FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2023/2024

Trabajo de Fin de Máster

TITULO: Clasificación de imágenes aplicada a géneros musicales

***Alumno*: Pablo Lozano Martín**

***Tutor*: Pablo Arcadio Flores Vidal**

Septiembre de 2024

UNIVERSIDAD COMPLUTENSE
MADRID

Agradecimientos

Quisiera dar las gracias a Pablo, por su paciencia y dedicación a lo largo de todo el proyecto.

A mi familia, por darme siempre fuerza. A mis amigos, que continuamente confiaron en mí. A mis profesores, por demostrarme la belleza en la estadística. Y a Cristina, por demostrarme la belleza en todo lo demás.

Resumen

Clasificación de imágenes aplicada a géneros musicales

En los últimos años, la inteligencia artificial ha avanzado considerablemente gracias al aumento significativo en la capacidad de procesamiento de los ordenadores. En esta situación, se han creado varios métodos que posibilitan que la propia máquina aprenda a prever o categorizar, a partir de ciertos datos específicos. El propósito de este estudio es emplear técnicas de aprendizaje automático para analizar y estudiar la relación entre los estilos visuales de las portadas de álbumes y el género musical. Con ese fin, se recolectarán las imágenes y los datos de la plataforma de Amazon Music y del conjunto de datos Million Song Dataset.

Con la variedad de datos disponibles, se llevará a cabo el entrenamiento de varios modelos de inteligencia artificial, para luego analizar los resultados obtenidos y las causas que explican cada comportamiento. Este proyecto se centra en comprender las relaciones entre variables, en lugar de buscar un rendimiento óptimo de los modelos.

Palabras clave

Inteligencia artificial, aprendizaje automático, aprendizaje profundo, clasificación de imágenes, visión artificial, correlación.

Código

El código relativo a este proyecto está disponible en el siguiente repositorio de GitHub:

github.com/pablo-lozano-martin/album-artwork-classification

Abstract

Image classification applied to musical genres

In recent years, artificial intelligence has advanced significantly due to the substantial increase in computer processing power. In this context, various methods have been developed that enable machines to learn to predict or categorize based on specific data.

The aim of this study is to employ machine learning techniques to analyze and investigate the relationship between album cover visual styles and music genres. To this end, images and data will be collected from the Amazon Music platform and the Million Song Dataset.

With the variety of available data, several artificial intelligence models will be trained, and the results will be analyzed to understand the behavior and the reasons behind it. This project focuses on understanding the relationships between variables, rather than seeking optimal model performance.

Keywords

Artificial intelligence, machine learning, deep learning, image classification, computer vision, correlation

Code

The code related to this project is available in the following GitHub repository:

github.com/pablo-lozano-martin/album-artwork-classification

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

La inteligencia artificial (IA) es una de las áreas más fascinantes y dinámicas de la ciencia contemporánea, pero sus raíces se remontan a mediados del siglo XX, un período marcado por avances tecnológicos y teóricos que sentaron las bases de lo que hoy conocemos como IA.

Uno de los pioneros en este campo fue Alan Turing, un matemático británico cuyo trabajo revolucionó nuestra comprensión de la computación. En 1950, Turing planteó una pregunta fundamental: "¿Pueden las máquinas pensar?", lo que dio origen al "Test de Turing", un criterio para determinar si una máquina puede exhibir comportamiento inteligente indistinguible del de un ser humano (Turing (1950)). Este cuestionamiento abrió las puertas al desarrollo de la inteligencia artificial, proponiendo que una máquina podría, en teoría, imitar los procesos mentales humanos.

Paralelamente, el desarrollo del transistor en 1947 por John Bardeen, Walter Brattain y William Shockley marcó un hito en la evolución de la electrónica (Bardeen (1949)). Este dispositivo, que permite controlar el flujo de corriente en circuitos electrónicos, fue crucial para la miniaturización y eficiencia de las computadoras, lo que a su vez hizo posible el procesamiento de grandes cantidades de datos necesarios para las primeras investigaciones en inteligencia artificial.

Con el paso de las décadas, la tecnología de los transistores se integró en circuitos integrados cada vez más complejos, lo que fue encapsulado en la Ley de Moore, formulada por Gordon Moore en 1965 (Moore (1965)). Esta ley predice que el número de transistores en un microprocesador se duplicará aproximadamente cada dos años, lo que implica un crecimiento exponencial en la capacidad de procesamiento de las computadoras. Este crecimiento ha sido un factor clave en el avance de la inteligencia artificial, permitiendo que algoritmos más complejos sean ejecutados en tiempos razonables.

Otro pilar fundamental en el desarrollo de la IA fue la creación y expansión de Internet. En los últimos años, se sabe que alrededor del 64 % de las personas del mundo usa Internet (Datareportal (2023)). Esta red global ha facilitado el acceso a grandes cantidades de datos y ha permitido la colaboración entre investigadores de todo el mundo, acelerando el desarrollo de nuevas técnicas y aplicaciones de inteligencia artificial.

Dentro de la IA, uno de los campos más prometedores es el aprendizaje automático o *machine learning*, que permite a las máquinas mejorar su rendimiento en tareas específicas mediante la experiencia. A través de algoritmos que aprenden de los datos, el aprendizaje automático ha impulsado avances significativos en áreas como la visión por computadora, el procesamiento del lenguaje natural y la robótica, acercándonos cada vez más a la posibilidad de máquinas verdaderamente inteligentes.

El aprendizaje automático, relacionado con la estadística, se basa en el análisis de datos. Existen diversos tipos de algoritmos, como el aprendizaje supervisado, no supervisado y por refuerzo.

En este estudio emplearemos el aprendizaje supervisado, que permite clasificar imágenes mediante etiquetas, en este caso, portadas de álbumes de música.

1.1. Arte visual en las portadas de álbumes

A lo largo de la historia, el arte ha evolucionado con las distintas corrientes artísticas, incluyendo las portadas de discos. En el siglo pasado, surgieron géneros musicales como jazz, rock, pop, hip-hop y electrónica, acompañados por estilos artísticos visuales que reflejaron estos géneros, como el arte psicodélico para el rock en los años 60, el minimalismo para el techno en los 90 y el graffiti para el hip-hop.

Cada década ha sido testigo de la dominancia de diferentes géneros: el rock en los 70, el pop en los 80, el grunge en los 90 y el reggaetón en los 2000. Estos géneros han evolucionado y adoptado influencias de épocas pasadas, reflejando cambios tanto en la música como en el arte visual de las portadas de álbumes, que a menudo capturan la esencia cultural del tiempo.

Para clasificar las portadas de álbumes por género y entender estas relaciones, utilizamos modelos de aprendizaje automático. Aunque las redes neuronales profundas suelen ser más precisas, su capacidad para explicar sus decisiones es limitada. Por eso, también recurrimos a modelos más interpretables, como Random Forest. Estos modelos nos permiten obtener información sobre qué elementos visuales están asociados con ciertos géneros musicales, ayudándonos a comprender las conexiones culturales y artísticas entre el arte visual y la música.

1.2. Objetivos

El principal objetivo de este trabajo es el de la aplicación de distintos modelos de inteligencia artificial para tratar de clasificar portadas de LPs, comparando cuales lo hacen de forma más adecuada utilizando diversas medidas para medir la precisión de la clasificación. El acometer el objetivo principal requiere de la consecución de una serie de pasos que pueden considerarse los objetivos secundarios. Estos son:

- Obtener, y procesar los datos de las imágenes y etiquetas.

- Efectuar un análisis completo del dataset obtenido.
- Entrenar los modelos elegidos.
- Analizar los resultados obtenidos por los distintos modelos, usando para ello distintas métricas.

1.3. Plan de trabajo

El proyecto ha sido desarrollado siguiendo una metodología ágil, lo que ha permitido adaptarse rápidamente a los cambios y necesidades del trabajo. Desde el inicio, se priorizó la recopilación de datos, creación del código y la obtención de resultados, asegurando que estos elementos clave estuvieran listos en las primeras etapas. A lo largo del proceso, se implementaron mejoras continuas, lo que permitió optimizar el flujo de trabajo y la calidad del producto final. Las tareas se completaron en un orden planificado, garantizando que cada fase se construyera sobre la anterior de manera eficiente. En particular, se dio prioridad al procesamiento de datos y al entrenamiento de modelos, sentando una base sólida antes de avanzar hacia el análisis de resultados.

Las tareas se han planificado temporalmente, como se detalla en la Figura 1.1.

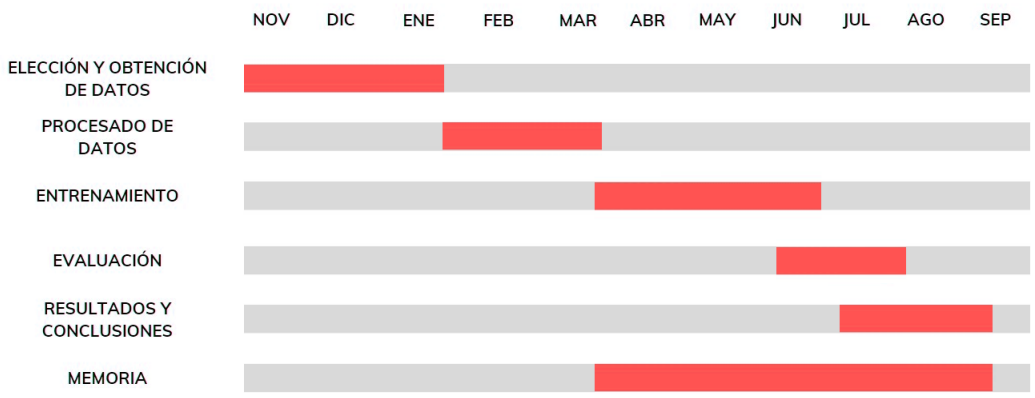

Figura 1.1: Plan de proyecto

Capítulo 2

Herramientas

En el presente capítulo se describen las herramientas empleadas para llevar a cabo tanto el preprocesado de los datos como el entrenamiento y evaluación de los modelos.

2.1. Python

Python (Van Rossum y Drake (2009)) es un lenguaje de programación muy popular en el campo de la inteligencia artificial, ofreciendo diversas bibliotecas que incluyen implementaciones de los modelos que se evalúan. Como un lenguaje de alto nivel, Python simplifica la complejidad de los modelos, lo que nos permite concentrarnos en aspectos clave como la preparación de datos, el ajuste de parámetros importantes y la evaluación de resultados. Además, Python facilita la gestión y manipulación de conjuntos de datos.

2.2. Git

Git (Git (2023)) es una herramienta de control de versiones ampliamente utilizada en el desarrollo de software. Permite a los desarrolladores gestionar y registrar los cambios en el código a lo largo del tiempo, facilitando el seguimiento de modificaciones en la implementación del preprocesamiento y el entrenamiento de los modelos.

2.3. GitHub

GitHub (Github (2023)) es una plataforma colaborativa que se basa en el sistema de control de versiones Git. Ha sido empleada para almacenar y compartir el código desarrollado para este proyecto. Además, GitHub facilita la difusión y colaboración en el código, permitiendo un acceso sencillo para otros interesados.

2.4. Jupyter Notebook

Jupyter Notebook (Kluyver et al. (2016)) es una aplicación que permite crear documentos que combinan secciones de código Python ejecutables con texto explicativo. Esta herramienta es fundamental para el desarrollo de este proyecto, ya que nos ha facilitado la implementación del código necesario para el preprocesamiento de datos y la manipulación de los modelos utilizados. Además, Jupyter Notebook permite integrar anotaciones en formato Markdown, lo que ayuda a mejorar la comprensión y documentación del código.

2.5. Librerías de Python

Para llevar a cabo este proyecto, se han utilizado varias librerías de Python que facilitan tanto la preparación como la implementación de los modelos. Estas librerías proporcionan herramientas para el preprocesamiento de datos y la ejecución de modelos, cada una con funciones específicas que contribuyen al desarrollo del proyecto. A continuación, se detalla la utilidad de cada librería en el contexto de este trabajo:

- Pandas (Pandas (2023)): Esta librería es esencial para el análisis y la manipulación de datos. Se ha utilizado para gestionar y transformar los conjuntos de datos necesarios para el entrenamiento de los modelos.
- Numpy (Harris et al. (2020)): Proporciona funciones para el manejo de vectores y matrices, y está integrada en otras librerías como Pandas y Matplotlib. Se utiliza para realizar operaciones matemáticas y científicas eficientes.
- Matplotlib (Hunter (2007)): Permite la creación de gráficos bidimensionales. Ha sido empleada para visualizar los datos y las predicciones generadas por los modelos.
- Scikit-learn (Pedregosa et al. (2011)): Esta librería es clave para el entrenamiento de modelos de aprendizaje automático. Se ha utilizado para implementar el modelo de Random Forest, facilitando su entrenamiento y evaluación.
- TensorFlow (Abadi et al. (2015)): Es una librería para el entrenamiento de redes neuronales, diseñada para simplificar y organizar este proceso.
- Keras (Chollet (2024)): Biblioteca de código abierto que proporciona una interfaz de alto nivel para el desarrollo y entrenamiento de modelos de aprendizaje profundo, funcionando sobre TensorFlow y otras librerías similares.
- Joblib (Joblib (2023)): Facilita la serialización y el almacenamiento de funciones de Python. Se ha utilizado para guardar los resultados del entrenamiento de los modelos, permitiendo su posterior recuperación y uso.

