Universidad de Valladolid

Escuela de Ingeniería Informática de Valladolid

**TRABAJO FIN DE GRADO**

Grado en Ingeniería Informática

Mención Ingeniería del Software

# Análisis Comparativo de Modelos de Aprendizaje Supervisado para el Reconocimiento de Emociones en el Habla


Autor:

**Víctor González Núñez**

Tutor:

**Joaquín Adiego Rodríguez**

Cotutora:

**Beatriz Juanes Mayfield**

“La inteligencia artificial no sustituirá a los humanos, pero aquellos que la usen sustituirán a quienes no lo hagan.” - Garry Kasparov

# Agradecimientos

Mi más sincero agradecimiento a mi tutor, Joaquín Adiego Rodríguez, por su dedicación, orientación y confianza desde el primer momento. Su capacidad para acompañarme y para guiarme ha sido clave en el desarrollo de este proyecto. Gracias por estar siempre disponible, incluso en los momentos en los que más me costaba avanzar.

También quiero dar las gracias a Natalia Martín Cruz, que ha estado muy presente durante todo el proceso, aportando su mirada, su criterio técnico y su cercanía. Ha sido una supervisión doblemente valiosa: por lo que sabía y por cómo lo transmitía.

Agradezco de manera especial Beatriz Juanes Mayfield, autora del Trabajo de Fin de Máster en el que se inspira y apoya gran parte de este trabajo. Su proyecto ha sido una referencia constante, y sin su claridad, su documentación y su generosidad compartiendo materiales, nada de esto habría sido igual.

A mis amigos y seres queridos, por supuesto. Gracias por sostenerme sin hacer preguntas cuando no podía responderlas, por entender mi silencio cuando estaba metido en el código, por celebrar conmigo los pequeños logros como si fueran grandes victorias. Vuestra compañía ha sido el verdadero sistema de soporte.

Y no quiero terminar sin extender mi reconocimiento a todo el profesorado del Grado en Ingeniería Informática, por compartir su conocimiento, por exigirnos más de lo que a veces creíamos posible y, sobre todo, por enseñarnos a pensar con criterio.

A todos, gracias.

# Resumen

Detectar emociones en la voz no es solo un reto técnico: es también una forma de acercarse a lo que las personas comunican sin decirlo. Este trabajo explora esa idea desde la inteligencia artificial, desarrollando un sistema capaz de analizar grabaciones y reconocer estados como la alegría, la tristeza, la ira o el miedo. A lo largo del proyecto se han combinado técnicas de procesamiento acústico y modelos de clasificación para encontrar una forma eficaz de interpretar la voz desde un punto de vista emocional. No se trata solo de obtener buenos resultados, sino de hacerlo con rigor, cuidando los datos, entendiendo los límites y asumiendo que detrás de cada señal hay una persona. El sistema es funcional, pero más allá de eso, este trabajo abre una línea que busca conectar tecnología y emoción sin perder de vista lo esencial: tratamos de enseñar a una máquina a escuchar, no solo a oír.

# Abstract

Detecting emotions in voice is not just a technical challenge: it's also a way of getting closer to what people communicate without saying it directly. This project explores that idea through the lens of artificial intelligence, developing a system capable of analyzing audio recordings and recognizing emotional states such as joy, sadness, anger or fear. Throughout the process, acoustic feature extraction and classification models have been combined to find an effective way of interpreting speech from an emotional perspective. It's not only about achieving good results, but about doing it rigorously: taking care of the data, understanding the limitations, and remembering that behind every signal, there's a person. The system is functional, but beyond that, this project opens a path that seeks to connect technology and emotion without losing sight of what really matters: we are trying to teach a machine to listen, not just to hear.

# Índice

# Índice de tablas

# Índice de figuras

# Capítulo 1 – Introducción y objetivos

## 1.1 Introducción

En los últimos años, el avance de los sistemas inteligentes ha transformado la forma en la que las personas se relacionan con la tecnología [80]. Uno de los retos más importantes que ha surgido en este proceso es dotar a estos sistemas de la capacidad para interpretar los estados emocionales de los usuarios. Esta habilidad resulta esencial para construir soluciones más empáticas, personalizadas y adaptativas, particularmente en áreas como la atención al cliente, la salud mental, la educación o los asistentes virtuales.

Entre los distintos canales de comunicación humana, la voz destaca por su capacidad para transmitir no solo contenido verbal, sino también una rica dimensión emocional. Aspectos como la entonación, el ritmo o la intensidad permiten inferir emociones con notable eficacia, incluso en ausencia de señales visuales. Esta cualidad convierte a la voz en una vía especialmente valiosa para el desarrollo de tecnologías que buscan comprender el estado afectivo de los interlocutores.

El objetivo principal de este Trabajo de Fin de Grado es diseñar y validar un sistema de detección automática de emociones a partir de grabaciones de voz, centrado en la identificación de cuatro emociones básicas: alegría, tristeza, miedo e ira. El proyecto se enmarca en una línea de investigación más amplia sobre análisis emocional, complementando trabajos previos como el Trabajo Fin de Máster (TFM) de Beatriz Juanes Mayfield, que propone una herramienta híbrida basada en voz, texto e imagen [1]. Sin embargo, con la idea de seguir avanzando en las líneas abiertas del trabajo previamente mencionado, este proyecto se focaliza exclusivamente en las señales vocales, explorando sus posibilidades técnicas y metodológicas para maximizar su aplicabilidad en entornos reales.

Este trabajo no se limita a la construcción de un sistema funcional, sino que también persigue comparar enfoques técnicos diversos para detectar emociones a partir de la voz, identificando qué métodos son más robustos y generalizables. Para ello, se han entrenado y evaluado cinco modelos representativos de distintos paradigmas: cuatro de Machine Learning (Random Forest, HistGradientBoosting, XGBoost y LinearSVC), basados en árboles de decisión o clasificadores lineales, y uno de Deep Learning (MLPClassifier), que emplea una red neuronal multicapa. De este modo, se busca sentar las bases para futuros desarrollos más amplios, que puedan integrarse en contextos reales (véase capítulo 3.2).

A lo largo de la memoria, se presenta una revisión del estado del arte, se describe el diseño del sistema y se analizan los resultados obtenidos, poniendo en valor las conclusiones alcanzadas y señalando líneas

claras de mejora. Así, este TFG contribuye no solo a nivel práctico, sino también teórico, al aportar conocimiento en un área emergente de gran relevancia tecnológica, social y empresarial.

## 1.2 Contexto

En el panorama actual de la inteligencia artificial, el análisis de emociones se ha consolidado como un área de investigación en expansión, con aplicaciones que van desde la mejora de la experiencia de usuario hasta la toma de decisiones personalizadas en tiempo real [2]. A medida que los sistemas conversacionales y los asistentes virtuales se integran cada vez más en la vida cotidiana, su habilidad para interpretar y responder a las emociones del usuario se vuelve clave para una interacción más humana y eficaz.

Diversas tecnologías como la visión artificial, el análisis de texto y el reconocimiento de voz se han aplicado con éxito para detectar emociones. Entre ellas, el análisis acústico del habla ofrece una vía especialmente prometedora por su carácter no intrusivo, al basarse en la voz natural del usuario. Esta propiedad lo hace adecuado para entornos en los que se valora la accesibilidad, como sistemas de atención automatizada o plataformas adaptativas.

En este marco, el presente Trabajo de Fin de Grado se sitúa en la intersección entre el procesamiento de señales de audio, la ingeniería del aprendizaje automático y la investigación aplicada a la interacción humano-computadora. La propuesta consiste en entrenar modelos supervisados capaces de identificar emociones a partir de grabaciones vocales, contribuyendo así al desarrollo de sistemas tecnológicos más empáticos, accesibles y funcionales, con potencial de aplicación directa en múltiples sectores.

## 1.3 Motivación

Este Trabajo de Fin de Grado nace de una oportunidad real de colaboración con el grupo de investigación INSISOC de la Universidad de Valladolid. Tras cursar la asignatura “Sistemas móviles” con el profesor Joaquín Adiego Rodríguez, surgió la posibilidad de que me tutorizara en este proyecto. La propuesta que me planteó se enmarca en una línea de investigación dirigida por la profesora Natalia Martín Cruz, centrada en el análisis del efecto de las emociones sobre las decisiones empresariales. En particular, se me propuso retomar y profundizar en una de las vías abiertas en el Trabajo Fin de Máster desarrollado por la profesora Beatriz Juanes Mayfield, centrado en el reconocimiento emocional multimodal.

A partir de ese contexto, la motivación principal de este TFG es contribuir al desarrollo de sistemas inteligentes que sean capaces de interpretar señales humanas de forma natural, con el objetivo de incrementar su utilidad práctica y su integración en entornos reales. La detección emocional a través de la voz abre nuevas posibilidades en múltiples ámbitos: desde mejorar la interacción con asistentes virtuales, hasta ofrecer apoyo en el ámbito educativo o complementarse con enfoques terapéuticos en contextos psicológicos. Además, en el marco específico de este proyecto, se busca que la tecnología desarrollada pueda aplicarse también al análisis de la toma de decisiones en entornos empresariales, donde el componente emocional a menudo influye de forma decisiva.

Desde un punto de vista académico y personal, este proyecto representa la oportunidad de integrar y aplicar conocimientos adquiridos a lo largo del grado en inteligencia artificial, análisis de datos y desarrollo de sistemas, dando forma a una solución funcional, con proyección práctica y vocación de continuidad investigadora.

## 1.4 Objetivo general

El objetivo general de este Trabajo de Fin de Grado es diseñar e implementar un sistema de Inteligencia Artificial capaz de detectar de forma automática las emociones de alegría, tristeza, miedo e ira a partir de grabaciones de voz. Para ello, se emplearán técnicas de aprendizaje automático y procesamiento de señales acústicas, con el fin de identificar patrones vocales representativos de cada estado emocional.

## 1.5 Objetivos específicos

Para alcanzar el objetivo general planteado, se definen los siguientes objetivos específicos:

- **Analizar el estado del arte** en el ámbito del reconocimiento de emociones a partir de la voz, incluyendo técnicas, desafíos actuales y aplicaciones relevantes.
- **Diseñar e implementar un flujo completo de procesamiento**, que abarque las etapas de preprocesamiento de audio y extracción de características acústicas significativas.
- **Desarrollar y entrenar distintos modelos de clasificación automática** mediante técnicas de aprendizaje supervisado.

- **Evaluar y comparar el rendimiento** de los modelos propuestos utilizando métricas estandarizadas, con el fin de seleccionar el enfoque más robusto y eficaz.
- **Documentar exhaustivamente el proceso técnico desarrollado** y plantear posibles líneas futuras de mejora, optimización e implementación real.

# Capítulo 2 - Planificación del Proyecto

## 2.1 Metodología de trabajo

Para estructurar adecuadamente el desarrollo de este Trabajo de Fin de Grado, se ha optado por combinar dos marcos metodológicos ampliamente aceptados en el ámbito de la ciencia de datos y la ingeniería de proyectos: CRISP-DM y SCRUM. Esta decisión responde a la necesidad de integrar un enfoque técnico riguroso para el tratamiento de datos con una gestión de proyecto ágil, que permita adaptarse a los cambios y al trabajo iterativo propio del desarrollo de sistemas de inteligencia artificial.

### Metodología CRISP-DM

CRISP-DM (*Cross Industry Standard Process for Data Mining*) es una de las metodologías más utilizadas a nivel industrial y académico para el desarrollo de proyectos de análisis de datos. Fue concebida para proporcionar un modelo estándar, independiente de herramientas o sectores, que sirviera como guía estructurada para abordar problemas complejos mediante técnicas de minería de datos o aprendizaje automático.

La metodología está formada por seis fases principales, como se muestra en la Ilustración 1, cuya ejecución no necesariamente sigue un orden lineal. En la práctica, el proceso admite iteraciones, retrocesos y ajustes continuos a medida que se profundiza en el análisis o surgen nuevas necesidades. A continuación, se describen en detalle cada una de sus fases:

1. **Comprensión del negocio**: el primer paso consiste en entender el problema que se quiere resolver desde una perspectiva organizativa y estratégica. Es decir, se traduce la necesidad del entorno (empresa, institución, investigación, etc.) en objetivos analíticos concretos. Esta fase implica la identificación de metas, restricciones, prioridades y métricas de éxito. También se define el alcance del proyecto, los criterios de finalización y los posibles riesgos.
2. **Comprensión de los datos**: una vez definido el problema, el siguiente paso es familiarizarse con los datos disponibles. Esto incluye tareas de recopilación, exploración inicial, identificación de patrones generales, análisis de calidad, y detección de valores anómalos o inconsistencias. El objetivo es evaluar la relevancia de los datos con respecto a los objetivos planteados, entender sus características internas y determinar si es necesario adquirir información adicional o transformarla.

3. **Preparación de los datos**: esta etapa representa uno de los bloques de trabajo más intensos en cualquier proyecto de ciencia de datos. Se encarga de transformar los datos brutos en una estructura que pueda ser procesada por los algoritmos de modelado. Involucra tareas como selección de atributos relevantes, limpieza de registros erróneos o incompletos, integración de múltiples fuentes, transformación de formatos, generación de nuevas variables derivadas, normalización o codificación de etiquetas. Una preparación adecuada es clave para el rendimiento posterior del modelo.

4. **Modelado**: en esta fase se eligen las técnicas y algoritmos de aprendizaje automático más adecuados para abordar el problema. Esto incluye definir qué tipo de modelo se necesita (clasificación, regresión…), configurar sus parámetros, entrenarlo con los datos preparados y evaluar su rendimiento preliminar. A menudo es necesario realizar múltiples iteraciones con diferentes algoritmos o ajustes para mejorar los resultados. Dependiendo del enfoque, también pueden implementarse pipelines automáticos, optimización de hiperparámetros o validación cruzada.

5. **Evaluación**: antes de considerar el modelo como definitivo, es necesario evaluar su rendimiento con rigor, utilizando métricas cuantitativas adecuadas al tipo de problema (por ejemplo, accuracy, precision, recall, F1-score, etc.). Esta fase no solo mide la eficacia del modelo, sino que también revisa si responde realmente al objetivo de negocio inicial. En muchos casos, se contrastan varios modelos y se escoge el más equilibrado entre precisión, interpretabilidad y eficiencia. También se identifican posibles limitaciones o sesgos, y se valora la estabilidad de los resultados.

6. **Despliegue**: la última fase consiste en la puesta en marcha del sistema desarrollado en su entorno final de aplicación. Esto puede implicar la creación de informes técnicos, la entrega del modelo para su integración en una plataforma, o la automatización de procesos para el uso continuo del sistema. Dependiendo del contexto, el despliegue puede ser técnico (por ejemplo, implementación en un servidor), organizativo (formación de usuarios finales), o documental (generación de informes y evidencias de validación). Además, en proyectos reales se contempla la necesidad de mantenimiento, retraining y actualización periódica del modelo con nuevos datos.

En el contexto de este proyecto, CRISP-DM permite articular de forma clara los apartados técnicos, aportando coherencia entre la teoría, el diseño del sistema y la validación experimental [71].

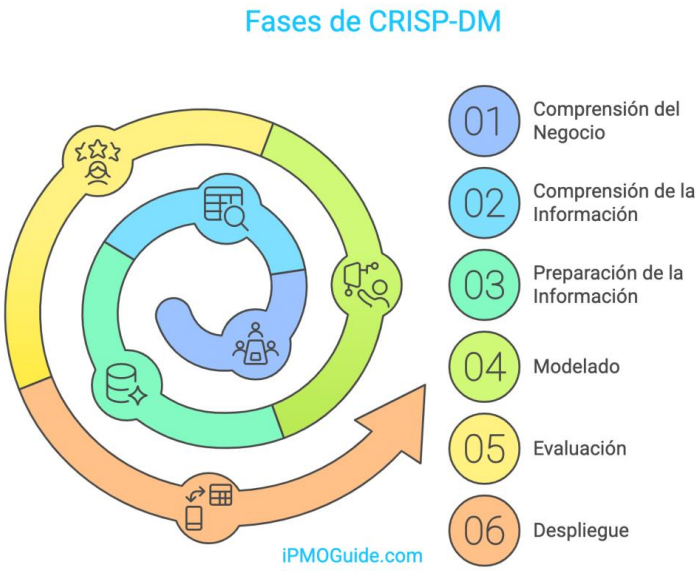


*Ilustración 1-Fases metodología CRISP-DM. Fuente [100]*

## Marco de trabajo ágil SCRUM

SCRUM es un marco de trabajo ágil orientado a la gestión iterativa e incremental de proyectos complejos. Aunque su aplicación suele darse en equipos multidisciplinares, en este proyecto se ha adaptado al contexto individual con muy buenos resultados [3].

El enfoque SCRUM se ha utilizado como estructura de gestión temporal, dividiendo el desarrollo en sprints aproximadamente quincenales, cada uno con objetivos específicos y entregables parciales. La planificación del backlog inicial, la priorización de tareas y las revisiones periódicas del trabajo con el tutor se han alineado con esta filosofía.

Además, algunos roles de SCRUM se han reinterpretado de forma simbólica en el marco del TFG:

- **Product Owner**: Representado por el propio estudiante en colaboración con el tutor, encargándose de mantener el enfoque del proyecto alineado con los objetivos académicos y técnicos.
- **Scrum Master**: Aunque no ha existido como figura formal, este rol ha sido asumido indirectamente por el estudiante, velando por la organización del trabajo, la resolución de bloqueos y la mejora continua del flujo de trabajo.

- **Equipo de desarrollo**: En este caso, constituido únicamente por el propio autor del TFG, responsable de la implementación técnica, la toma de decisiones y la ejecución práctica del sistema.

Esta aproximación ha permitido una gestión ágil y flexible, adaptada a los ritmos reales del calendario académico y a la naturaleza cambiante del desarrollo experimental. Gracias a SCRUM, ha sido posible ajustar objetivos, reorganizar prioridades y mantener una dinámica constante de avance, incluso en momentos de incertidumbre técnica.

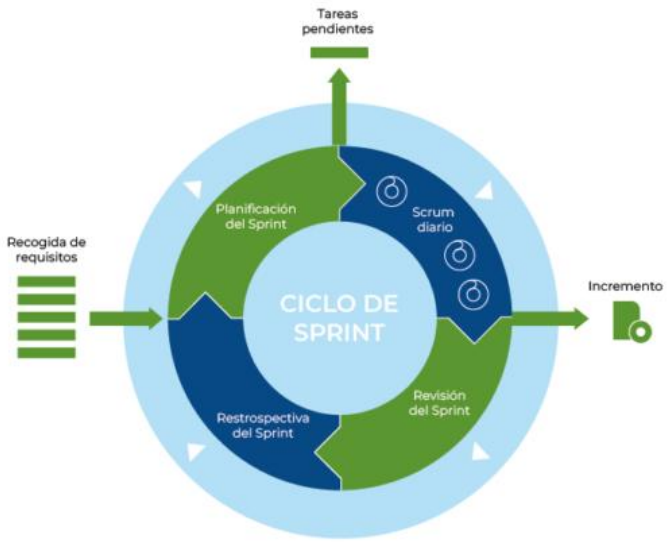


*Ilustración 2-Ciclo de sprint SCRUM. Fuente [101]*

La elección de CRISP-DM + SCRUM se justifica por su complementariedad. El primero ofrece un marco estructurado para los aspectos analíticos y técnicos, garantizando una trazabilidad clara en el proceso de desarrollo del modelo. El segundo permite acompasar la carga de trabajo a los ritmos del calendario académico y a la naturaleza iterativa de la experimentación en inteligencia artificial. Juntos, ofrecen una combinación eficaz de orden técnico y flexibilidad práctica.

## 2.2 Relación entre las fases CRISP-DM y los contenidos del TFG

Para estructurar de forma clara el desarrollo técnico de este Trabajo de Fin de Grado, se ha relacionado cada fase de la metodología CRISP-DM con los apartados específicos de la memoria. Esto permite seguir el proceso de trabajo con mayor facilidad, entender en qué momento se toman ciertas decisiones y ver cómo cada etapa contribuye al sistema final.

- **Comprensión del negocio**: Definición del problema, selección de emociones, revisión del estado del arte (capítulos 1 y 3).
- **Comprensión de los datos**: Estudio de los datasets (RAVDESS, SAVEE, TESS), duración media, sample rates (capítulo 5.1).
- **Preparación de los datos**: Extracción de características acústicas (capítulo 5.2).
- **Modelado**: Entrenamiento de los cinco clasificadores, optimización de parámetros con Optuna y GridSearchCV, uso de validación cruzada, generación de métricas (capítulos 5.3 y 6.1).
- **Evaluación**: Comparación de resultados, análisis de robustez, interpretación de matrices de confusión e importancias de características (capítulo 7).
- **Despliegue**: Propuesta de nuevas líneas de trabajo a futuro, predicción de audios externos etc. (capítulo 8).

## 2.3 Planificación temporal por sprints

La gestión del tiempo en este Trabajo de Fin de Grado no ha sido una simple cuestión de repartirse tareas. Se ha buscado una forma de trabajar que permitiera avanzar con orden, pero sin perder flexibilidad. Para lograrlo, se ha combinado una metodología estructurada como CRISP-DM con una metodología ágil, marco de trabajo como Scrum, adaptado a las particularidades de un proyecto académico y personal.

Scrum ha ofrecido un enfoque práctico: dividir el trabajo en bloques breves, llamados sprints, con objetivos definidos y margen para reajustar si algo no iba según lo previsto. Aunque esta metodología suele aplicarse en equipos, en este caso ha funcionado como una guía de ritmo. Cada sprint ha actuado como una pequeña etapa con entregables claros, revisiones regulares con el tutor y espacio para adaptarse cuando ha sido necesario.

A la vez, CRISP-DM ha dado forma a la parte más analítica del proyecto. Este modelo, habitual en ciencia de datos, organiza el desarrollo por fases lógicas: entender el problema, conocer los datos, prepararlos, construir modelos, evaluarlos y pensar en su posible despliegue. Cada uno de esos pasos ha tenido su reflejo directo en el trabajo realizado, y se ha traducido en un sprint específico, con tareas y objetivos ajustados a lo que exigía ese momento, como se observa en la Tabla 2.1:

| Sprint | Fase | Periodo | Objetivos principales | Resultados esperados |
|---|---|---|---|---|
| S1 | Comprensión del negocio | 18 - 25 feb | Reunión inicial con el tutor, definición del problema, revisión preliminar del TFM anterior, objetivos generales | Objetivos del TFG definidos, comprensión inicial del dominio y enfoque heredado del TFM |
| S2 | Comprensión de los datos | 26 feb - 10 mar | Estudio de los datasets (RAVDESS, SAVEE, TESS), análisis de *sample rates* y duración media, primeras cargas | Conocimiento de estructura, calidad y características generales de los datos |
| S3 | Preparación de los datos | 11 - 24 mar | Implementación de funciones de extracción de características (MFCC, Chroma, etc.), codificación de etiquetas | Pipeline básico de preprocesamiento funcional y probado |
| S4 | Modelado (Iteración 1) | 25 mar - 7 abr | Entrenamiento inicial de modelos (HistGB, XGB, RF, LinearSVC, MLP), visualización de métricas | Modelos funcionales con métricas base y predicciones sobre conjunto de prueba |
| S5 | Modelado (Iteración 2) | 8 - 21 abr | Optimización de hiperparámetros con Optuna, GridSearchCV y validación cruzada (5 y 10 *folds*) | Modelos optimizados y evaluados |
| S6 | Evaluación | 22 abr - 5 may | Generación de matriz de confusión, importancia de características y comparación | Informe de evaluación por modelo, identificación de puntos fuertes y limitaciones |
| S7 | Despliegue | 6 - 12 may | Predicción sobre audios nuevos y generación de archivos de resultados | Sistema completo con capacidad de predecir con audios |
| S8 | Documentación | 13 - 26 may | Redacción de la memoria: capítulos 1 - 4 | Primera versión sólida de los capítulos iniciales del TFG |
| S9 | Documentación | 27 may - 9 jun | Redacción de la memoria: capítulos 5 - 7 | Resultados integrados y listos para discusión |
| S10 | Documentación | 10 - 23 jun | Redacción del capítulo 8, segunda lectura, estandarización de estilo, revisión bibliográfica, anexos | Versión casi final de la memoria, lista para revisión externa |
| S11 | Documentación | 24 jun - 4 jul | Revisión final del tutor, preparación de defensa | Materiales preparados para entrega y exposición |

*Tabla 2.1-Sprints desarrollados en el proyecto*

### 2.3.1 Diagrama de Gantt

El siguiente diagrama de Gantt (ver Ilustracion 3) refleja la planificación temporal del proyecto de fin de grado, combinando un enfoque ágil basado en Scrum con las fases tradicionales del modelo CRISP-DM utilizado en ciencia de datos.

A continuación, se destacan algunos elementos clave del diagrama:

- Las fases principales del proyecto, correspondientes a las etapas de CRISP-DM (Comprensión del negocio, Comprensión de los datos, Preparación, Modelado, Evaluación y Despliegue), están representadas como épicas en color rojo. Además, se incluye una fase final de Documentación, resaltada en azul por su relevancia de cierre.
- Cada épica se descompone en sprints numerados del S1 al S11, todos con una duración aproximada de dos semanas, manteniendo una cadencia constante que favorece la planificación, revisión y entrega continua de valor, como propone Scrum.
- Dentro de cada sprint se detallan las tareas específicas previstas para ese periodo. Estas tareas se corresponden con acciones concretas como reuniones, implementación de funciones, entrenamientos de modelos o redacción de la memoria, entre otras.

Esta estructura jerárquica no solo facilita la visualización del flujo de trabajo a lo largo del tiempo, sino que también permite vincular cada bloque de actividad con los objetivos que persigue en el marco metodológico. Además, favorece el control de avances y la trazabilidad de decisiones a lo largo del desarrollo del proyecto.

El diagrama ha sido elaborado con *GanttProject*, una herramienta de planificación de proyectos de código abierto que permite representar tareas, dependencias y cronogramas de forma clara y exportable, facilitando así su integración en esta memoria [72].

