del modelo y los valores reales en el conjunto de entrenamiento.

   Este ajuste de parámetros puede llevar a que el modelo se ajuste demasiado o no se ajuste en absoluto a los datos de entrenamiento, esto es lo que se conoce como *overfitting* y *underfitting*. El ***overfitting*** ocurre cuando el modelo se ajusta demasiado bien al conjunto de entrenamiento, capturando el ruido y las fluctuaciones aleatorias en los datos en lugar de aprender la relación subyacente. Esto puede conducir a un rendimiento deficiente en datos no vistos, ya que el modelo no generaliza bien. Para mitigar el overfitting, se pueden aplicar técnicas como la regularización o el uso de más datos de entrenamiento. Por otro lado, el ***underfitting*** ocurre cuando el modelo es demasiado simple para capturar la complejidad de los datos subyacentes. Esto se manifiesta en un rendimiento deficiente tanto en el conjunto de entrenamiento como en

datos no vistos. Para abordar el underfitting, se pueden probar modelos más complejos, ajustar los hiperparámetros del modelo o utilizar características más informativas.

7. **Validación del Modelo:** Durante y después del entrenamiento, es importante evaluar el rendimiento del modelo para garantizar que esté generalizando bien. Esto implica calcular métricas y aplicar técnicas en las que entraremos más a fondo en una sección posterior.

8. **Ajuste de Hiperparámetros:** En algunas ocasiones, es necesario ajustar los hiperparámetros del modelo para mejorar su rendimiento. Esto puede implicar la selección de diferentes valores de hiperparámetros, como la tasa de aprendizaje en un modelo de redes neuronales, o la profundidad máxima en un árbol de decisión.

Una vez que se han seguido todos los pasos previos, se puede utilizar el modelo para realizar predicciones sobre nuevos datos. Durante el proceso de predicción, se proporciona al modelo un conjunto de características de entrada y este genera una predicción o estimación sobre el resultado. Hay que tener en cuenta que estas predicciones no son siempre correctas y deben ser interpretadas y utilizadas de manera adecuada, teniendo en cuenta sus limitaciones y el contexto.

Además, si el modelo recibe nuevos datos de forma periódica, es importante realizar un seguimiento del rendimiento, evaluándolo con nuevos datos. De esta forma se podrán hacer reajustes para garantizar su precisión y continuidad futura.

2.3. Tipos de algoritmos

Los algoritmos se pueden clasificar en diferentes categorías según el tipo de aprendizaje que emplean y la naturaleza de los datos con los que trabajan. A continuación, se presentan los principales tipos de algoritmos utilizados en Machine Learning.

2.3.1. Aprendizaje supervisado

El aprendizaje supervisado se basa en datos etiquetados, donde cada dato de entrada tiene una etiqueta que indica su salida esperada. Estas etiquetas son fundamentales para entrenar modelos predictivos, ya que permiten al algoritmo aprender la relación entre las características de entrada y las salidas deseadas. Algunos de los algoritmos más empleados en este tipo de aprendizaje incluyen:

- **Regresión Lineal:** Este algoritmo busca encontrar la mejor línea de ajuste para predecir una variable continua a partir de una o más variables independientes. En un contexto matemático, se basa en minimizar la suma de los cuadrados de las diferencias entre las predicciones del modelo y los valores reales, utilizando el método de mínimos cuadrados.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \varepsilon$$

  donde y es la variable dependiente, x_i son las variables independientes, β_i son los coeficientes del modelo y ε representa el error aleatorio.

- **Máquinas de Vectores de Soporte para clasificación (SVM):** SVM es un algoritmo de aprendizaje supervisado utilizado para clasificar datos en diferentes categorías. Su objetivo es encontrar el hiperplano que mejor separa las clases en un espacio de características. El fundamento matemático detrás de SVM implica maximizar la distancia entre el hiperplano y los puntos de datos más cercanos de cada clase, conocidos como vectores de soporte.

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$
$$\text{sujeto a } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \text{ para todo } i$$

- **Máquinas de Vectores de Soporte para regresión (SVR):** SVR es una variante del SVM utilizada para problemas de regresión. En lugar de encontrar el hiperplano

que mejor separa las clases, SVR busca encontrar la función que mejor se ajusta a los datos, minimizando al mismo tiempo la cantidad de puntos que están fuera del margen de tolerancia definido por un valor epsilon.

$$\min_{\mathbf{w},b,\zeta,\zeta^*} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*)$$

$$\text{sujeto a } y_i - \mathbf{w}\cdot\mathbf{x}_i - b \leq \varepsilon + \zeta_i \text{ y } \mathbf{w}\cdot\mathbf{x}_i + b - y_i \leq \varepsilon + \zeta_i^* \text{ para todo } i$$

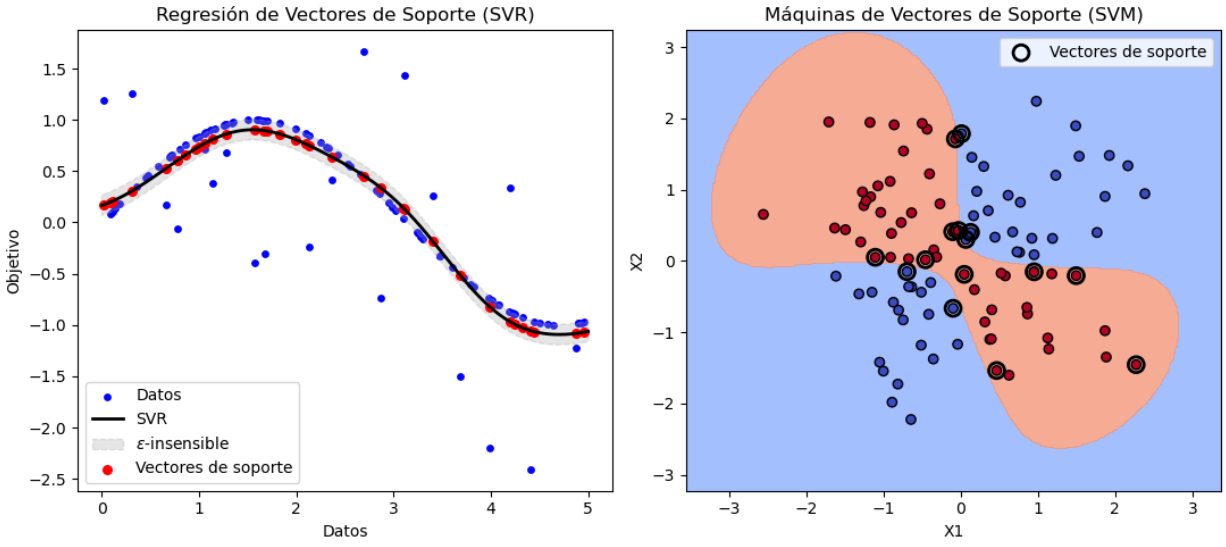

Figura 2.1: Comparación de un modelo SVR y un modelo SVM.

- **Redes Neuronales:** Las redes neuronales son modelos compuestos por capas de nodos interconectados, que aprenden a partir de los datos mediante el ajuste de los pesos de las conexiones entre las neuronas. Cada neurona en una red neuronal está asociada con una función de activación que determina su salida en función de la suma ponderada de las entradas recibidas más un sesgo. Una red neuronal típica consta de varias capas, incluida una capa de entrada, una o más capas ocultas y una capa de salida. Cada capa está formada por un conjunto de neuronas interconectadas, y la información fluye de la capa de entrada a través de las capas ocultas hasta la capa de salida.

  Matemáticamente, el cálculo realizado en una red neuronal se puede expresar como sigue. Sea x el vector de entrada de la red neuronal, W la matriz de pesos que conecta las neuronas en una capa con las neuronas en la siguiente capa, b el vector de sesgos

y f la función de activación. La salida de una neurona en una capa dada se calcula como:

$$z = Wx + b \ , \quad a = f(z) \tag{2.1}$$

donde z es la suma ponderada de las entradas más el sesgo, y a es la salida de la neurona después de aplicar la función de activación. Este proceso se repite para todas las neuronas en la capa, y la salida de la capa se convierte en la entrada de la siguiente capa hasta llegar a la capa de salida.

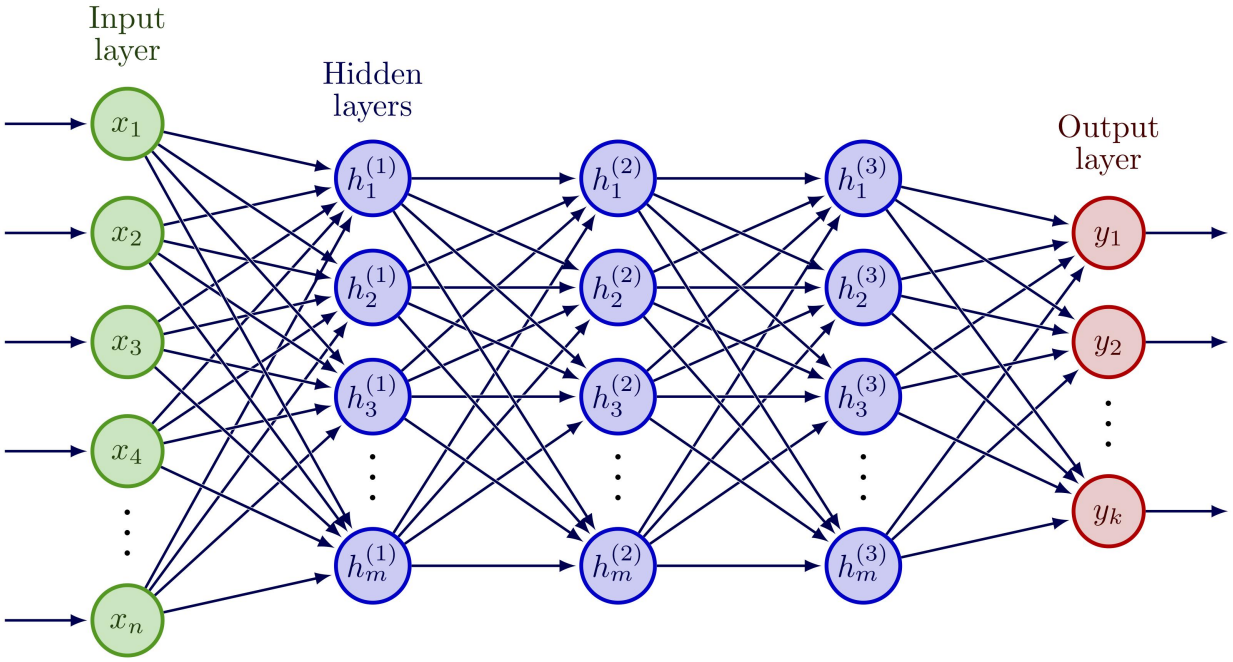

Figura 2.2: Esquema de funcionamiento de una red neuronal, los valores de *Input* x se corresponden a los datos de entrada, mientras que los valores y o *Output* se corresponden a los de salida.

2.3.2. Aprendizaje no supervisado

El aprendizaje no supervisado se centra en descubrir patrones y estructuras en conjuntos de datos no etiquetados. Algunos de los algoritmos más relevantes en este tipo de aprendizaje son:

- **Análisis de Componentes Principales (PCA):** PCA es una técnica utilizada para

reducir la dimensionalidad de conjuntos de datos, preservando al mismo tiempo la mayor cantidad posible de variabilidad. Su fundamento matemático implica calcular los vectores propios de la matriz de covarianza de los datos y proyectar los datos originales en un nuevo espacio de menor dimensión definido por estos vectores propios.

$$\text{Var}(\mathbf{z}_i) = \lambda_i \tag{2.2}$$

donde $\mathbf{z}_i$ es la i-ésima componente principal y λ_i es el i-ésimo valor propio de la matriz de covarianza.

- **Agrupamiento K-Means:** K-Means es un algoritmo de agrupamiento que divide un conjunto de datos en k grupos basados en la similitud entre las observaciones.

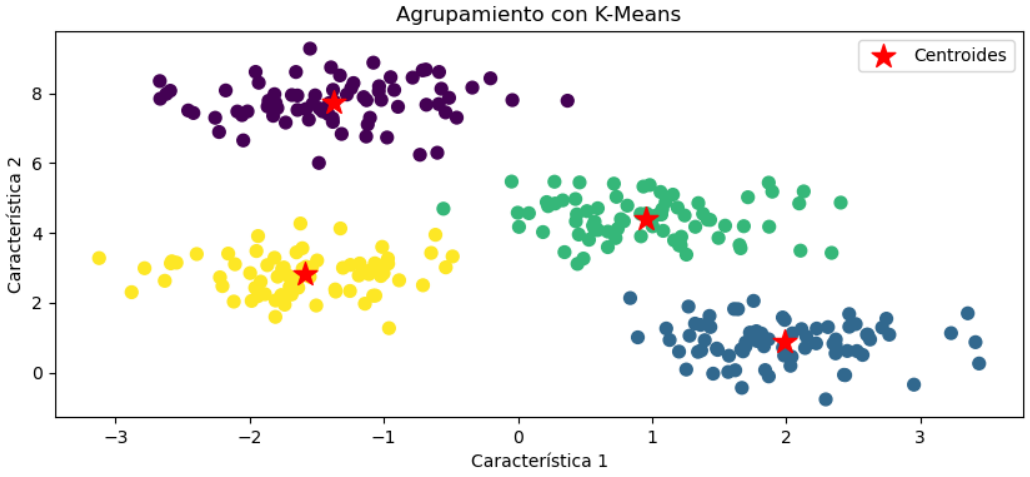

Figura 2.3: Esquema de funcionamiento de kmeans.

Su fundamento matemático implica minimizar la suma de las distancias al cuadrado de cada punto de datos al centroide más cercano de su grupo asignado.

$$J = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \mu_i||^2 \tag{2.3}$$

donde C_i es el i-ésimo grupo y μ_i es el centroide del grupo C_i.

2.3.3. Aprendizaje por refuerzo

El aprendizaje por refuerzo se centra en aprender a través de la interacción con un entorno, maximizando una recompensa acumulativa, como se ejemplifica en la figura 2.4.

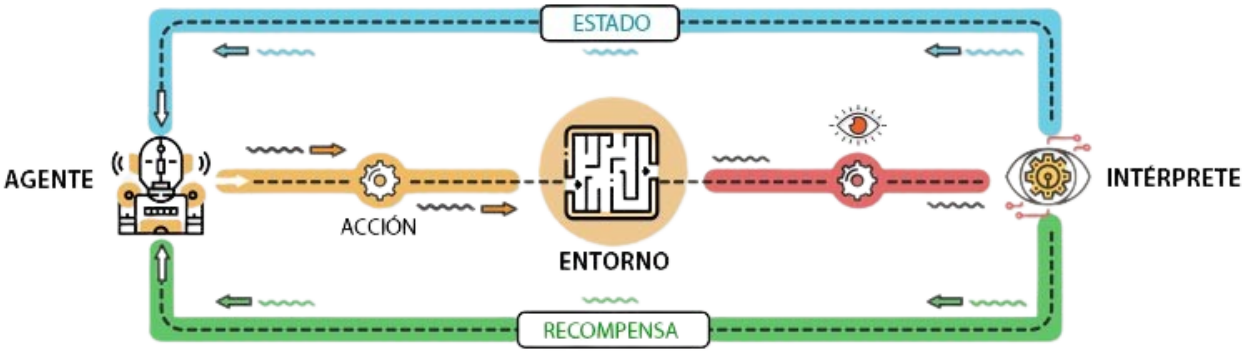

Figura 2.4: Esquema de funcionamiento del aprendizaje por refuerzo. [21]

Algunos algoritmos destacados en este tipo de aprendizaje son:

- **Q-Learning:** Q-Learning es un algoritmo de aprendizaje por refuerzo que busca aprender la función $Q(s, a)$, que representa la calidad de tomar una acción a en un estado s. Su fundamento matemático se basa en la ecuación de Bellman, que relaciona el valor de $Q(s, a)$ con el valor de Q para el siguiente estado s' y todas las posibles acciones a' en ese estado.

$$Q(s, a) = (1 - \alpha) \cdot Q(s, a) + \alpha \cdot [r + \gamma \cdot \max_{a} Q(s', a')] \tag{2.4}$$

  donde r es la recompensa recibida, α es la tasa de aprendizaje y γ es el factor de descuento.

- **Deep Q-Networks (DQN):** DQN es una extensión de Q-Learning que utiliza redes neuronales para aproximar la función $Q(s, a)$. Su fundamento matemático implica el uso de la técnica de descenso de gradiente para minimizar la diferencia entre las predicciones de la red neuronal y los objetivos calculados utilizando la ecuación de Bellman.

$$L(\theta) = \mathbb{E}[(r + \gamma \cdot \max_{a'} Q(s', a'; \theta^{-}) - Q(s, a; \theta))^2] \tag{2.5}$$

donde θ son los parámetros de la red neuronal y θ^- son los parámetros de una red objetivo utilizada para calcular los objetivos.

2.4. Evaluación y validación de modelos

La evaluación y validación adecuada de los modelos es fundamental para garantizar su fiabilidad y generalización a nuevos datos.