Capítulo 3

Preprocesado de Datos

El preprocesamiento de datos es conocida como una de las etapas más importantes. Esta fase, se llevan a cabo varias operaciones:

- **Limpieza de Datos:** Es el primer paso en el preprocesamiento y se enfoca en identificar y corregir errores, inconsistencias o valores atípicos en los datos. Este proceso incluye la eliminación de duplicados, corrección de errores tipográficos, manejo de outliers, y la corrección de formatos inconsistentes. La calidad de los datos es esencial, ya que datos sucios pueden llevar a modelos inexactos o ineficientes. El objetivo principal es asegurar que los datos sean precisos y fiables antes de proceder a las siguientes etapas del análisis.

- **Normalización:** se refiere al proceso de escalar los valores de las características para que tengan una escala comparable. La normalización asegura que las características se distribuyan de forma parecida, mejorando la estabilidad y el rendimiento del modelo.

- **Selección y Transformación de Características:** Implica identificar y retener las variables más relevantes para el modelo, eliminando aquellas que son redundantes o irrelevantes. Esto ayuda a reducir la dimensionalidad del conjunto de datos, lo que puede mejorar la eficiencia y la interpretabilidad del modelo. La transformación de características, por otro lado, implica modificar las variables existentes para mejorar la representación del modelo.

- **Manejo de Datos Nulos:** Los valores faltantes pueden distorsionar los resultados del modelo, por lo que existen estrategias como la eliminación de filas o columnas con valores nulos, la imputación de valores faltantes utilizando la media, la mediana, o métodos más avanzados como la imputación múltiple o el uso de modelos predictivos. La elección del método depende del porcentaje de datos faltantes y de la importancia de las variables involucradas. Un manejo adecuado de los datos nulos garantiza que el modelo no se vea sesgado por la ausencia de información.

El preprocesamiento de datos es fundamental para asegurar que los datos sean de alta calidad, estén bien formateados y sean relevantes para el análisis. Este proceso incluye la obtención de datos, su limpieza para eliminar errores y redundancias, y el preprocesamiento para transformar los datos en un formato adecuado para el modelo. Al realizar un preprocesamiento exhaustivo, se optimiza la capacidad del modelo para aprender y generalizar a partir de los datos proporcionados.

A continuación comprobaremos el proceso de obtención, limpieza y preprocesamiento de los datos, además de la preparación de los conjuntos de datos que se utilizarán en el entrenamiento y evaluación de los diferentes modelos.

3.1. Elección de los datos

Para este proyecto se necesitarán varios tipos de datos. Primero, las imágenes de las carátulas de los álbumes son esenciales, ya que contienen información visual clave para la clasificación. Además, las etiquetas que indican el género o los géneros musicales asociados a cada álbum son fundamentales para el entrenamiento supervisado de los modelos. Estas etiquetas permiten que los modelos aprendan a asociar características visuales específicas con géneros musicales particulares.

Otra información útil podría incluir el texto de los títulos de los álbumes y los nombres de los artistas, que podrían proporcionar contexto adicional o características auxiliares para mejorar la precisión de la clasificación. Sin embargo, en este proyecto específico, se ha decidido priorizar el uso de imágenes y etiquetas de género musical como los datos principales para entrenar los modelos, con el objetivo de desarrollar un sistema de clasificación eficiente basado en las características visuales distintivas de las carátulas de los álbumes.

3.2. Obtención de los datos

Para la realización de este trabajo se ha utilizado el conjunto de datos MuMu (Oramas S. (2017)), una colección multimodal de música con anotaciones de géneros de múltiples etiquetas. MuMu resulta de la combinación de dos fuentes principales de datos: el conjunto de datos de Reseñas de Amazon y el Million Song Dataset (MSD). Cada uno de estos conjuntos proporciona información valiosa y complementaria que, al ser combinada, enriquece significativamente el análisis y experimentación.

El conjunto de datos de Reseñas de Amazon incluye millones de reseñas de álbumes realizadas por clientes, junto con metadatos detallados sobre los álbumes recopilados desde la propia web. Estas reseñas aportan una rica fuente de información textual y valoraciones de usuarios que son cruciales para comprender la percepción del público sobre los diferentes géneros musicales. Por otro lado, el MSD ofrece metadatos y características de audio precomputadas para un millón de canciones, lo cual permite analizar

aspectos técnicos y objetivos de la música que se están evaluando.

Para integrar la información de ambos conjuntos de datos, se emplea MusicBrainz, una base de datos comunitaria que proporciona un estándar para la identificación de música. Este mapeo cuidadoso entre las bases de datos de Amazon, MusicBrainz y MSD produce un conjunto final de 147,295 canciones pertenecientes a 31,471 álbumes. De estos álbumes, se tienen 447,583 reseñas de clientes, lo que proporciona una extensa fuente de datos textuales y contextuales.

En nuestro caso, el uso del conjunto de datos MuMu en este trabajo es particularmente interesante por tres razones principales:

- **Anotaciones de género de múltiples etiquetas:** Anotaciones de género de múltiples etiquetas: MuMu no solo clasifica la música en géneros únicos, sino que también admite la clasificación en múltiples géneros por canción o álbum. Esta capacidad refleja de manera más precisa la realidad del ámbito musical, donde las canciones a menudo cruzan los límites de géneros y no se pueden encasillar en una sola categoría.

- **Tamaño y diversidad del conjunto de datos:** Con 147,295 canciones y 31,471 álbumes, MuMu ofrece un conjunto de datos lo suficientemente grande y variado para entrenar y evaluar modelos de clasificación de géneros musicales con robustez y generalización. La gran cantidad de reseñas de clientes (447,583) también proporciona una rica fuente de datos textuales que pueden ser utilizados para análisis de sentimiento, extracción de características textuales y otras técnicas de procesamiento de lenguaje natural.

- **Evaluación y benchmarking:** MuMu ha sido utilizado en experimentos previos de clasificación de géneros musicales de múltiples etiquetas, lo que permite comparar y evaluar nuevos enfoques y modelos con referencia a trabajos anteriores. Esta capacidad de benchmarking es crucial para medir el progreso y la efectividad de las nuevas técnicas propuestas en este trabajo.

Aunque en este proyecto no vamos a hacer uso de la multimodalidad del dataset, también debemos destacar que es algo muy interesante: la combinación de la información textual de las reseñas de Amazon con las características de audio del MSD permite un análisis que abarca tanto la percepción subjetiva de los oyentes como las propiedades objetivas de la música.

3.3. Procesado de los datos

Como bien es sabido, el procesado de los datos es, si no el paso más importante, uno de los más importantes del aprendizaje automático.

Esta etapa garantiza la calidad y consistencia de los datos utilizados y, en este caso, se han llevado a cabo varias tareas de limpieza y simplificación sobre el conjunto de datos MuMu para asegurar que esté en óptimas condiciones para el análisis posterior.

Primero, se eliminaron los registros que no contenían identificadores de Amazon o anotaciones de géneros, ya que estos son campos esenciales para el análisis. Esto asegura que cada registro tenga la información mínima necesaria para ser útil. A continuación, se redujo el conjunto de datos a solamente dos campos: *genres* y *amazon_id*. De esta forma, simplemente se mapea una imagen y sus géneros mediante este útimo campo.

Se verificó la existencia de archivos de imagen asociados a cada canción. Los registros sin imágenes o con imágenes en formatos incorrectos fueron eliminados. Además, se comprobó la inexistencia de duplicados para eliminar toda la información redundante o innecesaria.

Para cada carátula, se contó la cantidad de géneros anotados y se redujo a solamente tres géneros principales como máximo. Esta reducción se realiza para simplificar la fase de modelado sin sacrificar el buen funcionamiento de los modelos, ya que se conservan los géneros más representativos en lugar de seleccionar tres géneros al azar. Este enfoque facilita el análisis y mejora la interpretabilidad de los resultados, al conservar las etiquetas más relevantes.

Figura 3.1: Ejemplos de álbumes y sus géneros musicales

Las etiquetas de género fueron analizadas para identificar las 20 más frecuentes,

descartando aquellas con menos de 1000 ocurrencias. Por lo tanto, se eliminaron los registros que, después del filtrado de etiquetas poco frecuentes, quedaron sin géneros asignados. Este filtrado elimina el ruido y se enfoca en las etiquetas más representativas y comunes, haciendo el análisis más manejable y relevante.

Finalmente, para probar el modelo en un enfoque de clasificación multi-etiqueta (*multi-label*), se realizó una normalización de las etiquetas utilizando el MultiLabelBinarizer (Pedregosa (2011)) (MLB). Esta técnica es especialmente útil cuando un registro puede pertenecer a múltiples géneros simultáneamente, algo común en la música. El *MultiLabelBinarizer* transforma las etiquetas en vectores binarios de 0 y 1, donde cada posición del vector representa la presencia o ausencia de un género específico. Por ejemplo, si un registro pertenece a los géneros Rock y Pop, su representación binaria podría ser algo como [1, 0, 1, ..., 0], donde las posiciones con 1 indican los géneros a los que pertenece. Esta normalización facilita la interpretación y el procesamiento de las etiquetas en el modelo, permitiendo que este aprenda a identificar múltiples géneros de manera simultánea en lugar de limitarse a una única clasificación. Además, este formato binario es compatible con la función de pérdida utilizada en modelos de aprendizaje profundo para problemas multi-etiqueta.

Además de la normalización de etiquetas, las imágenes fueron preprocesadas para adecuarlas al modelo de clasificación. Todas las imágenes se redimensionaron a un tamaño estándar de 224x224 píxeles para los modelos de CNN (común para modelos de visión por computadora como las redes neuronales convolucionales) y a una resolución de 150x150 para el modelo de Random Forest (primando la simplificación y reducción de dimensiones de los datos). Esta dimensión es un compromiso ideal entre la resolución suficiente para capturar detalles importantes y la eficiencia computacional. Las imágenes se mantuvieron con sus tres bandas RGB, lo que permite que el modelo aproveche la información de color. Además, los valores de los píxeles se normalizaron dividiendo cada valor por 255, lo que escala los valores originales de 0 a 255 (que representan la intensidad de color en cada canal) a un rango entre 0 y 1. Esta normalización facilita la convergencia del modelo durante el entrenamiento, ya que ayuda a estabilizar los gradientes y mejora el desempeño del modelo al manejar los datos de entrada de manera consistente.

Estos preprocesamientos conjuntos de etiquetas e imágenes no solo optimizan el rendimiento del modelo, sino que también aseguran que el análisis y la predicción se realicen sobre datos consistentes y representativos, maximizando la eficacia del proceso de clasificación tanto de etiqueta de valor único como multievaluado.

Capítulo 4

Análisis descriptivo de los datos

En este capítulo, realizaremos un análisis exhaustivo del conjunto de datos final con el objetivo de comprender su estructura y características clave antes de proceder a las etapas más avanzadas de modelado con redes neuronales. Este análisis preliminar no solo sirve como una guía para el diseño y la implementación de los modelos, sino que también establece una base sólida para poder evaluar los resultados de una forma crítica. En la Figura 4.1 se pueden ver los principales géneros del dataset.

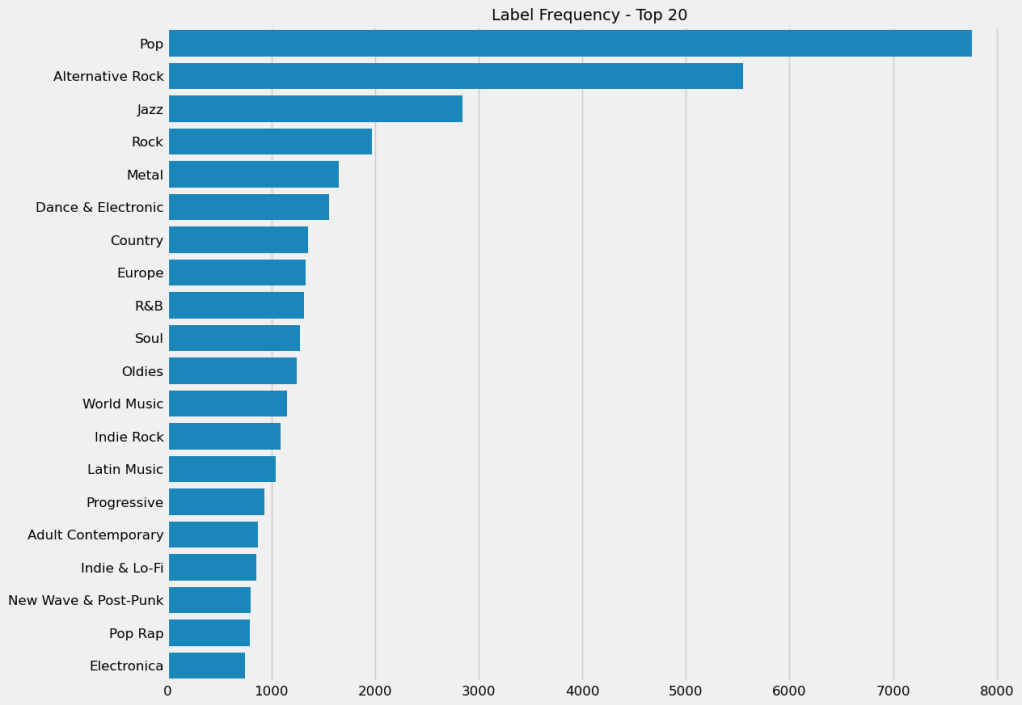

Figura 4.1: Top 20 géneros musicales

En una primera impresión, se pueden observar la variedad de estilos que hay presentes. Los géneros Pop y Alternative Rock dominan frente al resto de géneros, en los cuales no vemos nada fuera de lo común, lo que nos facilitará el entendimiento de los resultados y las conclusiones.