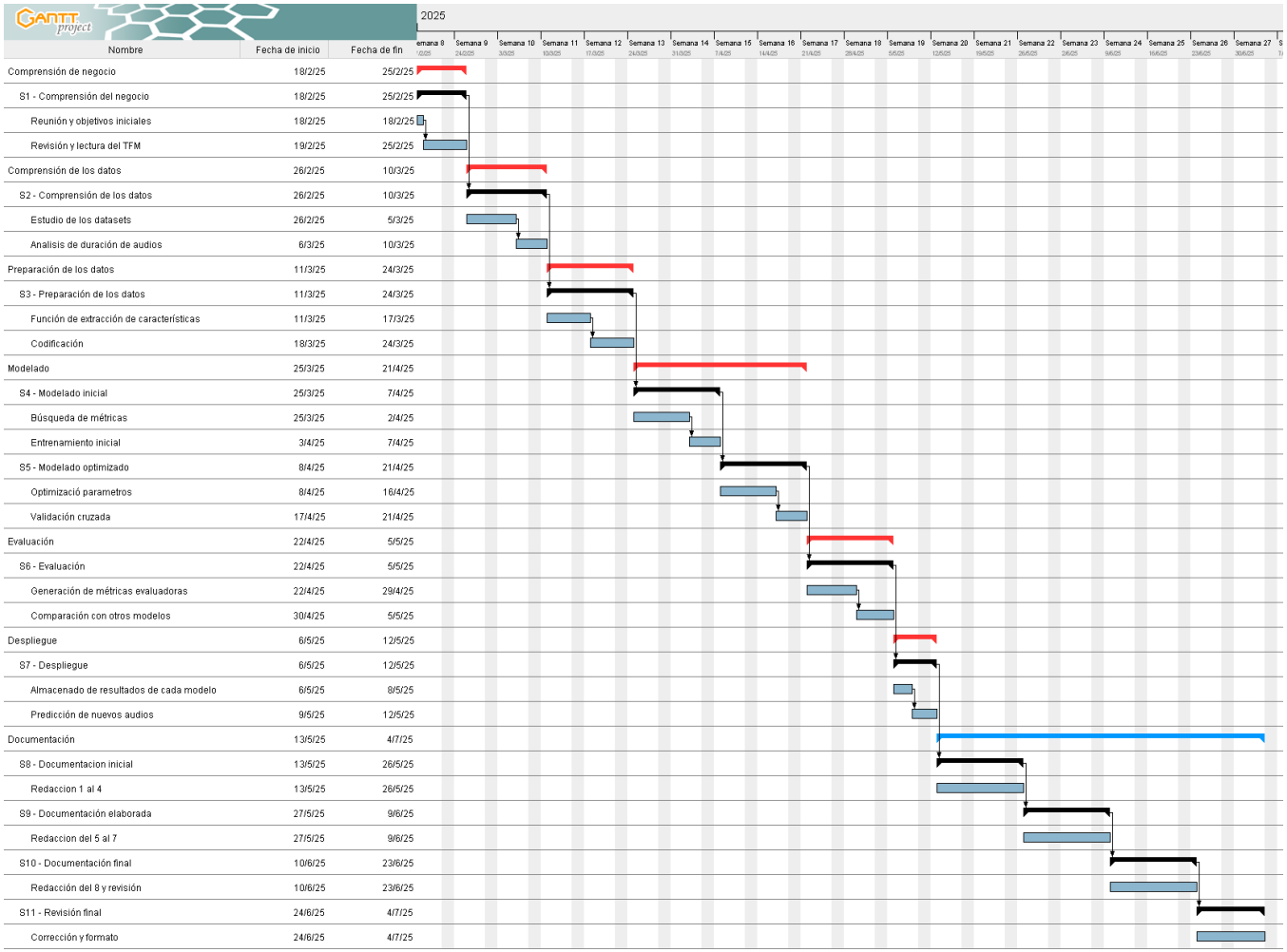


*Ilustración 3-Diagrama de Gantt. Creado con GanttProject*

## 2.4 Gestión de riesgos

Se identificaron y analizaron distintos tipos de riesgos que podían comprometer el cumplimiento de los objetivos establecidos o alterar la planificación prevista. Para su gestión, se siguió el enfoque propuesto por el Project Management Institute (PMI) en el estándar PMBOK, en el cual se define un riesgo como *"un evento o condición incierta que, de ocurrir, tiene un efecto positivo o negativo sobre uno o más objetivos del proyecto"* [4].

La gestión de riesgos en este TFG se abordó a través de un análisis cualitativo, en el que cada riesgo fue evaluado según su probabilidad de ocurrencia y su impacto potencial sobre el proyecto. Ambos criterios se codificaron mediante una escala trivalente: Alta, Media o Baja.

Este proceso de evaluación resultó especialmente relevante en el contexto académico, donde los recursos y tiempos son limitados, y el proyecto requiere la ejecución de múltiples tareas técnicas (implementación de modelos, análisis de resultados, redacción estructurada, entre otras) bajo un calendario acotado.

### 2.4.1 Matriz de probabilidad e impacto

Para sistematizar la priorización de los riesgos se utilizó una matriz de probabilidad e impacto, una herramienta clásica en la gestión de proyectos que permite visualizar el nivel de severidad combinando ambos factores [5].

A continuación, se presenta dicha matriz (Tabla 2.2), utilizada como base para categorizar los riesgos del proyecto:

| Impacto \ Probabilidad | Baja | Media | Alta |
|---|---|---|---|
| Alta | Media | Alta | Alta |
| Media | Baja | Media | Alta |
| Baja | Baja | Baja | Media |

*Tabla 2.2-Matriz de probabilidad e impacto*

Esta matriz establece que los riesgos ubicados en la esquina superior derecha (Alta probabilidad y Alto impacto) deben ser tratados con la máxima prioridad, mientras que aquellos en la zona inferior izquierda (Baja probabilidad y Bajo impacto) pueden simplemente monitorizarse.

### 2.4.2 Principales riesgos del proyecto

Con base en este análisis, se identificaron los riesgos más relevantes, mostrados en la Tabla 2.3 que podrían afectar al desarrollo del TFG. En la siguiente tabla se recogen estos riesgos clasificados según su probabilidad e impacto, junto con las estrategias adoptadas para su tratamiento. Los niveles están codificados con colores: rojo (Alta), amarillo (Media), verde (Baja), con el objetivo de facilitar su lectura visual.

| ID | Riesgo | Probabilidad | Impacto | Severidad | Estrategia y acciones |
|---|---|---|---|---|---|
| R1 | Dificultades en la comprensión del TFM original y sus conceptos | Media | Alta | Alta | Lectura activa, reuniones con la autora del TFM |
| R2 | Problemas técnicos en la implementación de modelos de IA | Alta | Media | Alta | Enfoque incremental con pruebas aisladas |
| R3 | Falta de rendimiento computacional durante entrenamiento de modelos | Media | Alta | Alta | Optimización de código |
| R4 | Desequilibrio entre clases o baja calidad de algunos audios | Alta | Media | Alta | Validación cruzada y métricas ponderadas |
| R5 | Falta de tiempo para redacción final y maquetación del documento | Media | Alta | Alta | Sprints específicos de redacción y margen temporal |
| R6 | Cambios de criterio o nuevos requisitos por parte del tutor | Baja | Media | Baja | Comunicación constante y adaptación tras revisiones |
| R7 | Errores acumulados por no versionar bien el código | Media | Media | Media | Versionado por carpetas y copias de seguridad semanales |
| R8 | Dificultades para identificar las librerías adecuadas para el desarrollo en Python | Media | Media | Media | Investigación inicial y apoyo del tutor para definir el stack; se utilizó una lista de librerías sugeridas por el tutor |
| R9 | Retrasos en la comunicación con el tutor o falta de feedback a tiempo | Media | Media | Media | Planificación con margen, seguimiento proactivo por correo y reuniones periódicas |

*Tabla 2.3-Riesgos del proyecto*

## 2.5 Estimación de costes

La estimación de costes en un proyecto de ingeniería permite valorar de forma aproximada los recursos necesarios para su desarrollo, lo que resulta especialmente útil en contextos profesionales donde se desea escalar o replicar el trabajo. Aunque este Trabajo de Fin de Grado no ha requerido financiación externa ni adquisición de licencias específicas, se ha llevado a cabo una evaluación estructurada de los costes implicados, tanto materiales como humanos. Esta valoración incluye el hardware y software utilizados, así como una estimación del esfuerzo temporal y económico si este sistema se desarrollara en un entorno profesional.

### 2.5.1 Costes materiales

El desarrollo del sistema se ha llevado a cabo íntegramente en un entorno local, utilizando un equipo personal sin necesidad de infraestructuras en la nube ni licencias de pago. Tanto las herramientas como las librerías utilizadas han sido de código abierto, lo que permite reproducir el entorno de trabajo sin costes adicionales.

#### Hardware

El sistema se ha desarrollado íntegramente en un equipo local sin necesidad de servidores externos ni entornos cloud. El dispositivo utilizado ha sido un portátil ASUS TUF Gaming A15 FA507NVR, que por sus prestaciones ha permitido entrenar modelos de aprendizaje automático, procesar señales acústicas y generar visualizaciones sin limitaciones de rendimiento.

Este equipo incorpora las siguientes especificaciones técnicas:

- **Procesador**: AMD Ryzen 7 7435HS (8 núcleos, 16 hilos)
- **Memoria RAM**: 32 GB DDR5 4800 MHz (2x16GB)
- **Almacenamiento**: SSD NVMe de 1 TB
- **Tarjeta gráfica**: NVIDIA GeForce RTX 4060 con 8 GB GDDR6
- **Sistema operativo**: Windows 11 Home

Aunque se ha utilizado un equipo personal ya disponible, se ha estimado un coste proporcional por desgaste asociado al uso durante el proyecto. Para ello, se ha considerado una vida útil razonable de 7 años para este tipo de dispositivo, con un uso medio de 5 horas diarias durante 5 días a la semana. Esto supone aproximadamente 9.100 horas de uso total en ese periodo. Dado que el TFG ha requerido unas 300 horas efectivas de trabajo, se ha calculado el coste proporcional con base en el precio estimado del equipo (1.300 €):

$$Coste\ de\ desgaste = \left(\frac{300\ h}{9100\ h}\right)\ x\ 1300\ € = 42{,}86€$$

Por tanto, el coste estimado de uso del equipo durante el desarrollo del sistema asciende a 42,86 €, como referencia orientativa para valorar su contribución técnica dentro del conjunto de costes del proyecto

### Software

Todo el software ha sido gratuito y de código abierto, por lo tanto, el coste para este apartado es de 0€. A continuación, se detallan las principales herramientas utilizadas en la Tabla 2.4:

| Software \ Librería | Función principal | Licencia | Coste |
|---|---|---|---|
| Python 3.12 | Lenguaje de programación | Open Source | 0 € |
| Visual Studio Code | Entorno de desarrollo (IDE) | Open Source | 0 € |
| NumPy | Manejo de vectores de características | BSD | 0 € |
| Librosa | Extracción de características acústicas | ISC | 0 € |
| Scikit-learn | Clasificadores, validación, métricas | BSD | 0 € |
| XGBoost | Modelo de boosting | Apache 2.0 | 0 € |
| Matplotlib | Visualización de gráficos y métricas | PSF | 0 € |
| Tqdm | Visualización de progreso en procesos | MPL | 0 € |

*Tabla 2.4-Estimación de coste software*

## 2.5.2 Costes humanos (esfuerzo estimado)

Desde el 18 de febrero hasta el 4 de julio se han ejecutado once sprints de trabajo, cubriendo todas las fases del modelo CRISP-DM. El esfuerzo total invertido se estima en 300 horas, repartidas entre

desarrollo, análisis, validación y documentación. También se adjuntan las horas que realmente se han aplicado.

En la Tabla 2.5 se muestra la distribución de tiempo por cada fase del CRISP-DM, y se incluye la documentación:

| Fase CRISP-DM | Horas estimadas | Horas reales |
|---|---|---|
| Comprensión del negocio | 20 h | 20 h |
| Comprensión de los datos | 43 h | 50 h |
| Preparación de los datos | 34 h | 40 h |
| Modelado | 67 h | 70 h |
| Evaluación | 29 h | 30 h |
| Despliegue | 17 h | 17 h |
| Documentación | 90 h | 120 h |
| Total | **300 h** | **347 h** |

*Tabla 2.5-Estimación temporal y real por fase (se incluye Documentación)*

## 2.5.3 Simulación de costes profesionales

Aunque este proyecto ha sido realizado de forma individual en el marco de un Trabajo de Fin de Grado, es posible estimar su coste humano si se hubiera desarrollado en un entorno profesional. Para ello, se han considerado perfiles cualificados asociados a cada fase del ciclo CRISP-DM, junto con una estimación salarial media por hora basada en informes del portal de InfoJobs [82].

A efectos de simulación, se ha utilizado un coste medio por hora en función del perfil técnico, como se detalla en la Tabla 2.6:

| Fase CRISP-DM | Perfil profesional simulado | Salario medio estimado (€/h) |
|---|---|---|
| Comprensión del negocio | Analista funcional de IA | 28 €/h |
| Comprensión de los datos | Ingeniero de datos | 26 €/h |
| Preparación de los datos | Técnico en procesamiento de señales | 25 €/h |
| Modelado | Ingeniero de Machine Learning | 30 €/h |
| Evaluación | Científico de datos | 28 €/h |
| Despliegue | Técnico de integración / Dev ML | 24 €/h |
| Documentación | Redactor técnico especializado | 22 €/h |

*Tabla 2.6-Salarios profesionales estimados*

## Estimación de coste por fase

A partir de las horas estimadas en cada fase del proyecto y del coste medio por perfil profesional, se ha elaborado la Tabla 2.7 que desglosa el coste simulado por etapa. Esta distribución permite visualizar de forma clara qué partes del desarrollo implicarían un mayor esfuerzo económico si el sistema hubiera sido implementado en un entorno profesional.

| Fase CRISP-DM | Horas estimadas | Precio / hora | Cálculo | Coste estimado |
|---|---|---|---|---|
| Comprensión del negocio | 20 h | 28 €/h | 20 × 28 | 560 € |
| Comprensión de los datos | 43 h | 26 €/h | 43 × 26 | 1.118 € |
| Preparación de los datos | 34 h | 25 €/h | 34 × 25 | 850 € |
| Modelado | 67 h | 30 €/h | 67 × 30 | 2.010 € |
| Evaluación | 29 h | 28 €/h | 29 × 28 | 812 € |
| Despliegue | 17 h | 24 €/h | 17 × 24 | 408 € |
| Documentación | 90 h | 22 €/h | 90 × 22 | 1.980 € |
| Total simulado | | | | **7.738 €** |

*Tabla 2.7-Coste simulado por etapa*

Aunque este trabajo no ha requerido financiación externa ni contratación de servicios, estimar los costes permite hacerse una idea más concreta del esfuerzo real que hay detrás. Tanto el análisis de herramientas como la simulación del tiempo de dedicación aportan una visión cercana a lo que supondría llevar este mismo desarrollo a un entorno profesional. Más allá de los números, este ejercicio sirve para valorar el trabajo invertido, medir su impacto potencial y dejar la puerta abierta a que, en un futuro, el sistema pueda evolucionar, escalarse o integrarse como parte de un proyecto más grande

# Capítulo 3 - Estado del arte

Este capítulo reúne los conocimientos teóricos y técnicos que sirven de base para el desarrollo del proyecto. A lo largo de los siguientes apartados se revisan los fundamentos del reconocimiento de emociones en el habla, los modelos de clasificación más habituales, los principales retos que plantea este campo y los criterios que se han seguido en este trabajo para seleccionar las emociones a estudiar. También se incluyen algunas aplicaciones prácticas que demuestran el valor real de esta tecnología en distintos contextos.

## 3.1 Fundamentos del reconocimiento de emociones en el habla

El reconocimiento de emociones en el habla (*Speech Emotion Recognition*, SER) es una disciplina de la inteligencia artificial centrada en identificar el estado emocional de una persona a partir de su voz. Diversos estudios han demostrado que la voz humana no solo transmite información verbal, sino también señales acústicas que reflejan el estado afectivo del hablante. Como señala el estándar académico, *"Speech Emotion Recognition (SER) is defined as the process of inferring human emotions from speech signals using techniques such as feature extraction, selection, and classification, often employing machine learning algorithms"* [6].

El proceso SER se estructura habitualmente en dos fases. En la primera se realiza la extracción de características relevantes de la señal de audio, como los coeficientes cepstrales en la escala Mel (MFCC), la energía, el *zero-crossing rate* o el centroide espectral. En la segunda, estas características se emplean como entrada para modelos de aprendizaje automático que clasifican la emoción predominante en el fragmento de voz analizado [2].

Una de las ventajas más destacadas del SER es su carácter no intrusivo, lo que permite su uso en entornos reales sin necesidad de imagen o texto. Según Wang y Yin (2023), *"acoustic-based emotion recognition is more robust in real-time applications and requires less intrusive data"* [2].

En el Trabajo Fin de Máster de Beatriz Juanes Mayfield (2024) se presenta un sistema híbrido de análisis emocional que integra voz, texto e imagen. Dentro de ese enfoque, la autora dedica un capítulo específico al reconocimiento emocional a partir del habla, destacando su relevancia y justificación técnica. En sus palabras:

*"Este campo se conoce como Reconocimiento de Emociones en el Habla (SER, por sus siglas en inglés: Speech Emotion Recognition) y se centra en desarrollar algoritmos y modelos de Deep Learning para analizar la prosodia de las palabras."* [1].

*"Entonar de distinta manera las palabras está asociado a distintos estados de ánimo como alegría, tristeza, enfado, miedo, entre otras."*[1].

Estas afirmaciones refuerzan el valor de las características prosódicas como indicadores emocionales. A diferencia del contenido semántico, que puede ser ambiguo o neutro, la modulación del habla constituye una vía eficaz para detectar emociones básicas como la alegría, la tristeza, el miedo o la ira.

Este TFG adopta dicha perspectiva y se centra exclusivamente en la dimensión acústica como canal de entrada. Se plantea el diseño, entrenamiento y validación de modelos que, a partir de grabaciones vocales, sean capaces de detectar automáticamente emociones, incluso en contextos variados y con diferentes locutores.

### 3.1.1 Aspectos técnicos avanzados del SER

El reconocimiento de emociones en el habla se apoya en la idea de que las señales acústicas, a través de variaciones prosódicas como el tono, la intensidad y el ritmo, permiten identificar estados emocionales.

En términos prácticos, un sistema SER se estructura habitualmente en cuatro fases:

1. **Preprocesamiento del audio**: Se lleva a cabo la limpieza de ruido, normalización de amplitud y segmentación temporal. El TFM describe este paso como esencial para garantizar una entrada limpia a los algoritmos, señalando la importancia de *"ajustar el volumen y la amplitud del audio para garantizar la consistencia en los datos"* [1].
2. **Extracción de características acústicas**: Se extraen parámetros como la frecuencia fundamental, el espectro de frecuencia y la duración de los segmentos. Este proceso incluye la Transformada de Fourier.
3. **Entrenamiento del modelo**: Se selecciona una arquitectura neuronal, como redes convolucionales (CNN), recurrentes (RNN) o híbridas. En el TFM se recomienda esta última opción por su capacidad para capturar tanto patrones espaciales como temporales en la señal de audio [1].
4. **Evaluación del modelo**: Se valida el sistema con datos nuevos no utilizados en el entrenamiento, evaluando la precisión en la detección emocional [1].

La autora menciona explícitamente la Transformada de Fourier como la herramienta que *"permite transformar la función de la onda desde el dominio del tiempo al dominio de las frecuencias"*, descomponiéndola en *"sus frecuencias puras"* para facilitar su análisis [1]. También detalla cómo la escala de Mel transforma las frecuencias para alinearlas con la percepción humana, destacando que *"los valores introducidos a la red neuronal sean más representativos de cómo los percibe una persona"* [1].

### 3.1.2 Introducción a los modelos de clasificación empleados

En este apartado se realiza una breve introducción a los modelos de clasificación que se han decidido utilizar en este trabajo (consultar el capítulo 5.3 para más información), los cuales son:

- **Random Forest**: Algoritmo que combina múltiples árboles de decisión para mejorar la precisión y reducir el riesgo de sobreajuste. Cada árbol se entrena con una muestra aleatoria del conjunto de datos, y la predicción final se obtiene por votación mayoritaria [7].
- **XGBoost**: Implementación optimizada del algoritmo de *gradient boosting* que destaca por su eficiencia y capacidad de regularización [8].
- **HistGradientBoostingClassifier**: Variante del *gradient boosting* que utiliza histogramas en lugar de valores continuos, lo que permite un entrenamiento más rápido y eficiente, especialmente en conjuntos de datos grandes [9].
- **Support Vector Machine (SVM)**: Algoritmo de aprendizaje supervisado que busca el hiperplano que maximiza la separación entre clases. Es eficaz incluso en espacios de alta dimensión [10].

  En este trabajo, la implementación utilizada corresponde a LinearSVC de scikit-learn [76], una versión optimizada del clasificador SVM con kernel lineal. Esta clase permite un entrenamiento eficiente en problemas multiclase sin necesidad de transformar explícitamente los datos mediante kernels no lineales (consultar sección 5.3.6 para más detalle).
- **Perceptrón Multicapa (MLP)**: Tipo de red neuronal *feedforward* compuesta por varias capas de nodos conectados, capaz de modelar relaciones no lineales entre las características de entrada y las salidas [11].

### 3.1.3 Desafíos en el reconocimiento emocional

A pesar de los avances significativos, el reconocimiento de emociones en el habla (SER) enfrenta una serie de desafíos técnicos que aún limitan su aplicación en entornos reales.

Uno de los principales retos es la variabilidad en la expresión emocional entre individuos. Factores como el idioma, la cultura o la personalidad afectan la forma en que se manifiestan las emociones. *"Variability in emotional expressions across cultures, languages, and individuals poses a substantial hurdle"* [12].

El rendimiento de los sistemas de SER se ve afectado negativamente por ambientes acústicos ruidosos o no controlados. La presencia de ruido de fondo, reverberaciones y otras distorsiones acústicas puede degradar la calidad de las señales de voz, dificultando la extracción precisa de características emocionales. Como se indica en la literatura, *"[...] when background noise is present because it can obstruct voice signals"* [13].

Además, existe una falta de datos reales y espontáneos. Aunque abundan los datasets con grabaciones actuadas, los que contienen emociones naturales en contextos cotidianos son escasos. Un estudio lo resume así: *"Spontaneous datasets for Speech Emotion Recognition (SER) are scarce and frequently derived from laboratory environments or staged scenarios, such as TV shows"* [14].

Para paliar estos problemas, se investigan estrategias como la normalización acústica y, especialmente, la generación de datos sintéticos mediante técnicas de aumento de datos. Una de las aproximaciones más recientes y efectivas en este ámbito es el uso de redes generativas adversarias (GANs). Tal y como señalan los autores de un estudio específico: *"We propose modifications in the original network architecture and the training process to improve the quality of the generated spectrograms. [...] To the best of our knowledge, this is the first time GANs are used to address the problem of data imbalance through data augmentation in the context of SER or other audio classification task"* [15].

Este enfoque consiste en generar espectrogramas artificiales que representan clases emocionales minoritarias, con el objetivo de mitigar el desbalance en los conjuntos de entrenamiento. Según el mismo trabajo, los resultados experimentales demuestran que esta técnica mejora el rendimiento general de los modelos de clasificación emocional.

### 3.1.4 Criterios de selección de emociones

Este trabajo se ha centrado en el análisis de cuatro emociones: alegría, tristeza, miedo e ira. La decisión no es aleatoria ni se basa solo en su frecuencia de estudio, sino en cómo se expresan estas emociones a través de la voz y en qué medida resultan útiles en contextos reales.

El psicólogo Paul Ekman propuso una clasificación ampliamente reconocida que incluye seis emociones básicas: alegría, tristeza, miedo, ira, asco y sorpresa [69]. Todas ellas, según sus estudios, son universales, se reconocen en diferentes culturas y comparten ciertas expresiones faciales. Pero no todas funcionan igual en el plano acústico. El asco, por ejemplo, se manifiesta más a través del rostro que del habla, lo que complica su detección usando únicamente la señal de voz. La sorpresa, por su parte, tiende a ser breve, difusa y a veces difícil de diferenciar de otras emociones con patrones similares. Por eso, quedaron fuera del conjunto analizado.

En paralelo, la teoría de la inteligencia emocional desarrollada por Daniel Goleman pone el foco en emociones como la alegría, la tristeza, el miedo y la ira [70]. Las considera claves porque afectan directamente a cómo las personas se relacionan, deciden o enfrentan situaciones cotidianas. Esta perspectiva no solo refuerza su importancia psicológica, también las convierte en candidatas idóneas para modelos que busquen entender el estado emocional en tiempo real.

La elección final responde, por tanto, a una combinación de factores. Se han priorizado emociones que, además de tener un respaldo teórico sólido, presentan rasgos acústicos claros y son relevantes en la interacción persona-máquina. Su análisis permite construir sistemas más útiles, adaptables y cercanos a cómo las personas realmente comunican cómo se sienten.

## 3.2 Aplicaciones prácticas del SER

El reconocimiento de emociones en el habla (SER) no solo es objeto de estudio académico, sino que ya se aplica activamente en sectores clave donde comprender el estado emocional del usuario mejora la interacción, la seguridad o la eficacia del servicio.

#### Asistentes virtuales y atención al cliente

Los asistentes de voz y chatbots modernos integran SER para interpretar el tono emocional del usuario y adaptar sus respuestas. Esto permite que los sistemas respondan de forma empática en situaciones de

frustración, confusión o satisfacción. Según Wired (2018), empresas como MetLife utilizan análisis emocional en tiempo real para mejorar el rendimiento de sus agentes de atención: *"Agents see a notification if they start speaking more quickly, a caller is silent for a long time, or the caller and agent talk over each other"* [16].

## Educación y aprendizaje en línea

En plataformas educativas, el SER puede detectar señales de aburrimiento, motivación o confusión durante clases virtuales. Así, se adapta el ritmo o contenido para mejorar el aprendizaje. Esta tecnología forma parte del llamado affective computing, que busca *"recognize, interpret, process, and simulate human affects [..] This is done using machine learning techniques that process different modalities, such as speech recognition…"* [17].

## Salud mental y bienestar emocional

El SER también se usa para monitorear estados emocionales en contextos de salud, como la detección temprana de depresión o ansiedad mediante patrones vocales. El estudio de Dhuheir (2021) indica que *"Automatically identifying the emotions can help build smart healthcare centers that can detect depression…"* [18].

## Automoción y seguridad vial

Los fabricantes de automóviles están integrando SER en sus asistentes de conducción. Estos sistemas detectan estados como estrés o somnolencia en la voz del conductor, y pueden ajustar funciones del vehículo o emitir alertas para prevenir accidentes. Cerence, por ejemplo, ha desarrollado asistentes de voz que interpretan emociones para mejorar la seguridad y experiencia al volante [19].

## 3.3 Retos y desafíos actuales en el SER

A pesar del progreso constante en los sistemas de reconocimiento de emociones en el habla (SER), siguen existiendo importantes desafíos que dificultan su aplicación práctica a gran escala. Estos retos abarcan desde limitaciones técnicas hasta cuestiones éticas y sociales.

### Variabilidad interindividual y cultural

La forma en que las emociones se expresan vocalmente varía en función de múltiples factores como el idioma, la cultura, la edad o la personalidad. Esto dificulta la generalización de los modelos [12].

### Condiciones acústicas adversas

La presencia de ruido de fondo, reverberaciones o mala calidad de grabación puede afectar seriamente la fiabilidad del sistema. Como advierten varios estudios, *"Noise has a significant impact on speech-emotion recognition systems' accuracy"* [20].

### Datasets limitados y poco representativos

La mayoría de los conjuntos de datos usados para entrenar modelos SER incluyen emociones actuadas, registradas en entornos controlados. Esto contrasta con las emociones reales expresadas en contextos espontáneos. En el estudio de EMOVOME, se afirma que *"Laboratory settings provide controlled environments with some degree of spontaneity when used to simulate real-life scenarios [...]These datasets prioritize ecological validity but are typically restricted and come with challenges such as background noise and variability in recording conditions, called in-the-wild conditions."* [14].

## Diversidad lingüística

Muchos modelos SER están entrenados únicamente con datos en inglés, lo cual impide su eficacia en otros idiomas o acentos. Esto ha motivado investigaciones sobre modelos adaptativos multilingües: *"Most of the work in literature dealing with cross-cultural or cross-corpus studies is centered around training with one language/corpus and testing with another"* [21].