2.4.1. Validación cruzada

La validación cruzada es una técnica utilizada para evaluar el rendimiento de un modelo al dividir el conjunto de datos en múltiples subconjuntos llamados *folds*. El modelo se entrena en varios de estos *folds* y se evalúa en el *fold* restante, repitiendo este proceso varias veces para obtener una estimación del rendimiento del modelo independiente a la selección de conjuntos de entrenamiento y test. La forma más común de validación cruzada es el método *k-fold*, donde el conjunto de datos se divide en k *folds* para realizar las iteraciones, tal y como se muestra en la figura 2.5.

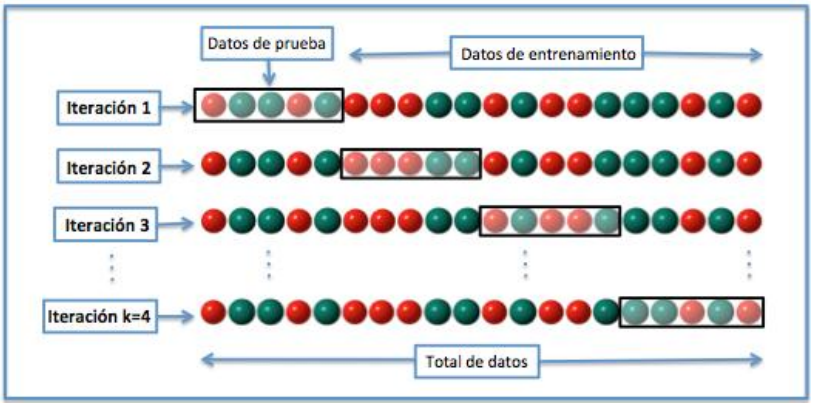

Figura 2.5: Esquema de funcionamiento de la técnica de validación cruzada *K-Folds*. [22]

2.4.2. Curva de aprendizaje

La curva de aprendizaje es una herramienta útil para evaluar la capacidad de generalización de un modelo en función del tamaño del conjunto de datos de entrenamiento. Graficar el rendimiento del modelo en función del tamaño del conjunto de datos puede proporcionar información sobre si el modelo se beneficia de más datos o si está sufriendo de sobreajuste u *overfitting*. La curva de aprendizaje suele mostrar la variación del error del modelo (por ejemplo, el error cuadrático medio en problemas de regresión o la precisión en problemas de clasificación) en función del tamaño del conjunto de entrenamiento.

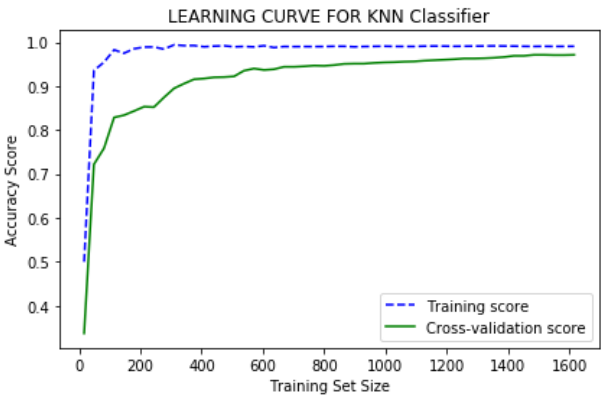

Figura 2.6: Curva de aprendizaje para un modelo de clasificación KNN. [23]

2.4.3. Métricas de evaluación

Existen varias métricas para evaluar el rendimiento de un modelo dependiendo del tipo de problema que se esté abordando. A continuación, se explicarán las más comunes:

2.4.3.1. Métricas para modelos de clasificación

En modelos de clasificación, el objetivo es asignar una etiqueta o clase a cada instancia de datos. Un ejemplo muy común utilizado en el mundo del ML es la clasificación de una lista de pasajeros del Titanic en supervivientes o no supervivientes en función de datos como la clase en la que viajaban, la edad o el sexo. Durante la explicación y como simplificación, nos centraremos en los modelos de clasificación binaria, donde los datos se definen como pertenecientes o no pertenecientes a una clase, aunque existen modelos para la clasificación en varias categorías. Antes de entrar en detalle en las diferentes métricas de los modelos de clasificación, debemos definir varios conceptos:

- **Umbral de clasificación (umbral de decisión)**: Es un valor que se utiliza para

separar las clases en un problema de clasificación binaria. Cuando se realiza una predicción, si la probabilidad de que una instancia sea verdadera (true) supera el umbral, se clasifica como verdadera; de lo contrario, se clasifica como falsa (false).

- **Verdadero positivo (TP)**: Se refiere a los casos en los que el modelo predice correctamente una instancia como perteneciente a una clase específica y esta instancia realmente pertenece a esa clase.
- **Verdadero negativo (TN)**: Se refiere a los casos donde se predice correctamente una instancia como no perteneciente a una clase específica.
- **Falso positivo (FP)**: Se refiere a los casos en los que el modelo predice incorrectamente una instancia como perteneciente a una clase específica cuando en realidad no pertenece a esa clase.
- **Falso negativo (FN)**: Se refiere a los casos en los que el modelo predice incorrectamente una instancia como no perteneciente a una clase específica cuando en realidad pertenece a esa clase.

Para visualizar de forma gráfica las predicciones correctas e incorrectas del modelo, se utiliza la matriz de confusión. Esta matriz es una tabla que muestra la cantidad de predicciones correctas e incorrectas, organizadas por cada clase. En el contexto de la clasificación binaria, la matriz de confusión se estructura de la siguiente manera:

	Predicción negativa	Predicción positiva
Valor negativo	Verdadero negativo (TN)	Falso positivo (FP)
Valor positiva	Falso negativo (FN)	Verdadero positivo (TP)

Cuadro 2.1: Esquema de la matriz de confusión

Conociendo estos conceptos podemos definir las métricas mas comunes. En primer lugar tenemos la **exactitud (accuracy)**, que es una medida de la proporción de predicciones correctas realizadas por el modelo. Se calcula como:

$$\text{Accuracy} = \frac{\text{Número de predicciones correctas}}{\text{Número total de predicciones}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.6}$$

Por otra parte, tenemos la **precisión (precision)**, la cual es una medida de la proporción de identificaciones positivas y la **recuperación (recall)**, que es una medida del número de verdaderos positivos que se identificaron de forma correcta. Se calculan como:

$$\text{Precision} = \frac{\text{Verdaderos positivos}}{\text{Verdaderos positivos} + \text{Falsos negativos}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.7}$$

$$\text{Recall} = \frac{\text{Verdaderos negativos}}{\text{Verdaderos negativos} + \text{Falsos positivos}} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{2.8}$$

También tenemos la **puntuación F1**, que es una medida que combina la precisión y la recuperación en una sola métrica. Se calcula como la media armónica de ambas métricas:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.9}$$

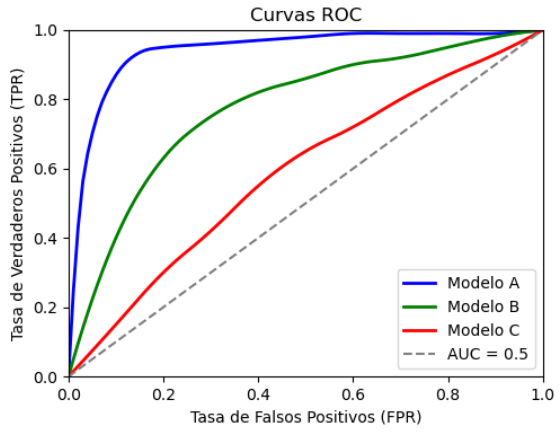

Figura 2.7: Curvas ROC para varios modelos. El modelo A presenta el mejor rendimiento.

Por último tenemos la **curva ROC** (curva de característica operativa del receptor), que es una medida de la capacidad de discriminación de un modelo en problemas de clasificación binaria. De forma gráfica, se dibuja como la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR). De esta forma, cada punto en la curva representa el rendimiento del modelo para un umbral de clasificación específico.

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.10}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{2.11}$$

Para aplicar la información dada por ROC, se emplea el **área bajo la curva (AUC)**, que proporciona una medida agregada del rendimientos para todos los umbrales de clasificación posibles y, como su nombre indica, es el área definida por la curva ROC. El valor 1 indicaría un modelo perfecto, donde para cualquier umbral de clasificación los positivos son clasificados correctamente, un valor de 0.5 indica un rendimiento similar al azar, y valores inferiores a 0.5 indican un rendimiento peor que el azar.

2.4.3.2. Métricas para problemas de regresión

En problemas de regresión, el objetivo es predecir un valor numérico continuo a partir de un conjunto de variables independientes. Las métricas de evaluación más comunes para estos incluyen:

- **R^2 (Coeficiente de determinación)**: Es una medida estadística que indica la proporción de la variabilidad de la variable dependiente explicada por el modelo de regresión. Es decir, R^2 indica qué tan bien el modelo de regresión se ajusta a los datos observados. Un valor de 1 significaría que el modelo se ajusta perfectamente a los datos, y un valor de 0 sería lo contrario.

- **MSE (Error cuadrático medio)**: Es el promedio de los cuadrados de los errores entre los valores observados y los valores predichos por el modelo. Hay que tener en cuenta que no se encuentra en la misma escala que las medidas, pero puede ser útil para penalizar de manera más significativa los errores grandes, ya que se elevan al cuadrado.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.12}$$

- **RMSE (Raíz del error cuadrático medio)**: Es la raíz cuadrada del MSE. Devuelve la métrica a la misma escala que los datos originales, lo que facilita la interpretación.

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.13}$$

- **MAE (Mean Absolute Error)**: Mide la magnitud promedio de los errores en las predicciones del modelo, sin considerar su dirección.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.14}$$

Además de estas métricas, es común encontrarse con las gráficas de valores predichos frente a valores reales y gráficas de residuos, un ejemplo se ilustra en la figura 2.8. En la primera los valores predichos por el modelo se representan en el eje x, mientras que los

valores reales se representan en el eje y, siendo un buen modelo aquel donde los puntos caen cerca de la diagonal. Por otra parte, la gráfica de residuos representa los residuos (diferencias entre los valores reales y los valores predichos) en el eje y, y los valores predichos en el eje x. Idealmente, los residuos deberían distribuirse aleatoriamente alrededor de cero.

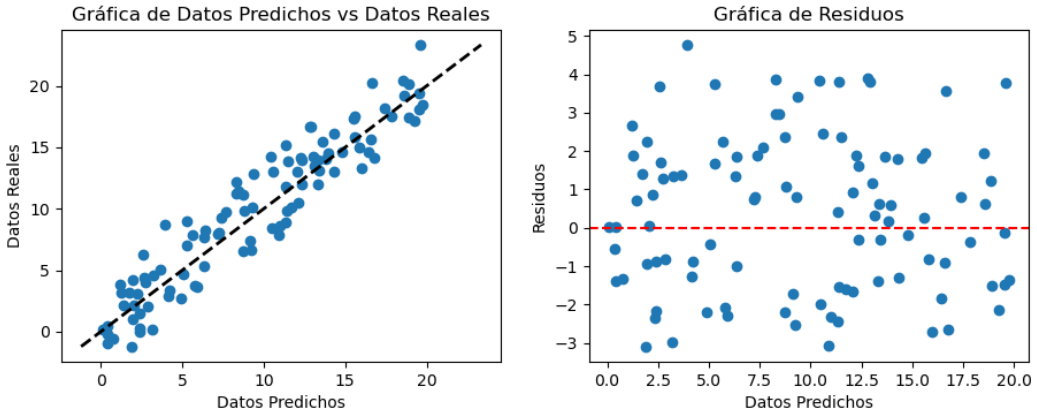

Figura 2.8: A la izquierda, se muestra una gráfica de valores reales frente a los predichos por un modelo, observándose que los puntos se agrupan en torno a la diagonal. A la derecha se muestra la gráfica de residuos, que se distribuyen aleatoriamente entorno a 0.

2.4.4. Técnicas para el ajuste de hiperparámetros

El ajuste de hiperparámetros es un paso crucial en el desarrollo de los modelos para optimizar su rendimiento y generalización. Estos son configuraciones preestablecidas al modelo que afectan su comportamiento y rendimiento durante el entrenamiento, como la complejidad del modelo, la velocidad de entrenamiento o la regularización. Para poder definir los mejores valores para cada modelo y problema existen varias técnicas.

La primera de ellas es el **Grid Search**, que consiste en especificar una lista de valores para cada hiperparámetro y evaluar el rendimiento del modelo para todas las combinaciones posibles. Esto se hace construyendo una malla de todas las combinaciones y evaluando el modelo para cada punto. Esta técnica no es óptima cuando se quiere ajustar múltiples hiperparámetros con muchos valores a la vez. Si tenemos n hiperparámetros con m_i opciones posibles para el hiperparámetro i, entonces se evaluarían $m_1 \times m_2 \times \ldots \times m_n$ combinaciones.

El **Random Search** es una técnica que nos permite simplificar la anterior. Teniendo la malla del Grid Search, esta técnica evalúa aleatoriamente un número de combinaciones, lo que permite ajustes más eficientes aunque no tan precisos.

Por último encontramos la **optimización bayesiana**. Esta técnica utiliza el teorema de Bayes para actualizar iterativamente la distribución de probabilidad de los hiperparámetros del modelo a medida que se realizan evaluaciones adicionales del mismo. Inicialmente, se tiene una primera distribución de probabilidad $P(A)$ que representa una primera distribución de los hiperparámetros. A medida que se evalúa el modelo con diferentes conjuntos y se observan los resultados (B), el teorema de Bayes se utiliza para actualizar esta primera distribución a otra $P(A|B)$, que representa la distribución de los hiperparámetros después de observar los datos. Para ello se aplica:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.15}$$

Donde $P(A|B)$ es la distribución posterior dada la observación de los datos, $P(B|A)$ es la probabilidad de observar los datos dado un conjunto específico de hiperparámetros, $P(A)$ es la primera distribución de los hiperparámetros, y $P(B)$ es la probabilidad de observar los datos.

La optimización bayesiana utiliza esta distribución posterior para decidir qué conjunto de hiperparámetros analizar a continuación, enfocándose en áreas del espacio de hiperparámetros donde la distribución posterior es más alta. Esta iteración continua de evaluación permite encontrar de manera eficiente conjuntos óptimos de hiperparámetros para el modelo.

2.4.5. Técnicas de *under* y *oversampling*

En problemas de clasificación con desequilibrio de clases, donde una clase representa una porción significativamente mayor de los datos que las otras, los modelos de aprendizaje automático (machine learning) pueden sesgarse hacia la clase mayoritaria, lo que resulta en un bajo rendimiento en las clases minoritarias. Para abordar este problema, se utilizan técnicas de *undersampling* y *oversampling* para equilibrar las clases y mejorar la precisión del modelo en las clases minoritarias.

El *undersampling* implica reducir el tamaño de la clase mayoritaria para que sea comparable al de la clase minoritaria. Se pueden utilizar diferentes métodos, como el muestreo aleatorio, donde se eliminan aleatoriamente instancias de la clase mayoritaria, o el muestreo de submuestreo basado en la distancia, donde se seleccionan instancias de la clase mayoritaria que se encuentran a una mayor distancia de las instancias de la clase minoritaria. Por otro lado, el *oversampling* implica aumentar el tamaño de la clase minoritaria duplicando o generando nuevas instancias de la misma. Entre las técnicas de *oversampling* más comunes se encuentran la replicación de instancias, donde se duplican las de la clase minoritaria, y la generación de instancias sintéticas, donde se utilizan algoritmos para generar nuevas instancias similares a las existentes en la clase minoritaria.

Capítulo 3

Desarrollo práctico

Este capítulo profundiza en el desarrollo central del trabajo, consistente en aplicar diferentes técnicas de machine learning que traten de predecir que materiales son potencialmente superconductores a partir de la densidad de estados. Hay que tener en cuenta que estos modelos no son más que métodos heurísticos que nos permiten reconocer patrones invisibles a simple vista, no por ello son correctos siempre ni se han de tomar como verdad absoluta.

Durante el desarrollo se explorarán dos aspectos clave. En primer lugar, se examinará el proceso de obtención y procesamiento de los datos, así como la información que estos proporcionan. Posteriormente, se aplicarán las algunas de técnicas y pasos detallados en el capítulo anterior para desarrollar diversos modelos y obtener resultados.

3.1. Elaboración del código

Antes de empezar con el desarrollo del trabajo, es importante realizar varias aclaraciones sobre el código, el cual se encontrará de forma completa y descargable en el repositorio de GitHub: https://github.com/olayafgon/ML-and-Superconductors.

En primer lugar, todo el código ha sido escrito en Python, optando por una estructura modular, separando las distintas partes y procesos del proyecto para facilitar su comprensión

y mantenimiento. Además, se ha implementado un archivo de configuración para facilitar la personalización de ciertos aspectos del programa, como la ruta de almacenamiento de los resultados o la selección de determinadas redes de Bravais.