La Figura 4.2 muestra las 10 combinaciones de géneros más comunes en el conjunto de datos después del preprocesado. En ella, podemos observar que la combinación de géneros más frecuente es Alternative Rock, Indie Rock, Indie & Lo-Fi, que destaca significativamente sobre las demás, con más de 700 registros. Esto sugiere una fuerte tendencia hacia estilos de música alternativa e indie dentro del dataset.

La segunda combinación más frecuente es Pop y Rock, que también tiene una presencia considerable, aunque menor en comparación con la primera combinación. Esta mezcla de géneros más *mainstream*, como el Pop y el Rock, es coherente con la popularidad general de estos géneros en la música contemporánea.

Otras combinaciones notables incluyen Alternative Rock, Indie & Lo-Fi, Pop o géneros como Dance & Electronic y Gangsta & Hardcore, Pop Rap, Rap & Hip-Hop también aparecen entre las combinaciones más comunes, mostrando la diversidad de estilos presentes en el conjunto de datos.

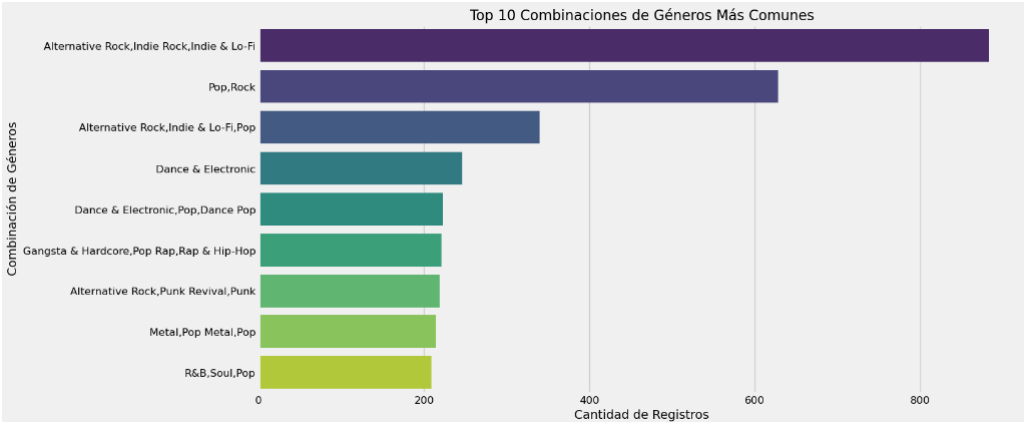

Figura 4.2: Principales combinaciones de géneros

A continuación, comentaremos el aspecto multievaluado de los datos. En la Figura 4.3, vemos la distribución original del número de géneros por registro antes de aplicar cualquier tipo de limitación. Esta distribución muestra que un número significativo de registros tiene entre 4 y 6 géneros anotados, con una moda establecida en 4 estilos musicales. Esta variabilidad puede reflejar la complejidad y diversidad de las anotaciones, pero también podría introducir ruido en el análisis, ya que algunos registros llegan hasta los 9.

Para conseguir una visión más simplificada de los datos decidimos limitar el número máximo de géneros por registro a 3, como vemos en la Figura 4.4, en la que la distribución cambia drásticamente. Ahora, la mayoría de los registros tienen exactamente 3 géneros, y desaparecen los registros con más de 3 géneros. Este cambio simplifica el dataset, reduciendo la complejidad del problema de clasificación, lo que puede mejorar la eficacia de los modelos de aprendizaje automático, al concentrar la información en los géneros más relevantes para cada registro.

Cabe destacar que la forma de limitación ejecutada es en base al orden de las eti-

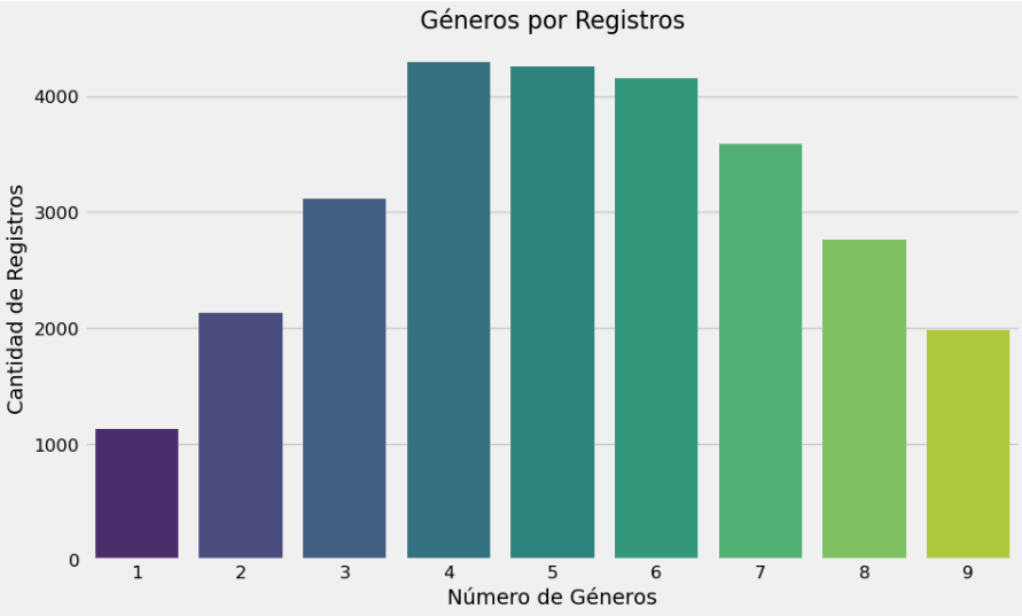

Figura 4.3: Distribución de géneros original

quetas del propio dataset, en el que el primer género es el más representativo y el último el menos importante.

La comparación entre ambas gráficas pone de manifiesto el efecto del preprocesado sobre la estructura del dataset. La segunda gráfica, más simplificada, reduce la variabilidad en el número de géneros por registro, lo que puede ayudar a los modelos a generalizar mejor sin perder demasiada información importante. Esto refuerza la importancia de las decisiones de preprocesado en la preparación de los datos para el análisis y modelado, asegurando un balance adecuado entre simplicidad y representatividad.

De esta forma, el proyecto estará realizado sobre dos datasets diferenciados: El primero, con hasta 3 etiquetas posibles por imagen, y el segundo, con un solo valor por imagen.

Finalmente, comentaremos la forma del dataset en su versión de etiqueta de valor único. En la Figura 4.5 se aprecia que el estilo Alternative Rock es el predominante. Esta claro desbalance de pesos entre etiquetas puede ocasionar muchos problemas en el funcionamiento de los modelos. Recordemos que, cuando la cantidad de opciones a clasificar es tan dispar, los modelos de aprendizaje automático suelen desarrollar una preferencia hacia la etiqueta o el grupo de etiquetas principal.

A lo largo de este trabajo vamos a implementar diversas estrategias para resistir y mitigar este desbalance de clases con el fin de evitar que los modelos desarrollen sesgos hacia la etiqueta predominante.

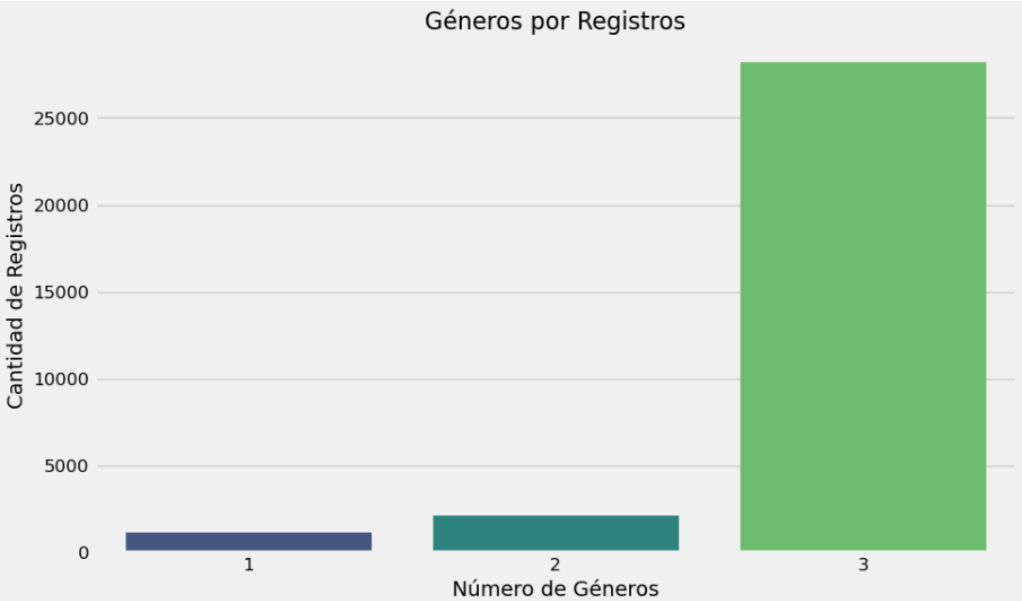

Figura 4.4: Distribución de géneros capada a 3

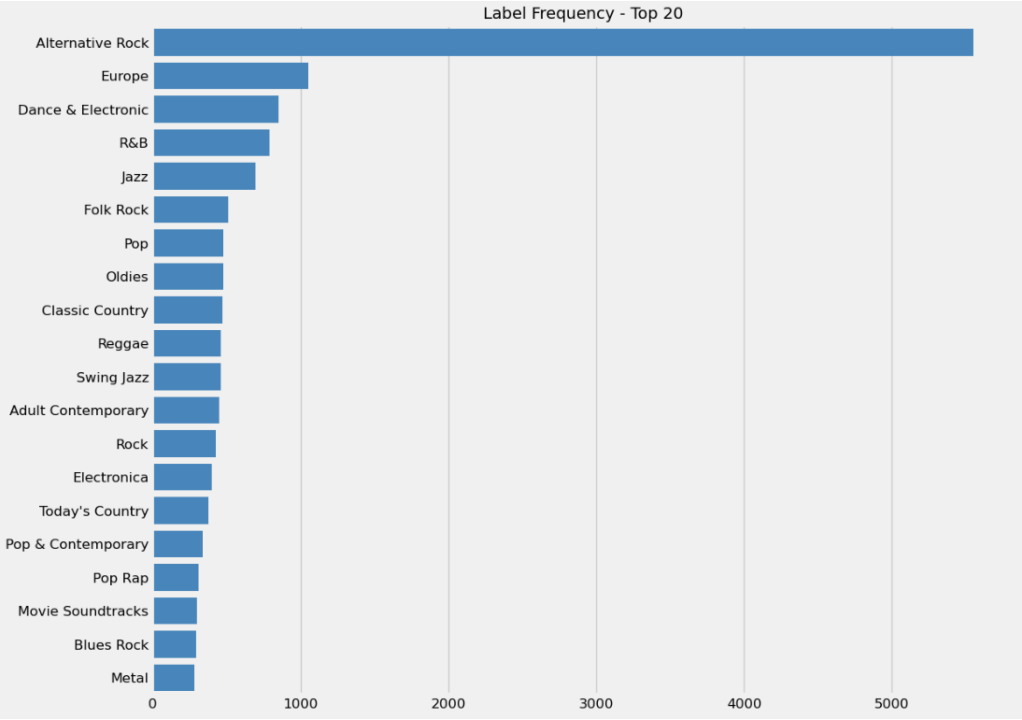

Figura 4.5: Top 20 géneros musicales capado a 1

Capítulo 5

Modelos de aprendizaje

El problema a abordar en este trabajo se enmarca en el ámbito del aprendizaje supervisado, una técnica fundamental en la ciencia de datos. En particular, el objetivo es entrenar un modelo capaz de predecir una variable de salida categórica a partir de un conjunto de datos que incluye tanto las entradas como las salidas esperadas. Este tipo de problema se entiende como un problema de clasificación, ya que la variable objetivo es discreta y se divide en categorías distintas.