## Emociones complejas y ambiguas

Detectar emociones básicas es una tarea relativamente bien resuelta, pero cuando se trata de identificar sentimientos complejos (como nostalgia, ironía o ansiedad leve), los modelos actuales aún presentan un rendimiento limitado. Según un análisis reciente, *"Recognizing emotions from speech is a daunting task due to the subtlety and ambiguity of expression"* [22].

## Privacidad y ética

El uso de SER plantea preocupaciones sobre la recopilación, almacenamiento y uso de datos emocionales sensibles. La legislación europea, como el Reglamento General de Protección de Datos (GDPR) y la Ley de Inteligencia Artificial (*AI Act)*, establece requisitos estrictos para garantizar que los sistemas de inteligencia artificial respeten los derechos fundamentales de las personas, la cual también refuerza la necesidad de transparencia y consentimiento informado [23]. (Para más información, consultar el apartado 8.9).

## Adaptación en tiempo real

Para que los sistemas de reconocimiento de emociones en el habla (SER) sean efectivos en aplicaciones prácticas como vehículos, atención al cliente o dispositivos portátiles, deben procesar señales de voz en tiempo real y adaptarse dinámicamente a cambios en el contexto, ruido ambiental y variaciones de acento. Esto requiere modelos altamente robustos y eficientes.

Recientes investigaciones han explorado enfoques multimodales que combinan características de audio y texto, enriquecidas con señales prosódicas y espectrales, para mejorar la precisión del SER en

condiciones naturales. Por ejemplo, el estudio de Ferreira (2025) [24] presenta un sistema que integra modelos de audio de última generación con codificadores de texto, utilizando características como la frecuencia fundamental (F0) y espectrogramas Mel para capturar mejor las emociones expresadas en el habla espontánea. Aunque este trabajo no se centra directamente en la adaptación en tiempo real, sus métodos contribuyen a la creación de sistemas SER más resilientes y adaptables en entornos del mundo real.

# Capítulo 4 - Análisis y diseño del sistema

El sistema desarrollado para la detección automática de emociones a partir de grabaciones de voz se ha estructurado siguiendo una arquitectura modular, reutilizable y estandarizada. Esta organización permite comparar con rigor distintos modelos de clasificación bajo un mismo entorno funcional y facilita la trazabilidad de los resultados obtenidos.

Para gestionar el desarrollo del sistema se adoptó un enfoque metodológico basado en SCRUM, adaptado al contexto de un proyecto individual. Dado que SCRUM suele organizar el trabajo en torno a historias de usuario, se optó por representar los objetivos mediante requisitos funcionales y no funcionales, una alternativa más adecuada en este caso por su claridad y facilidad de estructuración técnica.

## 4.1 Arquitectura general del sistema

El diseño general del sistema se organiza en torno a cinco bloques principales: (1) carga y etiquetado de datos, (2) extracción de características acústicas, (3) entrenamiento y validación, (4) evaluación de resultados, y (5) predicción sobre nuevos audios. Esta estructura común se aplica de manera uniforme a todos los modelos implementados, diferenciándose únicamente los parámetros específicos de cada uno.

1. **Carga y etiquetado de datos**: se integran tres bases de datos ampliamente utilizadas en reconocimiento emocional: SAVEE, RAVDESS y TESS. Cada conjunto se incorpora con su esquema propio de etiquetas, pero se unifican bajo un mapeo común centrado en cuatro emociones objetivo: alegría, tristeza, ira y miedo.

2. **Extracción de características acústicas**: todos los modelos utilizan una única función común de extracción basada en la biblioteca Librosa, lo que garantiza consistencia en los datos de entrada. Esta función calcula una serie de descriptores acústicos relevantes para el análisis emocional de la voz, que se tratarán con mayor detalle en capítulos posteriores.

3. **Entrenamiento y validación**: cada modelo se entrena con una división estratificada 80/20 y se evalúa adicionalmente mediante validación cruzada. Los clasificadores basados en árboles (Random Forest, XGBoost, HistGradientBoosting) utilizan validación de 5 pliegues, mientras que modelos como LinearSVC emplean 10 pliegues, adaptándose a sus características. Algunos modelos, como MLP y SVC, requieren además normalización previa de los datos (se definen Pipelines con StandarScaler).

4. **Evaluación de resultados**: se aplican métricas estandarizadas para todos los modelos: precisión general, matriz de confusión, informe de clasificación y análisis de importancia de características. Los resultados se almacenan de forma organizada en carpetas por modelo y exportados tanto como gráficos (.png) como archivos de texto (.txt), lo que facilita la revisión y comparación.
5. **Predicción sobre nuevos audios**: el sistema incluye una función común que permite aplicar cualquier clasificador entrenado sobre audios externos, reutilizando el pipeline de extracción y manteniendo la coherencia con el procesamiento aplicado en entrenamiento.

## 4.2 Requisitos funcionales

Los requisitos funcionales definen las capacidades esenciales que debe cumplir el sistema de detección emocional para garantizar su operatividad, escalabilidad y fiabilidad. A continuación, se agrupan en la Tabla 4.1 según las funcionalidades principales del sistema.

| ID | Nombre | Descripción |
|---|---|---|
| RF1 | Carga estructurada de audios | Permite cargar archivos .wav desde carpetas de datasets (SAVEE, RAVDESS, TESS) respetando su estructura. |
| RF2 | Etiquetado automático de emociones | Asigna emociones (alegría, tristeza, ira, miedo) automáticamente en función del nombre o carpeta del archivo, configurado para los datasets seleccionados. |
| RF3 | Extracción de características acústicas | Extrae MFCC, Chroma, RMS, ZCR y centroide espectral, generando un vector por audio. |
| RF4 | Entrenamiento de clasificadores | Entrena modelos supervisados configurables: HistGradientBoosting, XGBoost, Random Forest, MLP, LinearSVC. |
| RF5 | Validación cruzada | Evalúa los modelos mediante validación cruzada de 5 o 10 pliegues, devolviendo métricas promedio y desviación estándar. |
| RF6 | Generación de métricas e informes | Genera informes, precisión, matriz de confusión e importancia de características exportables (.txt, .png). |
| RF7 | Predicción sobre audios externos | Permite predecir emociones en nuevos archivos .wav mostrando resultado por archivo y exportándolo. |
| RF8 | Escalado de datos | Aplica normalización con StandardScaler para MLP y SVC en entrenamiento y predicción. |
| RF9 | Organización modular | El sistema está implementado de forma modular, con un único script principal (train.py) que permite seleccionar el modelo mediante un argumento de línea de comandos. Los resultados de cada clasificador se almacenan en carpetas separadas. |

*Tabla 4.1-Requisitos funcionales*

## 4.3 Requisitos no funcionales

Los requisitos no funcionales (RNF) definen las condiciones bajo las cuales debe operar el sistema, determinando aspectos como la portabilidad, modularidad, rendimiento, trazabilidad o usabilidad. A continuación, en la Tabla 4.2, se detallan los principales requisitos identificados en el desarrollo del sistema de detección emocional.

| ID | Nombre | Descripción |
|---|---|---|
| RNF1 | Portabilidad | Ejecutable en entornos con Python 3.12 y bibliotecas necesarias. Compatible con Windows, Linux y Mac. |
| RNF2 | Modularidad del código | Estructura modular que permite reutilización de componentes y fácil mantenimiento del sistema. |
| RNF3 | Trazabilidad de resultados | Resultados organizados por modelo y exportados en formatos estándar para análisis y comparación. |
| RNF4 | Rendimiento razonable | Debe funcionar en equipos personales sin necesidad de hardware avanzado, con tiempos de entrenamiento moderados. |
| RNF5 | Claridad visual y reproducibilidad | Gráficos legibles con etiquetas claras y uso de semilla fija para garantizar replicabilidad. |
| RNF6 | Usabilidad y operatividad | Uso sencillo desde línea de comandos o IDE; mensajes informativos durante la ejecución. |
| RNF7 | Organización por modelo | Cada clasificador genera sus propios resultados en una carpeta independiente, siguiendo un formato común, aunque todos se ejecutan desde un único script configurable. |

*Tabla 4.2-Requisitos no funcionales*

## 4.4 Reglas de negocio

Las reglas de negocio definen restricciones internas del sistema que deben cumplirse en todo momento para asegurar su coherencia, trazabilidad y alineación con los objetivos del proyecto, las cúales son mostradas en la Tabla 4.3. Estas reglas no dependen de la implementación concreta, sino que reflejan decisiones clave tomadas durante el diseño del sistema.

| ID | Nombre | Descripción |
|---|---|---|
| RN1 | Procedencia de los datasets | El sistema solo puede entrenarse con conjuntos de datos públicos y validados en la literatura (por ejemplo: RAVDESS, SAVEE y TESS), garantizando su uso ético y legal. |
| RN2 | Conjunto fijo de emociones | La clasificación emocional se limita a un conjunto de cuatro emociones básicas (alegría, tristeza, miedo e ira), definidas previamente para todo el sistema. |
| RN3 | Consistencia en la evaluación | Todos los modelos se evalúan sobre la misma partición de datos de prueba, y con la misma configuración experimental, para asegurar la comparabilidad de resultados. |

*Tabla 4.3-Reglas de negocio*

## 4.5 Casos de uso del sistema

Para representar de forma clara y estructurada la interacción del usuario con el sistema, se ha definido un único caso de uso principal, ya que todo el flujo de trabajo está automatizado dentro del script train.py. Este script engloba internamente todas las operaciones necesarias: desde la carga y procesamiento de los datos hasta el entrenamiento, la validación y la predicción de emociones.

El usuario únicamente debe seleccionar el modelo deseado mediante el argumento *--model* y, si lo desea, introducir audios en la carpeta TestAudios para obtener predicciones personalizadas.

A continuación, en la Tabla 4.4, se describe el caso de uso correspondiente en formato tabla:

| | |
|---|---|
| **ID** | CU01 - Clasificación emocional mediante modelo seleccionado |
| **Descripción** | El usuario lanza el script principal (train.py) para ejecutar de forma automática todo el flujo de procesamiento: carga de datos, extracción de características, entrenamiento del modelo, validación, generación de informes y predicción sobre audios externos (si los hay). |
| **Actor** | Usuario |
| **Precondiciones** | El usuario ha indicado correctamente el modelo deseado mediante el argumento --model y ha colocado, si lo desea, archivos .wav en la carpeta TestAudios. |
| **Postcondiciones** | El sistema genera métricas, gráficos e informes en una carpeta específica del modelo dentro del directorio Results/, y, si hay audios externos, un archivo con las predicciones. |
| **Flujo principal** | 1.- El usuario ejecuta el script train.py indicando el modelo mediante el parámetro --model.<br><br>2.- El sistema carga automáticamente los audios de los datasets RAVDESS, SAVEE y TESS. |

| | |
|---|---|
| | 3.- Se etiquetan las muestras en función del nombre del archivo o la carpeta.<br><br>4.- Se extraen las características acústicas (MFCC, Chroma, RMS, ZCR y centroide espectral).<br><br>5.- Se entrena el modelo seleccionado con una partición 80/20 estratificada.<br><br>6.- Se realiza validación cruzada (5-fold o 10-fold, según el modelo).<br><br>7.- Se generan y guardan el informe de clasificación, la matriz de confusión y la importancia de características.<br><br>8.- Si existen audios en TestAudios, se realiza la predicción sobre ellos y se guardan los resultados.<br><br>9.- El sistema finaliza la ejecución. |
| **Flujos alternativos** | 3A.- Si un archivo .wav tiene una duración muy corta (menos de 1000 muestras), se descarta automáticamente y no se incluye en el conjunto de entrenamiento.<br><br>6A.- Si el modelo seleccionado es LinearSVC, se aplica validación cruzada de 10 pliegues en lugar de 5.<br><br>8B.- Si no hay audios en TestAudios, el sistema omite la predicción externa y continúa sin errores. |
| **Excepciones** | 1B.- Si el usuario no especifica un modelo con --model, el sistema no se ejecuta y muestra un mensaje de error.<br><br>1C.- Si el modelo indicado no está en la lista de opciones válidas (histgb, xgb, rf, svc, mlp), se lanza una excepción y el sistema no arranca. |

*Tabla 4.4-Caso de uso del sistema*

Se muestra a continuación en la Ilustración 4 un diagrama de casos de uso en notación UML que resume la interacción principal del usuario con el sistema. Dado que el funcionamiento está completamente automatizado, se ha definido un único caso de uso que abarca todo el flujo:

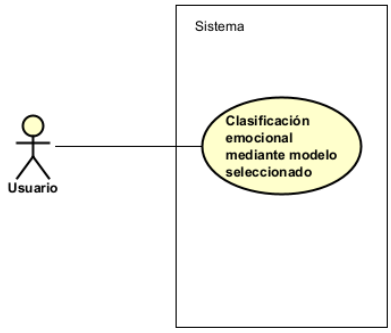


*Ilustración 4-Diagrama UML caso de uso*

## 4.6 Diagrama de secuencia

Para complementar el análisis estructural del sistema, se ha elaborado un diagrama de secuencia (ver Ilustración 5) que ilustra el comportamiento dinámico asociado al único caso de uso identificado: la ejecución completa del sistema desde la carga de datos hasta la predicción final. En este diagrama se representan los intercambios de mensajes más relevantes entre los distintos componentes principales del código, reflejando cómo se orquestan las funciones entre sí durante la ejecución.

Con el fin de mantener la claridad visual y centrarse en el flujo lógico del sistema, el diagrama se ha limitado únicamente a mostrar los mensajes que implican llamadas a clases o instancias desarrolladas dentro del propio proyecto, como los módulos train.py, o model_wrappers.py. Por tanto, no se representan explícitamente las funciones de bibliotecas externas como NumPy, scikit-learn o Librosa, que, si bien son esenciales para el procesamiento interno, no aportan valor descriptivo al comportamiento general del sistema en esta vista. A continuación, se muestra el diagrama:

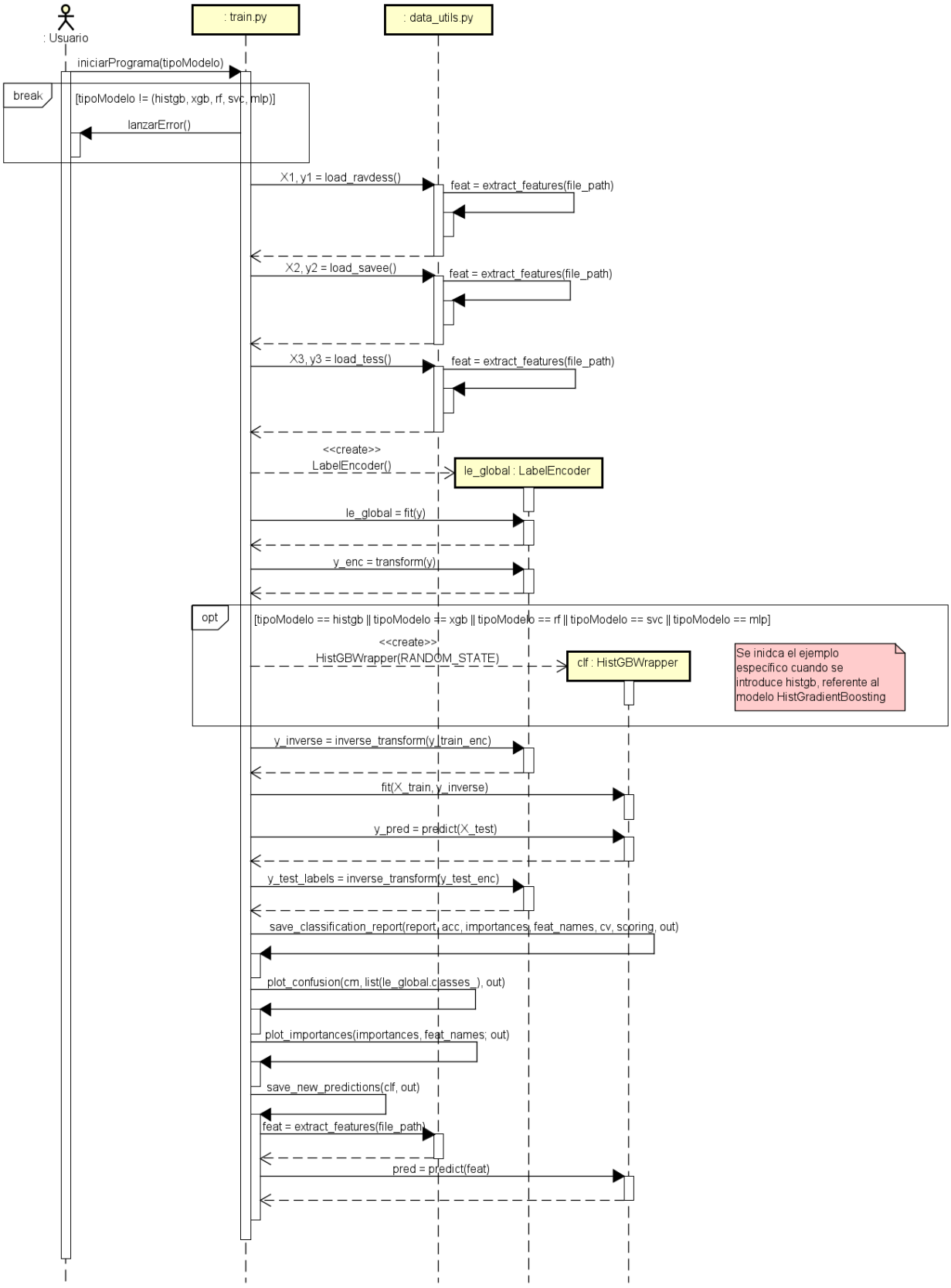


*Ilustración 5-Diagrama de secuencia del caso de uso*

## 4.7 Repositorio del código fuente

Con el objetivo de favorecer la transparencia, la reutilización y la reproducibilidad del sistema desarrollado, todo el código fuente del proyecto ha sido publicado en un repositorio público de GitHub, disponible en el siguiente enlace:

https://github.com/victorgleznun/supervised-ser-evaluation

En este repositorio se encuentra tanto la implementación completa como un archivo README.md que actúa como guía de uso.

# Capítulo 5 - Metodología experimental

## 5.1 Conjuntos de datos (Datasets)

Esta selección combina datasets de diversa longitud, características acústicas y condiciones de grabación, lo cual permite evaluar la robustez y generalización de los clasificadores. Al cubrir diferentes condiciones técnicas y demográficas, se enriquece el análisis experimental y se facilita la comparación con otros trabajos en el campo del *Speech Emotion Recognition* (SER).

### 5.1.1 Descripción general de los datasets utilizados

Para llevar a cabo un estudio robusto en el reconocimiento automático de emociones mediante el análisis acústico del habla, se seleccionaron cuidadosamente tres conjuntos de datos ampliamente validados y empleados en la literatura especializada: SAVEE, RAVDESS y TESS. Estos datasets destacan por su relevancia científica, calidad acústica, diversidad emocional y accesibilidad pública, factores esenciales para garantizar la reproducibilidad y la validez técnica del presente trabajo.

La Tabla 5.1 expone el número de muestras por emoción y por dataset que se ha analizado:

| Dataset | Tristeza | Alegría | Miedo | Ira | Total |
|---|---|---|---|---|---|
| SAVEE | 120 | 60 | 60 | 60 | 300 |
| RAVDESS | 184 | 184 | 184 | 184 | 736 |
| TESS | 400 | 400 | 400 | 400 | 1600 |

*Tabla 5.1-Muestras por emoción y dataset utilizado*

#### SAVEE (Surrey Audio-Visual Expressed Emotion)

El dataset SAVEE fue desarrollado por la Universidad de Surrey como parte de una iniciativa de investigación en reconocimiento de emociones humanas a partir de la voz y expresiones faciales. Este corpus fue grabado por cuatro hombres adultos nativos del inglés británico (identificados como DC, JE,

JK y KL), con edades comprendidas entre 27 y 31 años, todos ellos estudiantes de posgrado o investigadores de la propia universidad [25].

SAVEE contiene grabaciones correspondientes a siete emociones: ira, disgusto, miedo, alegría, tristeza, sorpresa y neutralidad. Las grabaciones se realizaron utilizando material textual cuidadosamente diseñado, compuesto por 15 frases distintas por emoción, incluyendo frases comunes, específicas y genéricas. En total, se generaron 120 grabaciones por locutor, resultando en un conjunto total de 480 archivos de audio.

Cada archivo tiene formato WAV, con alta calidad de audio, codificado a 44.1 kHz de tasa de muestreo. Debido a que se trata de un corpus compuesto exclusivamente por locutores masculinos, se recomienda su uso en combinación con otros datasets que incluyan voces femeninas, para mejorar la representatividad del sistema entrenado.

## RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song)

El dataset RAVDESS fue desarrollado por investigadores de la Universidad de Ryerson (Canadá) con el propósito de proporcionar un conjunto de datos de alta calidad para el análisis emocional tanto en el habla como en el canto. El corpus completo está disponible públicamente y ha sido validado perceptualmente por los autores en una publicación académica de acceso abierto [26].

Para este trabajo se empleó exclusivamente la porción de audio de voz (speech) en formato WAV, con codificación de 16 bits y 48 kHz de tasa de muestreo, lo que garantiza una calidad acústica excelente para el análisis de características espectro-temporales. Esta sección del dataset está compuesta por 1440 archivos de audio, resultantes de 60 grabaciones por cada uno de los 24 actores profesionales (12 hombres y 12 mujeres).

Cada locutor vocalizó dos frases distintas con múltiples expresiones emocionales. El conjunto incluye ocho emociones: neutral, calmada, feliz, triste, enfadada, temerosa, sorprendida y con disgusto. Cada emoción fue producida en dos niveles de intensidad (normal y fuerte), a excepción de la emoción neutral, que solo está disponible en intensidad normal. Los actores grabaron cada expresión con dos repeticiones y para dos frases distintas, lo que permite un diseño experimental balanceado y flexible.

## TESS (Toronto Emotional Speech Set)

El conjunto de datos TESS (Toronto Emotional Speech Set) fue desarrollado por la Universidad de Toronto con el propósito de estudiar la percepción de emociones en palabras habladas. Está compuesto

exclusivamente por voces femeninas, lo que lo convierte en un complemento valioso para otros conjuntos de datos más centrados en hablantes masculinos. Esta característica contribuye a mejorar la capacidad de generalización de los modelos entrenados, evitando posibles sesgos de género [27].

El dataset incluye grabaciones de dos actrices canadienses de 26 y 64 años, respectivamente. Ambas interpretaron 200 palabras objetivo que han sido insertadas en la frase portadora "Say the word _", produciendo cada palabra con siete emociones distintas: ira, disgusto, miedo, alegría, tristeza, sorpresa agradable y neutralidad. Las grabaciones se organizaron por actriz y emoción, en carpetas independientes, sumando un total de 2800 archivos de audio en formato .wav.

Cada archivo representa una única combinación de palabra, actriz y emoción. Las muestras de audio tienen una calidad elevada, aunque el formato de codificación presenta cierta variabilidad. El archivo fuente en Kaggle no especifica explícitamente la tasa de muestreo, pero según documentación adicional y versiones disponibles públicamente, la mayoría de las grabaciones están en 24414 Hz, con algunas excepciones a 96000 Hz.

### 5.1.2 Ventajas y desventajas

A continuación, se muestran las principales ventajas destacables de los sets de datos utilizados en este trabajo:

- **Alta calidad acústica**: los tres datasets seleccionados (SAVEE, RAVDESS y TESS) han sido grabados en condiciones acústicas controladas, una característica que la literatura considera esencial para asegurar la calidad de los datos utilizados en el reconocimiento emocional. La precisión en la extracción de descriptores espectro-temporales como MFCC, ZCR o centroides espectrales depende directamente de estas condiciones de grabación. Según Schuller, la robustez y reproducibilidad de los modelos está directamente relacionada con la fidelidad de los datos de entrada y la homogeneidad de las condiciones de captura [28].
- **Distribución equilibrada de emociones**: los tres conjuntos de datos han sido filtrados para contener una representación balanceada de las emociones básicas: alegría, tristeza, ira y miedo. Esta estructura permite diseñar experimentos justos, evitando el sobreajuste hacia clases mayoritarias.
- **Diversidad demográfica**: aunque no perfectamente equilibrado, RAVDESS incluye grabaciones de 12 hombres y 12 mujeres, mientras que TESS aporta dos voces femeninas de edades diferentes. Esta heterogeneidad favorece una mejor generalización de los modelos a distintas voces.

A pesar de sus ventajas, los conjuntos de datos empleados presentan también ciertas limitaciones:

- **Naturaleza actuada de las emociones**: los tres datasets contienen emociones actuadas por intérpretes en entorno controlado. Aunque estas grabaciones permiten replicabilidad experimental, no reflejan la espontaneidad ni el ruido emocional de interacciones reales, lo que limita su aplicabilidad directa en entornos naturales.
- **Pobre diversidad lingüística**: todos los datasets están grabados exclusivamente en inglés, lo cual reduce la capacidad del sistema para generalizar a locutores de otras lenguas o regiones culturales. Esto representa una barrera para su uso multilingüe sin retrabajo del pipeline. Como dicta un estudio: *"However, several challenges exist in the field of SER because of the subjective and context-dependent nature of emotion in addition to the individual differences and linguistic variations"* [29].
- **Inconsistencias técnicas (especialmente en TESS)**: a diferencia de SAVEE (44.1 kHz) y RAVDESS (48 kHz), TESS utiliza mayoritariamente una tasa de muestreo no estándar (24414 Hz), con algunas excepciones a 96000 Hz.
- **Limitada capacidad de generalización entre datasets**: uno de los desafíos más relevantes en el reconocimiento emocional a partir del habla es la pobre capacidad de generalización cuando un modelo se entrena en un conjunto de datos y se evalúa en otro diferente. Esto se debe a factores como las condiciones acústicas, los hablantes, la distribución emocional o el idioma de cada dataset. Según Parry (2019): "*The main drawback of this approach is that these models still tend to overfit the corpora in the training set and also display poor generalisation capabilities to out-of-domain data* " [30].

Esta última afirmación coincide con lo observado a lo largo del proyecto: incluso aquellos modelos que muestran un alto rendimiento durante el entrenamiento y la validación pueden ver reducido su desempeño cuando se enfrentan a grabaciones externas o procedentes de una base distinta. Este comportamiento revela una clara dependencia del modelo respecto al dominio de entrenamiento, lo que pone de manifiesto la necesidad de aplicar estrategias que mejoren su capacidad de generalización. Algunas de las vías más prometedoras en este sentido son la normalización entre datasets, el uso de técnicas de aumento de datos o incluso la aplicación de métodos de adaptación al dominio (para más información, ver Capítulo 8).

### 5.1.3 Consideración de otros datasets de interés

Durante la fase inicial de exploración se revisaron diversos conjuntos de datos ampliamente utilizados en la literatura sobre reconocimiento emocional en el habla. El objetivo era evaluar su idoneidad como

complemento o alternativa a los datasets finalmente seleccionados (SAVEE, RAVDESS y TESS). Sin embargo, tras un análisis preliminar, varios de estos corpus fueron descartados por distintas razones técnicas, metodológicas o prácticas.

- **CREMA-D (*Crowd-sourced Emotional Multimodal Actors Dataset*)**: este dataset cuenta con más de 7,000 clips de audio actuados por 91 actores (hombres y mujeres) en seis emociones básicas, lo que lo convierte en una base rica y diversa [31]. Sin embargo, en pruebas internas se observó una menor estabilidad en las métricas de precisión, especialmente cuando se integraba junto a SAVEE, RAVDESS y TESS. Esto podría atribuirse a diferencias en la codificación, calidad del audio o el tipo de entonación empleada por los actores. Por ello, se decidió excluirlo para mantener la coherencia técnica y garantizar resultados comparables y reproducibles.