Es importante subrayar que la escritura y estructuración del código se ha realizado siguiendo las recomendaciones establecidas en la obra *Clean Code* de Robert C. Martin [10]. En este sentido, se ha puesto especial atención en la creación de código claro, con nombres descriptivos y fáciles de entender, evitando la ambigüedad y favoreciendo la legibilidad. Se ha buscado que cada función tenga una única responsabilidad, manteniendo así un alto grado de cohesión y facilitando su reutilización. Siguiendo esta filosofía, se han aplicado también los principios SOLID [11], los cuales han servido como guía para la arquitectura del código. SOLID es un acrónimo de cinco conceptos fundamentales en la programación orientada a objetos listados a continuación:

- S - *Single Responsibility Principle* (Principio de Responsabilidad Única): Cada módulo o clase debe tener una única responsabilidad, un único motivo para cambiar.
- O - *Open/Closed Principle* (Principio Abierto/Cerrado): Las entidades de software (clases, módulos, funciones, etc.) deben estar abiertas para su extensión, pero cerradas para su modificación. Esto significa que se debe poder añadir nuevas funcionalidades sin necesidad de modificar el código existente.
- L - *Liskov Substitution Principle* (Principio de Sustitución de Liskov): Los objetos de una clase hija deben poder sustituir a los objetos de su clase padre sin que se produzca un comportamiento erróneo. Este principio garantiza la coherencia en la jerarquía de clases y facilita la reutilización del código.
- I - *Interface Segregation Principle* (Principio de Segregación de Interfaces): Es preferible tener varias interfaces específicas para cada cliente, en lugar de una única interfaz de propósito general. Esto evita que los clientes dependan de métodos que no utilizan.
- D - *Dependency Inversion Principle* (Principio de Inversión de Dependencias): Los módulos de alto nivel no deben depender de los módulos de bajo nivel. Ambos deben depender de abstracciones. Las abstracciones no deben depender de los detalles. Los

detalles deben depender de las abstracciones. Este principio promueve la independencia entre los módulos y facilita la reutilización del código.

3.2. Datos

3.2.1. Obtención y procesado

Quizás el paso más relevante en cualquier modelo es la obtención y procesado del dato. Para poder obtener resultados fiables y precisos es fundamental tener un dato de calidad con el que entrenar el modelo, además de procesarlo de la forma correcta para evitar valores nulos o valores con formatos incorrectos.

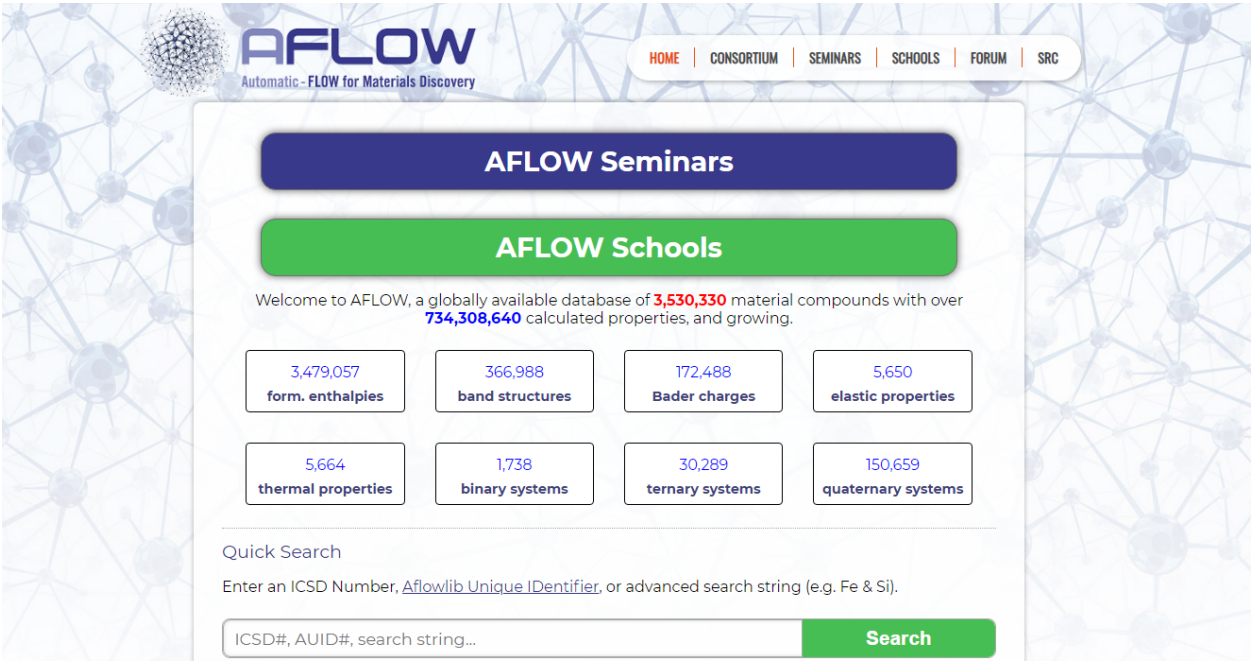

Figura 3.1: Interfaz web de la base de datos *Aflowlib.org*. [12]

En primer lugar, hemos obtenido una base de datos con numerosos materiales cristalinos e información sobre estos. En concreto, hemos obtenido los datos de la base *Aflowlib.org* [12], que proporciona acceso a una selección de datos sobre materiales inorgánicos recopilados de la base de datos *Identified Inorganic Crystal Structures* (ICSD). Proporciona información

sobre los materiales como la estructura, la energía y las propiedades electrónicas.

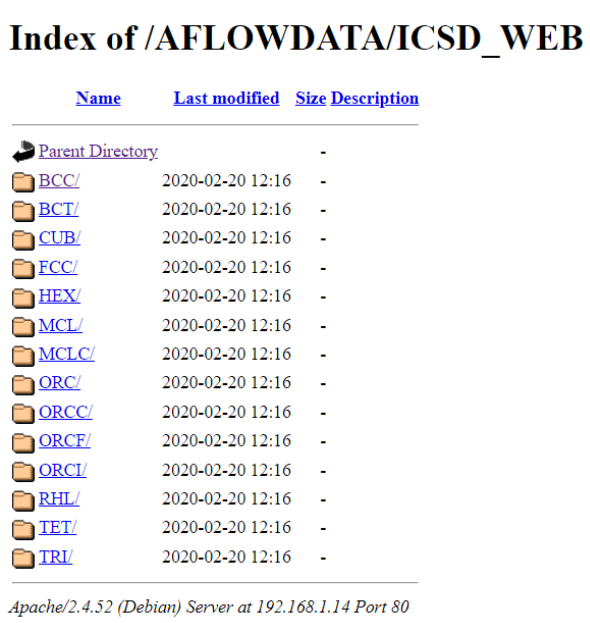

Figura 3.2: Directorio de la sección AFLOW-DATA_ICSD_WEB dentro de la base de datos *Aflowlib.org*.

En la figura 3.2 se muestra el directorio de la web https://aflowlib.duke.edu/AFLOWDATA/ICSD_WEB/, el cual es una parte del sitio web *Aflowlib*. Esta sección se estructura en torno a la búsqueda y descarga de información sobre los materiales, organizados en carpetas según su estructura cristalina. Para cada material disponible contiene archivos con información detallada, los cuales se pueden descargar en diferentes formatos, incluyendo archivos de texto, archivos de imagen o archivos con información en formato XML.

Entre los archivos disponibles se encuentran los del tipo *dosdata.json.xz*. Debido a que son archivos de fácil lectura y procesado, y a que contienen información sobre la densidad de estados electrónica, son los que se han escogido para obtener los datos. Hay que tener en cuenta que esta DOS es calculada mediante aproximaciones teóricas del estado fundamental, por lo cual no es la densidad del material en estado superconductor.

Además de la información de la DOS, proporcionan datos como la composición química del material, especificando los elementos presentes (etiquetados como *species*) y su estequiometría en la celda unidad (*composition*). También contienen datos energéticos del sistema en el cual se ha simulado la densidad de estados, incluyendo los límites inferior y superior del rango de energía considerado (*Emin* y *Emax*), el nivel de Fermi (*Efermi*) y la resolución empleada en el cálculo de la densidad de estados (*DOS_grid*).

La información sobre la DOS total (*tDOS*) se encuentra en la sección *tDOS_data*, donde se proporciona la *tDOS* en función de la energía (*energy*). La sección *pDOS_data* contiene información sobre la DOS parcial (*pDOS*) para cada orbital atómico (s, p, d) y cada especie

química presente en el material. Por ejemplo, para el material TiO2 podría contener subsecciones para los átomos de Ti y O. Cada subsección incluiría la pDOS para los orbitales s, p y d de cada átomo, organizada en función de la energía. El indicador *spin_polarized* señala si el cálculo de la DOS ha tenido en cuenta la polarización de espín. La entrada *total* en cada subsección representa la suma de la *pDOS* para todos los orbitales del elemento correspondiente.

Dada la envergadura de la base de datos, que contiene más de 60000 materiales, la descarga manual de los archivos resulta inviable. Por este motivo, se ha desarrollado un código en Python que automatiza el proceso de descarga y extracción de datos. Este código, dividido en dos etapas principales, primero accede a cada elemento de la base de datos mediante técnicas de *web scraping* y descarga los archivos *_dosdata.json.xz* correspondientes. Posteriormente, descomprime los archivos descargados para obtener los archivos *.json* que contienen la información en un formato legible. Todo este proceso se corresponde con el módulo del programa *src/data_handling/data_raw_dowload.py*.

Una vez descargados los archivos, se utiliza un segundo código, correspondiente al módulo *src/data_handling/data_raw_read.py*, para procesar la información almacenada. La clase MaterialRawDataRead se encarga de leer los archivos JSON, extraer la información relevante sobre la DOS y almacenar los resultados en un archivo CSV. Para cada material, se extraen la composición química, la energía de Fermi, la red de Bravais, si es un material magnético o no y los datos de la DOS total. En el caso de materiales magnéticos, la densidad de estados total se calcula como la diferencia entre la DOS de espín mayoritario y minoritario.

Dado que cada material tiene una energía de Fermi diferente, para poder comparar la densidad de estados se han tomado los valores en un intervalo de $15eV$ entorno a la energía de Fermi. En este caso, para cada material obtenemos 1999 puntos de densidad de estados, donde el punto central (1000) contiene la densidad de estados en el valor de la energía de Fermi correspondiente al material. Los puntos del 1 al 999 se correspondería al rango de energías $[E_F - 15eV, E_F)$, y los puntos del 1001 al 1999 se corresponden a $(E_F, E_F + 15eV]$.

Por otra parte, si lo que queremos es encontrar relaciones entre la densidad de estados y la superconductividad, necesitamos saber que materiales de nuestra base son superconductores.

Teniendo en cuenta que las simulaciones con las que se han obtenido las densidades de estados se han ejecutado con temperatura 0K, podemos ignorar la temperatura crítica de los materiales.

Para etiquetar los superconductores hemos tomado la información obtenida en el proyecto 3DSC [14]. En él, los autores desarrollaron un algoritmo de machine learning que relaciona los materiales de la base de datos SuperCon con estructuras cristalinas y la base de datos ICSD mediante la fórmula química. El código y los datos del proyecto se encuentran de forma pública en un repositorio de GitHub [15], conteniendo uno de los archivos (*3DSC_ICSD_only_IDs.csv*) un listado de superconductores dados por su código en el ICSD.

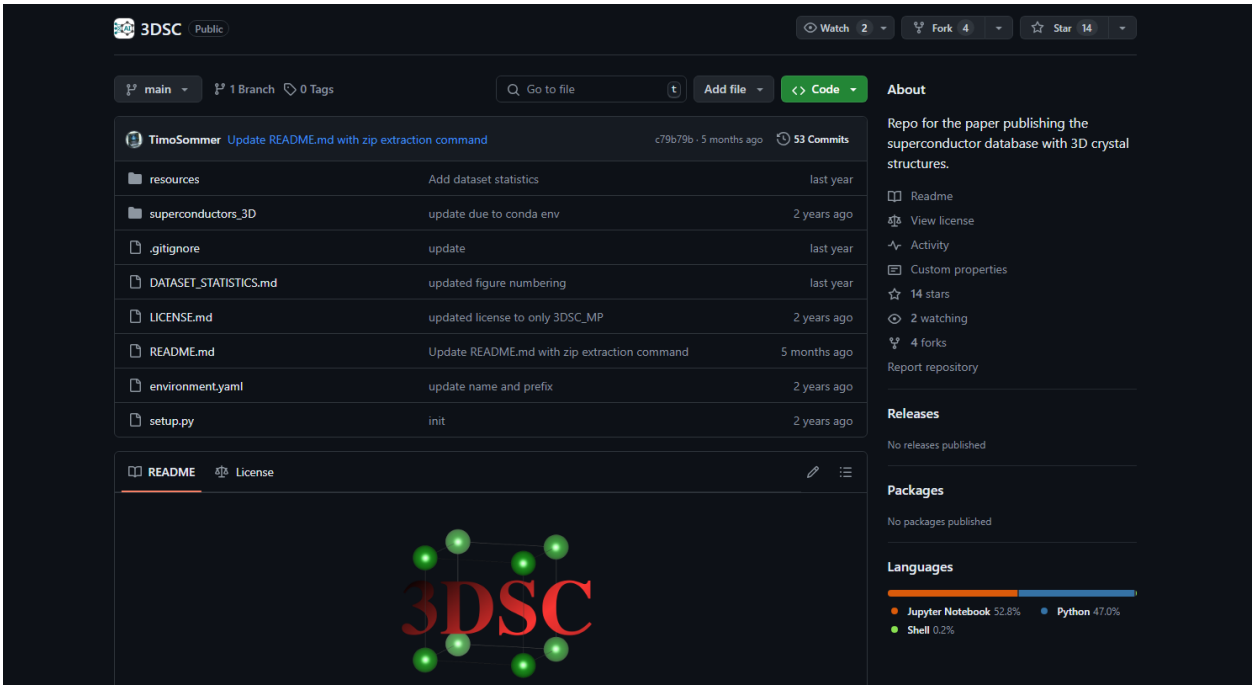

Figura 3.3: Repositorio para el artículo que publica la base de datos de superconductores con estructuras cristalinas en 3D (3DSC). [15]

Finalmente y mediante el módulo *src/data_handling/data_processing.py*, se relacionan los códigos ICSD de materiales superconductores con los datos obtenidos previamente de la base AFLOW. Cabe mencionar que la base de datos 3DSC solo nos proporciona una lista de materiales superconductores, lo que no significa que los materiales que no se encuentran

en esa lista no puedan serlo. Por simplificación en el desarrollo tomaremos como no superconductores aquellos materiales no presentes en la lista, pero si se precisara desarrollar un modelo con precisión sería fundamental contrastar con otras bases de datos que, por ejemplo, dieran información sobre materiales aislantes o conductores, y descartar aquellos que no pudiéramos etiquetar.

3.2.2. Análisis exploratorio

Como ya mencionamos en los fundamentos, el primer paso de cualquier modelo es el análisis y exploración de los datos para comprenderlos.

En nuestro caso, tras la obtención y procesado comentados anteriormente, se obtuvo un dataframe (figura 3.4) de 2005 columnas y 60216 filas. Cada una de esas filas se corresponde a un material con código ICSD único. Respecto a las columnas, contienen información sobre la red de Bravais del material, su fórmula química, el código en ICSD, la energía de Fermi en electronvoltios, si es magnético o no, si es superconductor y 1999 columnas correspondientes a las densidades de estados en el entorno de la energía de Fermi.

	bravais_lattice	material_name	ICSD	fermi_energy	is_magnetic	is_superconductor	DOS_m15_00	...	DOS_p15_00
0	BCC	Ag1F6Sb1	28676	-2.03066	False	False	0.0	...	23.260
1	BCC	Ag1F6Sb1	411795	-2.03253	False	False	0.0	...	13.800
2	BCC	Ag1Te3	37186	5.57824	False	False	0.0	...	4.651
3	BCC	Ag3Au1Se2	15734	2.21332	False	False	0.0	...	0.000
4	BCC	Ag3Au1Se2	171959	2.22832	False	False	0.0	...	0.000
...	...	...	...	...	...	...	...	...	...
60212	TRI	Se4Tl4Zr1	261209	3.16038	False	False	0.0	...	0.000
60213	TRI	Se6Si2Tl4	35042	2.64747	False	False	0.0	...	0.000
60214	TRI	Se11Ta2Tl4	412581	2.44024	False	False	0.0	...	0.000
60215	TRI	Si2Te6Tl6	416310	4.07369	False	False	0.0	...	0.000
60216	TRI	Tl3Yb8	104202	2.27586	False	False	0.0	...	0.000

60216 rows × 9 columns

Figura 3.4: DataFrame con los datos empleados en el trabajo.

Conviene apuntar que la propiedad magnética, aunque ha sido empleada como una variable más en el problema y se observará en las gráficas a continuación, no es el objetivo de este trabajo y podría requerir de un estudio en más profundidad en otros proyectos. En este

caso, como materiales magnéticos nos referimos a aquellos cuyas densidades de estados de espines arriba y abajo, o mayoritario y minoritario, son distintas.