Para abordar este desafío, se experimentará con dos enfoques metodológicos distintos: uno basado en técnicas de aprendizaje automático tradicionales y otro basado en modelos de aprendizaje profundo. El primer enfoque implicará el uso de algoritmos clásicos que han sido ampliamente utilizados y validados en tareas de clasificación. Por otro lado, el segundo enfoque explorará técnicas avanzadas de aprendizaje profundo, tales como redes neuronales convolucionales, que han demostrado un rendimiento sobresaliente en tareas complejas debido a su capacidad para aprender representaciones jerárquicas de los datos.

La comparación de estos dos enfoques permitirá evaluar sus respectivas ventajas y limitaciones en el contexto específico del problema de clasificación, proporcionando una visión integral sobre su eficacia y aplicabilidad en la tarea asignada.

5.1. Random Forest

Un Random Forest es un algoritmo de aprendizaje automático que combina múltiples árboles de decisión para mejorar la precisión y robustez de las predicciones. A diferencia de un solo árbol de decisión, que puede ser sensible a las variaciones en los datos y propenso al sobreajuste, un Random Forest construye una "bosque"de árboles de decisión independientes, entrenados en diferentes subconjuntos aleatorios del conjunto de datos. Cada árbol hace una predicción, y el Random Forest promedia estas predicciones (o usa votación mayoritaria en el caso de clasificación) para obtener una predicción final más precisa y estable. Este enfoque también conserva parte de la explicabilidad al permitir el análisis de importancia de características, aunque de manera menos directa que un

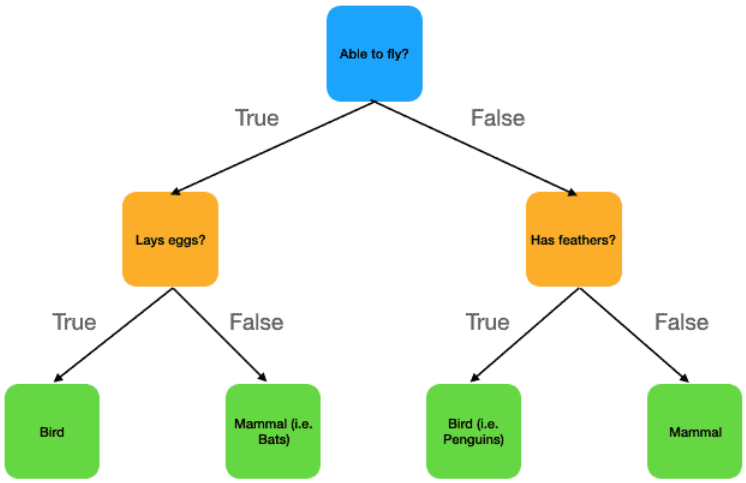

Figura 5.1: Ejemplo de árbol de decisión (Attard (2021))

único árbol de decisión.

Como ejemplo, consideremos la decisión de clasificar un animal como ave o mamífero. Primero, podríamos hacer la pregunta: ¿Puede volar? Si la respuesta es afirmativa, entonces el siguiente paso sería preguntar: ¿Pone huevos? Si el animal pone huevos, se clasifica como un ave; si no pone huevos, se clasifica como un mamífero. Este proceso de descarte mediante preguntas se ilustra en la Figura 5.1, que muestra un árbol de decisión que clasifica un animal en función atributos.

Un solo árbol de decisión puede ser susceptible al sobreajuste cuando se enfrenta a datos complejos, lo que limita su capacidad para generalizar a nuevos datos. Para mitigar este problema, el algoritmo Random Forest combina múltiples árboles de decisión, cada uno entrenado con una porción diferente de los datos y con ligeras variaciones en el proceso de entrenamiento. Esta técnica de ensamblaje mejora significativamente la generalización del modelo al promediar las predicciones de varios árboles, en lugar de depender de uno solo.

El algoritmo Random Forest emplea la técnica de *bagging*, que implica la selección aleatoria de subconjuntos de datos para entrenar cada árbol. Cada árbol se construye a partir de un subconjunto de datos con reemplazo, lo que ayuda a reducir la varianza y mejorar la estabilidad del modelo. Además, se introducen dos hiperparámetros clave para ajustar el rendimiento del Random Forest: el número de árboles en el bosque y el número máximo de características consideradas para dividir cada nodo. La profundidad máxima de cada árbol también se puede limitar para evitar un sobreajuste excesivo.

En problemas de clasificación, las predicciones de todos los árboles se combinan mediante un proceso de votación. La clase que recibe el mayor número de votos es seleccionada como la predicción final del Random Forest, lo que proporciona una solución robusta y menos propensa al sobreajuste en comparación con un solo árbol de decisión.

Este algoritmo de aprendizaje automático ofrece una buena explicabilidad al permitir

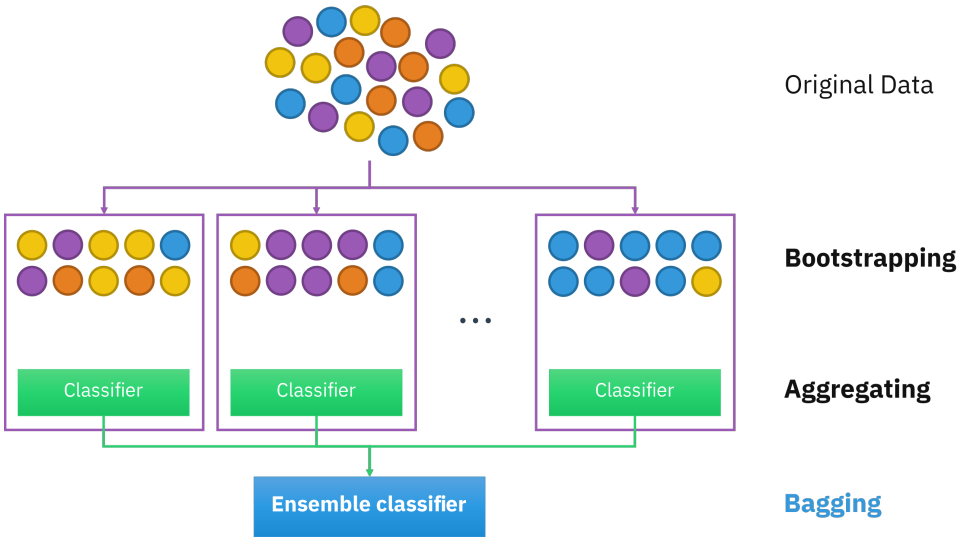

Figura 5.2: Esquema del proceso de bagging (Sirakorn (2020))

la identificación de la importancia de cada variable en las predicciones. Dicho proceso se realiza a través de medidas como la disminución de la impureza, que evalúa cuánto contribuye cada característica a la mejora de la precisión del modelo. Aunque Random Forest es más complejo que un solo árbol de decisión, ya que combina múltiples árboles de decisión para hacer predicciones, sigue siendo relativamente ligero en comparación con modelos más avanzados como las redes neuronales profundas.

Sin embargo, su costo computacional y en términos de recursos es mayor debido al entrenamiento y combinación de estos múltiples árboles. Cada árbol en el bosque se entrena con un subconjunto aleatorio de datos y características, lo que añade una capa adicional de complejidad en el proceso de entrenamiento y en la inferencia. A pesar de esto, la capacidad de Random Forest para manejar grandes volúmenes de datos y su robustez frente al sobreajuste lo convierten en una opción valiosa en muchos contextos.

5.2. Modelos de aprendizaje profundo

Los modelos de aprendizaje profundo son una categoría avanzada dentro del aprendizaje automático, que se basa en redes neuronales artificiales organizadas en múltiples capas. Estas redes se inspiran en la estructura y el funcionamiento del cerebro humano, imitando su capacidad para aprender y realizar tareas complejas. La estructura básica de una red neuronal profunda incluye una capa de entrada, varias capas ocultas y una capa de salida.

La capa de entrada recibe los datos iniciales del modelo, que pueden ser cualquier tipo de información estructurada, como imágenes, texto o datos numéricos. Esta capa simplemente transmite los datos a las siguientes capas sin realizar cálculos complejos. Las capas ocultas, situadas entre la capa de entrada y la capa de salida, son donde se

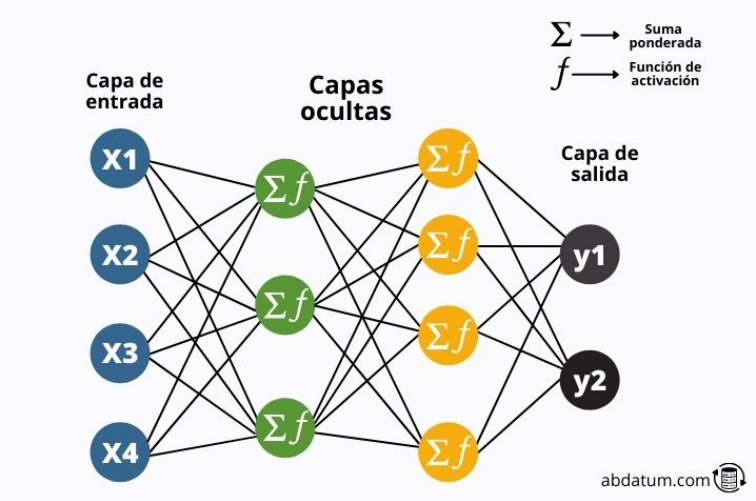

Figura 5.3: Esquema de una red neuronal (Calvo (2017))

realiza la mayor parte del procesamiento. Estas capas están compuestas por neuronas artificiales que aplican funciones matemáticas no lineales a las entradas ponderadas para extraer características cada vez más abstractas y representativas de los datos.

Cada neurona en una capa oculta toma las entradas de la capa anterior, las multiplica por pesos específicos, suma un sesgo y pasa el resultado a través de una función de activación no lineal, como ReLU o Sigmoid. Este proceso permite a la red aprender patrones complejos y no lineales en los datos. Finalmente, la capa de salida produce la predicción final del modelo, que puede tener diferentes configuraciones según el tipo de tarea que se esté realizando, como clasificación, regresión o generación de datos.

La Figura 5.3 ilustra cómo se interconectan las capas y cómo fluye la información a través de la red neuronal, mostrando la complejidad y la profundidad de estos modelos avanzados.

En una red neuronal, cada neurona desempeña un papel crucial en el procesamiento de datos a través de un sistema de pesos y sesgos. Estos parámetros son esenciales para la capacidad de la red de aprender y hacer predicciones precisas. Durante el entrenamiento de la red neuronal, se utiliza el algoritmo de retropropagación para ajustar estos pesos y sesgos de manera que se minimice el error entre las predicciones generadas por la red y las etiquetas reales de los datos.

El proceso comienza con la fase de propagación hacia adelante, en la cual los datos de entrada se pasan a través de la red neuronal. Cada neurona en la red toma estos datos de entrada, los multiplica por sus pesos asignados, les suma un sesgo, y luego aplica una función de activación para obtener una salida. Esta salida se transmite a la siguiente capa de neuronas hasta que se obtiene la predicción final.

Una vez que se realiza la predicción, se compara con la etiqueta real mediante una función de pérdida, que mide el error de la predicción. La fase de retropropagación se encarga de ajustar los pesos y sesgos en función del error calculado. Este proceso uti-

liza el gradiente descendente u otros algoritmos de optimización para determinar cómo cambiar los parámetros para reducir el error. Específicamente, se calcula el gradiente del error con respecto a cada peso y sesgo mediante el algoritmo de retropropagación, que emplea la regla de la cadena para distribuir el error a través de la red. Las neuronas que contribuyen más significativamente al error tienen sus parámetros ajustados en mayor medida, mientras que las que tienen menor influencia tienen ajustes menores.

Este ciclo de propagación hacia adelante y retropropagación se repite durante múltiples iteraciones, o épocas, para entrenar la red neuronal. A medida que el entrenamiento avanza, los pesos y sesgos se ajustan continuamente, mejorando la capacidad de la red para hacer predicciones precisas y reduciendo el error en sus salidas.

Sin embargo, debido a la complejidad de las redes neuronales y su arquitectura, interpretar cómo cada variable de entrada influye en las decisiones de la red puede ser complicado. Aunque el proceso de entrenamiento es bien comprendido, la çaja negraïnherente a las redes neuronales dificulta la explicación clara de cómo se toman decisiones específicas basadas en los datos de entrada. Esto plantea desafíos adicionales en aplicaciones donde la transparencia y la interpretabilidad son cruciales.

5.2.1. Arquitecturas escogidas

Para este proyecto, hemos decidido crear un modelo basado en la arquitectura MobileNetV2 y compararlo con el modelo (Koenig (2019)).

Usar modelos preentrenados para reconocimiento y clasificación de imágenes es preferible porque ya han sido entrenados en grandes conjuntos de datos y pueden capturar características generales útiles, como bordes y texturas. Esto ahorra tiempo y recursos al evitar el entrenamiento desde cero, que requeriría grandes cantidades de datos y potencia computacional. Además, los modelos preentrenados suelen ser más precisos y robustos al tener una base sólida, permitiendo ajustes finos para tareas específicas con menos datos y esfuerzo, mejorando así la eficiencia y el rendimiento del proyecto.