- **IEMOCAP (*Interactive Emotional Dyadic Motion Capture Database*)**: este corpus es ampliamente valorado por su riqueza multimodal y su anotación emocional detallada, incluyendo categorías complejas como frustración o neutralidad [32]. No obstante, su uso requiere un proceso de solicitud y autorización específico, lo cual dificultó su inclusión en los plazos establecidos de este Trabajo Fin de Grado. Asimismo, su estructura orientada a conversaciones espontáneas introduce un grado de complejidad que difiere del enfoque controlado del resto de los datasets utilizados.

- **Emo-DB (*Berlin Database of Emotional Speech*)**: aunque es uno de los corpus más conocidos en Europa, el idioma de las grabaciones (alemán) limita su aplicabilidad directa en un sistema enfocado al inglés. Además, contiene un número reducido de muestras por emoción y solo nueve locutores, lo que podría comprometer la capacidad de generalización del modelo entrenado en contextos angloparlantes [33].

- **ASED (*Arabic Speech Emotion Dataset*)**: el dataset ASED contiene grabaciones de voz en árabe y ha sido utilizado en investigaciones centradas en contextos regionales [34]. A pesar de su calidad, se optó por excluirlo debido a la barrera lingüística, que afectaría negativamente la consistencia del pipeline técnico y dificultaría la comparación directa con otros trabajos centrados en inglés.

- **EmoV-DB (*Emotional Voices Database*)**: este conjunto de datos fue considerado por su enfoque en la síntesis de voz emocional y su disponibilidad en inglés. Sin embargo, al intentar reproducir los archivos de audio, se encontró que estaban codificados en un formato de punto flotante (*Float*) incompatible con las herramientas utilizadas en este proyecto. Esta incompatibilidad técnica impedía su integración sin realizar una conversión previa de los archivos, lo que habría requerido un tiempo adicional no contemplado en el cronograma del trabajo [66].

- **ASVP-ESD (*Audio, Speech and Vision Processing Lab Emotional Sound Database*)**: este corpus incluye una amplia variedad de sonidos emocionales, tanto hablados como no hablados, recopilados de fuentes como películas, programas de televisión y plataformas en línea. Aunque su riqueza y diversidad lo hacen atractivo, presenta ciertas limitaciones para este proyecto. En primer lugar, muchos de los audios contienen risas, sonidos no verbales o interacciones en chino, lo que introduce una variabilidad difícil de controlar. Además, la presencia de audios sin contenido lingüístico dificulta su uso en un sistema centrado en el análisis del habla [68].
- **JL-Corpus**: este conjunto de datos fue desarrollado para estudiar emociones primarias y secundarias en el habla del inglés de Nueva Zelanda. Aunque su diseño es interesante, se identificó que carece de muestras correspondientes a la emoción de miedo, lo que genera un desequilibrio en la representación emocional. Esta ausencia podría afectar negativamente al entrenamiento y evaluación del modelo, especialmente si se busca una clasificación equilibrada entre diferentes emociones [67].

En conclusión, la elección final de trabajar con SAVEE, RAVDESS y TESS se fundamenta en criterios de accesibilidad inmediata, calidad acústica uniforme, claridad en la estructura emocional y compatibilidad lingüística.

## 5.2 Extracción de características acústicas

Una vez preparado y etiquetado el conjunto de datos, el siguiente paso fundamental en el sistema consiste en transformar las grabaciones de audio en una representación numérica que pueda ser interpretada por los modelos de aprendizaje automático. Esta etapa, conocida como extracción de características acústicas, tiene un papel clave en la eficacia del reconocimiento emocional, ya que determina qué información del sonido se conserva y cómo se estructura.

### 5.2.1 Características utilizadas

La extracción de características acústicas constituye un componente fundamental en cualquier sistema de reconocimiento emocional en el habla. Estas características resumen, de forma cuantificable, las propiedades espectrales y temporales del audio, permitiendo a los modelos de aprendizaje automático capturar patrones vocales vinculados a distintos estados afectivos.

En este trabajo se ha optado por un conjunto de características comúnmente utilizadas en la literatura especializada en SER, tanto por su capacidad discriminativa como por su bajo coste computacional. Las características seleccionadas son:

- **Coeficientes Cepstrales en la Escala Mel (MFCC):** los MFCC son descriptores espectrales que representan la envolvente del espectro en una escala perceptual similar a la del oído humano. Se han calculado 13 coeficientes por archivo, promediados a lo largo del tiempo para obtener un vector de representación fijo. Estos coeficientes son altamente sensibles a las modulaciones vocales típicas de emociones como la tristeza (curvas descendentes), la ira (ataques espectrales marcados) o la alegría (entonación variable y amplia dinámica) [35].

  El cálculo de los MFCC parte de dividir el audio en ventanas cortas (20 - 40 ms) donde la señal puede considerarse estacionaria. A cada ventana se le aplica una Transformada Discreta de Fourier:

  $$S_i(k) = \sum_{n=1}^{N} x_i(n) \cdot h(n) \cdot e^{-j2\pi kn/N}$$

  Donde:

  - $S_i(k)$: componente espectral compleja del *frame* i en la frecuencia k
  - $x_i(n)$: muestra de la señal de audio en la posición n dentro de la ventana i
  - h(n): función de ventana (por ejemplo, ventana de Hamming), usada para suavizar los bordes del *frame*
  - N: número total de muestras en la ventana

  Esto tiene como objetivo obtener el espectro de frecuencias, al que se le aplica posteriormente un banco de filtros Mel, distribuidos de forma no lineal para simular la sensibilidad de la cóclea humana. Esta transformación se define como:

  $$M(f) = 1125 \cdot \ln(1 + \frac{f}{700})$$

  Tras filtrar el espectro, se toma el logaritmo de la energía de cada banda Mel (dado que la percepción de volumen es logarítmica, ver Ilustración 6), y se aplica una Transformada Discreta del Coseno (DCT) para obtener los coeficientes finales. Los 13 primeros son los que resumen la información relevante del espectro, siendo sensibles a variaciones vocales asociadas a emociones. Por ejemplo, la tristeza tiende a generar coeficientes más suaves y descendentes, mientras que la ira muestra picos más abruptos y la alegría presenta mayor dinamismo tonal [37].

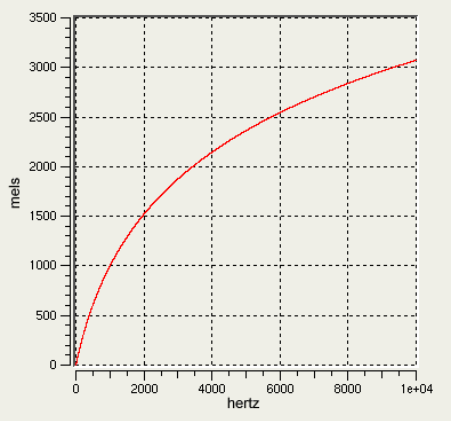


*Ilustración 6-Relación entre frecuencia en Hz y la escala Mel. Fuente [102]*

- **Chroma (12 bandas):** esta característica mide la energía contenida (ver un ejemplo concreto en la Ilustración 7) en cada uno de los 12 semitonos de la escala musical dentro de un espectro. Es útil para captar la estructura armónica del habla, particularmente en emociones que implican variaciones melódicas, como la alegría. El Chroma permite distinguir entre señales más tonalmente estables (como la tristeza) y otras más variables [36].

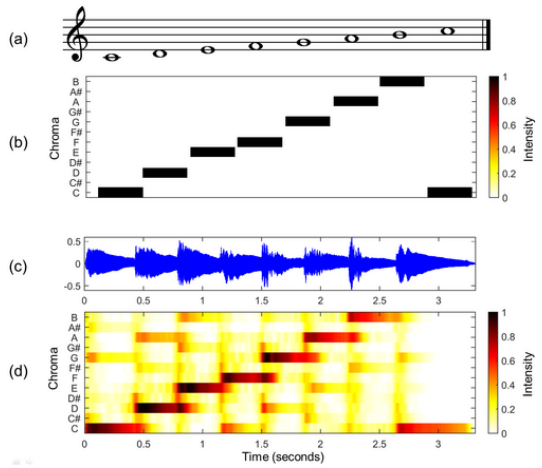


*Ilustración 7-(a) Escala ascendente pentagrama. (b) Vector de características Chroma para una escala ideal. (c) Representación temporal señal acústica. (d) Matriz Chroma obtenida. Fuente [103]*

- **Energía RMSE (*Root Mean Square Energy*):** proporciona una medida de la intensidad general del sonido en la grabación. Emociones como la ira o la alegría suelen tener valores RMS más altos, mientras que emociones como el miedo o la tristeza tienden a presentar una energía vocal reducida. Esta característica se asocia directamente con el volumen percibido y la fuerza expresiva de la emoción.

En otras palabras, mide la magnitud cuadrática media (como se muestra en la Ilustración 8) de la señal en una ventana, que se calcula a través de esta fórmula [37]:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}|x(n)|^2}$$

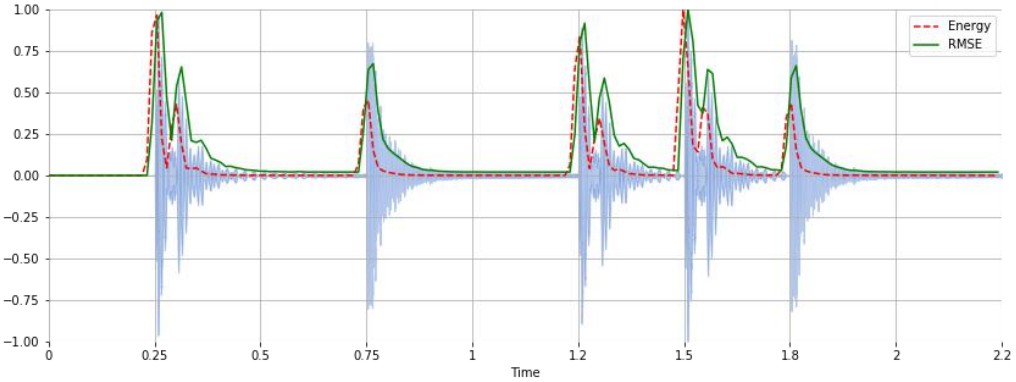


*Ilustración 8-Gráfico comparativo entre energía total y RMSE. Fuente [104]*

- **Tasa de Cruce por Cero (*Zero-Crossing Rate*, ZCR):** indica la frecuencia con que la señal cruza el eje cero (ejemplo mostrado en la Ilustración 9), lo que se relaciona con la aspereza o la complejidad espectral de la voz. Altos valores de ZCR son comunes en emociones agitadas (como la ira), debido a una mayor presencia de ruido y una alternancia rápida de señal.

  Se calcula como la frecuencia con que la señal cruza el cero [38]:

$$ZCR = \frac{1}{T-1}\sum_{n=1}^{T-1}1_{\{x[n]x[n-1]<0\}}$$

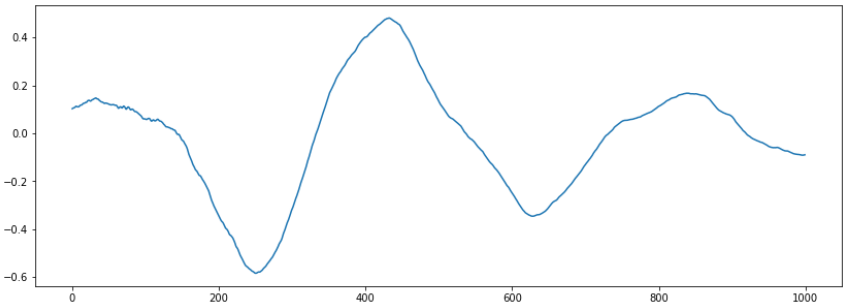

*Ilustración 9-ZCR de una señal audio. Fuente [113]*

- **Centroide Espectral:** este descriptor indica el punto medio del espectro, ponderado por amplitud (ver Ilustración 10). Se interpreta como el "brillo" o claridad percibida del sonido. Las emociones positivas como la alegría suelen presentar centroides más elevados, mientras que emociones como la tristeza o el miedo se asocian a frecuencias medias más bajas y patrones más oscuros o apagados [39].

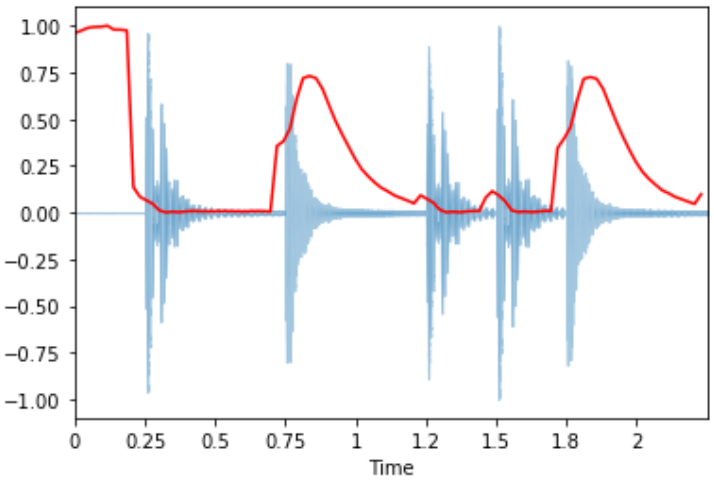


*Ilustración 10-Gráfico del Centroide Espectral de una señal de audio. Fuente [113]*

Este conjunto de cinco características ha sido implementado de forma uniforme para todos los modelos del sistema, a fin de mantener condiciones homogéneas en el entrenamiento, validación y predicción.

## 5.2.2 Justificación técnica para la selección de estas características

La selección de las características acústicas utilizadas en este sistema se fundamenta en su eficacia demostrada en numerosos trabajos de investigación en el ámbito del SER. Se optó por un conjunto equilibrado de descriptores que, en conjunto, permiten capturar las dimensiones clave de la señal emocional: contenido espectral, variación tonal, energía vocal y comportamiento frecuencial.

Los MFCC han sido ampliamente validados como descriptores fundamentales en la representación espectral del habla, ya que condensan información fonética y articulatoria que resulta crítica para diferenciar emociones, especialmente aquellas que alteran la prosodia o la dinámica espectral de la voz.

El Chroma, por su parte, permite capturar aspectos armónicos y tonales del habla, útiles en emociones como la alegría o el miedo, que presentan mayor variabilidad melódica frente a estados como la tristeza.

Las métricas de energía RMSE y ZCR ofrecen información cuantitativa sobre la intensidad vocal y la variabilidad temporal de la señal, dos aspectos esenciales para identificar emociones con carga expresiva elevada, como la ira, frente a emociones más apagadas como el miedo.

Finalmente, el centroide espectral complementa el análisis incorporando una medida relacionada con la distribución del brillo o claridad del sonido, reforzando así la discriminación emocional desde una perspectiva de percepción auditiva.

### 5.2.3 Funciones para extraer características acústicas

La extracción de las características acústicas se implementó utilizando la biblioteca Librosa, una herramienta ampliamente reconocida por su eficiencia y precisión en el procesamiento de señales de audio. En este trabajo se emplearon las siguientes funciones (mostradas en la Ilustración 11) para obtener las características utilizadas en los modelos:

```python
def extract_features(file_path):
    audio, sr = librosa.load(file_path, sr=None)
    if len(audio) < 1000:
        return None
    feats = []
    # MFCC (13)
    mfccs = librosa.feature.mfcc(y=audio, sr=sr, n_mfcc=13)
    feats.extend(np.mean(mfccs, axis=1))
    # Chroma (12)
    chroma = librosa.feature.chroma_stft(y=audio, sr=sr)
    feats.extend(np.mean(chroma, axis=1))
    # RMS Energy
    feats.append(np.mean(librosa.feature.rms(y=audio)))
    # Zero Crossing Rate
    feats.append(np.mean(librosa.feature.zero_crossing_rate(y=audio)))
    # Spectral Centroid
    feats.append(np.mean(librosa.feature.spectral_centroid(y=audio, sr=sr)))
    return np.array(feats)
```

*Ilustración 11-Captura función extract_features de data_utils.py de este proyecto*

- **librosa.feature.mfcc(y, sr, n_mfcc=13):** extrae 13 coeficientes cepstrales en la escala Mel. Requiere la señal de audio (y) y su tasa de muestreo (sr), que indica cuántas muestras por segundo tiene el audio.
- **librosa.feature.chroma_stft(y, sr):** calcula un cromagrama de 12 bandas a partir del espectro de la señal.
- **librosa.feature.rms(y):** obtiene la energía de la señal como valor cuadrático medio (RMS) por cada ventana de análisis.

- **librosa.feature.zero_crossing_rate(y):** devuelve la frecuencia con la que la señal cruza el eje cero.
- **librosa.feature.spectral_centroid(y, sr):** calcula el centroide espectral ponderado, que representa el "brillo" del sonido.

En todos los casos, la tasa de muestreo (sr) se detecta automáticamente a partir del archivo .wav original usando librosa.load(file_path, sr=None), lo que permite conservar la resolución y fidelidad original de cada grabación sin forzar un valor fijo. Estas funciones procesan los audios mediante ventanas cortas, aplicando transformaciones que permiten extraer descriptores relevantes de manera eficiente y sistemática [40].

Como medida de control de calidad, se descartaron automáticamente aquellos audios cuya duración era inferior a 1000 muestras, al considerarse demasiado cortos para una extracción de características fiable.

### 5.2.4 Preprocesamiento y codificación

Antes de entrenar cualquier modelo de clasificación, es fundamental preparar adecuadamente los datos tanto en términos de etiquetas como de escala de las variables numéricas. En este trabajo, se aplicaron dos transformaciones clave: la codificación de etiquetas emocionales mediante LabelEncoder [73], y la normalización de características numéricas utilizando StandardScaler [74]. Ambas operaciones se integraron en flujos consistentes mediante la clase Pipeline [75] de scikit-learn, cuando era necesario.

En primer lugar, las etiquetas correspondientes a las emociones (alegría, tristeza, ira, miedo) fueron convertidas a valores numéricos mediante la clase LabelEncoder de scikit-learn. Esta transformación permite representar cada categoría como un número entero, conservando su correspondencia original para poder decodificarla posteriormente. Este paso es necesario, ya que muchos modelos de aprendizaje automático no pueden trabajar directamente con variables categóricas en formato texto.

En segundo lugar, algunas arquitecturas empleadas (en concreto, el clasificador LinearSVC y la red neuronal MLP) requieren que las variables de entrada estén normalizadas para funcionar correctamente. Por ello, en estos modelos se integró un paso adicional de escalado mediante StandardScaler, que transforma las variables para que tengan media cero y desviación estándar uno. Esta estandarización es fundamental en modelos sensibles a la magnitud de los datos, ya que evita que variables con mayor rango dominen el proceso de aprendizaje. Para los modelos basados en árboles de decisión no fue necesario aplicar esta transformación, ya que su rendimiento no depende de la escala de las variables.

Ambos pasos (escalado y clasificación) se encapsularon en una estructura tipo Pipeline, lo que asegura que las transformaciones se apliquen de forma coherente durante el entrenamiento y en las

predicciones posteriores. Este enfoque es ampliamente recomendado para garantizar la consistencia y reproducibilidad del sistema.

## 5.3 Modelos de clasificación empleados

El sistema de detección emocional se ha construido íntegramente en Python. A lo largo de este apartado se describen, en primer lugar, las razones por las que se ha elegido este entorno de desarrollo, así como las principales librerías utilizadas durante la implementación. A continuación, se detallan los cinco modelos de clasificación empleados, incluyendo su funcionamiento básico y ventajas técnicas.

### 5.3.1 Introducción a Python como herramienta principal en la IA

Python se ha consolidado como una de las herramientas más versátiles y extendidas en el campo de la inteligencia artificial (IA) y el aprendizaje automático (Machine Learning), tanto en contextos académicos como profesionales.

Una de las principales ventajas de Python es su facilidad de aprendizaje y uso. Su sintaxis clara y legible permite a investigadores y desarrolladores centrarse en la lógica del modelo y los objetivos del experimento, en lugar de invertir tiempo en la gestión del lenguaje o en estructuras complejas. Esto resulta especialmente útil en entornos de investigación, donde la agilidad en la implementación y prueba de hipótesis es clave.

Además, Python cuenta con una amplia comunidad científica y técnica que ha generado una gran cantidad de recursos, desde documentación y ejemplos prácticos hasta foros de soporte y publicaciones académicas.

Otro aspecto destacable es la disponibilidad de librerías optimizadas para tareas concretas. Estas bibliotecas están ampliamente validadas en entornos académicos, lo que garantiza fiabilidad, eficiencia computacional y reproducibilidad de los resultados.

Por último, Python es portátil, gratuito y de código abierto, lo que lo convierte en una elección ideal para entornos de investigación universitaria. Su capacidad de integración con otros lenguajes y entornos permite escalar fácilmente los prototipos hacia soluciones más completas o entornos de producción [41].

### 5.3.2 Bibliotecas especializadas empleadas

Para llevar a cabo eficazmente la implementación, entrenamiento y evaluación rigurosa de los modelos predictivos desarrollados en este trabajo, se han utilizado diversas bibliotecas especializadas de Python. Estas herramientas proporcionan funcionalidades avanzadas en áreas clave como el aprendizaje automático, el procesamiento de señales acústicas, la optimización de hiperparámetros y la validación estadística.

Las bibliotecas seleccionadas han sido escogidas por su eficiencia computacional y su compatibilidad con proyectos de análisis de voz. A continuación, se describen las más relevantes:

#### NumPy

NumPy es una biblioteca esencial para el cálculo científico en Python, diseñada para el manejo eficiente de estructuras de datos numéricos multidimensionales llamadas *arrays*. Ofrece operaciones vectorizadas, álgebra lineal, estadística básica y funciones matemáticas avanzadas sobre grandes volúmenes de datos numéricos [42].

En este proyecto, NumPy se ha utilizado como soporte principal para almacenar y procesar las características acústicas extraídas de los audios. Su estructura *ndarray* permite gestionar grandes volúmenes de datos con rapidez y eficiencia, especialmente durante las etapas de preprocesamiento y escalado numérico antes del entrenamiento de modelos.

La elección de NumPy se justifica por su alto rendimiento computacional, su compatibilidad con otras bibliotecas clave como Librosa y scikit-learn, y su amplio respaldo en la comunidad científica. Estas cualidades garantizan un flujo de datos robusto, reproducible y técnicamente sólido en todas las fases del sistema.

#### Librosa

Librosa es una biblioteca especializada en Python para el análisis de señales de audio. Está diseñada para facilitar tareas como la extracción de características acústicas, el procesamiento musical y el análisis del habla, siendo ampliamente utilizada en proyectos científicos de reconocimiento emocional y clasificación auditiva.

En este proyecto, Librosa se ha empleado para extraer descriptores clave directamente desde archivos de audio, incluyendo MFCC, Chroma, energía RMS, tasa de cruce por cero (ZCR) y centroide espectral [40]. Estas características permiten representar de forma numérica y precisa el contenido acústico emocional de cada muestra.

Su elección se justifica por su alta eficiencia computacional, precisión en el análisis espectro-temporal y compatibilidad con bibliotecas como NumPy y scikit-learn.

## Tqdm

Tqdm es una biblioteca ligera de Python diseñada para mostrar barras de progreso en tiempo real durante la ejecución de tareas intensivas. Su integración sencilla la convierte en una herramienta muy útil para visualizar y monitorizar el avance del procesamiento de datos en aplicaciones científicas y de ingeniería [43].

En este proyecto, Tqdm se ha utilizado en la fase de carga y análisis de audios, donde era necesario procesar cientos de archivos por cada dataset. Su uso permitió tener una visión clara del progreso durante la extracción de características acústicas.

Gracias a esta visualización continua, Tqdm también ayudó a detectar posibles cuellos de botella o errores en tiempo de ejecución, lo que facilitó la depuración eficiente.

## Scikit-learn

Scikit-learn (sklearn) es una biblioteca de código abierto para aprendizaje automático en Python, especialmente diseñada para modelos supervisados y no supervisados, además de ofrecer utilidades de preprocesamiento, selección de modelos y evaluación [44].

En el proyecto, Scikit-learn desempeña un papel central en el pipeline técnico: gestiona la estratificación de datos, la validación cruzada, la optimización de parámetros (GridSearchCV) y el cálculo de métricas como *accuracy*, *recall*, *precision*, *F1-score* y matrices de confusión.

Se eligió por su eficiencia y extensiva documentación, siendo valorada en entornos académicos y profesionales por su robustez, consistencia y facilidad de integración con NumPy. Además, su licencia BSD (*Berkeley Software Distribution*) permite su uso libre en investigación y aplicaciones comerciales, garantizando resultados técnicamente sólidos y replicables en tareas de reconocimiento de emociones acústicas.

## Matplotlib

Matplotlib es una biblioteca de Python diseñada para la generación de gráficos y visualizaciones estáticas, animadas e interactivas. Es ampliamente utilizada en entornos científicos y académicos por su capacidad para representar datos de forma clara y personalizable [79].

En este trabajo, Matplotlib se ha empleado para la visualización de dos elementos clave: la matriz de confusión, que muestra gráficamente los aciertos y errores de cada modelo en la clasificación emocional; y la importancia de las características, que permite identificar qué descriptores acústicos tienen mayor peso en las predicciones del sistema.

Su elección se justifica por su compatibilidad directa con estructuras NumPy, su facilidad de integración en scripts automáticos y su capacidad para exportar gráficos en alta calidad, lo que ha permitido documentar visualmente los resultados obtenidos con cada clasificador.

## XGBoost

XGBoost (Extreme Gradient Boosting) es una de las bibliotecas más utilizadas actualmente cuando se trata de construir modelos de clasificación potentes y rápidos. Su popularidad no es casual: ofrece una implementación optimizada del algoritmo de *gradient boosting*, con mejoras clave en rendimiento, eficiencia y prevención del sobreajuste [81].

En este proyecto se ha utilizado como uno de los modelos base para el reconocimiento de emociones en voz, gracias a su robustez y flexibilidad. Permite ajustar con detalle aspectos como la profundidad de los árboles, la tasa de aprendizaje o la proporción de datos y características usados en cada iteración, lo que lo convierte en una herramienta ideal para tareas de afinado y experimentación.

Además, XGBoost se integra perfectamente con bibliotecas como scikit-learn y NumPy, lo que ha facilitado su uso en el pipeline general del sistema. Su rendimiento, incluso en entornos locales, ha sido excelente, y su amplia documentación ha ayudado a sacarle partido sin grandes complicaciones. En definitiva, ha sido una pieza importante del conjunto de modelos evaluados.

### 5.3.3 HistGradientBoostingClassifier

#### Introducción al modelo

El modelo HistGradientBoostingClassifier es una herramienta avanzada de clasificación automática que forma parte de la biblioteca Scikit-learn [9]. Este clasificador se basa en una técnica llamada *Gradient Boosting*. Esta técnica consiste en construir muchos árboles de decisión (modelos simples) de forma secuencial, de manera que cada nuevo árbol intenta corregir los errores que cometieron los anteriores. Al final, el modelo es el resultado de combinar todos los árboles, lo que permite obtener una predicción mucho más precisa.