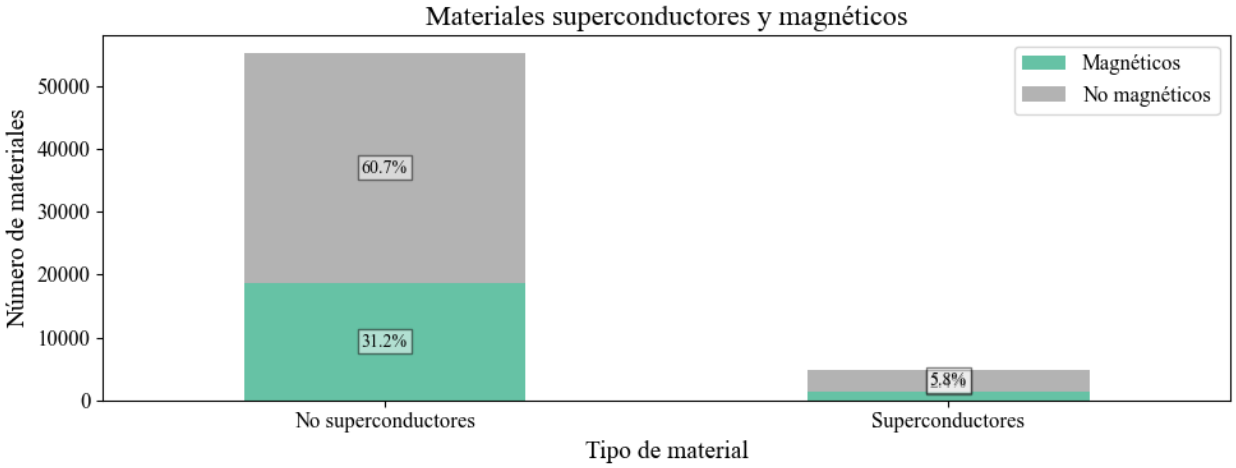

Figura 3.5: Gráfico de barras de la propiedad superconductor y magnética de los materiales.

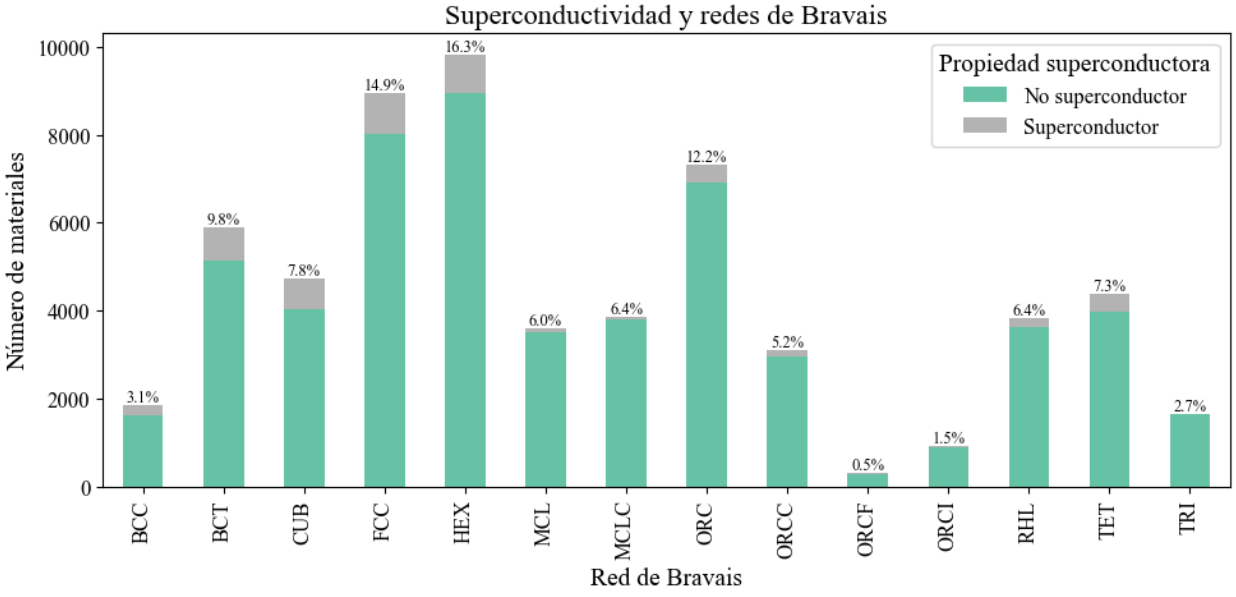

Figura 3.6: Distribución de los materiales en las diferentes redes de Bravais.

En cuanto a las propiedades de los materiales presentes en el conjunto de datos, en la figura 3.5 observamos que la mayor parte de los materiales son clasificados como superconductores, representando solo un 8.2 % del total. Respecto a la propiedad magnética, se observa que un 33.6 % de los materiales la presentan. Por otro lado, la figura 3.6 muestra la distribución del número de materiales en las diferentes redes de Bravais, siendo las estructu-

ras mayoritarias la hexagonal (HEX) con un 16.3 %, la cúbica centrada en las caras (FCC) con un 14.9 % y la ortorrómbica simple (ORC) con un 12.2 %.

Otro dato que puede ser de interés a la hora de estudiar las propiedades de los materiales es la composición de estos. Para analizar este aspecto, se examinó la fórmula química de los diferentes materiales buscando la presencia de cada elemento. En la figura 3.7 se representa el número de materiales que contienen cada elemento químico. Se observa que el oxígeno está presente en el 11.3 % de los materiales, seguido por el silicio (3.5 %) y el azufre (3.3 %).

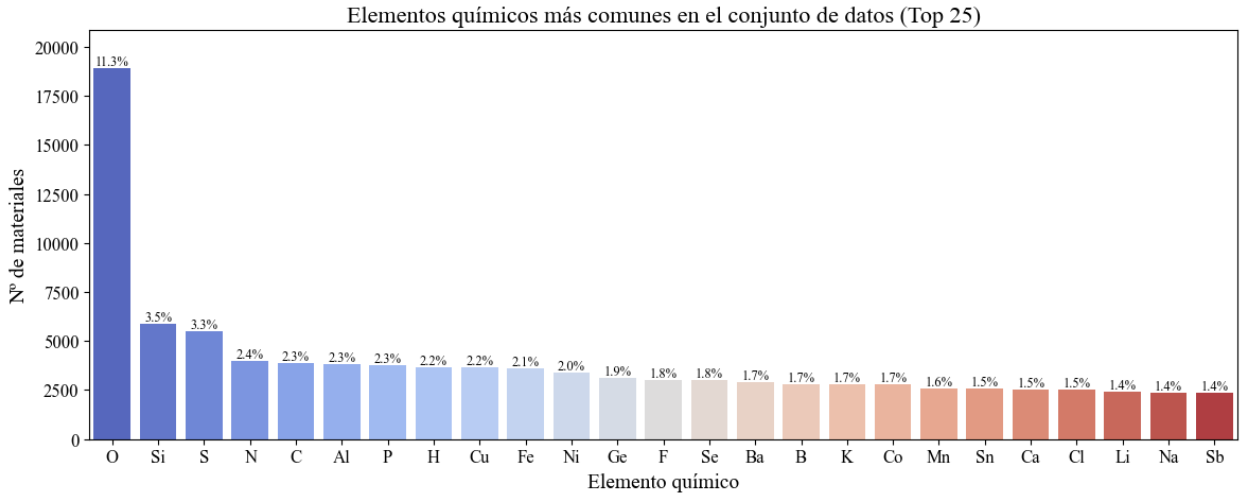

Figura 3.7: Elementos químicos más frecuentes en los materiales del conjuntos de datos.

Al analizar los materiales superconductores, se observa que el oxígeno (O) y el silicio (Si) son los elementos más presentes, como se muestra en la Figura 3.8. Sin embargo, al considerar la proporción de materiales superconductores que contienen un elemento específico en relación con el número total de materiales con dicho elemento, el torio (Th), el osmio (Os) y el rodio (Rh) adquieren mayor relevancia. Un aspecto interesante que comparten estos elementos más presentes en los superconductores es su pertenencia a la serie de los actínidos y metales de transición, los cuales suelen presentar orbitales f y d parcialmente llenos.

Una vez visto todo esto solo queda observar como se comportan las densidades de estados. Las gráficas de la figura 3.9 representan las densidades de estados (DOS) en función de la energía para dos materiales con códigos ICSD 189400 (Si) y 608582 (Al12Mo1), con estructuras cristalinas ortorrómbica centrada en el cuerpo (ORCC) y cúbica centrada en el

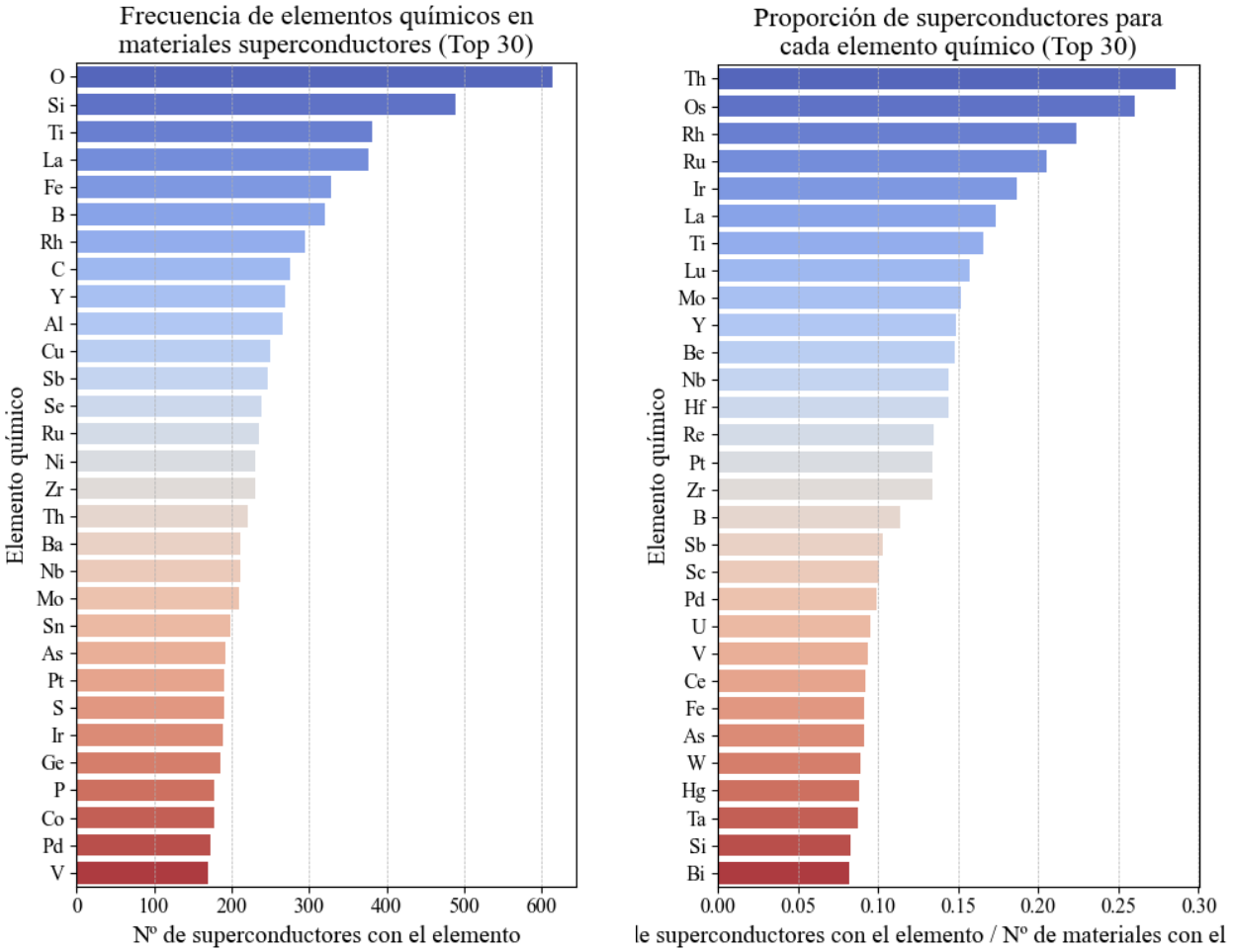

Figura 3.8: A la izquierda se muestran los elementos químicos más presentes en los materiales superconductores y a la derecha la proporción de materiales superconductores que contiene cada elemento químico respecto al total de materiales que contienen dicho elemento.

cuerpo (BCC) respectivamente. El primero, el Si, es no superconductor y presenta una banda de energías prohibidas por encima de la energía de Fermi mientras que Al12Mo1, clasificado como superconductor, no presenta banda prohibida.

La figura 3.10 presenta la mediana de la densidad de estados (DOS) para superconductores y no superconductores con estructuras cristalinas HEX y ORC, respectivamente. Las gráficas muestran que la mediana de la DOS en la energía de Fermi es significativamente mayor para los superconductores en ambos casos, lo que sugiere una posible correlación entre la DOS y la superconductividad en estas estructuras cristalinas. Aunque la mediana representa una medida robusta para ver la tendencia general, hay que tener en cuenta que

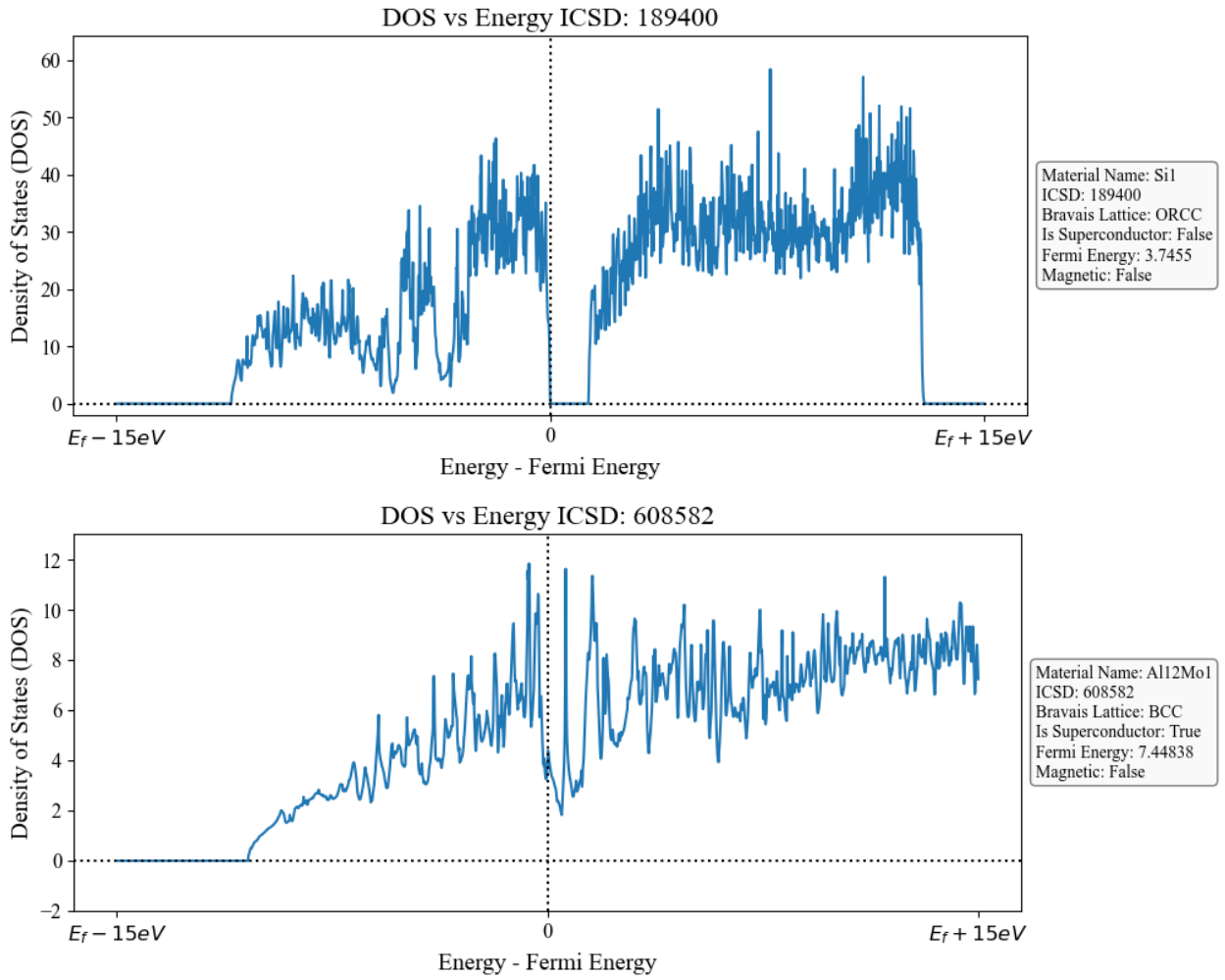

Figura 3.9: Densidad de estados frente a la energía relativa a la energía de Fermi para el Si (ICSD_189400) y el el Al12Mo1 (ICSD_608582).

la muestra puede ser heterogénea y que la DOS individual de cada material puede variar considerablemente.