El modelo de CNN que se ha desarrollado utiliza MobileNetV2 como base para la clasificación multietiqueta de imágenes. En primer lugar, se carga el modelo MobileNetV2 preentrenado con pesos de ImageNet, pero sin la capa de clasificación final, lo que permite aprovechar las características previamente aprendidas sin entrenar toda la red desde cero. Esta base de MobileNetV2 está congelada, es decir, sus pesos no se actualizan durante el entrenamiento, lo que ahorra tiempo y recursos. A continuación, se añade una capa de entrada que redimensiona las imágenes a un tamaño de 224x224 píxeles y normaliza los valores de los píxeles para que estén en el rango de 0 a 1. Luego, las características extraídas por MobileNetV2 se pasan a través de una capa de agrupamiento global, que reduce las dimensiones espaciales a un vector de características de tamaño fijo. Este vector se procesa con una capa de dropout para prevenir el sobreajuste y, finalmente, se pasa a través de una capa densa con activación sigmoide para realizar la clasificación multietiqueta, generando una salida con un nodo por cada etiqueta de

género.

La elección de MobileNetV2 permite aprovechar su eficiencia y capacidad para extraer características significativas con un menor uso de recursos, mientras que las capas adicionales adaptan el modelo para la tarea específica de clasificación multietiqueta. Otras opciones incluyen:

- **VGGNet:** Famosa por su simplicidad y eficacia, pero tiene muchos parámetros y es más pesada computacionalmente.
- **ResNet:** Ofrece profundidad y precisión mediante conexiones residuales, pero puede ser más costosa en términos de computación.
- **DenseNet:** Mejora la eficiencia al conectar capas de manera densa, pero también es más intensiva en recursos.

El modelo a comparar (Koenig (2019)) en es una red neuronal convolucional que sigue una estructura similar a VGG-16. Usa seis bloques de capas convolucionales que ayudan a extraer características importantes de las imágenes. Cada bloque aplica convoluciones con filtros de 3x3, seguido de normalización por lotes, activación ReLU y max pooling para reducir el tamaño de las características. Después, hay tres capas totalmente conectadas que combinan estas características y realizan la clasificación final. La salida tiene 446 nodos, cada uno para una etiqueta de género diferente.

Comparar este modelo con tu modelo basado en MobileNetV2 es interesante por varias razones.

Primero, MobileNetV2 está diseñado para ser más eficiente en términos de cálculo y memoria, utilizando una arquitectura de convoluciones separables en profundidad que puede ser más adecuada para aplicaciones en dispositivos con recursos limitados. En contraste, el modelo del trabajo ajeno, con su estructura similar a VGG-16, puede ofrecer una mayor capacidad de extracción de características gracias a su profundidad y tamaño, pero a costa de un mayor consumo de recursos.

Comparar ambos modelos puede revelar cómo el equilibrio entre precisión y eficiencia afecta el rendimiento en tareas de clasificación multietiqueta, proporcionando información valiosa sobre cuál puede ser más adecuado para diferentes contextos y limitaciones de hardware.

Capítulo 6

Entrenamiento de los modelos

En el presente capítulo se describe el proceso de entrenamiento y puesta a prueba de los modelos elegidos. Este es el paso previo a la evaluación de dichos modelos, lo que a su vez nos permitirá determinar la idoneidad de cada uno de ellos para realizar las clasificaciones de las imágenes que tenemos.

Todos los modelos se han entrenado siguiendo el mismo esquema general. Este esquema se ha plasmado en un documento de Jupyter Notebook para cada modelo entrenado. El código incluido en dichos documentos permite configurar los diferentes parámetros de los modelos. Las operaciones más importantes realizadas para entrenar los modelos se resumen en los siguientes puntos:

- **Carga y adaptación de los datos al modelo.** Se cargan los datos y se adaptan en un dataframe con la información de la imagen y sus etiquetas. Además se normalizan los datos para que puedan ser procesados correctamente por los distintos modelos.
- **Separación de los datos en conjunto de entrenamiento y conjunto de evaluación.** Dependiendo del modelo empleado, se ha optado por dividir el conjunto de datos en 2 o 3 subconjuntos. La razón para utilizar diferentes divisiones del conjunto de datos para el modelo de Random Forest y el modelo de Red Neuronal Convolucional (CNN) radica en las características y necesidades específicas de cada tipo de modelo.

Para el Random Forest, que es un modelo basado en árboles de decisión que no suele requerir un ajuste extenso de hiperparámetros, una simple división en 90 % para entrenamiento y 10 % para prueba es generalmente suficiente. Esto se debe a que el Random Forest puede manejar bien la variabilidad en los datos y la necesidad de ajuste fino es menor en comparación con modelos más complejos. La separación realizada se puede visualizar en el gráfico correspondiente a la Figura 6.1.

Por otro lado, las CNNs son modelos más complejos con muchos hiperparámetros que deben ser ajustados cuidadosamente para evitar el sobreajuste y maximizar el rendimiento. En este caso, la división en tres subconjuntos — 80 % para entrenamiento, 10 %

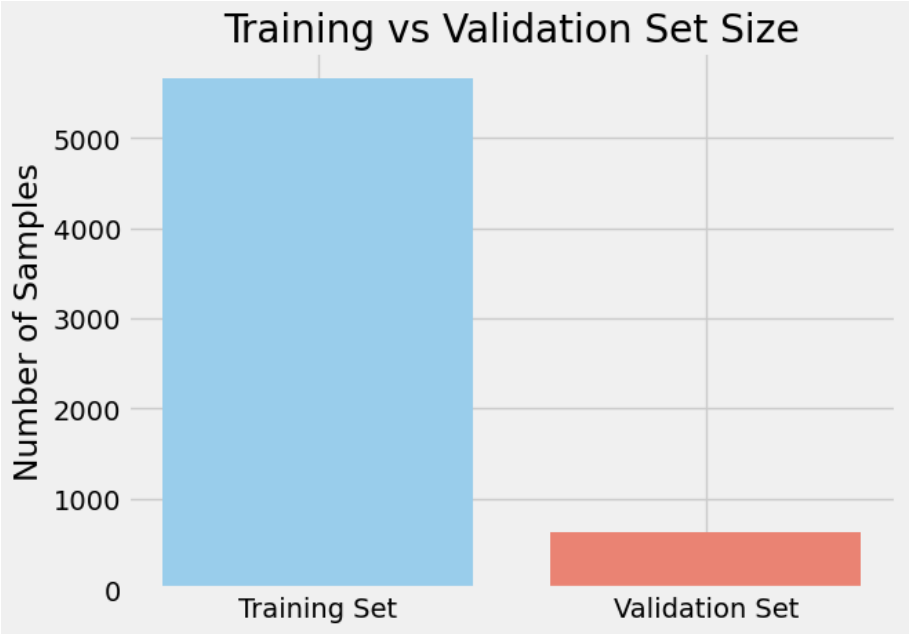

Figura 6.1: Separación de los conjuntos de entrenamiento y evaluación para Random Forest

para validación y 10 % para prueba — permite una separación adecuada para ajustar hiperparámetros y realizar una evaluación robusta. La validación con un conjunto separado ayuda a ajustar el modelo durante el entrenamiento, mientras que el conjunto de prueba proporciona una evaluación final completamente independiente para medir el rendimiento general del modelo. En la Figura 6.2 se comprueba la separación de los conjuntos.

La separación se hace con aleatoriedad ya que estamos ante un caso de clasificación, en el que cada elemento se debe tratar de forma independiente al resto.

Entrenamiento del modelo. Para ello se hace uso de las interfaces que nos ofrecen las herramientas de Scikit-learn y TensorFlow según el modelo que se desea entrenar. Se configuran los hiperparámetros con los que se quiere entrenar el modelo.

Evaluación de los resultados. Se calculan varias métricas para medir el error que comete el modelo sobre los datos de evaluación y se visualizan las predicciones. Tras esto se puede decidir entrenar nuevamente el modelo con distintos valores para los hiperparámetros con el objetivo de optimizar el entrenamiento del modelo.

Guardado del modelo en un fichero. Esto sirve para evitar tener que volver a entrenar el mismo modelo en caso de que se quiera hacer uso de él en un futuro.

Todos los modelos entrenados mediante este esquema se han puesto a prueba tanto con etiquetas de un único valor como multievaluadas.

Aunque el proceso de entrenamiento es muy similar para los distintos modelos, existen algunas características específicas de cada modelo que merecen ser mencionadas por separado. A continuación se explican para cada modelo las diferencias en su entre-

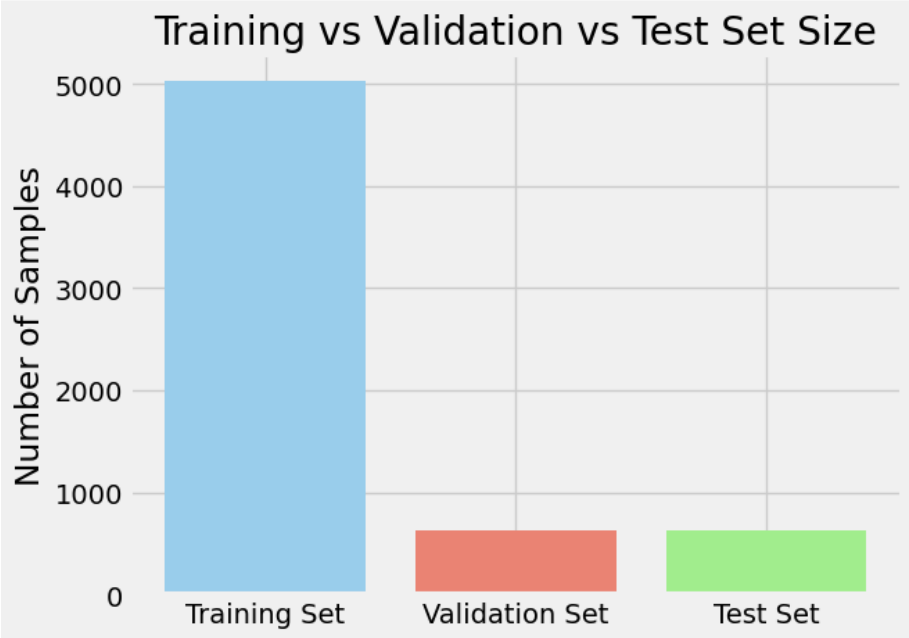

Figura 6.2: Separación de los conjuntos de entrenamiento, validación y evaluación para CNN

namiento respecto al esquema general.

6.1. Random Forest

Para el entrenamiento de este modelo, se ha decidido probar con el modelo base y otro en el que se han calibrado los siguientes hiperparámetros:

- El número de árboles
- La profundidad máxima de los árboles
- El número máximo de variables a usar en el entrenamiento de cada árbol.

Para calibrar los hiperparámetros, tal y como se menciona en el esquema general, se han ido probando distintos valores hasta encontrar alguno que mostrara un rendimiento adecuado. De esta forma, se ejecuta el modelo con diferentes variaciones de estos parámetros para poder observar qué combinación es la óptima en nuestro caso.

6.2. CNN

Para el entrenamiento de este modelo se han calibrado los siguientes hiperparámetros:

- El número de épocas.

- El tamaño de lote.
- El número de neuronas por capa.
- El número de capas.
- El optimizador.

El modelo se entrena utilizando el optimizador Adam, que es conocido por su eficiencia en el ajuste de pesos y su capacidad para manejar tasas de aprendizaje adaptativas. La función de pérdida empleada es la entropía cruzada binaria, que se utiliza comúnmente en problemas de clasificación binaria.

Durante el proceso de entrenamiento, es crucial que el modelo aprenda a asignar predicciones significativas en todos los casos, en lugar de simplemente predecir el mismo valor para todas las instancias, como por ejemplo, asignar siempre un valor de cero. Este comportamiento puede ser problemático, ya que puede indicar que el modelo no está capturando correctamente la variabilidad de los datos y, por ende, no está generalizando bien.

Para abordar este problema, hemos añadido un mecanismo de penalización que se activa cuando el modelo predice el mismo valor (como ceros) para todas las instancias. Este mecanismo añade una penalización adicional a la función de pérdida estándar, incrementando el costo de tales predicciones uniformes. De esta manera, el modelo es incentivado a explorar y asignar una gama más amplia de valores, lo que ayuda a mejorar su capacidad para aprender patrones complejos y ofrecer predicciones más variadas y útiles.

Capítulo 7

Métodos de Evaluación

Tras haber realizado el entrenamiento de todos los modelos, es el momento de compararlos. Para ello precisamos de métodos de evaluación que contrasten el acierto de las predicciones de cada uno de los modelos. El modo más eficaz de comparar objetivamente varios modelos distintos es calculando métricas que sean capaces de representar el comportamiento y los aciertos de cada uno de ellos. Además, es importante realizar la comparación sobre el mismo conjunto de datos. Como hemos mencionado anteriormente, el modelo de Random Forest entrenado se han separado los datos con un conjunto del 90 % - 10 % y en el modelo de CNN en un conjunto de 80 % - 10 % - 10 % para comprobar la capacidad de clasificación del modelo y calcular las métricas. Es importante recordar que la separación de los datos es aleatoria para evitar cualquier tipo de sesgo.