Para entender este enfoque, es útil pensar en un ejemplo: si intentáramos adivinar la emoción en una grabación de voz y fallamos, el siguiente árbol que construimos pondrá más atención en esos errores y tratará de acertarlos. Al repetir este proceso muchas veces, se va mejorando progresivamente. Esta idea fue propuesta por el investigador Jerome Friedman en 2001 [45] y se ha convertido en uno de los métodos más eficaces y utilizados en problemas de clasificación complejos, como el reconocimiento de emociones.

La versión que se utiliza en este proyecto, llamada HistGradientBoostingClassifier, introduce una mejora muy importante: en lugar de usar los valores reales de los datos, los agrupa en intervalos o "cajones" llamados histogramas. Por ejemplo, si tenemos una característica como el volumen de la voz, este modelo no analiza cada número por separado, sino que los agrupa en rangos (por ejemplo, 0-10, 10-20, etc.). Esto permite que el modelo trabaje mucho más rápido sin perder precisión. Esta idea se llama entrenamiento basado en histogramas y es especialmente útil cuando se tienen muchos datos o muchas características, como ocurre en este proyecto, donde se utilizan descriptores acústicos como MFCC, Chroma, etc.

Además, este modelo tiene la capacidad de trabajar con valores ausentes (es decir, cuando falta algún dato) de manera automática, sin necesidad de rellenarlos manualmente. También es capaz de detectar relaciones complejas entre características, como por ejemplo cómo se combinan ciertos tonos de voz con la energía para expresar una emoción específica.

En la Ilustración 12 se muestra de forma esquemática el algoritmo de *boosting* basado en histogramas. Cada árbol de decisión se construye utilizando intervalos o *bins* en lugar de valores individuales, lo que permite acelerar el entrenamiento sin perder precisión:

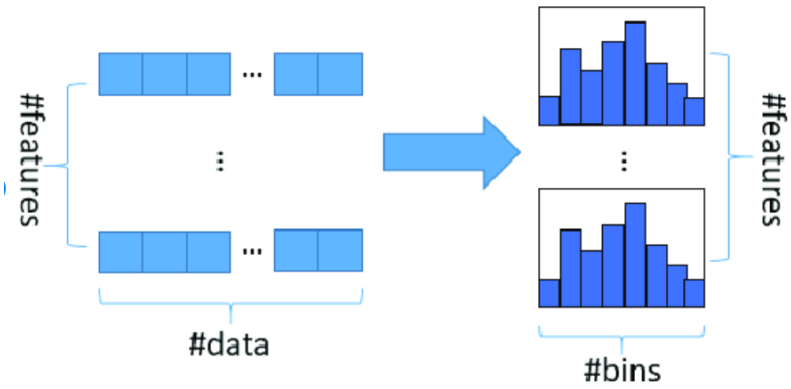


*Ilustración 12-Algoritmo de boosting basado en histogramas. Fuente [105]*

## Ventajas del modelo

El uso del HistGradientBoostingClassifier ofrece muchas ventajas importantes en este proyecto:

- **Precisión alta**: este modelo ha demostrado ser muy preciso en la clasificación de emociones, ya que es capaz de aprender patrones complejos que otros modelos más simples no detectan.
- **Velocidad de entrenamiento**: gracias al uso de histogramas, entrena más rápido que otros modelos similares, lo que permite probar más combinaciones de configuraciones en menos tiempo.
- **Manejo de datos incompletos**: si falta algún dato (por ejemplo, una característica mal extraída en un audio), el modelo puede seguir funcionando sin errores.
- **Capacidad para relaciones no lineales**: puede aprender combinaciones complejas de características acústicas (como variaciones en el tono, energía y frecuencia) sin necesidad de indicarle cómo hacerlo.
- **Requiere poca preparación previa de los datos**: a diferencia de otros modelos que necesitan normalizar o transformar todos los datos antes de entrenar, el HistGradientBoostingClassifier puede trabajar directamente con los vectores de características extraídos de los audios.

## Parámetros utilizados y explicación detallada

En este proyecto, se configuró el modelo con varios hiperparámetros, que son ajustes internos que controlan cómo aprende el modelo. Estos valores no se ajustan automáticamente durante el entrenamiento, sino que deben ser definidos de antemano. Para ello, se utilizó la herramienta Optuna, una biblioteca especializada en la búsqueda automática de combinaciones óptimas de hiperparámetros [46]. Optuna prueba diferentes configuraciones y elige aquellas que ofrecen mejores resultados en validación. A continuación, se explican uno a uno los parámetros seleccionados:

- **max_iter = 171**: este valor indica el número total de árboles de decisión que se van a construir. Cada árbol intenta corregir los errores cometidos por los árboles anteriores. Se eligió el valor 171 porque fue el que ofreció mejores resultados en precisión y estabilidad, evitando el riesgo de sobreajuste (es decir, que el modelo se adapte demasiado a los datos de entrenamiento y pierda capacidad de generalización).
- **max_depth = 10**: la profundidad máxima de cada árbol representa el número de niveles de decisiones que puede tomar. Cuanto mayor es la profundidad, más detalladas son las reglas que aprende el árbol. Sin embargo, profundidades excesivas pueden hacer que el modelo sea demasiado específico y pierda eficacia al enfrentarse a datos nuevos.
- **min_samples_leaf = 20**: este parámetro establece cuántos ejemplos como mínimo debe contener una hoja del árbol, es decir, una decisión final. Si el modelo pudiera tomar decisiones basadas en muy pocos ejemplos, podría cometer errores o aprender ruido en lugar de patrones reales. Al fijar un mínimo de 20 muestras por hoja, se garantiza que las decisiones estén basadas en datos estadísticamente más representativos.
- **random_state = 42**: este valor fija la semilla aleatoria usada internamente en el entrenamiento. No afecta al rendimiento del modelo, pero asegura que los resultados sean siempre iguales si se repite el entrenamiento bajo las mismas condiciones. Esto es muy útil para poder comparar resultados de forma justa o repetir experimentos en trabajos científicos.

## 5.3.4 XGBoostClassifier

### Introducción al modelo

El modelo XGBoostClassifier, también conocido como Extreme Gradient Boosting, es un algoritmo de aprendizaje automático muy potente basado en la técnica de *Gradient Boosting* (al igual que el modelo explicado en el apartado anterior: HistGradientBoostingClassifier). Fue desarrollado por Tianqi Chen y presentado en 2016 [47], y desde entonces ha sido ampliamente adoptado tanto en la investigación como en la industria. XGBoost destaca por su excelente equilibrio entre precisión, velocidad de entrenamiento y capacidad para generalizar correctamente en tareas complejas de clasificación.

XGBoost lleva este enfoque más allá, introduciendo varias mejoras internas. Una de las principales es el uso de regularización, un conjunto de técnicas que permiten controlar el tamaño y la complejidad del modelo para evitar que se ajuste demasiado a los datos de entrenamiento (un problema conocido como sobreajuste). Además, XGBoost está diseñado para trabajar de forma eficiente con grandes volúmenes de datos, permitiendo realizar entrenamiento paralelo y optimizando el uso de memoria, lo que reduce significativamente los tiempos de entrenamiento. También incluye mecanismos inteligentes para manejar automáticamente valores ausentes.

Por estas razones, XGBoost ha sido utilizado con éxito en competiciones de ciencia de datos y en múltiples áreas prácticas, como predicción médica, sistemas financieros, análisis de voz y, en este caso, reconocimiento de emociones en señales acústicas.

En la Ilustración 13 se muestra un ejemplo ilustrativo del funcionamiento de un árbol de decisión individual, uno de los componentes básicos en los que se apoya el algoritmo XGBoost. Como puede observarse, el árbol realiza una serie de preguntas binarias sobre las características de entrada, y en función de las respuestas recorre distintas ramas hasta llegar a una predicción.

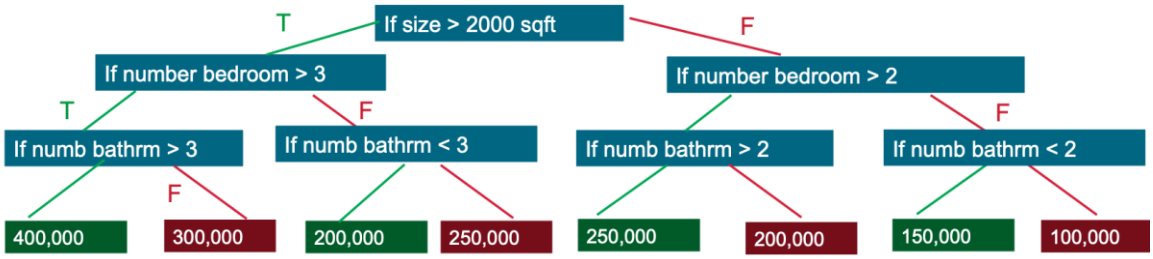


*Ilustración 13-Esquema funcionamiento de un árbol de decisión. Fuente [106]*

Además de comprender el funcionamiento interno de un árbol de decisión individual, es importante visualizar cómo el algoritmo XGBoost los combina mediante la técnica de *boosting*. En la Ilustración 14 se representa de forma conceptual cómo múltiples modelos débiles (como árboles simples) se van añadiendo secuencialmente, cada uno corrigiendo los errores del anterior. El resultado es un modelo conjunto (*ensemble*) más fuerte y preciso, capaz de generalizar mejor. Esta estrategia progresiva de aprendizaje permite que el sistema "aprenda de sus errores", reforzando aquellas muestras donde falló anteriormente y ajustando el peso relativo de cada modelo en la predicción final.

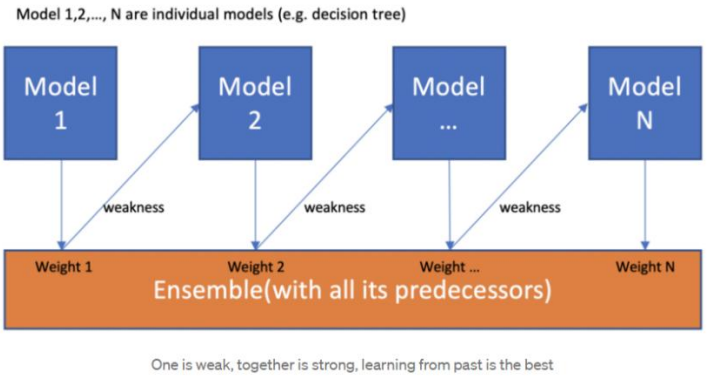


*Ilustración 14-Esquema del funcionamiento del algoritmo de boosting. Fuente [105]*

## Ventajas del modelo

El uso del modelo XGBoostClassifier aporta numerosos beneficios al proyecto, entre ellos:

- **Rendimiento predictivo excelente**: XGBoost es conocido por alcanzar altos niveles de precisión en tareas de clasificación gracias a su capacidad de aprender de errores anteriores y ajustar continuamente su predicción.
- **Rapidez y escalabilidad**: Está diseñado para entrenarse de forma rápida incluso con grandes conjuntos de datos, aprovechando técnicas de paralelización y optimización de memoria.
- **Manejo eficaz del sobreajuste**: Incorpora varios mecanismos de regularización, como penalizaciones en la complejidad de los árboles, que ayudan a mantener el modelo generalizable.
- **Compatibilidad con datos faltantes**: Puede trabajar directamente con características incompletas, asignando de forma automática la mejor dirección de una muestra con valores ausentes durante el entrenamiento.

- **Versatilidad en tareas complejas**: Puede adaptarse fácilmente a características numéricas complejas como los vectores acústicos derivados de MFCC, Chroma, ZCR, energía (RMSE) y centroide espectral.

## Parámetros utilizados y explicación detallada

Todos los hiperparámetros del modelo XGBoostClassifier fueron optimizados mediante búsqueda automática (Optuna), y se seleccionaron aquellos que proporcionaron el mejor equilibrio entre precisión, estabilidad y tiempo de entrenamiento [46]. A continuación, se explican los parámetros:

- **colsample_bytree = 0.8**: este valor indica que cada árbol de decisión solo podrá usar el 80% de las características disponibles. Esto introduce variabilidad entre árboles y reduce el riesgo de que todos aprendan exactamente lo mismo, lo que mejora la robustez del modelo.
- **learning_rate = 0.1**: también conocida como tasa de aprendizaje, controla cuánto afecta cada nuevo árbol al modelo final. Un valor de 0.1 es considerado estándar y proporciona un buen equilibrio entre velocidad de entrenamiento y precisión. Si fuera muy alto, el modelo aprendería demasiado rápido y podría cometer errores; si fuera muy bajo, se necesitarían muchos más árboles.
- **max_depth = 6**: define cuántos niveles de decisiones puede tomar cada árbol. Cuanto más profundo, más complejas son las reglas que puede aprender. Sin embargo, una profundidad excesiva puede hacer que el modelo se vuelva demasiado específico.
- **n_estimators = 200**: este número indica cuántos árboles se van a construir en total. Este valor es suficiente para alcanzar un rendimiento alto sin aumentar en exceso el tiempo de cómputo. Más árboles podrían mejorar ligeramente el resultado, pero con un coste computacional elevado.
- **subsample = 0.8**: este parámetro controla la fracción de datos que se usa para construir cada árbol. En este caso, el modelo utiliza el 80% de los datos en cada iteración, lo cual introduce variedad entre árboles y mejora la capacidad de generalización.
- **random_state = 42**: fija una semilla para el generador de números aleatorios interno.

### 5.3.5 Random Forest Classifier

#### Introducción al modelo

El Random Forest es un modelo de aprendizaje automático que pertenece a la familia de los métodos de ensamble, es decir, aquellos que combinan múltiples modelos simples para formar uno más robusto y preciso. Fue introducido formalmente por Leo Breiman en 2001 [48] y desde entonces se ha consolidado como uno de los algoritmos más fiables y utilizados tanto en contextos académicos como industriales.

A diferencia de otros clasificadores que construyen un único árbol de decisión, Random Forest genera muchos árboles de forma independiente. Cada uno de estos árboles se entrena con un subconjunto diferente de los datos, seleccionados aleatoriamente con repetición (técnica conocida como *bootstrap*). Además, en cada nodo del árbol, solo se consideran un subconjunto aleatorio de características para decidir cómo dividir los datos. Esta aleatoriedad introducida reduce la correlación entre árboles y mejora considerablemente el rendimiento del modelo.

La predicción final se realiza por votación mayoritaria en clasificación, es decir, se elige la clase más votada por los árboles del conjunto. Esta estrategia reduce la varianza del modelo, haciéndolo menos sensible al ruido o a los valores atípicos en los datos.

En comparación con los dos modelos previamente descritos (HistGradientBoostingClassifier y XGBoostClassifier), cuya construcción de árboles es secuencial y basada en la corrección de errores previos, Random Forest emplea una estrategia paralela conocida como *bagging* (*bootstrap aggregating*). Mientras que los métodos de boosting tienden a obtener mayor precisión en problemas muy complejos, Random Forest destaca por su estabilidad, su menor riesgo de sobreajuste y su facilidad de configuración, lo que lo convierte en una alternativa muy sólida y complementaria dentro del conjunto de clasificadores evaluados en este proyecto.

En la Ilustración 15 se ilustra el funcionamiento general del algoritmo Random Forest mediante un ejemplo visual. Cada muestra (en este caso, una combinación de frutas) se evalúa por distintos árboles de decisión construidos a partir de subconjuntos aleatorios de datos y características. Cada árbol emite una predicción individual, y la clase final se determina mediante votación mayoritaria entre todos los árboles del bosque. En el ejemplo, aunque uno de los árboles predice “banana”, la mayoría vota por “manzana”, por lo que esa será la decisión final del modelo. Esta estrategia colectiva es lo que le otorga al Random Forest su capacidad de generalización y robustez ante el ruido.

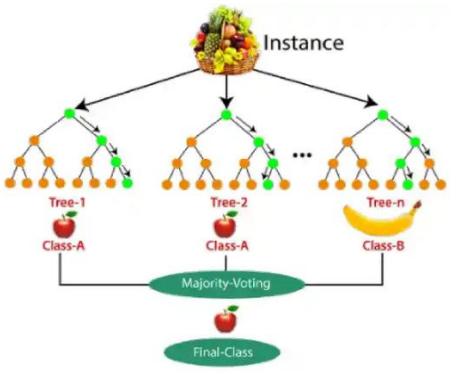


*Ilustración 15-Representación conceptual algoritmo del modelo Random Forest. Fuente [107]*

## Ventajas del modelo

El uso del modelo Random Forest Classifier aporta numerosos beneficios al proyecto, entre los que destacan:

- **Alta precisión y estabilidad**: Combinar muchos árboles reduce errores individuales y mejora la capacidad de generalización. Es especialmente útil en tareas complejas como la clasificación emocional por voz.
- **Resistente al ruido y valores extremos**: Gracias a su estructura basada en múltiples árboles, puede manejar grabaciones defectuosas o ruidosas sin que su rendimiento se degrade significativamente.
- **Versátil y eficiente en grandes volúmenes de datos**: Puede adaptarse fácilmente a tareas con muchas características, como los vectores acústicos extraídos en este proyecto.
- **Relativamente fácil de ajustar**: Aunque tiene varios parámetros, el modelo funciona bien con configuraciones estándar y es intuitivo para experimentar con diferentes combinaciones.

## Parámetros utilizados y explicación detallada

En este proyecto, los hiperparámetros del modelo se seleccionaron mediante búsqueda automática (Optuna). Se detallan a continuación:

- **n_estimators = 100**: este parámetro indica el número total de árboles que se incluirán en el modelo. Más árboles pueden mejorar la precisión, pero también aumentan el tiempo de

cómputo. El valor 100 es un equilibrio muy común entre rendimiento y eficiencia, y ha demostrado ofrecer resultados estables en esta tarea.

- **max_depth = 20**: controla la profundidad máxima que cada árbol puede alcanzar. Cuanto más profundo es un árbol, más decisiones puede tomar, lo que permite que aprenda reglas más específicas. En este caso, una profundidad de 20 ha resultado suficiente para capturar relaciones complejas sin caer en sobreajuste.
- **min_samples_split = 2**: indica el número mínimo de muestras necesarias para dividir un nodo en dos. El valor mínimo posible (2) permite que el modelo explore todas las divisiones posibles y se ajuste con precisión, lo cual es útil cuando se trabaja con conjuntos de datos etiquetados y múltiples clases.
- **class_weight = 'balanced'**: este parámetro ajusta automáticamente el peso de cada clase en función de su frecuencia. Es decir, si una emoción aparece mucho menos que otra en el dataset, el modelo lo tendrá en cuenta para no favorecer sistemáticamente a las clases mayoritarias. Esto es especialmente importante en tareas de clasificación multiclase como esta, donde los datasets pueden estar desbalanceados.
- **random_state = 42**: fijar la semilla permite que los resultados sean siempre los mismos cuando se repite el experimento.

## 5.3.6 Support Vector Machine (LinearSVC)

### Introducción al modelo

El modelo Support Vector Machine lineal (LinearSVC) es una técnica de aprendizaje automático supervisado ampliamente utilizada para tareas de clasificación. Fue desarrollado por Vladimir Vapnik y sus colaboradores [49], y su principio básico consiste en encontrar el hiperplano que mejor separa las clases en un espacio de características. En contextos donde se trabaja con vectores de alta dimensión, como en el reconocimiento de emociones a partir de voz, este enfoque resulta especialmente eficaz.

Un hiperplano es una superficie (línea, plano o generalización en muchas dimensiones) que divide los datos en regiones separadas por clase. LinearSVC busca el hiperplano que maximiza la distancia (margen) entre sí y los puntos más cercanos de cada clase, conocidos como vectores de soporte. Cuanto mayor sea esta distancia, mejor será la capacidad de generalización del modelo.

Para lograr esto, LinearSVC utiliza una función de pérdida llamada *hinge loss*, junto con un término de regularización que evita que el modelo se ajuste en exceso a los datos del entrenamiento. A diferencia de otros modelos más complejos, LinearSVC traza una frontera lineal entre las clases, lo que lo convierte en una opción especialmente rápida y estable para tareas con muchas variables y un número moderado de muestras.

En la Ilustración 16 se ilustra de forma conceptual el funcionamiento de un clasificador SVM lineal. Se puede observar cómo el modelo encuentra un hiperplano (la línea central sólida) que separa dos grupos de datos, y traza dos márgenes (líneas discontinuas) equidistantes a los puntos más cercanos de cada clase. Estos puntos, conocidos como vectores de soporte, determinan la posición óptima del hiperplano:

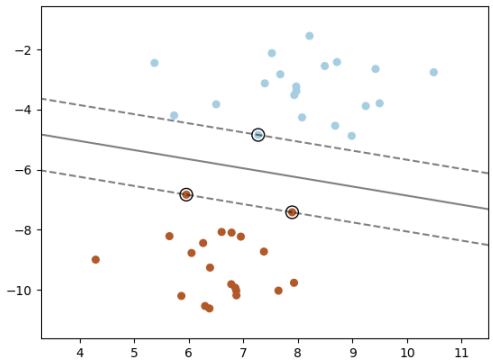


*Ilustración 16-Representación hiperplano que separa dos grupos de datos. Fuente [108]*

## Ventajas del modelo

El uso de LinearSVC aporta varias ventajas relevantes en este proyecto:

- **Alto rendimiento en espacios de alta dimensión**: es ideal cuando se trabaja con muchas variables y menos muestras, como en los vectores acústicos de este trabajo.
- **Velocidad de entrenamiento**: su implementación está optimizada para entrenamientos rápidos, lo que facilita experimentación iterativa.
- **Robustez frente a ruido**: la regularización ayuda a mantener una buena capacidad de generalización incluso con grabaciones imperfectas.
- **Simplicidad en la configuración**: requiere el ajuste de pocos parámetros, siendo fácil de implementar y optimizar.

## Parámetros del modelo y explicación detallada

Antes de realizar la búsqueda de hiperparámetros, se definió el modelo base que se utilizaría dentro de GridSearchCV. El modelo LinearSVC modelo se inicializó con los siguientes parámetros fijos:

- **class_weight = 'balanced'**: este parámetro ajusta automáticamente el peso de cada clase en función de su frecuencia en el conjunto de datos. Es especialmente útil en problemas multiclase con desequilibrio entre clases, ya que evita que el modelo favorezca las clases más numerosas.
- **random_state = 42**: se utiliza para garantizar la reproducibilidad del experimento. Al fijar la semilla del generador aleatorio, los resultados son consistentes en ejecuciones sucesivas del mismo código.

## Parámetros de GridSearchCV

GridSearchCV es una herramienta que se utiliza para buscar automáticamente la mejor configuración de hiperparámetros. En este caso, se aplicó sobre este modelo con la siguiente configuración:

- **param_grid = {'C': [0.01, 0.1, 1, 10, 100]}**: define los valores de C que se probarán. Cada uno representa una posible configuración del modelo LinearSVC, ajustando el nivel de penalización de errores de clasificación.

  El parámetro C controla el equilibrio entre permitir errores de clasificación y mantener un margen amplio entre clases. Valores pequeños de C permiten más flexibilidad al modelo (mayor margen y menos sobreajuste), mientras que valores grandes buscan clasificar todo correctamente, aunque con mayor riesgo de sobreajuste [50].
- **cv = 5**: indica que se utilizará validación cruzada de 5 pliegues para evaluar cada configuración. El conjunto de entrenamiento se divide en cinco partes: en cada iteración, se entrena con cuatro partes y se valida con la restante, repitiendo el proceso cinco veces.
- **scoring = 'accuracy'**: especifica que se usará la precisión como métrica de evaluación. Esta métrica mide la proporción de predicciones correctas respecto al total de muestras.

El proceso completo consiste en tomar el modelo base (LinearSVC) y entrenarlo múltiples veces, una por cada valor definido en param_grid['C']. En cada entrenamiento se realiza una validación cruzada de 5 pliegues, calculando la precisión media.

Una vez evaluadas todas las combinaciones, GridSearchCV selecciona automáticamente la que obtuvo mejor puntuación media. Este modelo se entrenó con el mejor valor de C (0.01), y esta parte fue eliminada del código entregado para no influir negativamente (hablando del tiempo de ejecución) en el resto del flujo del programa.

## 5.3.7 Perceptrón Multicapa (MLPClassifier)

### Introducción al modelo

El Perceptrón Multicapa (MLP) es una red neuronal artificial del tipo *feedforward*, lo que significa que la información fluye siempre en un único sentido: desde la capa de entrada hacia las capas ocultas y finalmente hasta la capa de salida. Fue desarrollado inicialmente en los años 80 como una extensión del perceptrón simple, que solo podía resolver problemas lineales. El MLP supera esta limitación incorporando una o más capas ocultas con funciones de activación no lineales, lo que le permite aprender relaciones complejas y no lineales entre las variables de entrada y la clase objetivo [51].

Cada neurona en una capa recibe múltiples entradas, calcula una combinación lineal de ellas aplicando pesos y un sesgo, y luego transforma ese resultado con una función no lineal. Esta transformación es fundamental, ya que permite que el modelo no solo separe los datos con una línea recta (como haría un modelo lineal), sino que pueda aprender fronteras de decisión curvas, segmentadas o complejas. Cuantas más capas y neuronas tenga la red, más complejos son los patrones que puede detectar.

Durante el entrenamiento, el MLP ajusta sus pesos mediante un proceso iterativo llamado descenso por gradiente, que minimiza una función de error entre las predicciones del modelo y las etiquetas verdaderas. En este proyecto se utilizó el optimizador Adam, una versión mejorada del descenso por gradiente estocástico (SGD) que adapta la tasa de aprendizaje automáticamente para cada peso. Esto hace que el proceso de aprendizaje sea más rápido y estable [52].

Otro aspecto clave de los MLP es que requieren que las entradas estén escaladas o normalizadas, ya que su rendimiento se ve afectado si una característica tiene valores mucho mayores que otras. Por eso, en este proyecto se aplicó una normalización previa mediante StandardScaler antes de entrenar el modelo.

La arquitectura utilizada en este trabajo incluye dos capas ocultas, configuradas para aprender representaciones internas intermedias entre los vectores acústicos y las emociones asociadas. Esto es especialmente útil en tareas como el reconocimiento emocional por voz, donde las relaciones entre

variables y el estado emocional no son evidentes ni lineales. Gracias a su capacidad para aprender representaciones profundas, el MLPClassifier es capaz de identificar patrones que otros modelos podrían pasar por alto, lo que justifica su inclusión como parte del conjunto de clasificadores evaluados en este proyecto.

En la Ilustración 17 se muestra la arquitectura básica de un Perceptrón Multicapa (MLP), una red neuronal compuesta por una capa de entrada, una o más capas ocultas y una capa de salida. Cada neurona de una capa está conectada a todas las de la siguiente, formando una red completamente conectada (*fully connected*). Estas conexiones están asociadas a pesos que se ajustan durante el entrenamiento. Como puede observarse, cada entrada atraviesa varias transformaciones no lineales en las capas ocultas antes de producir una salida. Este flujo de información unidireccional, desde las entradas hasta la salida, es característico de las redes de tipo *feedforward* como la utilizada en este trabajo.