Aunque en los ejemplos mostrados haya diferencias entre los materiales clasificados como superconductores y no superconductores, sin más información puede ser complicado clasificar los materiales. Esto es también debido al hecho de que la densidad de estados empleada no es calculada en un estado superconductor y, por tanto, no muestra explícitamente este estado. De todas formas, la disponibilidad de estados electrónicos juega un papel esencial en la formación de los pares de Cooper, lo que abre la posibilidad a la existencia de patrones ocultos. Por ello, en la sección a continuación, exploraremos diferentes modelos que nos permitirán

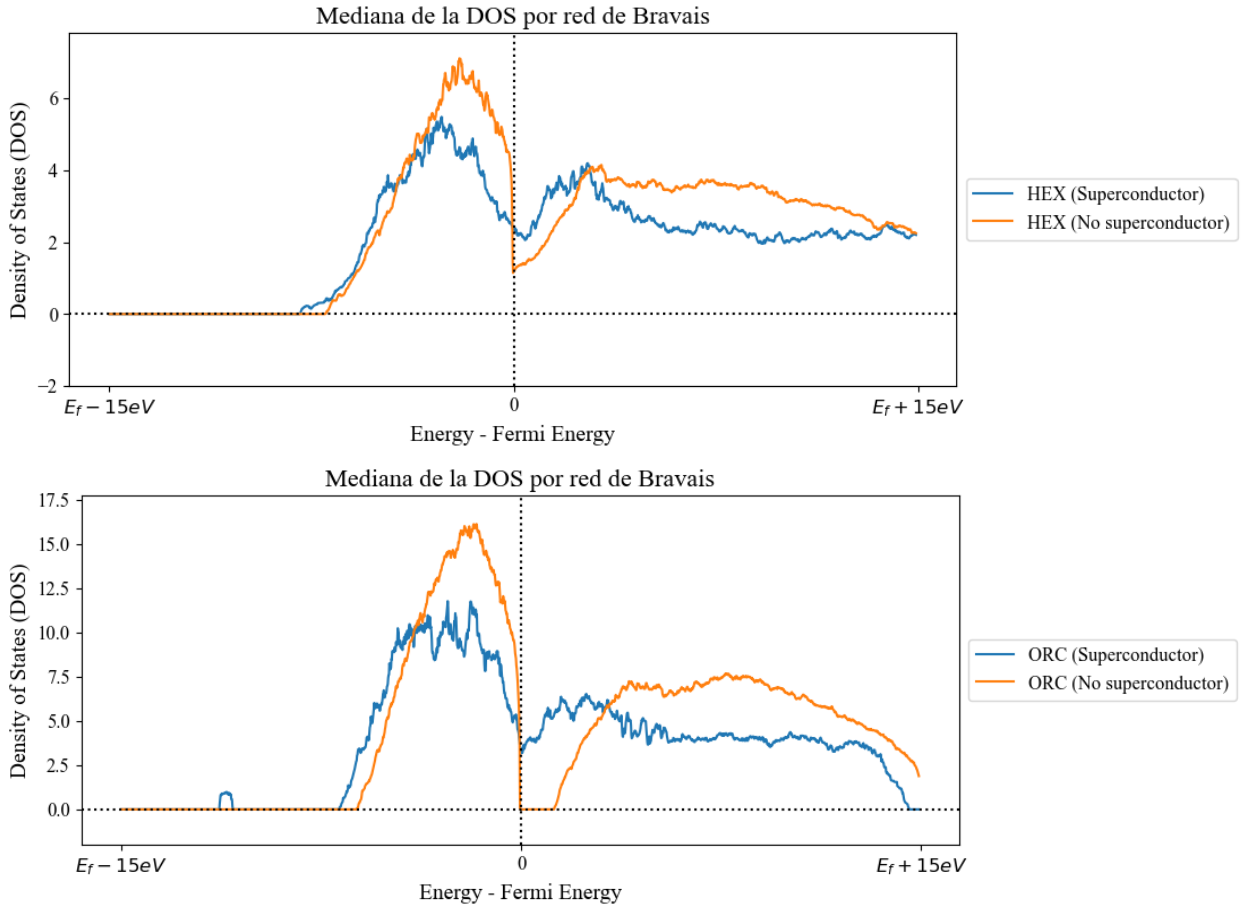

Figura 3.10: Medianas de la densidad de estados frente a la energía frente a la energía de Fermi para los materiales con estructura cristalina hexagonal (HEX) y ortorrómbica simple (ORC).

determinar si es posible saber si un material puede desarrollar o no la superconductividad a partir de esta DOS.

3.3. Métodos de Machine Learning

Como se desarrollo en el fundamento teórico, existen diversos modelos y algoritmos de aprendizaje automático (machine learning). En nuestro caso, el problema se centra en distinguir entre materiales superconductores y no superconductores, lo que corresponde a una clasificación binaria. En la siguiente sección exploraremos varios modelos y técnicas para abordar este problema.

3.3.1. Exploración inicial de algoritmos: Autogluon y PCAs

Existen numerosos algoritmos destinados a las clasificaciones binarias, pero no todos son efectivos en todos los problemas, y escoger uno sin tener referencias puede no ser lo mas preciso. Por esta razón, se ha utilizado la biblioteca AutoGluon, que facilita el entrenamiento y la comparación de varios modelos para un problema dado.

AutoGluon es una biblioteca de código abierto que simplifica el proceso de entrenamiento y despliegue de modelos. Su principal fortaleza radica en su capacidad de automatizar gran parte del trabajo, facilitando la exploración de una amplia gama de modelos sin necesidad de implementar cada uno manualmente. Entrena automáticamente varios algoritmos de aprendizaje automático, con opciones como incluir diferentes configuraciones de hiperparámetros o la optimización de estos, y selecciona el mejor modelo basado en una métrica de rendimiento predefinida.

En el siguiente código se muestra un preprocesado básico de los datos, en el que se usa el *RobustScaler* como método de escalado de las variables numéricas, para que el modelo no de mas importancia a unas variables que otras, y *OneHotEncoder* para las variables categóricas. Tras el preprocesado se aplica Autogluon sobre el conjunto. Como métrica de evaluación hemos usado la F1, ya que es la mas sensible a falsos negativos y, considerando que el conjunto de datos contiene un porcentaje muy bajo de materiales superconductores, es la mas adecuada.

```
data = materials_data.copy()
data.dropna(inplace=True)
data['is_superconductor'] = data['is_superconductor'].astype(bool)

dos_cols = data.loc[:, 'DOS_m15_00':'DOS_p15_00'].columns.tolist()
categorical_cols = ['bravais_lattice']
target_col = ['is_superconductor']

X = data.drop(target_col, axis=1)
X = X.fillna(0)
y = data[target_col]
y = y.iloc[:, 0]

categorical_transformer = OneHotEncoder(handle_unknown='ignore')
numerical_transformer = RobustScaler()
```

```
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', categorical_transformer, categorical_cols),
        ('num', numerical_transformer, ['fermi_energy', 'is_magnetic'] + dos_cols)
    ]
)

X_preprocessed = preprocessor.fit_transform(X)
X_preprocessed = pd.DataFrame(X_preprocessed,
↪  columns=preprocessor.get_feature_names_out())

X_train, X_test, y_train, y_test = train_test_split(X_preprocessed, y, test_size=0.2,
↪  random_state=42)

train_data = pd.concat([pd.DataFrame(X_train).reset_index(drop=True),
↪  pd.DataFrame(y_train).reset_index(drop=True)], axis=1)
test_data = pd.concat([pd.DataFrame(X_test).reset_index(drop=True),
↪  pd.DataFrame(y_test).reset_index(drop=True)], axis=1)

predictor = TabularPredictor(label='is_superconductor',
                             problem_type='binary',
                             eval_metric='f1',
                             path=os.path.join(path_final, 'autogluon_raw', 'Autogluon')
                             ).fit(
                                 TabularDataset(train_data),
                                 presets='medium_quality'
                                 )
performance = predictor.evaluate(TabularDataset(test_data))
leaderboard = predictor.leaderboard(TabularDataset(test_data),
↪  extra_metrics=['accuracy', 'roc_auc', 'precision', 'recall'], silent=True)
```

model	score_test	accuracy	roc_auc	precision	recall	score_val	eval_metric	pred_time_test	pred_time_val	fit_time
LightGBMXT	0.607211	0.948439	0.944582	0.768000	0.502092	0.628895	f1	0.198806	0.045526	62.790186
WeightedEnsemble_L2	0.607211	0.948439	0.944582	0.768000	0.502092	0.628895	f1	0.201306	0.047026	63.043919
LightGBM	0.598214	0.947692	0.942161	0.766340	0.490586	0.597101	f1	0.203279	0.049642	60.237896
LightGBMLarge	0.595527	0.947443	0.944748	0.765189	0.487448	0.612245	f1	0.175145	0.049613	66.553304
XGBoost	0.584576	0.946778	0.938747	0.768313	0.471757	0.605341	f1	0.713218	0.147273	127.501212
ExtraTreesGini	0.575032	0.944039	0.918897	0.723810	0.476987	0.582353	f1	0.281381	0.070143	7.535776
ExtraTreesEntr	0.574481	0.943789	0.923248	0.719685	0.478033	0.592375	f1	0.247290	0.071704	7.334228
RandomForestEntr	0.570347	0.943457	0.924635	0.718601	0.472803	0.587537	f1	0.233251	0.065557	41.417081
RandomForestGini	0.567602	0.943706	0.915982	0.727124	0.465481	0.585075	f1	0.280303	0.066550	48.284939
KNeighborsDist	0.467011	0.922866	0.808160	0.517154	0.425732	0.476190	f1	4.959379	1.083987	1.894165
KNeighborsUnif	0.376830	0.918715	0.799801	0.481301	0.309623	0.403509	f1	4.948307	1.085087	1.940960

Figura 3.11: Leaderboard obtenido para la ejecución del Autogluon sobre los datos de materiales con un preprocesado simple.

El resultado del *leaderboard* se presenta en la figura 3.11. En este caso, el mejor modelo

entrenado es el *LightGBMXT*, es decir, un modelo de *boosting*[1] basado en árboles de decisión. Este modelo obtuvo un valor F1 de 0.6072 en el conjunto de prueba y 0.6289 en el conjunto de validación. Aunque las métricas F1 de prueba y validación son similares, lo que indica que no hay *overfitting*, el resultado sigue estando lejos de ser óptimo. Este caso también demuestra que la precisión no siempre es la mejor medida. Se observa una precisión del 95 %, la cual se puede obtener si, por ejemplo, la clase positiva tiene una población muy baja y el modelo clasifica todos los materiales como negativos.

El primer problema que podemos identificar es el hecho de que solo el 8.2 % del conjunto de datos es clasificado como superconductor. Este desequilibrio, como se mencionó en la sección 2.4.5, puede sesgar el modelo hacia la clase mayoritaria (no superconductores), lo que podría resultar en un rendimiento deficiente para la clasificación de superconductores.

Para abordar este problema, se implementaron técnicas de remuestreo, incluyendo *undersampling* aleatorio mediante *RandomUnderSampler* y *oversampling* mediante *RandomOverSampler* y *SMOTE*. El *undersampling* aleatorio reduce el tamaño de la clase mayoritaria, mientras que el *oversampling* aumenta el tamaño de la clase minoritaria. En el caso de *SMOTE*, se generan nuevas muestras sintéticas de la clase minoritaria mediante la interpolación de las muestras existentes.

```
X_train, X_test, y_train, y_test = train_test_split(X_preprocessed, y, test_size=0.2,
↪  random_state=42)

test_data = pd.concat([pd.DataFrame(X_test).reset_index(drop=True),
↪  pd.DataFrame(y_test).reset_index(drop=True)], axis=1)

# Undersampling con RandomOverSampler
rus = RandomUnderSampler()
X_under_resampled, y_under_resampled = rus.fit_resample(X_train, y_train)
train_data_undersampled =
↪  pd.concat([pd.DataFrame(X_under_resampled).reset_index(drop=True),
↪  pd.DataFrame(y_under_resampled).reset_index(drop=True)], axis=1)
predictor_undersampled = TabularPredictor(label='is_superconductor',
                                          problem_type='binary', eval_metric='f1',
                                          path = os.path.join(path_final, 'resampling',
                                          ↪  'RandomUnderSampler')
```

[1]Los modelos de *boosting* construyen un modelo final combinando secuencialmente una serie de modelos base. Cada modelo se centra en corregir los errores del modelo anterior, mejorando el rendimiento del modelo final.

```
                                        ).fit(
                                            TabularDataset(train_data_undersampled),
                                            presets='medium_quality'
                                            )
leaderboard_undersampled = predictor_undersampled.leaderboard(TabularDataset(test_data),
↪  extra_metrics=['accuracy', 'roc_auc', 'precision', 'recall'], silent=True)

# Oversampling con RandomOverSampler
ros = RandomOverSampler()
X_over_resampled, y_over_resampled = ros.fit_resample(X_train, y_train)
train_data_oversampled =
↪  pd.concat([pd.DataFrame(X_over_resampled).reset_index(drop=True),
↪  pd.DataFrame(y_over_resampled).reset_index(drop=True)], axis=1)
predictor_oversampled = TabularPredictor(label='is_superconductor',
                                         problem_type='binary', eval_metric='f1',
                                         path=os.path.join(path_final, 'resampling',
                                         ↪  'RandomOverSampler')
                                         ).fit(
                                            TabularDataset(train_data_oversampled),
                                            presets='medium_quality'
                                            )
leaderboard_oversampled = predictor_oversampled.leaderboard(TabularDataset(test_data),
↪  extra_metrics=['accuracy', 'roc_auc', 'precision', 'recall'], silent=True)

# Oversampling con SMOTE
smote = SMOTE()
X_smote_resampled, y_smote_resampled = smote.fit_resample(X_train, y_train)
train_data_smote = pd.concat([pd.DataFrame(X_smote_resampled).reset_index(drop=True),
↪  pd.DataFrame(y_smote_resampled).reset_index(drop=True)], axis=1)
predictor_smote = TabularPredictor(label='is_superconductor',
                                   problem_type='binary', eval_metric='f1',
                                   path=os.path.join(path_final, 'resampling', 'SMOTE')
                                   ).fit(
                                       TabularDataset(train_data_smote),
                                       presets='medium_quality'
                                       )
leaderboard_smote = predictor_smote.leaderboard(TabularDataset(test_data),
↪  extra_metrics=['accuracy', 'roc_auc', 'precision', 'recall'], silent=True)
```

El código anterior muestra la aplicación de estas técnicas de remuestreo. Los resultados obtenidos muestran que el modelo *LightGBM* y el modelo *XGBoost* [2] obtuvieron los mejores rendimientos en general.

Se observó que el valor de *F1* sobre el conjunto de test fue de entorno al 0.4 para la técnica

[2]XGBoost (Extreme Gradient Boosting) es un algoritmo que utiliza el método de boosting para construir un modelo de predicción a partir de una combinación de árboles de decisión.

RandomUnderSampler y de entorno al 0.6 para la técnica *RandomOverSampler* y *SMOTE*, lo que nos indica un mejor rendimiento de estas últimas. Si nos fijamos en los resultados de validación, observamos valores mayores al 0.8, bastante alejados de los resultados de test. Esto sugiere un posible *overfitting*, ya que su rendimiento en validación es superior al del conjunto de prueba. Para mitigar este problema, se puede explorar el ajuste de hiperparámetros.

Antes de entrar en ese ajuste, observamos que otra de las cuestiones que puede estar causando estos resultados es la gran dimensionalidad del conjunto de datos, que posee 2005 columnas. Esto puede afectar a la capacidad del modelo para generalizar. Para abordar esto, a continuación se explorará un análisis de componentes principales (PCA), comentado en la sección 2.3.1, con el objetivo de reducir la dimensionalidad del conjunto de datos y evaluar su impacto en el rendimiento del modelo.

Para este estudio se realizaron entrenamientos de modelos en Autogluon para diferentes técnicas de remuestreo y distintos números de componentes principales (PCA), cuyos resultados se muestran en las tablas 3.1 y 3.2. Se observan las métricas de *F1* (2.9), tanto para el conjunto de validación como para el conjunto de prueba, y *accuracy* (2.6), *AUC* (figura 2.7), *precision* (2.7) y *recall* (2.8) para el conjunto de prueba.

Al analizar las tablas, se aprecia que las técnicas de remuestreo pueden tener un impacto significativo en las métricas de rendimiento. Por ejemplo, en la Tabla 3.1, *SMOTE* muestra un alto valor de F1 en el conjunto de validación para la mayoría de los valores de PCA, pero su rendimiento en el conjunto de prueba es ligeramente inferior. En contraste, *RandomUnderSampler* muestra un valor de F1 más bajo en validación, pero su desempeño en prueba es más estable.

Asimismo, se observa que el número de componentes principales utilizados en PCA también juega un papel crucial en la calidad del modelo. Se observa que, en general, un número mayor de componentes principales (n=50) conduce a mejores resultados en las métricas de prueba. Sin embargo, al aumentar el número de componentes más allá de 150 no siempre produce mejoras significativas. Esto podría deberse a que un número excesivo de componentes puede introducir ruido en los datos, afectando negativamente el rendimiento del modelo.