Dado que los modelos se pueden entrenar para realizar predicciones con un número de etiquetas variable, se ha decidido comparar los modelos para su versión de etiqueta de valor único como multievaluado de hasta 3 géneros. De este modo, damos la posibilidad a encontrar modelos que funcionan mejor al clasificar valores únicos o multi evaluados.

Existen varias métricas para evaluar un modelo al realizar las clasificaciones. Cada una de ellas presenta un punto de vista distinto y da más o menos importancia a los aciertos y fallos cometidos. Por ello es de especial relevancia conocer la manera en que se calculan y como afecta esto al resultado de la métrica. Cabe destacar que, a la hora de aplicarlas sobre los modelos, se hace uso de las implementaciones que ofrece Scikit-learn. A continuación se explican cada una de las métricas que se aplicarán.

7.1. Accuracy o exactitud

La Accuracy mide la proporción de predicciones correctas (tanto positivas como negativas) entre todas las predicciones realizadas por el modelo. Es una métrica global que indica qué tan bien está funcionando el modelo en general, pero puede ser engañosa si las clases están desbalanceadas.

La fórmula que determina el cálculo de esta métrica es la que se especifica a conti-

nuación:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.1}$$

Se suman los verdaderos positivos (TP) y verdaderos negativos (TN), y se divide por el total de predicciones (suma de TP, TN, falsos positivos (FP) y falsos negativos (FN)).

7.2. Precision o precisión

La Precision mide la proporción de verdaderos positivos (TP) entre todas las instancias que el modelo ha etiquetado como positivas. Esta métrica es crucial cuando el costo de un falso positivo es alto, ya que indica qué tan precisas son las predicciones positivas del modelo.

Su fórmula se presenta a continuación:

$$Precision = \frac{TP}{TP + FP} \tag{7.2}$$

Se divide el número de verdaderos positivos (TP) entre la suma de verdaderos positivos (TP) y falsos positivos (FP).

7.3. Recall o sensibilidad

El Recall mide la proporción de verdaderos positivos (TP) detectados por el modelo entre todas las instancias que son realmente positivas. Es fundamental cuando es crucial identificar todas las instancias positivas, como en problemas médicos.

Su fórmula resulta tal y como se muestra a continuación:

$$Recall = \frac{TP}{TP + FN} \tag{7.3}$$

Se divide el número de verdaderos positivos (TP) entre la suma de verdaderos positivos (TP) y falsos negativos (FN).

7.4. Área bajo la curva ROC

El Área bajo la curva ROC (AUC-ROC) mide la capacidad del modelo para distinguir entre clases positivas y negativas. La curva ROC representa la relación entre el Recall (Sensibilidad) y la Tasa de Falsos Positivos (FPR) a diferentes umbrales de clasificación. Un AUC cercano a 1 indica un modelo excelente, mientras que un AUC de 0.5 indica un modelo no mejor que el azar.

Su fórmula se calcula de la siguiente manera:

$$AUC = \int_0^1 TPR(FPR)\, d(FPR) \tag{7.4}$$

Capítulo 8

Resultados

Para cada uno de los modelos entrenados se han llevado a cabo una serie de experimentos utilizando diversos hiperparámetros. A continuación, se analizarán los resultados de los experimentos más relevantes para cada modelo, a través de los distintos métodos de evaluación. Principalmente se hará uso de la Accuracy y del AUC-ROC. La primera, porque es importante al ofrecer una visión general de cuán bien el modelo clasifica correctamente, mientras que la segunda ofrece una visión global y no depende del umbral de decisión, lo que es útil cuando se quiere evaluar el desempeño general del modelo en diferentes situaciones.

8.1. Random Forest

Como hemos comentado en puntos anteriores, para este modelo se han realizado diversos experimentos con variables de múltiple valor y de valor único. Los resultados de dichos experimentos son los detallados en las tablas 8.1 y 8.3.

ID	n_estimators	max_depth	accuracy
1	10	2	0.1156
2	20	10	0.1201
3	50	10	0.1277
4	100	20	0.1495
5	100	10	0.1423

Tabla 8.1: Resultados de Random Forest (Etiqueta de valor único)

Se puede observar que, como era de esperar, el resultado es muy pobre. Los valores de *accuracy* oscilan entre el 10 % y 15 %. Esto quiere decir que el modelo no funciona mucho mejor que el azar, ya que hemos acotado las posibles etiquetas a 10. Sin embargo, podemos establecer alguna conclusión más.

Por ejemplo, podemos reconocer que el resultado de los experimentos es peor, en líneas generales, en el caso de etiquetas multievaluadas que en el de etiquetas de valor

ID	n_estimators	max_depth	accuracy
1	100	20	0.1251
2	50	10	0.1075
3	50	5	0.1036
4	100	5	0.1185
5	50	20	0.1113

Tabla 8.2: Resultados de Random Forest (Etiqueta multievaluada)

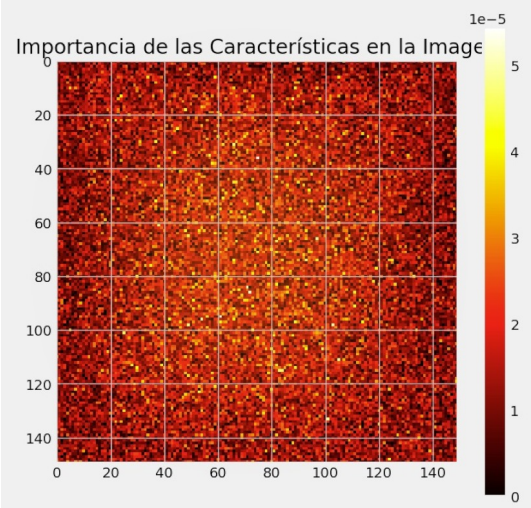

Figura 8.1: Mapa de calor de las carácteristicas de Random Forest

único.

En la clasificación de valor único, el modelo solo necesita aprender a diferenciar entre clases mutuamente excluyentes, lo cual es más directo y manejable. Sin embargo, en la clasificación multietiqueta, el modelo debe aprender a manejar la interdependencia y correlación entre las etiquetas, lo que aumenta significativamente la complejidad del problema. Además, cada etiqueta se trata de manera independiente en enfoques básicos, lo que puede llevar a una subóptima captura de las relaciones entre las etiquetas.

Cuando trabajamos con imágenes en un modelo como Random Forest, la interpretabilidad puede ser un desafío porque las características del modelo (píxeles o combinaciones de píxeles) no son fácilmente interpretables. Sin embargo, podemos visualizar un mapa de calor con la importancia de las características en la Figura 8.3.

Se comprueba que la zona central de la imagen tiene un mayor peso de media que las celdas cercanas al borde, tanto para la variante multievaluada como para la de valor único.

Además, vamos a crear con los resultados una matriz de confusión, ya que es una de las formas más indicadas para conocer el comportamiento clasificación del modelo. Idealmente, deberíamos observar una mayor presencia en las casillas de la diagonal principal de la matriz, pero la Figura 8.2 logra demostrar una vez más el mal comporta-

miento del modelo para este problema. Sin embargo, también nos logra aportar alguna información interesante.

El estilo musical Dance & Electronic se ha podido identificar correctamente en 9 ocasiones, aunque en 10 ocasiones se ha confundido con el género Oldies. Vemos que el modelo tiene cierta preferencia por este último estilo, ya que apuesta por él más que por el resto de géneros.

También podemos destacar la posible relación del arte de las portadas de Reggae y Folk Rock, o de Jazz y Oldies.

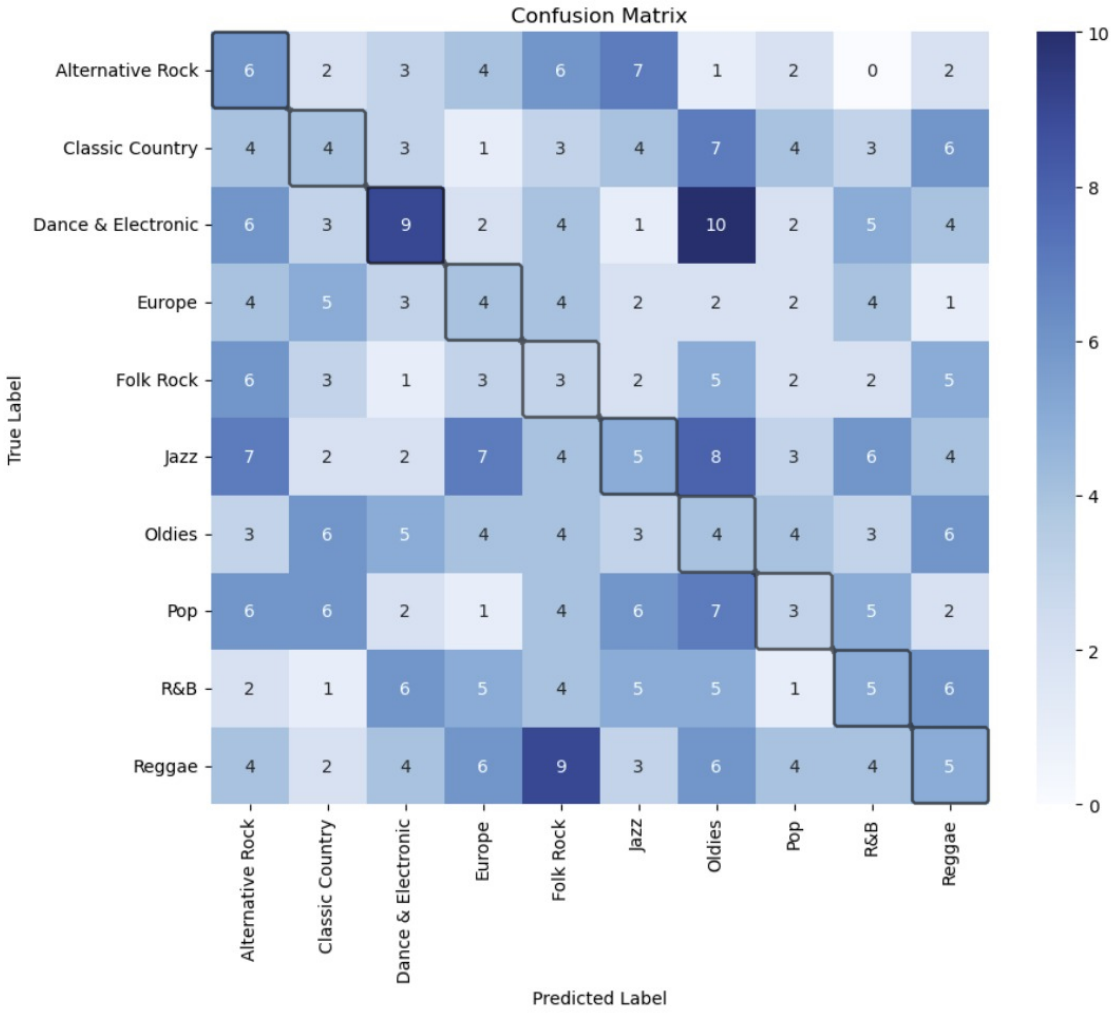

Figura 8.2: Matriz de confusión de Random Forest

Finalmente, vamos a recoger los valores de las principales métricas para las combinaciones óptimas que hemos encontrado. En la tabla 8.3 se comprueba que el modelo no es muy adecuado para este problema. Dividir las imágenes en vectores y usar estos datos como variables de entrada para el modelo no es muy funcional, como ya intuíamos al comienzo del proyecto.

En el caso de la métrica del AUC-ROC , podemos visualizarlo mediante la Figura 8.3. En este gráfico del modelo de etiquetas unievaluadas, los AUC varían entre 0.50 y 0.59

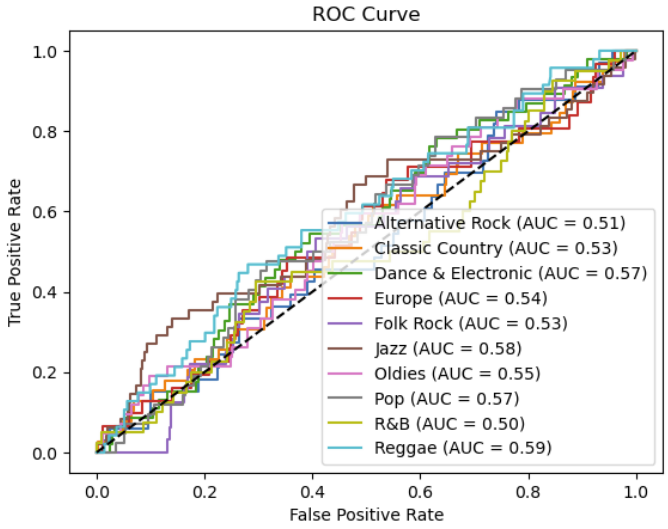

Figura 8.3: Gráfica de AUC-ROC de Random Forest (Etiqueta con valor único)

para los diferentes géneros, lo que sugiere que el modelo no tiene un buen rendimiento general. De hecho, muchos de los AUC están muy cerca de 0.5, lo que indica que el modelo está apenas por encima del azar para esos géneros.