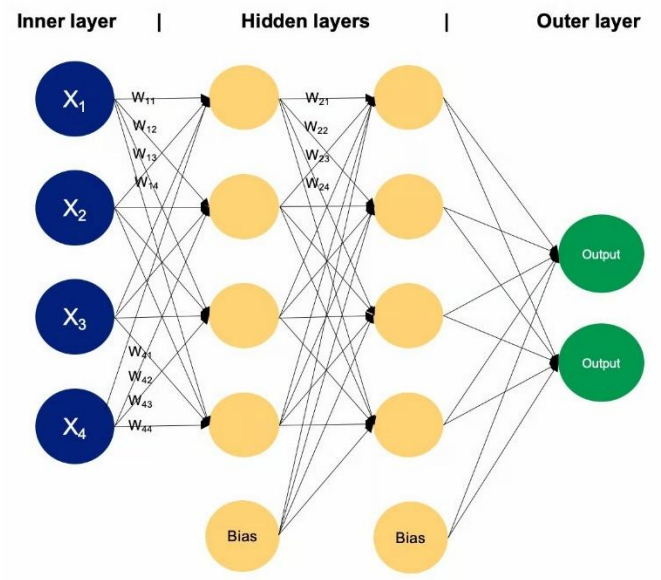


*Ilustración 17-Arquitectura de un MLP dos capas ocultas. Fuente [109]*

## Ventajas del modelo

- **Captura de patrones no lineales complejos**: adecuado para detectar combinaciones sutiles de características acústicas que expresan emociones.
- **Flexibilidad**: el número y tamaño de capas ocultas puede adaptarse a la complejidad del problema.
- **Buen aprovechamiento de grandes volúmenes de datos**: su rendimiento suele mejorar con más muestras, algo útil cuando se combinan RAVDESS, SAVEE y TESS.

## Parámetros utilizados y explicación detallada

Los hiperparámetros del modelo son los siguientes:

- **hidden_layer_sizes = (128, 64)**: define una arquitectura de dos capas ocultas. La primera tiene 128 neuronas, lo que permite al modelo aprender patrones generales. La segunda, con 64 neuronas, refina las representaciones aprendidas, reduciendo la dimensionalidad y ajustando mejor la frontera de decisión sin excesiva complejidad.
- **max_iter = 1000**: especifica el número máximo de iteraciones de entrenamiento. Un valor alto garantiza que el modelo tenga tiempo suficiente para converger, especialmente útil con optimizadores como Adam que ajustan la tasa de aprendizaje dinámicamente.
- **alpha = 0.0001**: corresponde al parámetro de regularización L2. Penaliza los pesos grandes en la red para evitar que el modelo se sobreajuste al conjunto de entrenamiento.
- **learning_rate_init = 0.001**: tasa de aprendizaje inicial. Define la velocidad a la que los pesos se actualizan en cada iteración. Un valor de 0.001 es estándar para Adam y suele proporcionar un buen equilibrio entre estabilidad y velocidad.
- **solver = 'adam'**: optimizador adaptativo que combina la media y la varianza de los gradientes. Es especialmente eficaz con datasets medianos y complejos como el de este proyecto, y requiere menos ajuste fino que otros métodos.
- **tol = 1e-6**: tolerancia mínima que controla cuándo detener el entrenamiento. Si el modelo deja de mejorar por debajo de este umbral, se considera que ha convergido. Un valor bajo como 1e-6 garantiza que el entrenamiento no finalice antes de tiempo.
- **random_state = 42**: establece una semilla fija para los procesos aleatorios, asegurando que los resultados sean reproducibles.

Inicialmente, se realizó una búsqueda sistemática de hiperparámetros mediante GridSearchCV, explorando combinaciones como:

- hidden_layer_sizes = [(128, 64), (256, 128, 64)]
- alpha = [1e-4, 1e-3]
- learning_rate_init = [1e-3, 5e-4].

Esta búsqueda se implementó con validación cruzada de 5 pliegues y precisión como métrica principal. Al finalizar esta búsqueda, se decidió por optar por los parámetros mencionados. Sin embargo, debido a que el entrenamiento de redes neuronales es computacionalmente intensivo y cada combinación

requería múltiples entrenamientos completos, el tiempo total de ejecución del programa se volvió excesivo. Por esta razón, se decidió retirar el bloque de búsqueda automática del código final.

### 5.3.8 Optimización de hiperparámetros con Optuna

Con el objetivo de mejorar el rendimiento de los clasificadores implementados en este proyecto, se optó por optimizar automáticamente sus hiperparámetros mediante la biblioteca Optuna, una herramienta moderna y eficiente para la búsqueda bayesiana [60]. Este proceso se aplicó a los tres modelos basados en árboles: HistGradientBoostingClassifier, XGBoostClassifier y RandomForestClassifier.

Los hiperparámetros son valores que controlan el comportamiento del modelo (como la profundidad máxima de los árboles o el número de iteraciones), pero que no se ajustan automáticamente durante el entrenamiento. Escogerlos de forma adecuada resulta fundamental para evitar tanto el sobreajuste como el infrarentrenamiento. Por esta razón, en lugar de utilizar valores arbitrarios o predeterminados, se realizó un proceso sistemático de optimización con Optuna.

El procedimiento general consistió en los siguientes pasos [61]:

1. Se definió una función objective(trial) para cada modelo, en la que se indicaban los hiperparámetros a explorar (por ejemplo, max_depth, n_estimators, min_samples_leaf, etc.).
2. En cada iteración, Optuna generaba una nueva combinación de hiperparámetros, entrenaba un modelo con esa configuración y evaluaba su rendimiento mediante validación cruzada de 5 pliegues (5-fold CV).
3. La métrica utilizada como función objetivo fue la precisión media (*accuracy*) obtenida durante la validación cruzada.
4. A partir de los resultados anteriores, Optuna aplicaba un algoritmo bayesiano llamado Tree-structured Parzen Estimator (TPE) [59], que permite guiar la búsqueda hacia las regiones del espacio de hiperparámetros con mayor probabilidad de mejorar el rendimiento.
5. Finalmente, tras un número definido de iteraciones (normalmente 50), se seleccionaba automáticamente la mejor combinación encontrada, que se integraba en el código definitivo del clasificador correspondiente.

Este enfoque basado en aprendizaje secuencial permite que Optuna “aprenda” qué combinaciones funcionan mejor y concentre sus esfuerzos en probar configuraciones prometedoras, en lugar de malgastar recursos en combinaciones que, con alta probabilidad, tendrán un rendimiento bajo. Así, se maximiza la eficiencia y se obtiene un ajuste fino del modelo sin necesidad de realizar búsquedas exhaustivas o aleatorias.

En la Ilustración 18 se ilustra de forma esquemática el flujo general del proceso de optimización de hiperparámetros en aprendizaje automático. A partir de un espacio de búsqueda definido por el usuario (por ejemplo, rangos para la tasa de aprendizaje o el número de estimadores), se aplican estrategias de muestreo y poda que interactúan cíclicamente para explorar configuraciones prometedoras. Estas estrategias se integran dentro de algoritmos de optimización tipo *black-box*, como Grid Search o métodos bayesianos, y permiten seleccionar automáticamente las combinaciones más eficaces. Además, se muestra cómo se descartan aquellas configuraciones poco prometedoras mediante mecanismos de parada temprana (*pruning*):

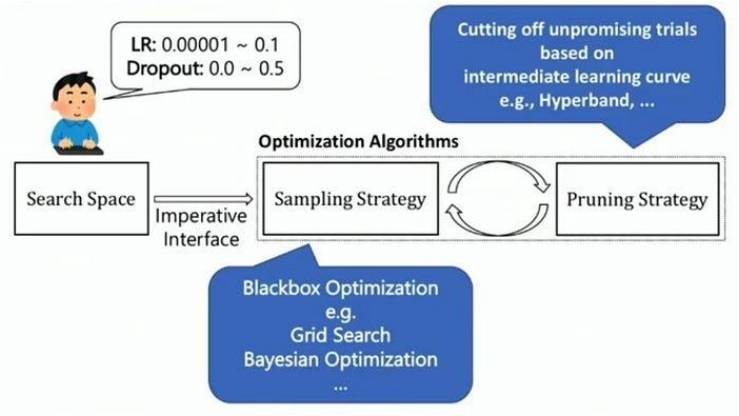


*Ilustración 18-Flujo esquemático de optimización de hiperparámetros. Fuente [110]*

# 5.4 Predicción sobre audios externos

El sistema no solo permite entrenar y evaluar modelos con datasets conocidos, sino que también da la opción de probar grabaciones nuevas. Es decir, el usuario puede añadir sus propios archivos de audio (en formato .wav) para que el modelo entrenado intente detectar la emoción que contienen.

Esta funcionalidad está integrada en el flujo general de ejecución. Si en el momento de lanzar el sistema hay audios colocados en la carpeta prevista para ello, se procesan de forma automática y se clasifican igual que cualquier otra muestra. No hace falta cambiar nada ni configurar parámetros adicionales.

Al terminar, el sistema guarda las predicciones en un archivo de texto que indica, para cada audio, la emoción detectada. De este modo, se puede comprobar fácilmente si el modelo acierta o no, especialmente con grabaciones reales, improvisadas o grabadas en condiciones distintas a las del entrenamiento.

Aunque es una función sencilla, permite interactuar con el sistema de forma más directa, y resulta útil tanto para validar su comportamiento como para mostrarlo en demostraciones prácticas.

# Capítulo 6 - Resultados y análisis

En este capítulo se recogen los resultados obtenidos tras poner a prueba los distintos modelos utilizados en el sistema de reconocimiento emocional por voz. La idea es analizar de forma clara y ordenada cómo se ha comportado cada clasificador, utilizando para ello métricas habituales que permiten comparar su rendimiento de manera objetiva.

Todos los resultados generados durante la fase experimental se han organizado en directorios independientes para cada modelo, respetando una estructura coherente dentro del proyecto. Cada carpeta se ubica dentro del directorio raíz definido por RESULTS_DIR y sigue el esquema: Results/[NombreDelModelo].

Dentro de cada carpeta se incluyen los siguientes elementos:

- **metrics.txt**: informe de clasificación completo, incluyendo métricas de precisión, *recall* y *F1-score* por clase, precisión global y resultados de validación cruzada (media y desviación estándar).
- **confusion_matrix.png**: visualización de la matriz de confusión, que permite identificar los errores más frecuentes entre clases emocionales.
- **feature_importances.png**: gráfico que muestra la importancia relativa de las características acústicas, calculada mediante el método de *Permutation Importance*.
- **new_predictions.txt**: archivo que recoge las predicciones realizadas sobre audios externos no incluidos en el conjunto de entrenamiento.

Los resultados obtenidos mediante informes de clasificación se pueden observar en la Tabla 6.2 (HistGradientBoosting), Tabla 6.5 (XGBoost), Tabla 6.8 (Random Forest), Tabla 6.11 (LinearSVC) y Tabla 6.14 (MLPClassifier).

Aplicando la validación cruzada, la media y desviación típica se observan en la Tabla 6.3, Tabla 6.6, Tabla 6.9, Tabla 6.12 y Tabla 6.15, respectivamente referentes a los modelos en el mismo orden que se definieron en el párrafo anterior.

Las matrices de confusión de cada modelo se muestran en la Ilustración 21, Ilustración 23, Ilustración 25, Ilustración 27 e Ilustración 29.

La información obtenida acerca de la importancia de cada característica se observa en gráficos de barras (Ilustración 22, Ilustración 24, Ilustración 26, Ilustración 28 e Ilustración 30) y en tablas, en las cuáles se muestra el top 10 de las más influyentes positivamente (Tabla 6.4, Tabla 6.7, Tabla 6.10, Tabla 6.13 y Tabla 6.16).

# 6.1 Definición de métricas extraídas

## 6.1.1 Precisión

La precisión es una de las métricas fundamentales para evaluar el rendimiento de un modelo de clasificación multiclase. En términos generales, mide cuánto de confiables son las predicciones positivas del modelo para una clase específica. Su definición formal es la siguiente:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

donde:

- **TP (True Positives)**: número de instancias correctamente clasificadas como pertenecientes a la clase.
- **FP (False Positives)**: número de instancias que fueron clasificadas como pertenecientes a la clase, cuando en realidad no lo eran.

Esta métrica resulta especialmente útil en tareas como el reconocimiento emocional, donde no todas las confusiones tienen el mismo impacto. Por ejemplo, confundir tristeza con ira puede tener implicaciones muy distintas a confundir alegría con miedo, dependiendo del contexto en el que se aplique el sistema. Incluir la precisión por clase en los informes permite evaluar qué tan "limpias" o específicas son las predicciones de cada emoción, lo cual es clave para aplicaciones prácticas que requieren sensibilidad emocional, como asistentes virtuales o sistemas de apoyo psicológico.

La precisión, por tanto, no mide la cobertura total de una clase, sino la fiabilidad de las predicciones positivas del modelo para esa clase concreta [53].

## 6.1.2 Sensibilidad (Recall)

La sensibilidad, también conocida como tasa de verdaderos positivos, es otra métrica esencial para evaluar el rendimiento de un clasificador. Mientras que la precisión se enfoca en la calidad de las predicciones positivas, la sensibilidad evalúa la capacidad del modelo para detectar correctamente todos los casos reales de una clase determinada.

Se define matemáticamente como:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

donde:

- **TP (True Positives)**: número de ejemplos de la clase correctamente identificados.
- **FN (False Negatives)**: número de ejemplos que pertenecen realmente a la clase, pero que el modelo ha clasificado erróneamente como otra.

En el contexto del reconocimiento de emociones, un alto valor de *recall* para la clase "tristeza" implica que el modelo es capaz de identificar la gran mayoría de grabaciones tristes sin omisiones. Por el contrario, un *recall* bajo en "alegría" indicaría que muchas grabaciones realmente alegres no están siendo reconocidas como tales, y en su lugar se clasifican incorrectamente, afectando la fiabilidad del sistema en aplicaciones sensibles.

Esta métrica resulta especialmente importante en escenarios donde la omisión de una emoción relevante puede ser más grave que una clasificación errónea ocasional [53].

### 6.1.3 F1-score

El F1-score es una métrica muy útil cuando se quiere saber si nuestro modelo mantiene un buen equilibrio entre lo que predice de manera correcta (precisión) y lo que es capaz de encontrar (*recall*). En lugar de valorar solo uno de esos aspectos, el F1 combina ambos en un único valor, utilizando una fórmula que se conoce como media armónica:

$$\text{F1} = 2 \times \frac{\text{Precision x Recall}}{\text{Precision} + \text{Recall}}$$

Esta media armónica tiene una particularidad: no premia los extremos. Es decir, si un modelo tiene una precisión altísima pero un *recall* muy bajo (o al revés), el F1-score no lo reflejará como un buen resultado. Más bien, mostrará que el modelo tiene una carencia importante, porque está sacrificando cobertura por exactitud, o viceversa.

Este equilibrio es especialmente relevante en problemas como el reconocimiento de emociones, donde las clases (alegría, tristeza, miedo, ira) no siempre están perfectamente balanceadas en los datos. Por ejemplo, si el modelo identifica bien los audios de "ira" cuando los encuentra, pero apenas los detecta en general, el F1-score será bajo. De este modo, actúa como un termómetro equilibrado que nos ayuda a detectar cuándo el modelo necesita mejorar en cobertura, precisión o ambas cosas.

Por estas razones, el F1-score es una de las métricas centrales utilizadas en la evaluación de los modelos en este trabajo [54].

### 6.1.4 Precisión global (Accuracy)

La precisión global es probablemente la métrica más conocida y utilizada cuando se evalúan modelos de clasificación. Básicamente, nos dice qué porcentaje total de predicciones han sido correctas al considerar todas las clases al mismo tiempo. Su fórmula es la siguiente:

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

Aquí intervienen:

- **TP (True Positives)**: aciertos al predecir correctamente una clase.
- **TN (True Negatives)**: aciertos al identificar que algo no pertenece a una clase.
- **FP (False Positives)**: errores al etiquetar algo como parte de una clase cuando no lo es.
- **FN (False Negatives)**: errores al no detectar correctamente una clase cuando sí lo era.

A simple vista, la *accuracy* parece una métrica muy clara: si el modelo acierta mucho, el valor será alto; si se equivoca a menudo, será bajo. Y en muchos casos es útil, porque da una idea general del comportamiento del sistema. Sin embargo, puede ser engañosa si el conjunto de datos está desbalanceado.

Imaginemos, por ejemplo, que el 70 % de las grabaciones del dataset son de "tristeza". Un modelo que simplemente dijera "tristeza" en todos los casos podría tener una precisión global elevada... sin haber aprendido absolutamente nada sobre las demás emociones. Por eso, aunque es una métrica intuitiva y siempre se incluye en los reportes, no conviene interpretarla de forma aislada. Es mucho más fiable cuando se analiza junto a otras métricas como el *recall* y el F1-score [55].

### 6.1.5 Promedios: Macro avg y Weighted avg

Cuando trabajamos con problemas de clasificación multiclase no basta con reportar una sola métrica global. En su lugar, se calculan los valores de precisión, *recall* y *F1-score* para cada clase individualmente (en las fórmulas se expone el ejemplo concreto de la métrica *F1-score*), y luego se generan dos promedios distintos para ofrecer una visión más amplia y equilibrada del rendimiento del modelo: *macro average y weighted average* [56].

## Macro average (macro avg)

El *macro average* es simplemente el promedio aritmético de la métrica evaluada considerando todas las clases por igual, sin importar cuántos ejemplos hay de cada una. Se calcula así (ejemplo para la métrica *F1-score*):

$$\text{MacroF1} = \frac{1}{\text{C}}\sum_{\text{i=1}}^{\text{C}} \text{F1}_\text{i}$$

Donde C es el número total de clases, y $F1_i$ es el valor de la métrica F1 para la clase i.

Este tipo de promedio resulta muy útil cuando queremos saber si el modelo está siendo justo con todas las clases, incluso con aquellas que tienen muy pocas muestras. Por ejemplo, si el modelo lo hace muy bien con "alegría" pero falla mucho en "miedo", el macro-F1 lo reflejará claramente, ya que cada emoción cuenta por igual, sin importar cuántas veces aparece en los datos.

## Weighted average (weighted avg)

El *weighted average*, en cambio, tiene en cuenta cuántos ejemplos hay de cada clase. Se calcula ponderando cada métrica individual por el número de muestras que tiene esa clase en el conjunto de datos:

$$WeightedF1 = \sum_{i=1}^{C}\left(\frac{Support_i}{\sum_{j=1}^{C} Support_j}\right) x\ F1_i$$

Aquí, $Support_i$ representa el número de muestras reales de la clase i. Esto significa que, si por ejemplo hay muchas más muestras de "tristeza" que de "ira", entonces el rendimiento del modelo sobre la tristeza influirá más en el resultado del weighted-F1.

Este promedio sirve para conocer el rendimiento general del modelo en condiciones reales, ya que refleja el peso que tiene cada clase en el conjunto de datos.

### 6.1.6 Matriz de confusión

La matriz de confusión es una herramienta muy útil cuando se quiere saber con detalle dónde exactamente se está equivocando un modelo de clasificación. A diferencia de métricas globales como el *accuracy*, que dan una visión general, la matriz permite ver clase por clase cómo se comporta el modelo.

En un problema con C clases, esta matriz se organiza como una tabla de tamaño C x C. Cada fila representa la clase real, mientras que cada columna indica la clase predicha. Así, la celda ubicada en la fila i y columna j contiene el número de veces que una muestra de la clase i fue predicha como clase j [57].

Por ejemplo, en este proyecto, donde trabajamos con cuatro emociones, el esquema típico sería el mostrado en la Tabla 6.1:

| | **Alegría** | **Ira** | **Miedo** | **Tristeza** |
|---|---|---|---|---|
| **Alegría** | TP alegría | FP alegría→ira | FP alegría→miedo | FP alegría→tristeza |
| **Ira** | FP ira→alegría | TP ira | FP ira→miedo | FP ira→tristeza |
| **Miedo** | FP miedo→alegría | FP miedo→ira | TP miedo | FP miedo→tristeza |
| **Tristeza** | FP tristeza→alegría | FP tristeza→ira | FP tristeza→miedo | TP tristeza |

*Tabla 6.1-Matriz de confusión aplicada a este proyecto*

Esta representación permite ver fácilmente qué clases se están confundiendo entre sí. Además, la matriz puede revelar patrones que otras métricas no muestran: si dos emociones muy parecidas aparecen confundidas con frecuencia, puede ser señal de que las características acústicas extraídas no están captando bien las diferencias entre ellas.

### 6.1.7 Validación cruzada (k-fold Cross-Validation)

La validación cruzada de k pliegues es una técnica fundamental para evaluar la capacidad de generalización de un modelo, es decir, su rendimiento sobre datos no vistos. En lugar de realizar una única división entre entrenamiento y prueba (lo que puede dar lugar a resultados poco representativos), esta técnica divide el conjunto de datos en k partes iguales (*folds*) y realiza el entrenamiento y la evaluación k veces, rotando cada vez el pliegue que se utiliza como conjunto de prueba [58].

El procedimiento general es el siguiente:

- Se divide el dataset en k pliegues del mismo tamaño.
- Para cada iteración i de 1 a k:
    - Se entrena el modelo con los k-1 pliegues restantes.
    - Se evalúa el rendimiento sobre el pliegue i.
- Finalmente, se obtienen k valores para cada métrica evaluada sobre los que se calcula:
    - La **media** (valor central del rendimiento estimado):

$$Media = \frac{1}{k}\sum_{i=1}^{k} Valor_i$$

    - Y la **desviación estándar** (medida de variabilidad entre pliegues):

$$Desviación\ Estándar = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(Valor_i - Media)^2}$$

Este enfoque ofrece dos ventajas clave: una estimación más robusta del rendimiento global y una visión clara de cuánto varía el modelo según los datos que le toquen en cada iteración. Por ejemplo, una desviación estándar muy alta puede indicar que el modelo es inestable o que ciertos pliegues están desequilibrados; mientras que una desviación baja combinada con buenos resultados medios sugiere que el modelo es consistente y fiable.

En este trabajo, se ha empleado por defecto una validación cruzada de 5 pliegues (*5-fold*) para la mayoría de los modelos, lo que implica que cada métrica reportada es la media de cinco evaluaciones independientes. Sin embargo, en el caso concreto del modelo LinearSVC, se ha optado por una validación de 10 pliegues (*10-fold*). Esta decisión se debe a que los modelos SVM suelen beneficiarse de evaluaciones más finas, y al contar con menos parámetros internos, toleran mejor dividir el conjunto de entrenamiento en subconjuntos más pequeños.

En la Ilustración 19 se representa visualmente el procedimiento de validación cruzada de 5 pliegues (*5-Fold Cross-Validation*). Como puede observarse, el conjunto de datos se divide en cinco subconjuntos iguales, y en cada iteración uno de ellos se reserva para prueba (marcado en rojo) mientras los otros cuatro se utilizan para entrenamiento (en azul). Este proceso se repite cinco veces.

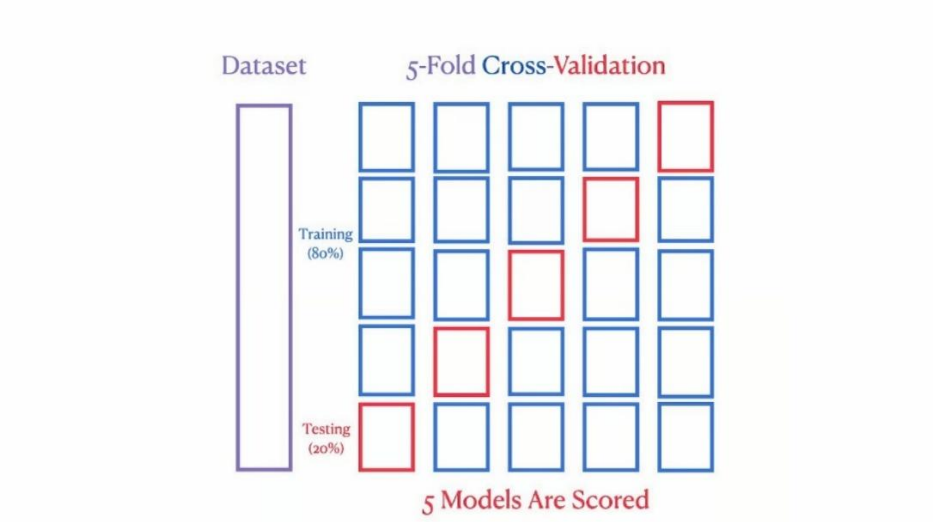


*Ilustración 19-Reparto de datos en validación cruzada de 5 pliegues. Fuente [111]*

### 6.1.8 Importancia de características

En este trabajo, se ha calculado la importancia de características empleando la técnica conocida como *Permutation Importance*. Esta técnica permite cuantificar la relevancia de cada característica individual, midiendo cómo cambia el rendimiento del modelo cuando se altera (o "rompe") la relación entre esa característica y la etiqueta de clase [62].

El procedimiento consiste en tomar el modelo ya entrenado y evaluar su rendimiento sobre el conjunto de prueba. Después, se selecciona una característica concreta, se barajan aleatoriamente sus valores en todas las muestras (el resto de las variables quedan sin tocar) y se vuelve a calcular la métrica de evaluación. Si la métrica empeora notablemente tras esta permutación, significa que dicha característica era importante para el modelo; si apenas cambia, su contribución es mínima. Este proceso se repite varias veces para obtener una media estable, que representa el peso relativo de esa variable.

Esta técnica se ha aplicado de forma uniforme a todos los modelos desarrollados. El resultado final es un ranking ordenado de características, donde destacan aquellas que el modelo considera más informativas para distinguir entre emociones.

El análisis de importancia de características tiene múltiples aplicaciones prácticas dentro del proyecto:

- **Mejora de interpretabilidad**: permite identificar qué rasgos acústicos (por ejemplo, ciertos coeficientes MFCC o el valor del ZCR) resultan más discriminativos entre emociones.

- **Reducción de dimensionalidad**: características con puntuaciones muy bajas podrían eliminarse en futuras versiones del sistema, reduciendo complejidad sin comprometer precisión.
- **Diseño más eficiente del sistema**: conocer qué variables aportan más valor orienta el esfuerzo hacia técnicas de extracción y preprocesamiento más especializadas.

Gracias a este análisis, no solo se mejora la calidad técnica del modelo, sino que se avanza hacia soluciones más transparentes, robustas y explicables, especialmente necesarias en aplicaciones sensibles.

En la Ilustración 20 se muestra un ejemplo de representación gráfica de la técnica de *Permutation Importance* aplicada a un modelo de clasificación (RandomForestClassifier) sobre un conjunto de datos. En este caso, se analiza cuánto disminuye la precisión del modelo cuando se permuta aleatoriamente cada variable. Cuanto mayor sea esta caída en la puntuación, más relevante es la característica para el modelo. Las barras horizontales representan la variabilidad de la importancia estimada a través de múltiples permutaciones. Este tipo de visualización facilita la identificación de las variables que más contribuyen a las predicciones y permite distinguir claramente entre características informativas y ruido aleatorio.