Técnica de Remuestreo	**PCA (n)**	**Validación**	**Test**				
		F1	**F1**	**Accuracy**	**AUC**	**Precision**	**Recall**
No	2	0.591358	0.593660	0.945035	0.923383	0.721386	0.501046
RandomUnderSampler	2	0.838868	0.409091	0.801395	0.912072	0.267788	0.866109
RandomOverSampler	2	0.986582	0.633171	0.943623	0.922098	0.654749	0.612971
SMOTE	2	0.947867	0.574552	0.919213	0.922195	0.493614	0.687238
No	5	0.626955	0.607735	0.948522	0.927208	0.737960	0.544979
RandomUnderSampler	5	0.859223	0.412995	0.801976	0.919960	0.270035	0.877615
RandomOverSampler	5	0.988533	0.646766	0.946945	0.928434	0.685815	0.611925
SMOTE	5	0.974339	0.634217	0.936981	0.927304	0.588025	0.688285
No	10	0.625231	0.640449	0.949435	0.939886	0.759342	0.531381
RandomUnderSampler	10	0.871122	0.418673	0.803554	0.923333	0.273603	0.891213
RandomOverSampler	10	0.987752	0.649591	0.946612	0.928965	0.678043	0.623431
SMOTE	10	0.974339	0.638663	0.937147	0.926888	0.587357	0.699791
No	20	0.623163	0.657459	0.946778	0.932297	0.711409	0.554393
RandomUnderSampler	20	0.880689	0.444326	0.826802	0.927125	0.298070	0.872385
RandomOverSampler	20	0.989315	0.649591	0.946612	0.928965	0.678043	0.623431
SMOTE	20	0.977408	0.654120	0.943540	0.930684	0.636634	0.672594
No	30	0.620648	0.664820	0.947526	0.934338	0.728169	0.540795
RandomUnderSampler	30	0.893513	0.424168	0.804550	0.931930	0.276820	0.906904
RandomOverSampler	30	0.990099	0.637989	0.946197	0.932691	0.684652	0.597280
SMOTE	30	0.980253	0.646122	0.942793	0.931729	0.634712	0.657950
No	40	0.625000	0.668524	0.949186	0.936140	0.754438	0.533473
RandomUnderSampler	40	0.856115	0.430156	0.812355	0.928072	0.283389	0.892259
RandomOverSampler	40	0.990884	0.635082	0.946280	0.931453	0.689106	0.588912
SMOTE	40	0.980159	0.638037	0.941216	0.937920	0.624000	0.652720
No	50	0.624402	0.668464	0.947858	0.933410	0.729050	0.546025
RandomUnderSampler	50	0.862651	0.421314	0.806128	0.924167	0.276064	0.889121
RandomOverSampler	50	0.990884	0.637128	0.946280	0.928315	0.686820	0.594142
SMOTE	50	0.980548	0.651813	0.944205	0.933066	0.645791	0.657950
No	75	0.604946	0.629412	0.948273	0.942159	0.768116	0.498954
RandomUnderSampler	75	0.858525	0.436410	0.817502	0.926800	0.289062	0.890167
RandomOverSampler	75	0.991670	0.632723	0.946695	0.926009	0.698232	0.578452
SMOTE	75	0.982567	0.650106	0.945035	0.925673	0.657051	0.643305
No	100	0.622331	0.644068	0.948605	0.943421	0.746706	0.533473
RandomUnderSampler	100	0.853365	0.409235	0.798157	0.920188	0.266540	0.880753
RandomOverSampler	100	0.990884	0.608037	0.944122	0.919044	0.685940	0.546025
SMOTE	100	0.982970	0.649198	0.945533	0.927860	0.664114	0.634937

Cuadro 3.1: Tabla con diferentes métricas de validación y de test para diferentes técnicas de remuestreo y diferente número de componentes principales.

Técnica de Remuestreo	**PCA (n)**	**Validación**	**Test**				
		F1	**F1**	**Accuracy**	**AUC**	**Precision**	**Recall**
No	150	0.611354	0.641834	0.948273	0.939551	0.757342	0.512552
RandomUnderSampler	150	0.853988	0.418350	0.806294	0.921336	0.274632	0.877615
RandomOverSampler	150	0.992063	0.624927	0.946778	0.935065	0.709163	0.558577
SMOTE	150	0.982512	0.640650	0.944952	0.915200	0.664792	0.618201
No	200	0.603917	0.631884	0.947941	0.938603	0.762360	0.500000
RandomUnderSampler	200	0.860697	0.420202	0.809366	0.919320	0.276964	0.870293
RandomOverSampler	200	0.992063	0.621125	0.945201	0.932681	0.688295	0.565900
SMOTE	200	0.977408	0.634658	0.945035	0.916376	0.671729	0.601464
No	250	0.594663	0.635569	0.947028	0.927558	0.757282	0.489540
RandomUnderSampler	250	0.843114	0.404085	0.796496	0.915758	0.263225	0.869247
RandomOverSampler	250	0.992063	0.602035	0.944786	0.913705	0.703497	0.526151
SMOTE	250	0.981775	0.629363	0.944454	0.917889	0.669022	0.594142
No	300	0.587026	0.607670	0.946612	0.929609	0.760399	0.478033
RandomUnderSampler	300	0.833741	0.400969	0.794587	0.914203	0.260870	0.866109
RandomOverSampler	300	0.992063	0.611374	0.945533	0.930916	0.704918	0.539749
SMOTE	300	0.981789	0.631925	0.944288	0.918747	0.664360	0.602510
No	400	0.572745	0.593939	0.946114	0.933634	0.772647	0.455021
RandomUnderSampler	400	0.830325	0.404500	0.797825	0.912723	0.263964	0.865063
RandomOverSampler	400	0.992063	0.571779	0.942046	0.908725	0.691395	0.487448
SMOTE	400	0.982957	0.613229	0.942710	0.919556	0.660628	0.572176
No	500	0.568615	0.594595	0.945450	0.932495	0.763668	0.452929
RandomUnderSampler	500	0.844828	0.399902	0.796164	0.909011	0.260925	0.855649
RandomOverSampler	500	0.992851	0.563467	0.941465	0.903214	0.690440	0.475941
SMOTE	500	0.986868	0.609402	0.944122	0.930763	0.684485	0.549163

Cuadro 3.2: Tabla con diferentes métricas de validación y de test para diferentes técnicas de remuestreo y diferente número de componentes principales.

3.3.2. Ajuste de hiperparámetros

Teniendo en cuenta los resultados obtenidos en la sección anterior, y de cara a maximizar el rendimiento de un modelo, se realizó un ajuste de hiperparámetros[3]. Se optó por entrenar dos modelos: *LightGBMXT* y *XGBoost*, debido a que ambos mostraron buen rendimiento general en las pruebas con AutoGluon. Además, se usaron con 10 componentes principales y *RandomOverSampler* como técnica de remuestreo.

Antes de entrar en el ajuste, se hizo un pequeño análisis adicional de los datos. En él se trato de observar si hay alguna correlación entre las variables dependientes y la variable objetivo. En la figura 3.12, se muestra la matriz de correlación que representa las relaciones lineales entre las variables del conjunto de datos tras el preprocesado. La variable objetivo *is_superconductor* presenta una correlación débil con las demás variables, con la excepción de *num_PC1*, *num_PC2*, *num_PC3* y *num_fermi_energy*, es decir, la energía de Fermi y las tres primeras componentes principales, donde se observa una correlación moderada. Esto indica que estas variables podrían tener un impacto en la predicción. Sin embargo, la baja correlación general sugiere que el modelo podría tener dificultades para identificar patrones fuertes y podría presentar una precisión limitada.

Para los ajustes se empleó tanto la técnica de *RandomizedSearchCV* como la de *BayesSearchCV*, con el objetivo de encontrar las mejores configuraciones para cada modelo. Como métrica de puntuación para elegir los parámetros se empleó la puntuación *F1*. Además, se emplearon 5 *folds* en la validación cruzada[4] y 100 iteraciones para cada modelo.

El primero de los hiperparámetros ajustados fue el *learning_rate*, también conocido como tasa de aprendizaje, que determina el paso de aprendizaje durante cada iteración del algoritmo de gradiente. También se ajustaron el *max_depth*, que controla la profundidad máxima de cada árbol de decisión en el modelo, y el *n_estimators*, que define el número de árboles de decisión que se combinan en el modelo. Por otro lado, se ajustaron el *subsample* y *colsample_bytree* que son parámetros de muestreo que ayudan a prevenir el sobreajuste

[3]Para más detalles ver la sección 2.4.4

[4]Para más detalle ver la sección 2.4.1

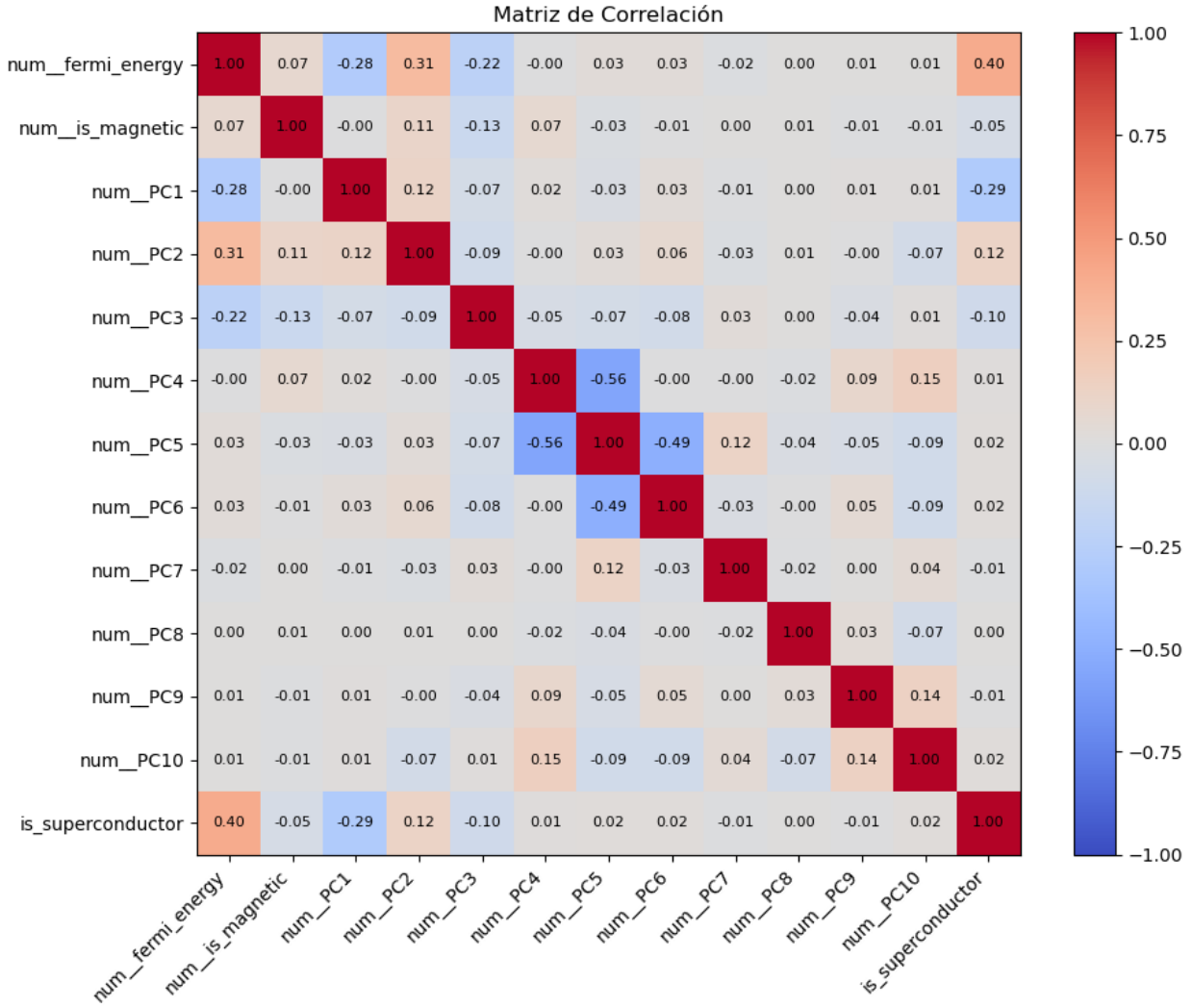

Figura 3.12: Matriz de correlaciones del conjunto de datos de entrenamiento.

mediante la reducción de la complejidad del modelo.

Los resultados obtenidos se muestran en la tabla 3.3. En términos de exactitud, ambos modelos, *LightGBM* y *XGBoost*, alcanzaron resultados muy similares, ligeramente superiores a 0.93. Observando las métricas de precisión, recuperación y puntuación F1, se puede apreciar que *XGBoost* presenta un rendimiento ligeramente mejor.

Modelo	**Método de Búsqueda**	**Accuracy**	**Precision**	**Recall**	**F1 score**	**AUC**
LightGBM	BayesSearchCV	0.9357	0.5864	0.6423	0.6131	0.9299
LightGBM	RandomizedSearchCV	0.9350	0.5817	0.6444	0.6114	0.9330
XGBoost	BayesSearchCV	0.9376	0.6039	0.6234	0.6135	0.9321
XGBoost	RandomizedSearchCV	0.9389	0.6118	0.6297	0.6206	0.9311

Cuadro 3.3: Métricas, sobre el conjunto de datos de prueba, obtenidas para los modelos entrenados con los hiperparámetros obtenidos en las búsquedas.

Para un análisis más profundo, las matrices de confusión[5] para cada modelo se presentan a continuación:

$$LightGBM - BayesSearchCV : \begin{pmatrix} 10655 & 433 \\ 342 & 614 \end{pmatrix}$$

$$LightGBM - RandomizedSearchCV : \begin{pmatrix} 10645 & 443 \\ 340 & 616 \end{pmatrix}$$

$$XGBoost - BayesSearchCV : \begin{pmatrix} 10697 & 391 \\ 360 & 596 \end{pmatrix}$$

$$XGBoost - RandomizedSearchCV : \begin{pmatrix} 10706 & 382 \\ 354 & 602 \end{pmatrix}$$

Las matrices de confusión permiten analizar con mayor detalle la clasificación realizada por los modelos. Por ejemplo, para *XGBoost* con *RandomizedSearchCV*, la matriz muestra que el modelo clasificó correctamente 10706 instancias como no superconductoras y 602 como superconductoras. Sin embargo, se observan 382 casos donde el modelo clasificó erróneamente instancias como no superconductoras cuando en realidad sí lo eran, y 354 casos donde clasificó erróneamente instancias como superconductoras cuando en realidad no lo eran. Es importante destacar que, en general, ambos modelos presentan una tendencia a confundir las instancias superconductoras con las no superconductoras.

[5]Para más detalle ver la tabla 2.1

3.3.3. Modelo final

Por último, se hizo una búsqueda más extensa para seleccionar los hiperparámetros finales para un modelo *XGBoost*. Además de aumentar el número de iteraciones en este ajuste, se implemento *early_stopping*, lo que ayuda a prevenir el sobreajuste del modelo. Por otra parte, se añadieron tres hiperarámetros más centrados en evitar el sobreajuste: *gamma*, que representa la reducción mínima de pérdida requerida para realizar una división en un árbol, y *reg_alpha* y *reg_lambda*, que implementan las regularizaciones para reducir la influencia de características irrelevantes y disminuir la sensibilidad a características individuales.

Los mejores hiperparámetros obtenidos fueron:

- 'colsample_bytree': 0.5
- 'learning_rate': 0.223468
- 'max_depth': 24
- 'n_estimators': 1500
- 'gamma': 0.01
- 'reg_alpha': 0.01
- 'reg_lambda': 0.01
- 'subsample': 0.824074

En la figura 3.13 se puede observar la evolución del entrenamiento del modelo con los parámetros previos. La curva azul representa el error en el conjunto de entrenamiento, mientras que la naranja representa el error en el conjunto de validación. Se puede apreciar que el modelo converge rápidamente en las primeras iteraciones, con una disminución significativa del error tanto en el conjunto de entrenamiento como en el de validación y parando el entrenamiento tras los pasos establecidos en *early_stopping* para evitar mayor separación entre ambos errores.

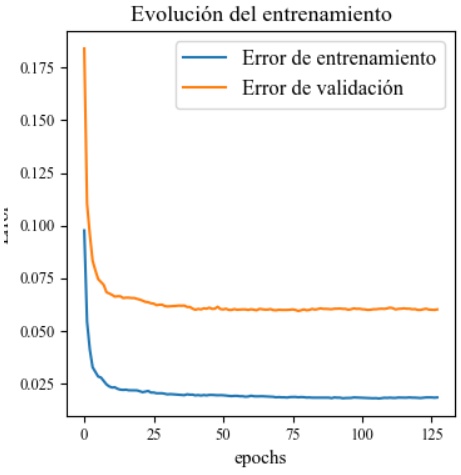

Figura 3.13: Evolución del entrenamiento del modelo.