Reggae (AUC = 0.59) y Jazz (AUC = 0.58) son los géneros con los AUC más altos, por lo que tienen un rendimiento ligeramente mejor en la clasificación de estos géneros en comparación con los demás. El R&B (AUC = 0.50) no se está clasificando de manera efectiva, actuando casi como si fuera al azar.

La combinación de *n_estimators* = 100 y *max_depth* = 20 puede funcionar de forma óptima tanto para clasificación de una sola etiqueta como para clasificación multietiqueta debido a un equilibrio adecuado entre la capacidad del modelo y la complejidad del problema. Con 100 árboles, el modelo es lo suficientemente robusto para capturar patrones sin ser excesivamente complejo, lo que previene el sobreajuste. Un *max_depth* de 20 permite que cada árbol en el bosque pueda capturar relaciones complejas sin volverse demasiado profundo, lo cual podría llevar a un sobreajuste a los datos de entrenamiento. Otras combinaciones pueden no haber logrado un equilibrio similar entre la complejidad del modelo y la capacidad de generalización, resultando en un rendimiento subóptimo.

n_label	n_estim	max_depth	accuracy	precision	recall	AUC
1	100	20	0.1495	0.3543	0.1139	0.5617
3	100	20	0.1251	0.0292	0.1062	0.5134

Tabla 8.3: Mejores resultados de Random Forest

8.2. CNN

Los resultados de los modelos de redes neuronales son significativamente mejores que los obtenidos en el apartado anterior, aunque presentan algunas particularidades que requieren un análisis más profundo.

En primer lugar, observamos en la tabla 8.4 que el accuracy de entrenamiento comienza en un valor relativamente bajo (alrededor del 59 %) en la primera época, pero se incrementa rápidamente a medida que avanza el entrenamiento, alcanzando aproximadamente un 87 % en las primeras 5 épocas. A partir de este punto, la accuracy de entrenamiento se mantiene relativamente estable alrededor del 87 %, con pequeñas fluctuaciones a lo largo del proceso de entrenamiento.

El accuracy de validación, por otro lado, muestra un comportamiento más estable desde el principio, alcanzando rápidamente un 87 % y manteniéndose constante en este valor a lo largo de todas las épocas. Este comportamiento sugiere que el modelo tiene una capacidad razonable para generalizar a los datos de validación, pero no parece mejorar más allá de un punto específico, lo que puede indicar que el modelo ha llegado a su capacidad máxima de aprendizaje con la arquitectura y los hiperparámetros actuales. Esto se puede ver visualmente en la Figura 8.4.

En cuanto a la pérdida o loss, los valores tanto en entrenamiento como en validación muestran una tendencia decreciente inicial, lo cual es típico en el entrenamiento de modelos de aprendizaje profundo. La pérdida de entrenamiento desciende rápidamente en las primeras épocas, y luego se estabiliza alrededor de 0.34, mientras que la pérdida de validación sigue un patrón similar, estabilizándose alrededor de 0.32. La diferencia marginal entre las pérdidas de entrenamiento y validación indica que el modelo no sufre de un sobreajuste significativo, lo que es un aspecto positivo.

El comportamiento del accuracy y la pérdida sugieren que el modelo está bien entrenado y no muestra señales evidentes de sobreajuste o subajuste. Sin embargo, el hecho de que el accuracy de validación se mantenga constante sugiere que el modelo podría estar limitado por su capacidad de aprendizaje, o que los datos de validación no están proporcionando una diversidad suficiente para continuar mejorando la generalización del modelo.

En la Figura 8.5, el modelo muestra un desempeño desigual según el género musical. El mejor rendimiento lo encontramos en la clasificación de carátulas de álbumes de Jazz, con un AUC de 0.70. Este resultado indica que el modelo tiene una buena capacidad para distinguir carátulas de Jazz de otros géneros, con una curva ROC que se aproxima más al borde superior izquierdo en comparación con las curvas de otros géneros. Los géneros R&B y Metal también muestran un desempeño aceptable, con AUC de 0.68 y 0.63 respectivamente. Esto sugiere que, aunque no sea perfecto, el modelo tiene una capacidad de discriminación razonable para estos géneros.

Por otro lado, la clasificación de carátulas en géneros como Pop y Rock presenta un rendimiento moderado, con AUC de 0.61 y 0.58, respectivamente. Si bien estos valores

Epoch	Train Accuracy	Train Loss	Validation Accuracy	Validation Loss
1	0.5921	0.6527	0.7739	0.4712
2	0.7567	0.4892	0.8750	0.3859
3	0.8399	0.4109	0.8750	0.3520
4	0.8746	0.3769	0.8750	0.3430
5	0.8746	0.3561	0.8750	0.3423
6	0.8746	0.3592	0.8750	0.3428
7	0.8750	0.3570	0.8750	0.3419
8	0.8711	0.3555	0.8750	0.3400
9	0.8686	0.3538	0.8750	0.3379
10	0.8612	0.3526	0.8750	0.3350
11	0.8711	0.3536	0.8750	0.3319
12	0.8697	0.3526	0.8750	0.3292
13	0.8725	0.3591	0.8750	0.3277
14	0.8715	0.3562	0.8750	0.3271
15	0.8757	0.3561	0.8750	0.3265
16	0.8732	0.3519	0.8750	0.3257
17	0.8746	0.3504	0.8750	0.3253
18	0.8715	0.3473	0.8750	0.3260
19	0.8708	0.3448	0.8750	0.3275
20	0.8715	0.3460	0.8750	0.3289
21	0.8686	0.3492	0.8750	0.3297
22	0.8693	0.3465	0.8750	0.3298
23	0.8725	0.3419	0.8750	0.3286
24	0.8739	0.3389	0.8750	0.3270
25	0.8718	0.3474	0.8750	0.3253
26	0.8722	0.3428	0.8750	0.3236
27	0.8711	0.3412	0.8750	0.3223
28	0.8743	0.3435	0.8750	0.3218
29	0.8729	0.3437	0.8750	0.3217
30	0.8711	0.3420	0.8750	0.3216

Tabla 8.4: Métricas de entrenamiento y validación

son superiores a 0.5, indicando que el modelo es mejor que un clasificador aleatorio, también revelan que hay margen de mejora, ya que el modelo no discrimina de manera óptima entre estos géneros y los demás.

El rendimiento más bajo se observa en los géneros Alternative Rock, Dance & Electronic, y Country, con AUC de 0.53, 0.45, y 0.34, respectivamente. Especialmente preocupante es el caso de Country, donde el AUC de 0.34 indica que el modelo no solo tiene dificultades para clasificar correctamente este género, sino que su rendimiento es peor

que un clasificador aleatorio en esta tarea específica.

El bajo rendimiento en géneros como Country y Dance & Electronic podría deberse a varias razones, como una representación insuficiente de características distintivas en las carátulas de los álbumes de estos géneros, o una posible similitud visual entre las carátulas de estos géneros y las de otros, lo que lleva al modelo a cometer errores. En cambio, géneros como Jazz o R&B pueden tener características visuales más definidas o distintivas que el modelo es capaz de capturar con mayor eficacia.

loss	auprc	auroc	precision_thresh_0.5	recall_thresh_0.5
0.087532	0.320698	0.901504	0.693627	0.210446

Tabla 8.5: Resultados del experimento

Para la clasificación multi-etiqueta, aunque el modelo obtuvo un buen puntaje AUROC, su AUPRC fue bajo, lo que indica que el modelo tiene alta precisión pero baja capacidad para recordar (recall). Esto sugiere que el modelo hace predicciones con confianza, pero a menudo falla en reconocer verdaderos positivos. Se menciona que, aunque el modelo supera a un modelo preentrenado en ImageNet en términos de AUROC, este no es un buen indicador en este caso debido a la gran cantidad de negativos en el dataset.

En este modelo, vamos a contar con la métrica AUPRC (Área Bajo la Curva de Precisión-Recall), que se enfoca en la relación entre la precisión y el recall, siendo más representativa en datasets desbalanceados, donde la clase positiva es rara. El AUPRC pone mayor énfasis en la clase positiva, siendo adecuado cuando se busca optimizar la detección de verdaderos positivos.

Nuestro modelo de red neuronal convolucional (CNN) presenta un AUC de 0.87, lo cual indica un rendimiento sólido en la clasificación de carátulas de álbumes musicales, aunque con margen de mejora. Comparado con el modelo de Koenig, que alcanzó un AUC de 0.90, nuestro modelo muestra un desempeño ligeramente inferior. La diferencia en el AUC puede atribuirse a la arquitectura y los hiperparámetros utilizados; el modelo de Koenig probablemente emplea técnicas más avanzadas o una arquitectura más sofisticada, lo que le permite capturar características más sutiles y mejorar la discriminación entre géneros musicales. Estos ajustes podrían haber contribuido a una mayor capacidad de generalización y precisión en la tarea.

Para la clasificación de una sola etiqueta, el modelo luchó por obtener buenos resultados. Sin embargo, aunque se probó con varias arquitecturas, el modelo simplemente logró predecir correctamente el género de Rock Alternativo, como vemos en la Figura 8.6. Esto se debe a su arquitectura limitada, que incluía solo capas convolucionales y una capa densa. Comprobando con el modelo de Koenig (2019), vemos que sucede lo mismo.

Este enfoque básico no fue suficiente para capturar la diversidad y complejidad de los distintos géneros musicales en el dataset. Además, el modelo sufrió de sobreajuste

en los datos de entrenamiento y no pudo generalizar bien a otros géneros, quedándose atascado en predecir principalmente Rock Alternativo, el género más dominante en los datos.

8.3. Posibles soluciones y mejoras en los resultados

El desempeño del modelo de red neuronal convolucional (CNN) en la clasificación de géneros musicales ha mostrado limitaciones significativas, particularmente en el contexto de la clase Rock Alternativo, que domina el conjunto de datos. Esto se debe a que el modelo, al ser entrenado con un desequilibrio notable en la representación de clases, tiende a sobreajustarse a la clase mayoritaria. A continuación, se presentan diversas estrategias que podrían mitigar este problema y mejorar el rendimiento global del modelo:

- **Rebalanceo del Conjunto de Datos**

  - Submuestreo (Undersampling): Esta técnica consiste en reducir el número de ejemplos de la clase mayoritaria (Rock Alternativo) para equilibrar el conjunto de datos con las clases minoritarias. Aunque esto puede ayudar a equilibrar la representación de las clases, existe el riesgo de perder información valiosa y reducir la cantidad total de datos de entrenamiento.
  - Sobremuestreo (Oversampling): Alternativamente, se puede aumentar el número de ejemplos de las clases minoritarias. Esto puede lograrse mediante la duplicación de ejemplos existentes o generando nuevas imágenes mediante técnicas de aumento de datos (data augmentation), como la rotación, el cambio de escala y las variaciones de color. El sobremuestreo ayuda a equilibrar la representación de las clases y permite que el modelo aprenda a identificar características distintivas de las clases minoritarias.

- **Uso de una Función de Pérdida Ponderada:** Para abordar el desequilibrio en la representación de clases, se puede ajustar la función de pérdida para penalizar más los errores en las clases minoritarias. Utilizando una función de pérdida ponderada, se asigna un mayor costo a las predicciones incorrectas de las clases menos representadas, incentivando al modelo a mejorar su rendimiento en estas clases. Esta técnica ayuda a equilibrar la influencia de cada clase en el proceso de entrenamiento, evitando que el modelo se sesgue hacia la clase mayoritaria.

- **Aumento de Datos Específico:** Implementar técnicas de aumento de datos específicamente para las imágenes de las clases minoritarias puede ser beneficioso. Aplicar transformaciones como rotaciones, traslaciones y escalado a las imágenes de las clases menos representadas incrementa su presencia en el conjunto de datos y, en consecuencia, proporciona una mayor diversidad de ejemplos para

el modelo. Esto facilita un aprendizaje más robusto y generalizable para todas las clases.

- **Balanceo de Clases en Minibatches:** Asegurar que cada minibatch durante el entrenamiento contenga una representación equilibrada de todas las clases es otra estrategia efectiva. Esta técnica garantiza que el modelo reciba ejemplos de todas las clases de manera continua, evitando que el entrenamiento se sesgue hacia la clase mayoritaria. El balanceo de clases en minibatches permite que el modelo se enfrente a una variedad más equitativa de ejemplos en cada iteración, promoviendo un aprendizaje más equilibrado y representativo.

Estas estrategias representan soluciones potenciales para mejorar la capacidad del modelo para clasificar correctamente todas las clases de géneros musicales, abordando el problema del desequilibrio en la representación de clases. Aunque estas técnicas se han probado en diversos contextos, no se han conseguido unos resultados convincentes. De esta forma no se han implementado completamente en el modelo actual, pero resulta interesante comentar que su incorporación podría resultar en una mejora significativa en la capacidad de generalización y precisión del modelo.