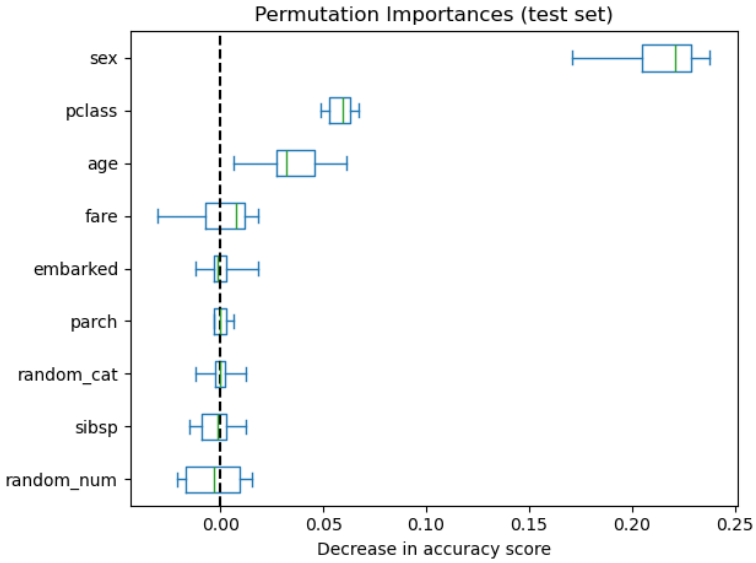


*Ilustración 20-Permutation Importance aplicada a un RandomForest sobre un conjunto de datos. Fuente [112]*

## 6.2 Resultados HistGradientBoostingClassifier

### Desempeño en el conjunto de prueba

| Clase | Precisión | Recall | F1-score | Soporte |
|---|---|---|---|---|
| Alegría | 0,91 | 0,89 | 0,9 | 129 |
| Ira | 0,93 | 0,84 | 0,89 | 129 |
| Miedo | 0,9 | 0,88 | 0,89 | 129 |
| Tristeza | 0,84 | 0,94 | 0,88 | 141 |
| Accuracy global | | | 0,89 | 528 |
| Macro avg | 0,89 | 0,89 | 0,89 | 528 |
| Weighted avg | 0,89 | 0,89 | 0,89 | 528 |

*Tabla 6.2-Informe de clasificación de HistGradientBoostingClassifier*

### Interpretación

- El modelo alcanza un **88,83 % de acierto global**, con valores muy equilibrados entre precisión y *recall* (ambos 0,89 en macro-promedio).
- Alegría e ira muestran la mejor precisión (> 0,90), aunque ira pierde cobertura (*recall* 0,84).
- Tristeza es la emoción con mayor cobertura (*recall* 0,94) pero menor precisión, lo que refleja cierta tendencia a sobre-predicción de esta clase.
- No hay ninguna emoción claramente rezagada: los *F1-scores* oscilan en un margen estrecho de 0,88 - 0,90.

### Validación cruzada (5-fold)

| Métrica | Media | Desv. típica |
|---|---|---|
| Accuracy | 0.7816 | ± 0.2414 |
| Precision macro | 0.8048 | ± 0.2133 |
| Recall macro | 0.7807 | ± 0.2429 |
| F1 macro | 0.7759 | ± 0.2485 |

*Tabla 6.3-Media y desviación típica obtenidas de la validación cruzada del HistGradientBoostingClassifier*

- El rendimiento promedio baja a **0,78 de *accuracy***, lo que indica que el modelo es algo sensible a cómo se reparten SAVEE, RAVDESS y TESS en cada pliegue.

- Las desviaciones estándar (0,24) siguen siendo elevadas; confirman que hay variación entre pliegues, típica cuando se mezclan tres corpus con calidades y locutores distintos.
- Aun así, todos los promedios superan 0,77, señal de que el modelo mantiene un comportamiento robusto frente a diferentes subconjuntos de datos.

## Matriz de confusión

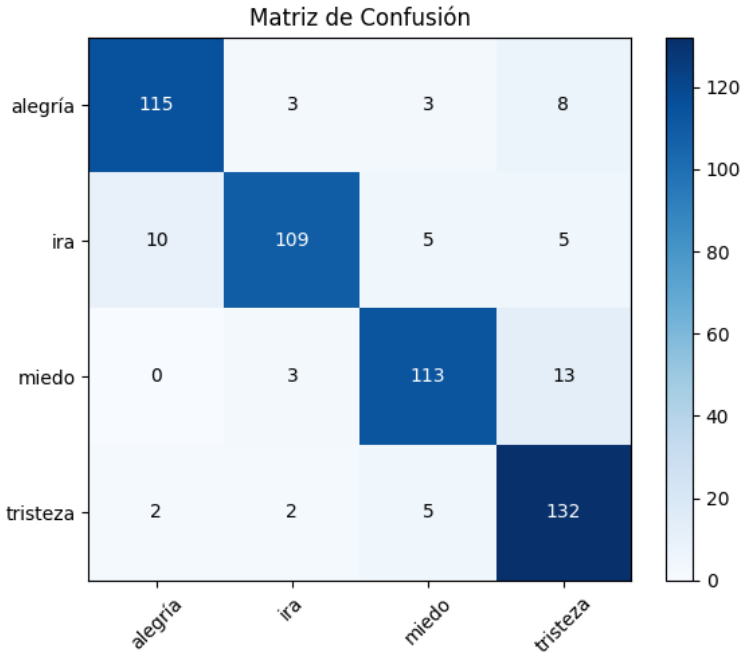


*Ilustración 21-Matriz de confusión del HistGradientBoostingClassifier*

- La emoción miedo es la que más se confunde con otra categoría, siendo 13 veces clasificada erróneamente como tristeza, lo que refuerza la proximidad prosódica entre ambas.
- Alegría se clasifica correctamente en 115 ocasiones, aunque en 8 casos se confunde con tristeza, lo que la convierte en su principal fuente de error.
- Ira muestra una dispersión más repartida, con 10 confusiones hacia alegría y 5 tanto hacia miedo como hacia tristeza, aunque mantiene una tasa de aciertos elevada (109).
- La diagonal domina claramente la matriz, con más de 100 aciertos por clase, lo que indica un buen rendimiento general del modelo y ausencia de errores sistemáticos graves.

## Importancia de características

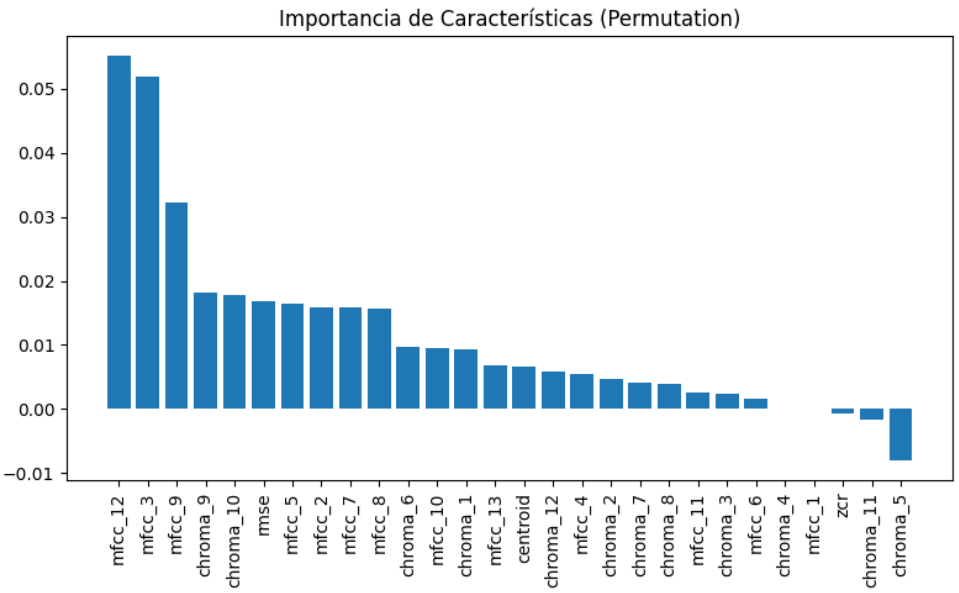


*Ilustración 22-Gráfico de barras permutation importance del HistGradientBoostingClassifier*

## Top 10 más influyentes

| Ranking | Característica | Importancia |
|---|---|---|
| 1 | mfcc_12 | 0.0551 |
| 2 | mfcc_3 | 0.0519 |
| 3 | mfcc_9 | 0.0322 |
| 4 | chroma_9 | 0.0182 |
| 5 | chroma_10 | 0.0178 |
| 6 | rmse | 0.0169 |
| 7 | mfcc_5 | 0.0165 |
| 8 | mfcc_2 | 0.0159 |
| 9 | mfcc_7 | 0.0159 |
| 10 | mfcc_8 | 0.0157 |

*Tabla 6.4-Top 10 características más influyentes de forma positiva del HistGradientBoostingClassifier*

- Los coeficientes MFCC (en especial el 12, 3 y 9) concentran la mayor parte de la información relevante, confirmando su utilidad para capturar matices espectrales vinculados a la prosodia emocional.
- Algunas cromas (chroma_4, chroma_11, chroma_5), ZCR y mfcc_1 muestran valores próximos a cero o negativos; podrían descartarse en futuras versiones para reducir dimensionalidad sin pérdida apreciable de rendimiento.

### Resumen de hallazgos

- **Rendimiento sólido**: 88,83 % de acierto y F1 equilibrado entre emociones en el conjunto de prueba.
- **Variabilidad moderada**: la validación *5-fold* revela oscilaciones atribuibles a la heterogeneidad de SAVEE, RAVDESS y TESS.
- **Dominio de MFCC**: los coeficientes cepstrales (sobre todo de orden alto) son los rasgos más determinantes para la clasificación.
- **Confusiones concretas**: alegría y miedo con tristeza son los pares con mayor solapamiento.

## 6.3 Resultados XGBoostClassifier

### Desempeño en el conjunto de prueba

| Clase | Precisión | Recall | F1-score | Soporte |
|---|---|---|---|---|
| Alegría | 0.90 | 0.87 | 0.88 | 129 |
| Ira | 0.93 | 0.85 | 0.89 | 129 |
| Miedo | 0.88 | 0.88 | 0.88 | 129 |
| Tristeza | 0.84 | 0.94 | 0.89 | 141 |
| Accuracy global | | | 0.88 | 528 |
| Macro avg | 0.89 | 0.88 | 0.88 | 528 |
| Weighted avg | 0.89 | 0.88 | 0.88 | 528 |

*Tabla 6.5-Informe de clasificación del XGBoostClassifier*

## Interpretación

- El modelo XGBoost logra un **88,45 % de acierto global**, con métricas muy equilibradas.
- Ira presenta la mayor precisión (0,93), aunque sacrifica algo de cobertura (*recall* 0,85).
- Tristeza vuelve a ser la emoción con mejor *recall* (0,94) pero relativamente menor precisión, lo que indica cierta sobre-predicción de esta clase.
- Los F1-scores permanecen estrechos (0,88 - 0,89), sin clases claramente rezagadas.

## Validación cruzada (5-fold)

| Métrica | Media | Desv. típica |
|---|---|---|
| Accuracy | 0.7816 | ± 0.2375 |
| Precision macro | 0.8054 | ± 0.2099 |
| Recall macro | 0.7802 | ± 0.2395 |
| F1 macro | 0.7767 | ± 0.2433 |

*Tabla 6.6-Media y desviación típica obtenidas de la validación cruzada del XGBoostClassifier*

## Observaciones

- El rendimiento medio cae a **0,78 de *accuracy***, patrón similar al HistGradientBoosting.
- Desviaciones estándar próximas a 0,24; confirman variabilidad entre pliegues, aunque todos los promedios superan 0,77.
- En conjunto, el modelo mantiene una consistencia aceptable frente a distintas particiones de datos.

## Matriz de confusión

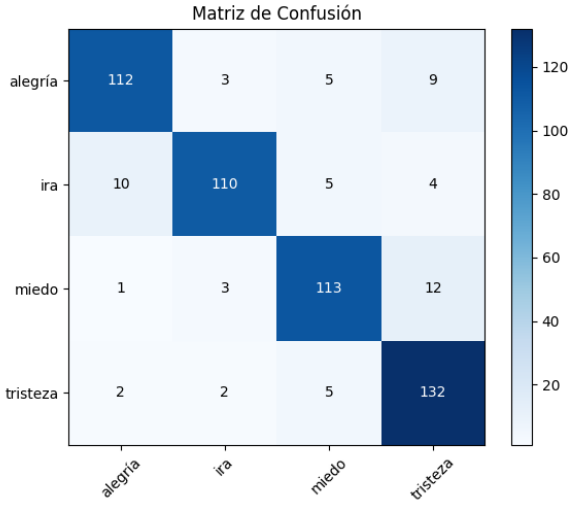


*Ilustración 23-Matriz de confusión del XGBoostClassifier*

- Alegría tiende a confundirse con tristeza en 9 ocasiones, siendo el error más frecuente para esa clase.
- Ira se clasifica erróneamente como alegría en 10 casos, repitiendo el patrón observado en otros modelos.
- El mayor solapamiento entre emociones se observa entre miedo y tristeza, con 12 errores en esa dirección.
- A pesar de estos desajustes, la diagonal sigue siendo claramente dominante: todas las emociones superan los 110 aciertos.

## Importancia de características

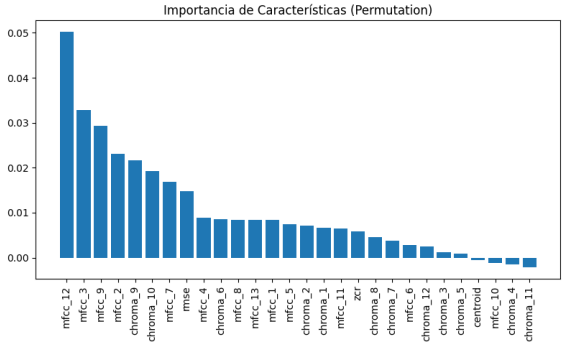


*Ilustración 24-Gráfico de barras permutation importance del XGBoostClassifier*

## Top 10 más influyentes

| Ranking | Característica | Importancia |
|---|---|---|
| 1 | mfcc_12 | 0.0502 |
| 2 | mfcc_3 | 0.0328 |
| 3 | mfcc_9 | 0.0294 |
| 4 | mfcc_2 | 0.0231 |
| 5 | chroma_9 | 0.0216 |
| 6 | chroma_10 | 0.0193 |
| 7 | mfcc_7 | 0.0169 |
| 8 | rmse | 0.0148 |
| 9 | mfcc_4 | 0.0089 |
| 10 | chroma_6 | 0.0085 |

*Tabla 6.7-Top 10 características más influyentes de forma positiva del XGBoostClassifier*

## Puntos destacados

- Los MFCC de orden alto (mfcc_12, mfcc_3, mfcc_9, mfcc_2) siguen liderando la relevancia, subrayando su capacidad para capturar los matices espectrales emocionales.
- Las cromas chroma_9 y chroma_10 mantienen peso considerable, lo que indica que la distribución armónica vocal también es discriminativa.
- rmse (energía) conserva influencia, coherente con el hecho de que la intensidad vocal distingue emociones de alta activación (ira, alegría) frente a tristeza.
- Varios rasgos (chroma_4, chroma_11, centroid, mfcc_10) exhiben importancias muy bajas o negativas, candidatos a eliminación en próximas iteraciones para aligerar el modelo.

## Resumen de hallazgos

- **Rendimiento robusto**: 88 % de accuracy y macro-F1 0,88 en el conjunto de prueba, sin grandes desequilibrios entre clases.
- **Variabilidad moderada**: la CV 5-fold muestra desviaciones de 0,24; evidencia de heterogeneidad entre corpora, aunque la media se mantiene ≥ 0,78.

- **MFCC predominantes**: confirman su valor como descriptor espectral principal; cromas selectos y la energía (rmse) complementan la información.
- **Confusiones clave**: Alegría y Tristeza e Ira y Alegría siguen siendo los pares problemáticos; Miedo y Tristeza muestra también cierto solapamiento.

## 6.4 Resultados RandomForestClassifier

### Desempeño en el conjunto de prueba

| Clase | Precisión | Recall | F1-score | Soporte |
|---|---|---|---|---|
| Alegría | 0.83 | 0.84 | 0.84 | 129 |
| Ira | 0.89 | 0.78 | 0.83 | 129 |
| Miedo | 0.88 | 0.88 | 0.88 | 129 |
| Tristeza | 0.84 | 0.93 | 0.88 | 141 |
| Accuracy global | | | 0.86 | 528 |
| Macro avg | 0.86 | 0.86 | 0.86 | 528 |
| Weighted avg | 0.86 | 0.86 | 0.86 | 528 |

*Tabla 6.8-Informe de clasificación del RandomForestClassifier*

### Interpretación

- El modelo alcanza un **85,8 % de precisión global**, con métricas balanceadas entre clases.
- Ira logra la mayor precisión (0,89), aunque su cobertura es la más baja (*recall* 0,78), lo que indica que es clasificada con confianza, pero también con más omisiones.
- Tristeza mantiene un patrón ya observado en modelos anteriores: el mayor *recall* (0,93) y un F1 muy alto (0,88), lo que sugiere una tendencia del modelo a acertar su detección incluso con cierto exceso.
- En conjunto, los valores de F1 están comprendidos entre 0,83 y 0,88, sin clases significativamente rezagadas.

## Validación cruzada (5-fold)

| Métrica | Media | Desv. típica |
|---|---|---|
| Accuracy | 0.7880 | ± 0.2298 |
| Precision macro | 0.8111 | ± 0.2024 |
| Recall macro | 0.7861 | ± 0.2325 |
| F1 macro | 0.7816 | ± 0.2381 |

*Tabla 6.9-Media y desviación típica obtenidas de la validación cruzada del RandomForestClassifier*

## Observaciones

- La media de precisión en validación cruzada (0,79 aproximadamente) confirma la solidez del modelo, aunque algo inferior al valor obtenido sobre el conjunto de prueba.
- Las desviaciones estándar, en torno a 0,23, sugieren que el rendimiento fluctúa según la partición
- Las métricas macro muestran un comportamiento bastante uniforme y por encima de 0,78, lo que refleja un equilibrio razonable entre precisión y cobertura entre clases.

## Matriz de confusión

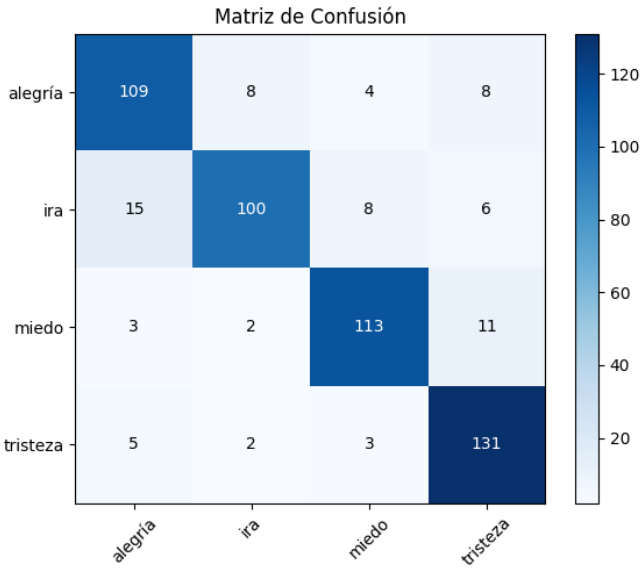


*Ilustración 25-Matriz de confusión del RandomForestClassifier*

- Alegría se confunde principalmente con ira y tristeza (8 errores en cada caso).
- El error más llamativo es la confusión de ira con alegría (15 casos), lo que sugiere proximidad espectral o energética entre estas emociones de alta activación.
- Miedo presenta una clasificación bastante precisa (113 aciertos), con pequeñas desviaciones hacia tristeza (11 casos).
- El modelo mantiene una matriz con diagonal dominante, sin errores sistemáticos graves.

## Importancia de características

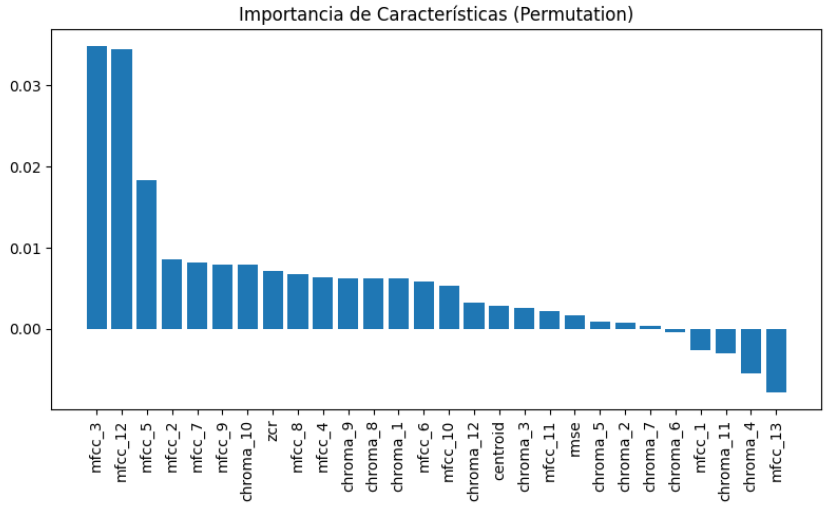


*Ilustración 26-Gráfico de barras permutation importance del RandomForestClassifier*

## Top 10 más influyentes

| Ranking | Característica | Importancia |
|---|---|---|
| 1 | mfcc_3 | 0.0348 |
| 2 | mfcc_12 | 0.0345 |
| 3 | mfcc_5 | 0.0184 |
| 4 | mfcc_2 | 0.0085 |
| 5 | mfcc_7 | 0.0081 |
| 6 | mfcc_9 | 0.0080 |
| 7 | chroma_10 | 0.0080 |
| 8 | zcr | 0.0072 |
| 9 | mfcc_8 | 0.0068 |
| 10 | mfcc_4 | 0.0064 |

*Tabla 6.10-Top 10 características más influyentes de forma positiva del RandomForestClassifier*

## Observaciones

- El modelo muestra nuevamente una fuerte dependencia de los coeficientes MFCC, destacando especialmente mfcc_3, mfcc_12 y mfcc_5.
- Algunas bandas cromáticas, como chroma_10, junto con zcr, también aparecen como relevantes.
- Variables como mfcc_13, chroma_4, chroma_11, mfcc_1 y chroma_6 presentan valores negativos o nulos, lo que sugiere que podrían eliminarse para mejorar la clasificación acústica.

## Resumen de hallazgos

- **Desempeño consistente**: *accuracy* del 86 % y valores macro-F1 de 0,86, con buen equilibrio entre precisión y *recall*.
- **Estabilidad aceptable**: la validación cruzada revela desviaciones moderadas (0,23 aproximadamente), en línea con modelos anteriores.
- **Predominio MFCC**: los coeficientes cepstrales continúan siendo los rasgos más influyentes; cromas y ZCR aportan matices adicionales.
- **Errores destacables**: las principales confusiones se dan entre emociones de alta activación (ira - alegría) y miedo - tristeza.

# 6.5 Resultados LinearSVC

## Desempeño en el conjunto de prueba

| Clase | Precisión | Recall | F1-score | Soporte |
|---|---|---|---|---|
| Alegría | 0.68 | 0.62 | 0.65 | 129 |
| Ira | 0.68 | 0.66 | 0.67 | 129 |
| Miedo | 0.72 | 0.74 | 0.73 | 129 |
| Tristeza | 0.79 | 0.85 | 0.82 | 141 |
| Accuracy global | | | 0.72 | 528 |
| Macro avg | 0.72 | 0.72 | 0.72 | 528 |
| Weighted avg | 0.72 | 0.72 | 0.72 | 528 |

*Tabla 6.11- Informe de clasificación del LinearSVC*

## Interpretación

- El modelo alcanza un **72,16 % de acierto global**, claramente por debajo de los clasificadores basados en árboles.
- Tristeza es la emoción mejor detectada (F1 = 0.82), mientras que alegría presenta el menor *recall* (0.62), síntoma de numerosas omisiones.
- El rango de F1-scores (0.65 - 0.82) revela disparidad de rendimiento entre clases, algo habitual en SVM lineal cuando los datos no son totalmente separables en el espacio lineal.

## Validación cruzada (10-fold)

| Métrica | Media | Desv. típica |
|---|---|---|
| Accuracy | 0.6763 | ± 0.2562 |
| Precision macro | 0.6928 | ± 0.2401 |
| Recall macro | 0.6741 | ± 0.2558 |
| F1 macro | 0.6714 | ± 0.2566 |

*Tabla 6.12-Media y desviación típica obtenidas de la validación cruzada del LinearSVC*

## Observaciones

- La media de *accuracy* cae a 0,68 aproximadamente, confirmando que el rendimiento es modesto y, además, muy variable entre pliegues (0,25 aproximadamente de desviación típica).
- Estas oscilaciones indican que la capacidad generalizadora del SVM lineal depende en gran medida de la combinación de datos que caiga en cada pliegue, reflejando menor robustez frente a los modelos de *ensemble*.

## Matriz de confusión

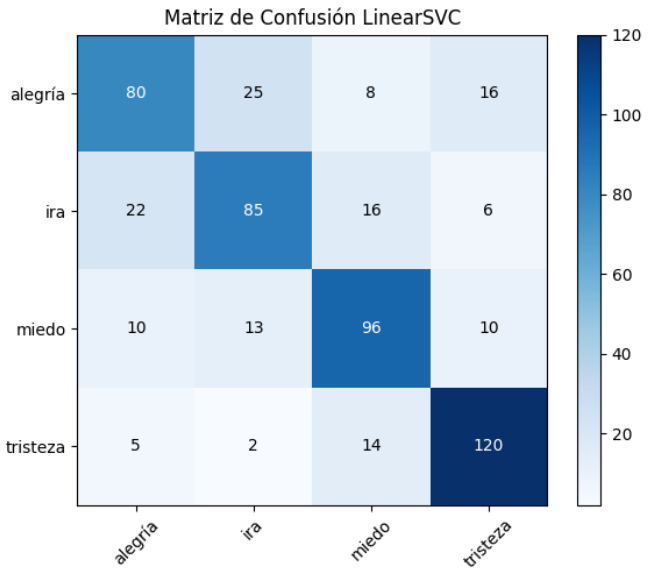


*Ilustración 27-Matriz de confusión del LinearSVC*

- Alegría e Ira es el par con mayor confusión (25 y 22 errores), señal de que el modelo lineal no separa bien estas emociones de alta activación.
- Miedo se confunde de forma considerable tanto con ira como con tristeza.
- Aun así, la diagonal conserva los valores máximos: el modelo reconoce cada emoción más veces de las que se equivoca.

## Importancia de características

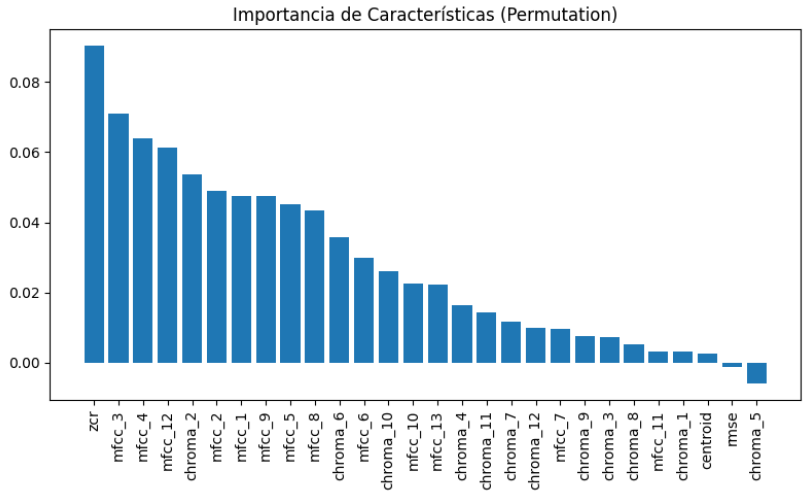


*Ilustración 28-Gráfico de barras permutation importance del LinearSVC*

## Top 10 más influyentes

| Ranking | Característica | Importancia |
|---|---|---|
| 1 | zcr | 0.0903 |
| 2 | mfcc_3 | 0.0710 |
| 3 | mfcc_4 | 0.0638 |
| 4 | mfcc_12 | 0.0614 |
| 5 | chroma_2 | 0.0538 |
| 6 | mfcc_2 | 0.0489 |
| 7 | mfcc_1 | 0.0475 |
| 8 | mfcc_9 | 0.0475 |
| 9 | mfcc_5 | 0.0453 |
| 10 | mfcc_8 | 0.0434 |

*Tabla 6.13-Top 10 características más influyentes de forma positiva del LinearSVC*

## Observaciones

- A diferencia de los modelos previos, la tasa de cruce por cero (zcr) encabeza la lista; su sensibilidad a cambios rápidos de signo en la señal parece relevante para la SVM lineal.