Las métricas de rendimiento del modelo se muestran en las tabla 3.4. Aunque se logró una alta exactitud de 0.9409 y un área bajo la curva

(AUC) de 0.9324, lo que indica una capacidad notable para clasificar correctamente muestras, la puntuación F1 refleja un desempeño moderado de 0.6330, señalando que el modelo tiene dificultades para identificar correctamente los materiales superconductores, tendiendo a clasificar los materiales como no superconductores, es decir, como la clase mayoritaria.

Métrica	**Valor**
Accuracy	0.9409
Precision	0.6240
Recall	0.6423
Puntuación F1	0.6330
AUC	0.9324

Cuadro 3.4: Métricas de rendimiento del modelo final.

Los valores *SHAP* (*SHapley Additive exPlanations*) permiten comprender la contribución individual de cada característica al valor de salida del modelo para cada instancia. En otras palabras, estos valores muestran cómo cada característica influye en la predicción final del modelo. Para el modelo entrenado, estos valores se muestran en la figura 3.14.

En el gráfico de la izquierda podemos ver los valores SHAP promedio para cada variable o característica. La variable *num_fermi_energy* tiene el mayor valor SHAP promedio positivo (0.71), lo que significa que esta característica tiene un fuerte impacto positivo en la predicción del modelo de media, como habíamos observado en la matriz de correlaciones 3.12.

El gráfico de la izquierda muestra la distribución de los valores SHAP para cada variable. Las características con un impacto positivo en el modelo tienen una distribución de valores SHAP inclinada hacia valores más altos (derecha), mientras que las características con un impacto negativo en el modelo tienen una distribución de valores SHAP inclinada hacia valores más bajos (izquierda). Por ejemplo, la característica *num_PC1* tiene una distribución de valores SHAP inclinada hacia la derecha, lo que sugiere que tiene un impacto hacia la clase positiva en la predicción del modelo en la mayoría de los casos. En contraste, *num_PC9* tiene una distribución de valores más simétrica, lo que sugiere que su impacto en la predicción es más variable y no tan consistente.

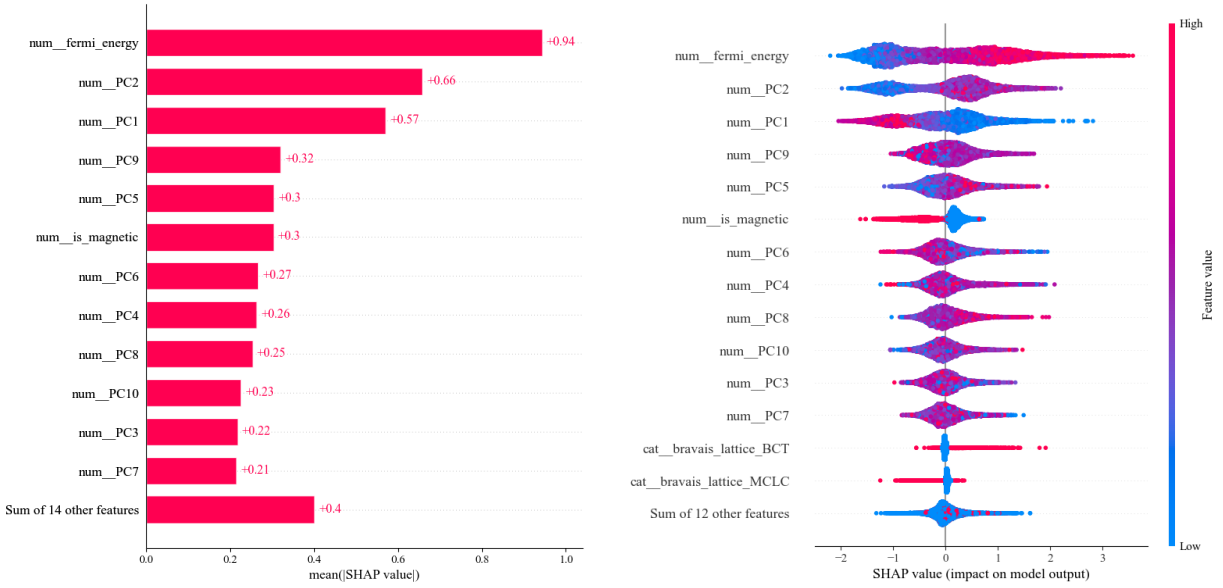

Figura 3.14: A la izquierda se muestra el gráfico del valor SHAP promedio de cada característica, lo que indica la importancia relativa de cada característica en la predicción del modelo. El gráfico de la derecha muestra la distribución de los valores SHAP para cada característica, lo que permite comprender cómo la influencia de cada característica varía en el conjunto de datos.

Por otra parte, la matriz de confusión de la tabla 3.5, nos aporta más información. A partir de ella se observa la clara dificultad del modelo de distinguir los superconductor y la tendencia a clasificar los materiales como no superconductores. Para ver que estos grupos en la matriz de confusión no ocultan subpatrones que nos podrían indicar, por ejemplo, la existencia de más de dos clases, haremos un análisis de los grupos de la matriz.

	Predicción: No superconductor	Predicción: Superconductor
No superconductor	10718	370
Superconductor	342	614

Cuadro 3.5: Matriz de confusión de rendimiento del modelo final.

En primer lugar vamos a ver la distribución de las diferentes redes de bravais en los grupos, ilustrada en la figura 3.15. En la gráfica superior, que muestra las predicciones de superconductores, se observa una mayor frecuencia de verdaderos positivos en las redes de

tipo HEX y BCT, indicando un buen desempeño en identificar correctamente los superconductores en estas estructuras cristalinas. Sin embargo, también se observan falsos negativos significativos en redes como TET y CUB. En la gráfica inferior, se muestran las predicciones de no superconductores. En general se observa que el modelo se comporta bien en este caso. No obstante, se ven falsos positivos en una menor proporción, pero algo significativos en redes como HEX y FCC, lo que implica que el modelo tiende a clasificar incorrectamente algunos no superconductores como superconductores en estas redes.

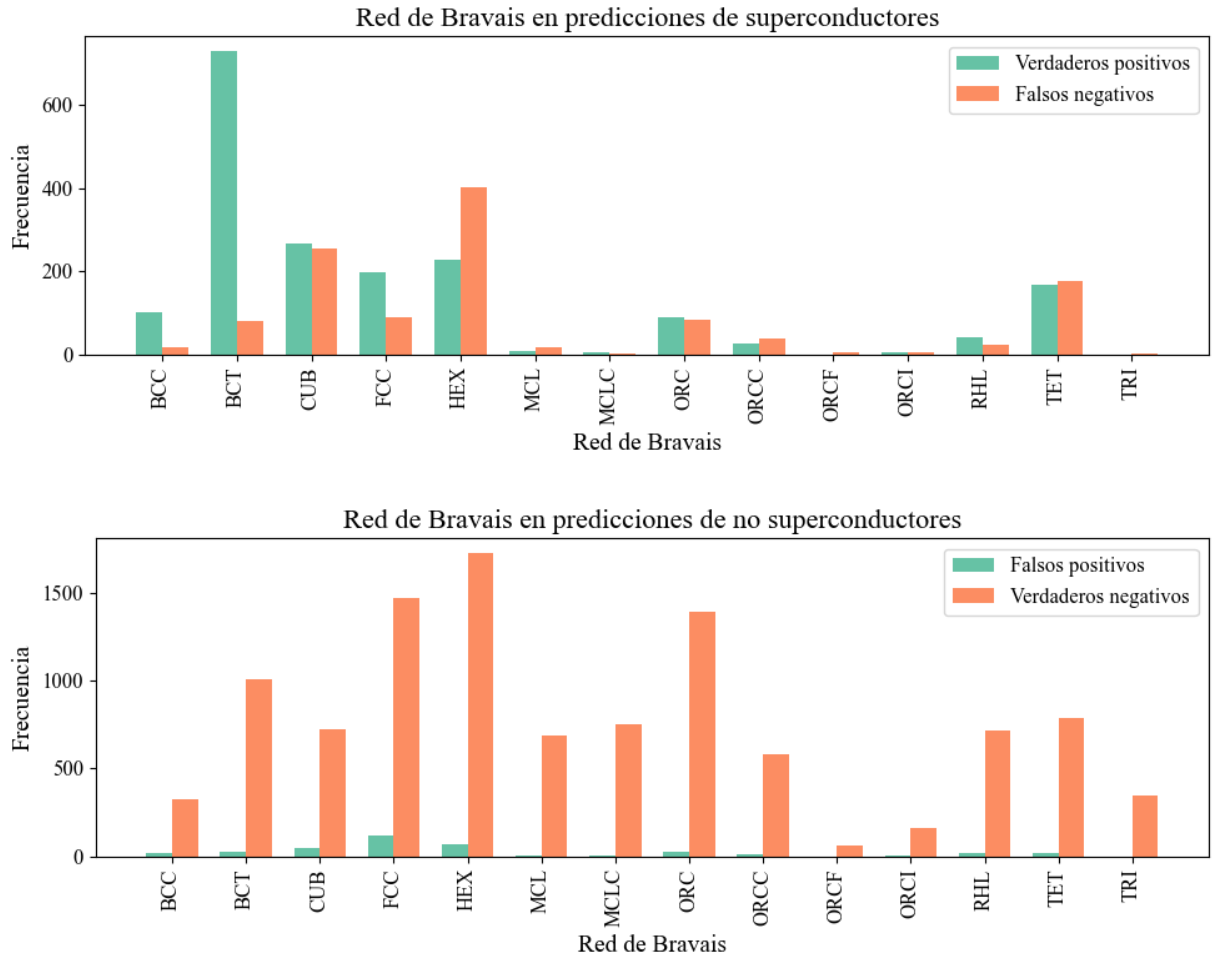

Figura 3.15: Las gráficas de barras muestran la distribución de las redes de Bravais en los diferentes grupos de las predicciones.

En la figura 3.16 se muestra la distribución de la propiedad magnética en los resultados. En la gráfica superior, que muestra las predicciones de superconductores, se observa que los materiales no magnéticos tienen una mayor proporción de verdaderos positivos en compa-

ración con los materiales magnéticos. Sin embargo, no se muestran discrepancias notables entre ambos grupos. La gráfica inferior muestra las predicciones de no superconductores. Aquí, la mayoría de los verdaderos negativos se encuentran en los materiales no magnéticos, lo que sugiere que el modelo es algo más eficaz para identificar correctamente los no superconductores en ausencia de la propiedad magnética. Sin embargo, ambos grupos muestran un buen desempeño general.

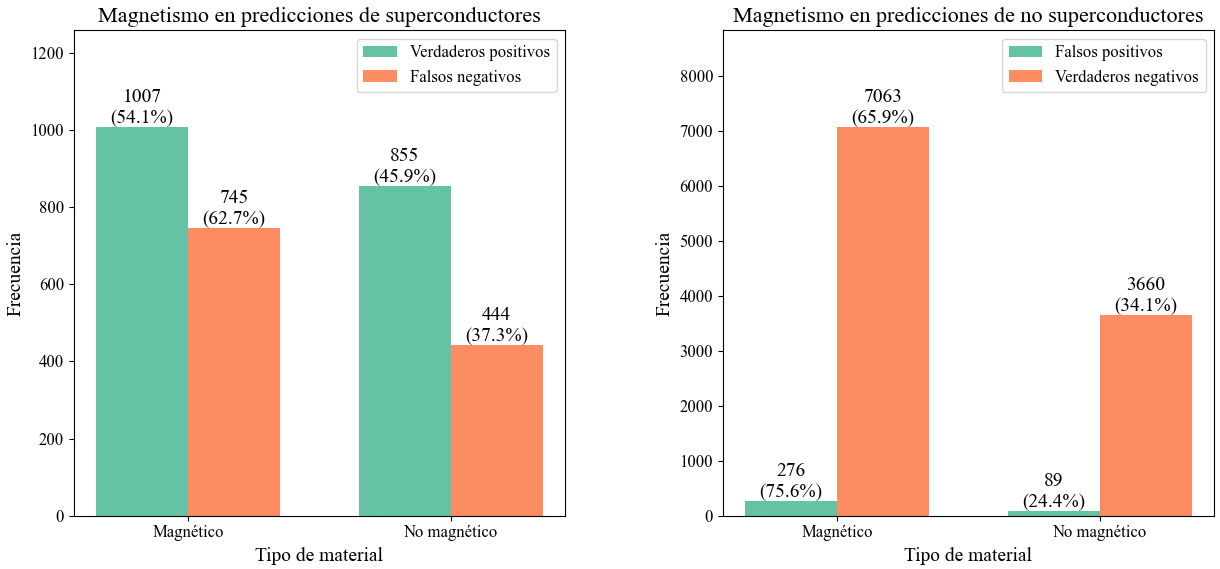

Figura 3.16: Las gráficas de barras muestran la distribución de la propiedad magnética en los diferentes grupos de las predicciones.

En cuanto a la energía de Fermi, que se ilustra en la figura 3.17, se observa un patrón distintivo entre los superconductores y no superconductores. Para los superconductores, las predicciones correctas tienen una distribución dispersa con un ligero pico entorno a los 4eV, mientras que las predicciones incorrectas presentan una distribución también dispersa con un pico más pronunciado entrono a los 7eV. En el caso de los no superconductores, se identifican agrupaciones claras: los verdaderos negativos se concentran alrededor de los 4 eV, y los falsos positivos se sitúan ligeramente por encima de los 5eV, mostrando una mayor agrupación. En general, se nota que las predicciones incorrectas ocurren con mayor frecuencia cuando la energía de Fermi es superior a la media correspondiente a los no superconductores.

Por último, vamos a comparar las representaciones de las medianas de las densidades de

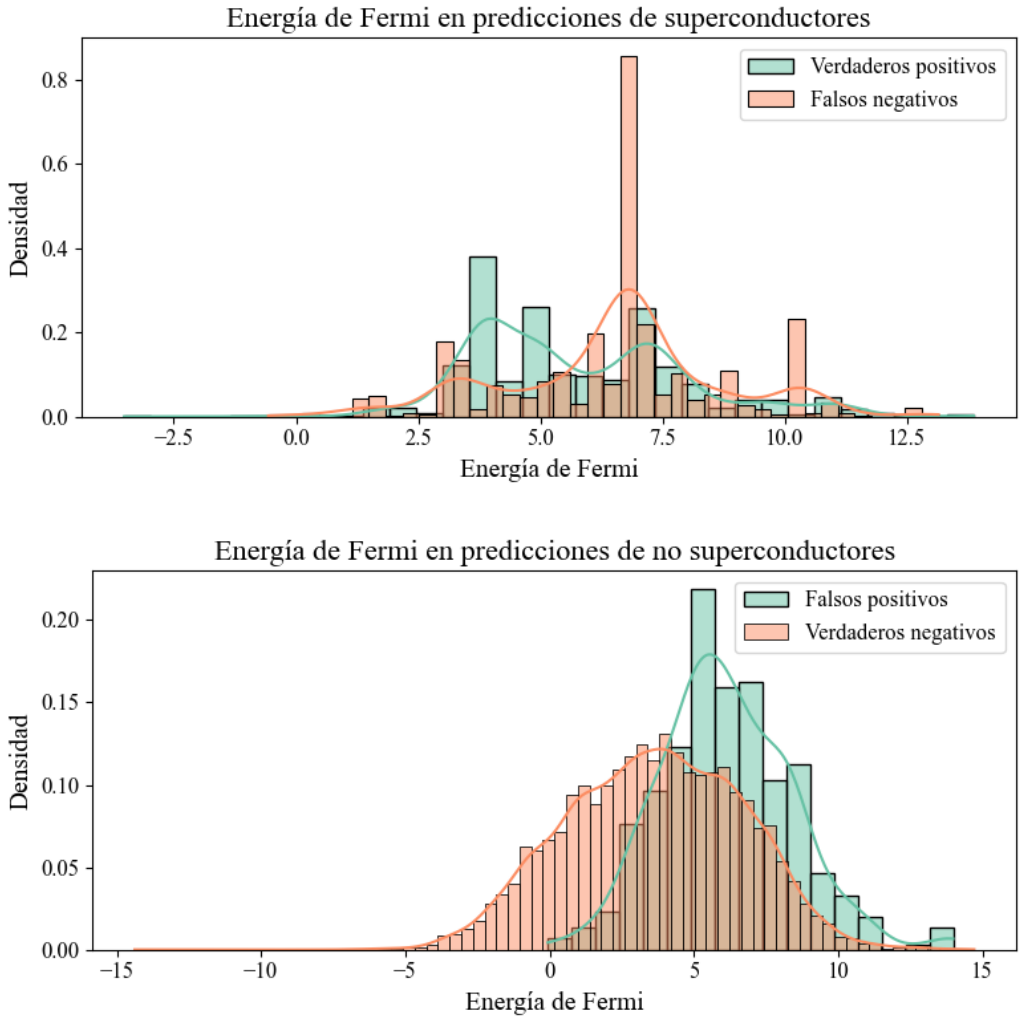

Figura 3.17: Las gráficas de barras muestran la distribución de la energía de Fermi en los diferentes grupos de las predicciones.

estados, que se muestran en la figura 3.18. Se observa que los verdaderos negativos, es decir, los no superconductores clasificados correctamente, muestran un patrón distintivo al resto de grupos, con un pico previo a la energía de Fermi y un pronunciado descenso a 0 en ese punto. Los otros tres grupos muestran un patrón similar entre ellos. Se observa que los falsos positivos tienen una DOS algo menor a la energía de Fermi que los verdaderos positivos, mientras que los falsos negativos tiene una densidad de estados similar. Esta representación nos muestra que el modelo tiene dificultades a la hora de clasificar los materiales cuando la densidad de estados en el entorno de la energía de Fermi es similar a la de los superconductores.