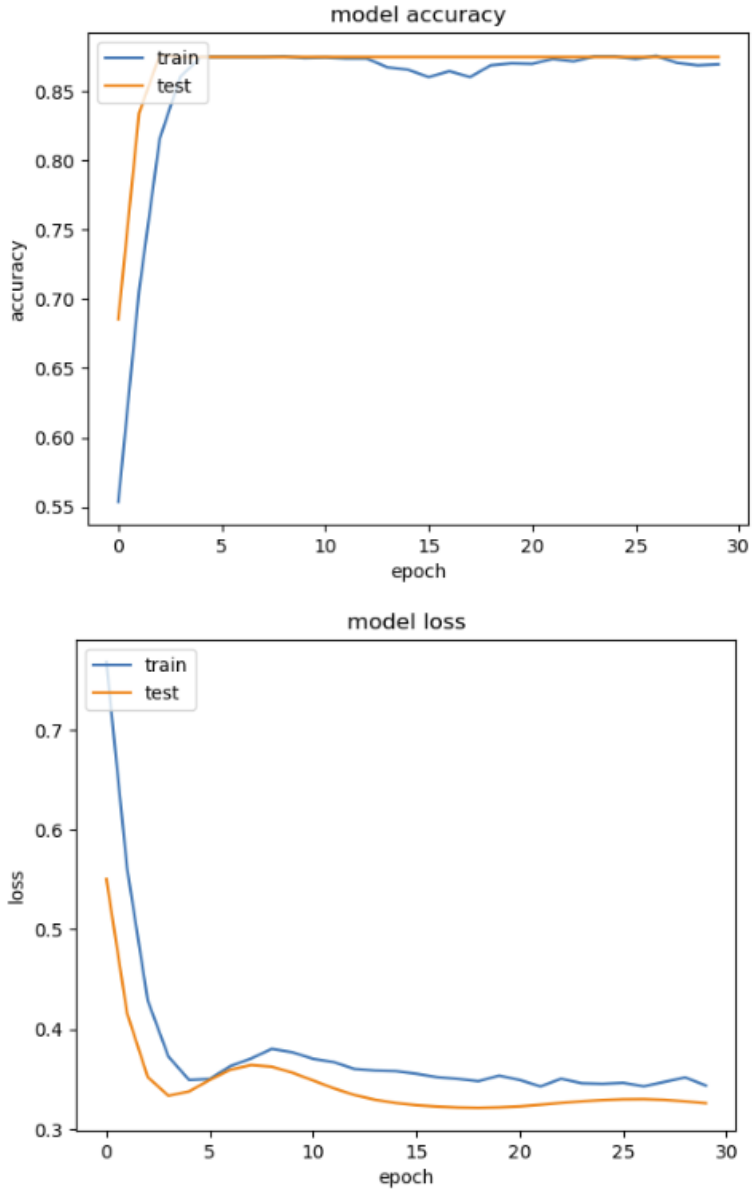

Figura 8.4: Gráfica de métricas del modelo de etiqueta con valor único

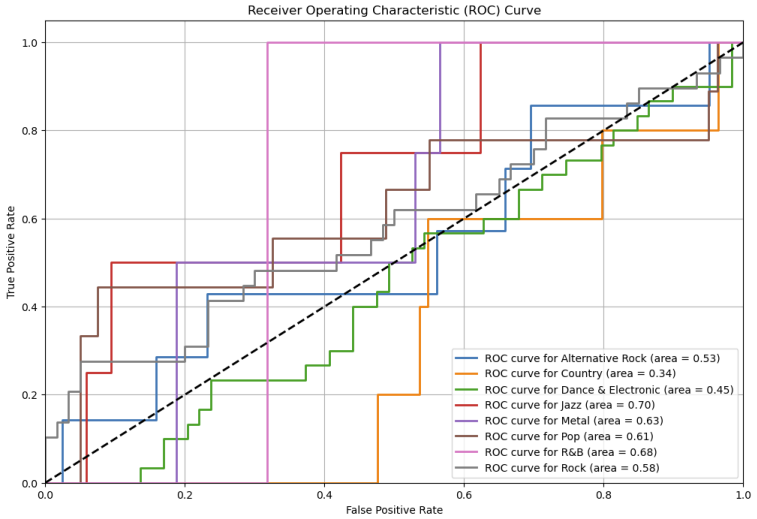

Figura 8.5: Gráfica de AUC - ROC de la red neuronal de MobileNetV2

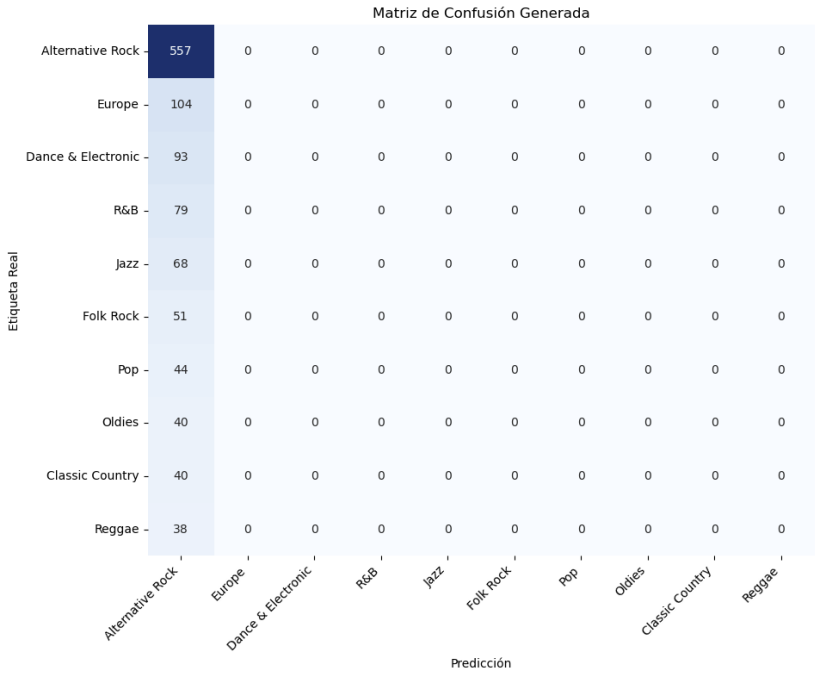

Figura 8.6: Matriz de confusión del modelo de etiqueta de valor único

Capítulo 9

Conclusiones y Trabajo Futuro

En este trabajo se ha pretendido explorar la aplicación de técnicas avanzadas de aprendizaje automático para analizar la correlación entre estilos visuales de portadas de álbumes y géneros musicales. El objetivo principal ha sido utilizar modelos de inteligencia artificial para comprender cómo las características visuales de las carátulas pueden relacionarse con el género musical.

Los resultados obtenidos revelan diferencias significativas en el rendimiento entre los modelos evaluados. El modelo de Random Forest mostró limitaciones en su capacidad para clasificar adecuadamente los géneros musicales, con AUC-ROC que variaban entre 0.50 y 0.59, indicando un desempeño cerca del azar para muchos géneros. Esto refleja una dificultad inherente del Random Forest para manejar datos de imágenes en comparación con enfoques más modernos. La elección de variables a partir de vectores de imagen no resultó ser efectiva, y el modelo mostró preferencia por ciertos géneros sin capturar correctamente la diversidad visual entre ellos.

En contraste, el modelo de red neuronal convolucional (CNN) demostró un rendimiento notablemente mejorado. Con un AUC de 0.87, el CNN logró una mayor capacidad de discriminación entre géneros musicales, aunque aún presenta áreas de mejora. El modelo alcanzó un AUC más alto en géneros como Jazz y R&B, pero mostró dificultades significativas en géneros como Country y Dance & Electronic. Esta diferencia en el rendimiento sugiere que algunas portadas de álbumes contienen características visuales menos distintivas o presentan similitudes que confunden al modelo.

A pesar del rendimiento superior del CNN, la capacidad de generalización del modelo fue limitada, y la precisión y el recall en la clasificación multi-etiqueta revelaron que el modelo tenía una alta precisión pero baja capacidad de recordar verdaderos positivos. Esta observación destaca la necesidad de mejorar la capacidad del modelo para identificar correctamente las clases menos frecuentes.

Para futuros trabajos, se podría tratar de explorar varias direcciones. En primer lugar, se podría mejorar la arquitectura del modelo CNN, incorporando técnicas más avanzadas como redes neuronales profundas o transfer learning con modelos preentrenados más robustos. La integración de métodos de regularización y técnicas para abordar el

desequilibrio en los datos también podría mejorar la capacidad de generalización y el rendimiento en géneros con menor representación. Además, la incorporación de características adicionales, como el análisis semántico de las imágenes o el uso de datos de audio complementarios, podría enriquecer el análisis y mejorar la precisión del modelo.

Otra línea de investigación podría involucrar la ampliación del dataset para incluir una mayor variedad de géneros y un número más equilibrado de ejemplos por género, lo que podría ayudar a los modelos a aprender mejor las diferencias visuales entre géneros. La creación de un modelo de clasificación híbrido que combine CNN con técnicas de aprendizaje de características específicas podría proporcionar una mejora adicional en la precisión y en la capacidad de capturar la diversidad visual en las portadas de álbumes.

Bibliografía

ABADI, M., AGARWAL, A., BARHAM, P., BREVDO, E., CHEN, Z., CITRO, C., CORRADO, G. S., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., GOODFELLOW, I., HARP, A., IRVING, G., ISARD, M., JIA, Y., JOZEFOWICZ, R., KAISER, L., KUDLUR, M., LEVENBERG, J., MANÉ, D., MONGA, R., MOORE, S., MURRAY, D., OLAH, C., SCHUSTER, M., SHLENS, J., STEINER, B., SUTSKEVER, I., TALWAR, K., TUCKER, P., VANHOUCKE, V., VASUDEVAN, V., VIÉGAS, F., VINYALS, O., WARDEN, P., WATTENBERG, M., WICKE, M., YU, Y. y ZHENG, X. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org.

ATTARD, M. https://insidelearningmachines.com/interpret_decision_trees/, 2021.

BARDEEN, B. W. H. . S. W., J. https://journals.aps.org/pr/abstract/10.1103/PhysRev.75.1208, 1949.

CALVO, D. https://abdatum.com/tecnologia/redes-neuronales-artificiales, 2017.

CHOLLET, F. Keras: A superpower for developers. 2024. Software available from keras.io.

DATAREPORTAL. Datareportal. https://datareportal.com/global-digital-overview, 2023.

GIT. Git. https://git-scm.com/, 2023.

GITHUB. Github. https://github.com/, 2023.

HARRIS, C. R., MILLMAN, K. J., VAN DER WALT, S. J., GOMMERS, R., VIRTANEN, P., COURNAPEAU, D., WIESER, E., TAYLOR, J., BERG, S., SMITH, N. J., KERN, R., PICUS, M., HOYER, S., VAN KERKWIJK, M. H., BRETT, M., HALDANE, A., DEL RÍO, J. F., WIEBE, M., PETERSON, P., GÉRARD-MARCHANT, P., SHEPPARD, K., REDDY, T., WECKESSER, W., ABBASI, H., GOHLKE, C. y OLIPHANT, T. E. Array programming with NumPy. *Nature*, vol. 585(7825), páginas 357–362, 2020.

HUNTER, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, vol. 9(3), páginas 90–95, 2007.

JOBLIB. Página oficial de la librería de Python joblib. 2023. https://joblib.readthedocs.io/en/stable/#(15/06/23).

KLUYVER, T., RAGAN-KELLEY, B., PÉREZ, F., GRANGER, B., BUSSONNIER, M., FREDERIC, J., KELLEY, K., HAMRICK, J., GROUT, J., CORLAY, S., IVANOV, P., AVILA, D., ABDALLA, S. y WILLING, C. Jupyter notebooks – a publishing format for reproducible computational workflows. En *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (editado por F. Loizides y B. Schmidt), páginas 87 – 90. IOS Press, 2016.

KOENIG, C. https://cs230.stanford.edu/projects_spring_2019/reports/18641024.pdf, 2019.

MOORE, G. E. https://www.cs.utexas.edu/~fussell/courses/cs352h/papers/moore.pdf, 1965.

ORAMAS S., B. F. . S. X., NIETO O. https://www.upf.edu/web/mtg/mumu, 2017.

PANDAS. Página oficial de la librería de Python Pandas. 2023. https://pandas.pydata.org/(15/06/23).

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M. y DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, vol. 12, páginas 2825–2830, 2011.

PEDREGOSA, V. G. G. A. M. V. T. B. G. O. . . D. , F. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MultiLabelBinarizer.html, 2011.

SIRAKORN. https://commons.wikimedia.org/wiki/File:Ensemble_Bagging.svg, 2020.

TURING, A. M. https://academic.oup.com/mind/article/LIX/236/433/986238, 1950.

VAN ROSSUM, G. y DRAKE, F. L. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.