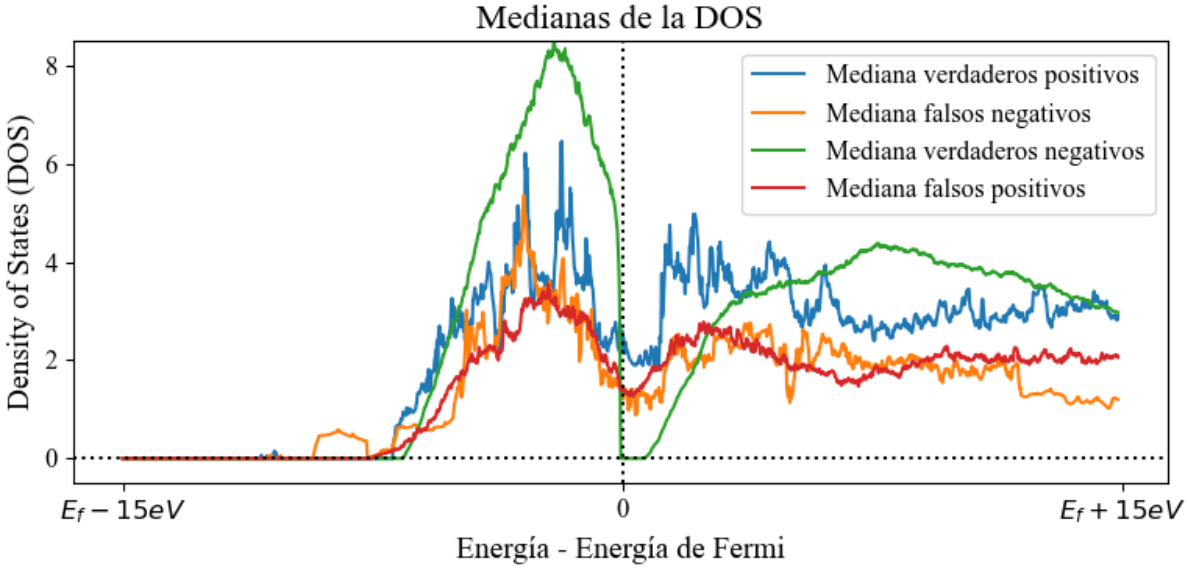

Figura 3.18: Mediana de las curvas de densidad de estados para cada grupo de la matriz de confusión.

En resumen, a pesar de que se identifican algunas tendencias en la densidad de estados y en la energía de Fermi de los diferentes grupos, no se evidencia un patrón claro que sugiera la existencia de múltiples clases. En la sección siguiente, se analizarán las posibles razones por las que el modelo encuentra dificultades para distinguir patrones en la DOS, y se propondrán mejoras y extensiones para abordar estas limitaciones.

3.4. Discusión de resultados y propuestas de ampliación

El modelo de Machine Learning desarrollado en este trabajo, a pesar de alcanzar una precisión notable en la clasificación general de materiales como superconductores o no, presenta dificultades para identificar correctamente los materiales superconductores. Esta dificultad se traduce en una puntuación F1 relativamente baja y en la tendencia a clasificar los materiales como no superconductores, incluso cuando en realidad sí lo son. Este comportamiento nos indica que el modelo, a pesar de su buen desempeño general, tiene dificultades para capturar la complejidad del fenómeno de la superconductividad.

Hay que tener en cuenta que la densidad de estados (DOS) proporcionada al modelo es calculada mediante aproximaciones teóricas del estado fundamental, las cuales no tienen en cuenta la interacción electrón-electrón. Por lo tanto, no se refleja de forma explícita la formación de los pares de Cooper. Pese a ello, la densidad de estados electrónica proporciona información sobre la disponibilidad de estados electrónicos a diferentes energías, que juega un papel esencial en la formación de estos pares. Esto abre la posibilidad a la existencia de patrones ocultos que podrían servir para diferenciar los materiales. Sin embargo, dados los resultados obtenidos, no parece ser suficiente.

Por otro lado, es importante recordar que es un fenómeno complejo que no se limita únicamente a la DOS. Existen otros factores cruciales que no han sido considerados en el modelo desarrollado. La interacción electrón-fonón, por ejemplo, es fundamental para la formación de pares de Cooper. Los fonones, vibraciones de la red cristalina, pueden mediar la atracción entre los electrones. La fuerza de esta interacción determina la temperatura crítica del material, por encima de la cual la superconductividad desaparece. La incorporación de la interacción electrón-fonón como variable en el modelo podría mejorar significativamente su capacidad para identificar materiales superconductores.

La estructura cristalina del material, representada en este caso por la red de Bravais, también juega un papel crucial. La red de Bravais describe la disposición tridimensional de los átomos en un material cristalino. Esta disposición afecta de manera significativa tanto a la interacción electrón-fonón como a la densidad de estados. Ciertas estructuras pueden facilitar la interacción electrón-fonón, lo que conduce a una mayor fuerza de atracción entre los electrones y, por lo tanto, a una mayor probabilidad de formación de pares de Cooper. Otras estructuras, en cambio, pueden dificultar la interacción electrón-fonón, reduciendo la fuerza de atracción y disminuyendo la posibilidad de superconductividad. Por otro lado, la DOS también se ve afectada. La estructura cristalina define la distribución de los átomos y, por lo tanto, la distribución de los estados electrónicos en el material. Ciertas estructuras cristalinas pueden concentrar la DOS en la energía de Fermi, aumentando la probabilidad de superconductividad. Otras estructuras, en cambio, pueden dispersarla en un rango más amplio de energías, dificultando el fenómeno.

La presencia de impurezas también puede afectar la superconductividad. Esto es debido a que pueden actuar como centros de dispersión, interrumpiendo el movimiento de los electrones y reduciendo la formación de pares de Cooper.

Por último, la composición química de los materiales también es un factor fundamental que afecta la superconductividad. La naturaleza de los elementos que componen un material determina la electronegatividad, el tipo de enlace químico y el número de electrones de valencia. Por ejemplo, la presencia de elementos con electrones de valencia desapareados puede favorecer la formación de pares de Cooper, mientras que elementos con estructuras electrónicas llenas pueden dificultarla. La influencia de la composición química es algo que se puede observar en la figura 3.8, donde se ve como hay ciertos elementos que aparecen con mayor frecuencia en los superconductores de la base de datos. La inclusión de información sobre las especies y composición química podría mejorar los resultados obtenidos.

Por otra parte, se podría realizar la implementación de modelos más avanzados y complejos. Por ejemplo, podría considerarse el empleo simultáneo de múltiples clasificadores, cada uno entrenado con conjuntos de datos diferentes sobre los materiales, cuyas predicciones se combinarían. Otra posibilidad sería plausible la aplicación de una Red Neuronal Recurrente (RNN), debido a su capacidad para modelar dependencias secuenciales en datos, en este caso las variaciones de la DOS en función de la energía. Los modelos empleados, como *XGBoost*, demuestran eficacia en la gestión de datos estructurados y variables independientes, mientras que las RNN destacan por su capacidad para capturar patrones secuenciales y modelar datos con dependencias temporales o posicionales, como es el caso de la densidad de estados.

En resumen, el modelo desarrollado en este estudio no considera propiedades de los materiales adicionales, al centrarse en la DOS. Esta simplificación limita su capacidad para abordar la complejidad del fenómeno de la superconductividad. Por otra parte, sería necesario emplear modelos más sofisticados para lograr predicciones más precisas.

Capítulo 4

Conclusiones

Este trabajo ha explorado la posibilidad de predecir la superconductividad de materiales cristalinos utilizando técnicas de aprendizaje automático (machine learning), con la densidad de estados (DOS) como principal descriptor. La investigación se basa en la premisa de que la DOS, una medida que describe la distribución de energías disponibles para los electrones, podría estar relacionada con la capacidad de un material para superconducir. Para ello, se han recogido datos de dos bases de datos: AFLOW, que contiene información sobre la DOS de un gran número de materiales, y 3DSC, que proporciona información sobre materiales superconductores.

Se ha realizado un análisis exploratorio de los datos para identificar patrones y relaciones entre la DOS y la superconductividad. Se observa una diferencia significativa en la mediana de la DOS en la energía de Fermi entre materiales superconductores y no superconductores, ejemplificada en estructuras cristalinas como la hexagonal (HEX) y la ortorrómbica simple (ORC). Sin embargo, la baja correlación general entre las variables y la variable objetivo sugiere que el modelo podría tener dificultades para identificar patrones fuertes y, por lo tanto, una precisión limitada.

Para abordar la clasificación binaria entre materiales superconductores y no superconductores, se han implementado y evaluado diversos modelos de aprendizaje automático. Se han explorado diferentes técnicas de remuestreo, incluyendo *undersampling* aleatorio mediante

RandomUnderSampler y *oversampling* mediante *RandomOverSampler* y *SMOTE*, para lidiar con el desequilibrio en los datos, donde la clase de superconductores representa solo un pequeño porcentaje del conjunto de datos.

La reducción de la dimensionalidad del conjunto de datos a través del análisis de componentes principales (PCA) también ha sido un elemento crucial en la búsqueda de modelos más efectivos. Se han realizado pruebas con diferentes números de componentes principales para determinar el impacto en el rendimiento del modelo. Se observa que, en general, un número mayor de componentes principales conduce a mejores resultados en las métricas de prueba, pero un número excesivo puede introducir ruido en los datos, afectando negativamente la precisión.

Los resultados de la evaluación de los modelos han demostrado que, aunque es posible lograr una alta exactitud en la clasificación de materiales, el modelo final presenta un rendimiento moderado en términos de puntuación F1, mostrando una tendencia a clasificar incorrectamente los materiales como no superconductores. Esta tendencia, observable en la matriz de confusión, podría explicarse por la baja correlación general entre las variables y la variable objetivo, así como por la distribución del conjunto de datos, donde la clase minoritaria (superconductores) tiene una representación significativamente menor. El análisis de los valores SHAP ha revelado la importancia de la energía de Fermi en la predicción de la superconductividad, corroborando la observación de la matriz de correlaciones. Sin embargo, el impacto de otras características es menos claro.

Estos resultados, aunque no representan un éxito total en la predicción de la superconductividad, demuestran el potencial de las técnicas de aprendizaje automático en la investigación de materiales. La combinación de técnicas de remuestreo, reducción de dimensionalidad y el análisis componentes principales ha permitido obtener un modelo con un rendimiento aceptable, pero aún queda mucho trabajo por hacer.

De cara a futuras investigaciones, se puede explorar la inclusión de otras características, mencionadas en la discusión de resultados, como la composición química o la interacción electrón-fonón. Además, se puede explorar la posibilidad de implementar modelos más complejos y sofisticados que permitan identificar relaciones no lineales entre las variables.

Además, se puede ampliar el conjunto de datos incluyendo información de otras fuentes para obtener una clasificación de no superconductores más limpia y mejorar la generalización del modelo. Actualmente, se considera que cualquier material no listado en la base de datos 3DSC como superconductor es no superconductor. Este enfoque podría ser impreciso, ya que existen materiales que son superconductores pero que no están necesariamente listados en bases de datos específicas de superconductores. Incluir información de bases de datos que clasifiquen materiales como conductores o aislantes permitiría una clasificación más precisa de los no superconductores, mejorando la calidad del conjunto de datos y, por lo tanto, la generalización del modelo.

Este último punto se concluye como un elemento crucial, ya que la calidad de los datos iniciales es un elemento fundamental en los resultados obtenidos a partir de técnicas de aprendizaje automático. Un conjunto de datos de mayor calidad, con una clasificación más precisa de los no superconductores y la inclusión de información relevante como la composición química, permitiría al modelo aprender patrones más robustos y generalizar mejor a nuevos datos.7

Bibliografía

[1] Ashcroft, N. W., & Mermin, N. D. (1976). *Solid State Physics*. Saunders College Publishing.

[2] Kittel, C. (2004). *Introduction to Solid State Physics*. Wiley.

[3] Bardeen, J., Cooper, L. N., & Schrieffer, J. R. (1957). Theory of Superconductivity. *Physical Review*, 108(5), 1175. https://doi.org/10.1103/PhysRev.108.1175

[4] Tinkham, M. (1996). *Introduction to Superconductivity*. McGraw-Hill Science, Engineering & Mathematics.

[5] Lee, P. A., Nagaosa, N., & Wen, X. (2004). Doping a Mott Insulator: Physics of High Temperature Superconductivity. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.cond-mat/0410445

[6] IBM. (2023, 14 noviembre). Aprendizaje automático. IBM Cloud. https://www.ibm.com/es-es/topics/machine-learning

[7] Google. (s.f.). *Machine Learning*. https://developers.google.com/machine-learning

[8] Google. (s.f.). *Skillboosting*. https://skillboosting.withgoogle.com/

[9] Kaggle: Your Machine Learning and Data Science Community. (s. f.). https://kaggle.com/

[10] Martin, R. C. (2009). *Clean code: A Handbook of Agile Software Craftsmanship*. Pearson Education.

[11] Olawanle, J. (2023, 10 abril). *SOLID Principles for Programming and Software Design.* freeCodeCamp.org. https://www.freecodecamp.org/news/solid-principles-for-programming-and-software-design/

[12] Aflow - Automatic FLOW for Materials Discovery. (s. f.). Copyright © 2015. https://aflowlib.org/

[13] Curtarolo, S., Setyawan, W., Wang, S., Xue, J., Yang, K., Taylor, R. H., Nelson, L. J., Hart, G. L., Sanvito, S., Buongiorno-Nardelli, M., Mingo, N., & Levy, O. (2012). AFLOWLIB.ORG: A distributed materials properties repository from high-throughput ab initio calculations. *Computational Materials Science*, 58, 227-235. https://doi.org/10.1016/j.commatsci.2012.02.002

[14] Sommer, T., Willa, R., Schmalian, J., & Friederich, P. (2022, 12 diciembre). *3DSC - A New Dataset of Superconductors Including Crystal Structures.* arXiv.org. https://arxiv.org/abs/2212.06071

[15] Aimat-Lab. (s.f.). *GitHub - aimat-lab/3DSC: Repo for the paper publishing the superconductor database with 3D crystal structures.* GitHub. https://github.com/aimat-lab/3DSC

[16] Materials Database Group, National Institute for Materials Science. (2022). Metallic, SuperCon, Superconductors, Oxide, Organic, Tc. National Institute for Materials Science. Creative Commons BY Attribution 4.0 International. Identifier: https://doi.org/10.48505/nims.3739

[17] Mujica-Schwahn, N. (2017). Position Dependence of High Efficiency Single Photon Detectors: A Route to Better Understanding of Transition Edge Sensors. *Journal of Low Temperature Physics*, 185(3-4), 1042-1047. [DOI: 10.1007/s10909-016-1192-8]

[18] Colaboradores de Wikipedia. (s.f.). Bravais lattice. In Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Bravais_lattice

[19] Colaboradores de Wikipedia. (s.f.). *Archivo:Magnetización según el tipo de superconductor.png - Wikipedia, la enciclopedia libre.* https://es.wikipedia.org/wiki/Archivo:Magnetización_según_el_tipo_de_superconductor.png

[20] Physics Girl. (s.f.). *Archivo: lattice point definition.* Physics Girl. https://physicsgirl.in/tag/lattice-point-definition/

[21] Instituto de Ingeniería del Conocimiento (IIC). (2023). Aprendizaje profundo por refuerzo. Recuperado en Abril de 2024, https://www.iic.uam.es/aprendizaje-profundo-por-refuerzo/

[22] Chacon, R. (2024, 14 abril). Validación cruzada. Interactive Chaos. Recuperado en Abril de 2024, https://interactivechaos.com/es/wiki/validacion-cruzada

[23] GeeksforGeeks. (2023, 11 septiembre). Using Learning Curves in Machine Learning. Recuperado en Abril de 2024, https://www.geeksforgeeks.org/using-learning-curves-ml/

[24] Wikipedia. (2024, 14 abril). *Banda de valencia.* https://es.wikipedia.org/w/index.php?title=Banda_de_valencia&oldid=140865580

[25] Quimitube. (2023, 9 noviembre). *Enlace metálico: Teoría de bandas.* Quimitube. https://www.quimitube.com/videos/enlace-metalico-teoria-de-bandas/

[26] Physics Stack Exchange. (2014, 28 enero). *Density of states (DOS) to energy graph.* https://physics.stackexchange.com/questions/554741/density-of-states-dos-to-energy-graph