- Los MFCC siguen ocupando la mayor parte del top 10, destacando órdenes medios-altos (mfcc_3, mfcc_4, mfcc_12).
- Rasgos con importancia negativa (chroma_5, rmse) podrían eliminarse para simplificar el modelo sin pérdida.

### Resumen de hallazgos

- **Rendimiento inferior**: *accuracy* de aproximadamente 72 % y macro-F1 con valor 0,72, claramente por debajo de los modelos basados en árboles.
- **Alta variabilidad**: la validación 10-fold refleja desviaciones estándar > 0,24, confirmando poca estabilidad entre particiones.
- **ZCR como líder**: a diferencia de los *ensemble*, la SVM lineal se apoya fuertemente en la tasa de cruce por cero, seguida de varios MFCC.
- **Confusiones marcadas**: el par Alegría - Ira y las mezclas Miedo - Tristeza evidencian la limitada capacidad separadora de un hiperplano lineal para datos acústicos complejos.

## 6.6 Resultados MLPClassifier

### Desempeño en el conjunto de prueba

| Clase | Precisión | Recall | F1-score | Soporte |
|---|---|---|---|---|
| Alegría | 0.91 | 0.91 | 0.91 | 129 |
| Ira | 0.91 | 0.91 | 0.91 | 129 |
| Miedo | 0.87 | 0.90 | 0.88 | 129 |
| Tristeza | 0.90 | 0.87 | 0.89 | 141 |
| Accuracy global | | | 0.90 | 528 |
| Macro avg | 0.90 | 0.90 | 0.90 | 528 |
| Weighted avg | 0.90 | 0.90 | 0.90 | 528 |

*Tabla 6.14-Informe de clasificación del MLPClassifier*

## Interpretación

- El MLP obtiene un **89,77 % de acierto global**, el valor más alto entre los modelos evaluados.
- Las cuatro emociones presentan F1-scores muy parejos (0,88 - 0,91), lo que indica un equilibrio ejemplar entre precisión y cobertura.

## Validación cruzada (5-fold)

| Métrica | Media | Desv. típica |
|---|---|---|
| Accuracy | 0.7839 | ± 0.2245 |
| Precision macro | 0.8118 | ± 0.1910 |
| Recall macro | 0.7836 | ± 0.2250 |
| F1 macro | 0.7801 | ± 0.2287 |

*Tabla 6.15-Media y desviación típica obtenidas de la validación cruzada del MLPClassifier*

## Observaciones

- El rendimiento medio en CV se sitúa en torno a 0,78 - 0,81, muy similar al de los modelos de árboles.
- Las desviaciones estándar (aproximadamente 0,22) revelan cierta variabilidad entre pliegues.
- Aun así, todas las métricas macro superan 0,77, lo que confirma la robustez general del MLP.

## Matriz de confusión

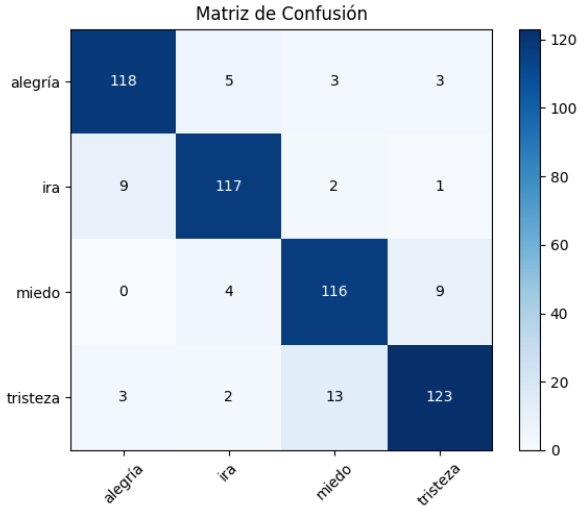


*Ilustración 29-Matriz de confusión del MLPClassifier*

- La diagonal domina con claridad, lo que indica un alto número de aciertos en todas las clases.
- El solapamiento más relevante se observa entre miedo y tristeza, con 9 errores en una dirección y 13 en la otra, lo cual es coherente con la similitud acústica de estas emociones de baja activación.
- Alegría e ira presentan errores muy puntuales (≤ 9 casos), lo que confirma la eficacia del MLPClassifier a la hora de diferenciar entre emociones de alta intensidad.

## Importancia de características

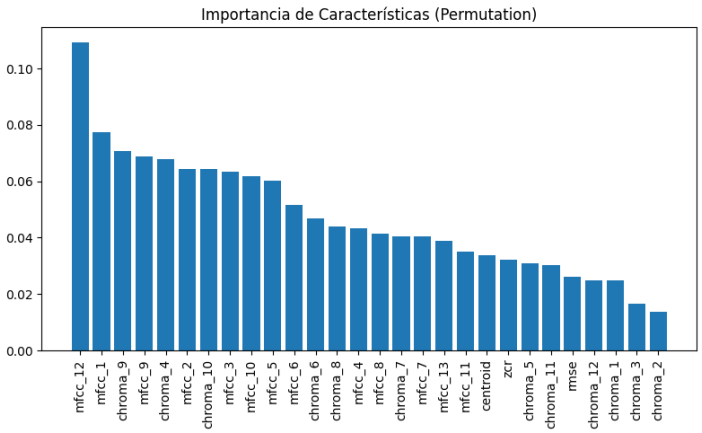


*Ilustración 30-Gráfico de barras permutation importance del MLPClassifier*

## Top 10 más influyentes

| Ranking | Característica | Importancia |
|---|---|---|
| 1 | mfcc_12 | 0.1093 |
| 2 | mfcc_1 | 0.0773 |
| 3 | chroma_9 | 0.0706 |
| 4 | mfcc_9 | 0.0687 |
| 5 | chroma_4 | 0.0680 |
| 6 | mfcc_2 | 0.0644 |
| 7 | chroma_10 | 0.0644 |
| 8 | mfcc_3 | 0.0633 |
| 9 | mfcc_10 | 0.0619 |
| 10 | mfcc_5 | 0.0602 |

*Tabla 6.16-Top 10 características más influyentes de forma positiva del MLPClassifier*

## Observaciones

- El MLP reafirma el dominio de los MFCC, siendo mfcc_12, mfcc_1 y mfcc_9 los más influyentes. El chroma_9 aparece como el tercero entre los mejores.
- Los chromas 1, 3 y 2 son los que menos información acústica aportan según este modelo.

## Resumen de hallazgos

- **Mejor desempeño absoluto**: *accuracy* aproximadamente del 90 % y macro-F1 con un 0,90 en el conjunto de prueba, superando al resto de modelos.
- **Equilibrio por clase**: diferencias de F1 muy estrechas; ninguna emoción se queda atrás.
- **Variabilidad moderada**: la CV muestra una desviación típica de valor aproximado 0,22; similar a los *ensemble*, pero la media se mantiene alta.
- **MFCC con Cromas**: el MLP combina eficazmente características espectrales (MFCC) con patrones armónicos (cromas), explicando su ventaja global.

# Capítulo 7 – Análisis comparativo de modelos

## 7.1 Síntesis de resultados

Tras entrenar y poner a prueba los cinco clasificadores, hemos podido comparar su rendimiento con todo detalle: no sólo cuánto aciertan, sino también cómo se comportan cuando cambian los datos, qué clase de errores cometen y qué rasgos acústicos consideran más importantes.

En el conjunto de prueba, el MLPClassifier encabeza la lista con un 89,77 % de acierto y un F1 muy parejo entre emociones. Le siguen de cerca los tres modelos basados en árboles (HistGradientBoosting, XGBoost y RandomForest), que se mueven entre el 85 % y el 88 % de precisión, con errores bien repartidos y un comportamiento estable.

El caso opuesto es LinearSVC: su precisión se queda en 72,16 % y se lía más a menudo con emociones que suenan parecidas, sobre todo al confundir alegría con ira.

La validación cruzada (cinco pliegues para la mayoría y diez en el SVC) confirma la foto general: todos los modelos rondan entre 0,78 y 0,82 de *accuracy* medio, pero con desviaciones estándar altas (entre 0,22 y 0,26). Esta dispersión refleja la mezcla de tres bases de datos muy distintas, con voces, duraciones y calidades de grabación variadas.

## 7.2 Comparación entre modelos

A continuación, se presentan los resultados comparativos entre los cinco modelos evaluados, analizando su precisión (Tabla 7.1), estabilidad (Tabla 7.2) y aportando una conclusión (Tabla 7.3). Esta comparación permite identificar fortalezas y debilidades específicas de cada enfoque.

### 7.2.1 Precisión en el conjunto de prueba

| Modelo | Accuracy | F1 macro |
|---|---|---|
| MLPClassifier | 0.8977 | 0.90 |
| HistGradientBoosting | 0.8883 | 0.89 |
| XGBoostClassifier | 0.8845 | 0.88 |
| RandomForestClassifier | 0.8580 | 0.86 |
| LinearSVC | 0.7216 | 0.72 |

*Tabla 7.1-Comparación de todos los modelos en cuanto a accuracy y f1-macro*

El MLPClassifier destaca como el modelo más preciso en este conjunto de pruebas, seguido muy de cerca por los clasificadores de árboles. Todos ellos superan el 85 % de precisión, salvo LinearSVC, cuyo rendimiento cae por debajo del 73 %.

### 7.2.2 Resultados en validación cruzada

| Modelo | Accuracy (CV) | Desv. típica |
|---|---|---|
| RandomForestClassifier | 0.7880 | ± 0.2298 |
| HistGradientBoosting | 0.7816 | ± 0.2414 |
| MLPClassifier | 0.7839 | ± 0.2245 |
| XGBoostClassifier | 0.7816 | ± 0.2375 |
| LinearSVC | 0.6763 | ± 0.2562 |

*Tabla 7.2-Comparación de todos los modelos respecto a validación cruzada (LinearSVC con 10 folds, resto 5)*

Aunque RandomForestClassifier obtiene la media más alta en validación cruzada, las diferencias con HistGradientBoosting, XGBoost y MLPClassifier son mínimas.

MLPClassifier, a pesar de no liderar en CV, destaca por tener la desviación estándar más baja, lo que sugiere una estabilidad razonable. Por otro lado, LinearSVC presenta tanto la media más baja como la mayor variabilidad, lo que indica menor fiabilidad ante cambios en los datos.

### 7.2.3 Equilibrio entre clases

Todos los modelos no lineales han mostrado un rendimiento bastante equilibrado a la hora de clasificar las distintas emociones, con especial precisión en la detección de tristeza, que ha sido reconocida correctamente en la mayoría de los casos.

En cambio, el modelo LinearSVC, al estar basado en una separación lineal de los datos, ha tenido más dificultades para distinguir entre emociones positivas como la alegría y otras de alta activación vocal como la ira. Aunque estas emociones tienen una carga emocional diferente, acústicamente pueden parecerse, ya que ambas suelen presentar niveles altos de energía, intensidad o variación en la voz. Esta similitud hace que un modelo lineal, el cual solo puede trazar fronteras rectas entre clases, tienda a confundirlas más fácilmente, algo que se ha reflejado con claridad en las matrices de confusión.

## 7.3 Evaluación de características acústicas

Los análisis de importancia de características realizados con la técnica de *Permutation Importance* revelan un patrón claro:

- Los MFCC, en especial mfcc_12, mfcc_3 y mfcc_9, son consistentes como los más informativos en todos los modelos.
- La ZCR ha resultado clave en modelos como LinearSVC y MLP, especialmente para diferenciar emociones de alta activación.
- La energía RMS muestra una relevancia moderada según el modelo, útil en combinación con otros rasgos.
- Algunas componentes cromáticas como chroma_9, chroma_10 o chroma_1 aportan valor en determinados clasificadores (especialmente en MLP y XGBoost), aunque su presencia es menos constante que la de los MFCC.
- El centroide espectral muestra una importancia muy baja en todos los modelos, sin superar nunca los puestos medios o bajo.

## 7.4 Variabilidad y robustez entre pliegues

Todos los modelos han mostrado una diferencia clara entre sus resultados en el conjunto de prueba y los valores medios obtenidos mediante validación cruzada. Esto refleja una variabilidad real causada por la diversidad de los tres datasets utilizados (SAVEE, RAVDESS y TESS). Las desviaciones estándar, especialmente en modelos como MLP y SVC, alcanzan hasta ± 0.25, señal de que el rendimiento puede fluctuar considerablemente entre particiones.

Este comportamiento indica que, aunque el sistema funciona bien, aún existen oportunidades de mejora, especialmente en la fase de preprocesamiento. Técnicas como la normalización de características entre corpus o el uso de *data augmentation* podrían ayudar a reducir esa variabilidad.

## 7.5 Análisis cualitativo de confusiones

Los errores más frecuentes observados en las matrices de confusión son coherentes con lo que se espera desde la psicología emocional:

- **Alegría ↔ Ira**: ambos estados presentan una alta activación vocal y dinámicas similares (entonación marcada, amplitud variable), lo que genera confusiones recurrentes, especialmente en modelos como LinearSVC.
- **Miedo ↔ Tristeza**: estas emociones suelen tener una prosodia más plana, menor intensidad y frecuencia fundamental, lo que favorece su solapamiento en varios clasificadores.
- **Ira ↔ Miedo y Alegría ↔ Tristeza**: aparecen como errores puntuales en la mayoría de los modelos, y se intensifican en aquellos que no capturan bien la no linealidad de los datos (de nuevo, LinearSVC es el ejemplo más claro).

Este tipo de errores no se producen al azar, sino que responden a lo complicado que resulta, incluso para los modelos, distinguir entre emociones que suenan muy parecidas desde el punto de vista acústico.

## 7.6 Conclusión comparativa

Resulta útil comparar los cinco clasificadores en términos cualitativos, atendiendo a dos dimensiones clave: rendimiento y robustez. La categoría de rendimiento se ha determinado a partir de la precisión en

el conjunto de prueba, la media de validación cruzada y el F1-macro general, mientras que la robustez se ha evaluado en función de la desviación estándar en validación cruzada y la estabilidad en la clasificación entre clases. Para asignar niveles como Máximo, Alta o Media, se han tomado como referencia los valores observados en las Tablas 7.1 y 7.2, con umbrales establecidos a partir del análisis conjunto de todas las métricas. La Tabla 7.3 resume de forma sintética estas observaciones.

| Modelo | Rendimiento | Robustez | Observación clave |
|---|---|---|---|
| MLPClassifier | Máximo | Media | Excelente precisión, algo sensible a los datos |
| HistGradientBoosting | Alta | Alta | Equilibrado y fiable |
| XGBoostClassifier | Alta | Alta | Muy preciso y estable |
| RandomForestClassifier | Media-Alta | Alta | Buen rendimiento, algo más disperso |
| LinearSVC | Baja | Baja | Simple pero limitado para este tipo de tarea |

*Tabla 7.3-Comparativa de rendimiento y robustez*

## 7.7 Modelo recomendado

A la luz de todos los resultados obtenidos, el MLPClassifier se posiciona como la opción más efectiva para esta tarea, gracias a su capacidad para capturar patrones complejos y no lineales en las señales de voz. Sin embargo, si se priorizan modelos más interpretables o con menor coste computacional, XGBoost y HistGradientBoosting también ofrecen un rendimiento excelente y estable, lo que los convierte en alternativas muy válidas para entornos de producción o recursos limitados.

# Capítulo 8 – Conclusiones y trabajo futuro

El trabajo realizado hasta este punto demuestra que es posible construir un sistema de reconocimiento emocional a partir de voz con resultados prometedores, tanto en precisión como en equilibrio entre clases. Aun con los buenos resultados obtenidos, está claro que este campo ofrece todavía muchas posibilidades por explorar. A lo largo del desarrollo del proyecto, han ido surgiendo ideas y direcciones que podrían ayudar a mejorar el sistema en futuras versiones. Algunas de estas propuestas buscan afinar el rendimiento técnico, pero muchas otras apuntan a hacer que el modelo sea más estable, más versátil y útil si algún día se aplica en situaciones reales.

## 8.1 Cumplimiento de objetivos

Después de varios meses de trabajo, puede decirse con certeza que los objetivos planteados al inicio del proyecto se han cumplido de forma satisfactoria. Desde la construcción del sistema hasta su validación técnica, cada etapa ha ido dando forma a una solución completa para el reconocimiento de emociones en voz. No solo se ha logrado que el sistema funcione, sino que también se ha comparado en profundidad cómo responden distintos modelos ante un mismo reto.

Entre los cinco clasificadores probados, ha sido el Perceptrón Multicapa (MLPClassifier) el que ha ofrecido mejores resultados en cuanto a precisión y equilibrio entre clases, aunque otros modelos como XGBoost o HistGradientBoosting han mostrado un rendimiento muy competitivo. Más allá de las cifras, el análisis detallado de métricas, validación cruzada y errores ha servido para entender mejor las fortalezas y debilidades de cada enfoque, lo que en sí mismo era uno de los objetivos del proyecto.

Ahora bien, como es habitual en este tipo de trabajos, el entrenamiento se ha realizado sobre bases de datos actuadas, en inglés y en condiciones controladas, lo que puede haber favorecido cierto sobreajuste al dominio de origen. Aun así, el comportamiento observado sugiere una buena capacidad de generalización dentro del marco experimental planteado, y deja abierta la puerta a mejoras futuras enfocadas en escenarios más realistas y diversos.

En resumen, el sistema no solo alcanza los objetivos propuestos, sino que además deja trazado un camino claro para futuras versiones más robustas, aplicables y sensibles a la complejidad del habla humana en entornos reales.

## 8.2 Mejora del preprocesamiento y normalización intercorpus

Uno de los aspectos que más influencia ha tenido sobre la estabilidad del sistema es la heterogeneidad de los datasets empleados. RAVDESS, SAVEE y TESS son bases de datos de calidad, pero difieren considerablemente en aspectos clave como el idioma, la intensidad de la actuación, la cantidad de locutores o el estilo de grabación. Estos contrastes pueden introducir ruido estructural en el entrenamiento, dificultando que el modelo generalice con fluidez.

Para abordar este problema, una de las primeras acciones recomendables en versiones futuras sería la implementación de un proceso de preprocesamiento más homogéneo. Esto podría incluir:

- **La normalización de volumen entre archivos**, para evitar que una emoción se relacione con una mayor intensidad simplemente por cómo fue grabada.
- **La igualación del *sample rate*** y de la duración media de los audios, aplicando recortes o rellenos controlados cuando sea necesario.
- **El uso de técnicas de ajuste vocal,** como filtros que reduzcan diferencias entre timbres o que compensen características acústicas propias de ciertas voces que podrían sesgar los resultados.

Este tipo de normalización no es trivial, ya que hay que preservar la riqueza emocional sin eliminarla por exceso de procesamiento. Sin embargo, armonizar ciertos aspectos técnicos comunes a todos los audios puede reducir el impacto de las fuentes de variabilidad artificial, mejorando la estabilidad de los modelos ante nuevas particiones de datos o ante posibles implementaciones en la vida real.

Además, en estudios recientes se enfatiza que técnicas como la normalización del volumen, la igualación de la tasa de muestreo y la duración de los audios, así como el uso de filtros para ajustar características vocales, son esenciales para mejorar la estabilidad y generalización de los modelos en contextos reales [63].

## 8.3 Aplicación de técnicas de data augmentation acústico

Uno de los caminos más prometedores para mejorar la capacidad de generalización del sistema es la incorporación de técnicas de *data augmentation* específicas para audio emocional. Esta técnica se ha aplicado con buenos resultados en otros trabajos relacionados con el habla y la expresión vocal, porque permite aumentar la cantidad y variedad de datos disponibles sin tener que grabar más audios desde cero [64].

La idea no es inventar datos aleatorios, sino hacer pequeños cambios intencionados en los audios que ya existen, para que el modelo aprenda a reconocer una emoción, aunque esta suene un poco diferente, como, por ejemplo, con otra entonación, a un volumen distinto o grabada en un entorno más ruidoso.

Entre las técnicas más habituales y aplicables al reconocimiento emocional destacan [65]:

- **Modificación del tono (*pitch shifting*)**: simula voces más graves o agudas, lo que ayuda al modelo a adaptarse mejor a diferentes perfiles de locutores.
- **Alteración de la velocidad de habla (*time stretching*)**: permite representar estilos más pausados o acelerados, ampliando la variabilidad prosódica sin cambiar el contenido emocional.
- **Adición de ruido de fondo (*noise injection*)**: simula condiciones de grabación menos ideales, como llamadas telefónicas o entornos urbanos. Esto entrena al sistema a mantener su rendimiento incluso con señales parcialmente degradadas.
- **Recortes y silencios artificiales (*random cropping* y *silence padding*)**: obligan al modelo a centrarse en las partes más relevantes del mensaje y a no depender de la longitud exacta del audio.

Estas técnicas no solo aumentan la robustez del sistema, sino que también reducen el riesgo de sobreajuste, ya que evitan que el modelo se acostumbre a una representación demasiado rígida de cada emoción. La incorporación de este tipo de variaciones es especialmente útil cuando se trabaja con bases de datos actuadas o grabaciones muy limpias, que pueden no reflejar la diversidad expresiva real en contextos cotidianos.

## 8.4 Evaluación con grabaciones espontáneas y naturales

Uno de los retos pendientes en el desarrollo de sistemas de reconocimiento emocional por voz es su capacidad de funcionar correctamente fuera del laboratorio. En este proyecto se han utilizado tres conjuntos de datos, todos ellos formados por emociones actuadas y grabadas en condiciones muy controladas. Aunque estos datos son útiles para entrenar modelos y realizar pruebas comparables, no reflejan fielmente cómo suenan las emociones en la vida real.

Por eso, una línea futura especialmente relevante sería evaluar el sistema con grabaciones espontáneas o contextos no guionizados. Esto incluiría:

- Conversaciones reales extraídas de entrevistas, llamadas telefónicas o reuniones.
- Audios recogidos en redes sociales, podcasts o vídeos informales.

- Grabaciones propias realizadas en condiciones naturales, sin actores ni guiones predefinidos.

Las emociones reales suelen ser más sutiles, menos exageradas y más mezcladas entre sí. Además, pueden estar influenciadas por factores como el entorno acústico, el estado físico del hablante o la dinámica de la interacción. Evaluar el sistema en estos escenarios permitiría comprobar su capacidad de generalización, así como detectar debilidades que no se manifiestan en los corpus artificiales.

## 8.5 Adaptación a entornos en tiempo real

Una de las mejoras más importantes que podría plantearse a futuro es adaptar el sistema para que funcione en tiempo real. Hasta ahora, las pruebas se han realizado sobre grabaciones ya terminadas, lo que permite un análisis completo y ordenado. Sin embargo, en la mayoría de las aplicaciones prácticas, lo ideal sería que el sistema detectara emociones mientras se está hablando, sin necesidad de esperar a que finalice el audio.

Este cambio de enfoque tiene sentido en muchos contextos reales, como los asistentes virtuales, las llamadas de atención al cliente, o incluso aplicaciones móviles que ofrezcan soporte emocional o seguimiento del estado anímico.

Para que eso sea posible, habría que abordar varios ajustes. Uno de los más evidentes es hacer que el sistema sea más ágil, tanto en el procesamiento del audio como en la respuesta del modelo. En lugar de analizar grabaciones completas, tendría que ser capaz de trabajar con fragmentos cortos (de unos pocos segundos) e ir actualizando su predicción de forma continua.

También sería útil que el modelo pudiera ejecutarse en dispositivos con menos potencia, por lo que convendría estudiar opciones más ligeras y eficientes. A esto se suma una dificultad adicional: en tiempo real no existe una “etiqueta oficial” para cada momento, así que el sistema tendría que generar estimaciones flexibles, que se ajusten en función del contexto y de lo que va detectando a lo largo de la conversación.

Estas adaptaciones no solo harían el sistema más útil, sino que también permitirían evaluar su comportamiento en situaciones menos controladas, como entornos con ruido o grabaciones de baja calidad. La capacidad de actuar al instante, además, abre la puerta a integraciones con dispositivos IoT, sistemas de asistencia en el coche o tecnologías centradas en la salud emocional del usuario.

## 8.6 Personalización y adaptación al usuario

Una posible mejora futura del sistema desarrollado consiste en permitir que se adapte progresivamente a las características individuales de cada usuario. Aunque el modelo ha ofrecido buenos resultados de forma general, es importante tener en cuenta que cada persona expresa las emociones a su manera. Cambios en el tono, en la forma de entonar, en la intensidad o en el ritmo al hablar pueden hacer que una misma emoción suene diferente de una voz a otra.

Actualmente, el sistema trata todas las voces por igual. Sin embargo, incorporar algún tipo de personalización podría mejorar significativamente la precisión en ciertos casos. Esto se podría implementar, por ejemplo, permitiendo que el modelo aprenda del propio uso que cada persona hace del sistema, ajustando su comportamiento en función de muestras previas o patrones frecuentes en su forma de hablar.

También sería útil que el sistema tenga cierta capacidad para recordar información contextual a corto plazo, de modo que no analice cada fragmento de voz de forma aislada, sino dentro del flujo de la conversación. De esta manera, podría interpretar mejor emociones menos marcadas o transiciones emocionales sutiles.

En conjunto, una mayor capacidad de adaptación al usuario final haría que el sistema no solo sea más preciso, sino también más flexible y aplicable en escenarios reales, donde la variabilidad entre personas es la norma.

## 8.7 Limitaciones del trabajo

Aunque los resultados obtenidos han sido consistentes y el sistema ha mostrado un comportamiento equilibrado entre precisión y generalización, es importante señalar una serie de limitaciones que han condicionado el alcance del proyecto. Estas limitaciones no restan valor al trabajo realizado, pero sí ayudan a entender en qué aspectos podría evolucionar el sistema en versiones futuras.

### Limitaciones relacionadas con los datos

Una de las principales restricciones ha estado en la naturaleza de los conjuntos de datos utilizados. Tanto RAVDESS como SAVEE y TESS son bases muy conocidas en el ámbito del reconocimiento emocional, pero tienen una característica común: las emociones han sido representadas por actores en

condiciones controladas. Además, la combinación de tres corpus diferentes ha introducido variaciones significativas en cuanto a idioma, número de locutores, estilo de interpretación y formato de grabación, lo que puede haber afectado a la estabilidad del modelo, especialmente en la validación cruzada.

## Limitaciones técnicas y de entorno de desarrollo

El desarrollo del sistema se ha llevado a cabo en un entorno local, utilizando como equipo principal un portátil ASUS TUF Gaming A15 FA507NVR. Aunque se han podido entrenar y validar modelos de tamaño moderado sin incidencias críticas, algunas tareas (permutaciones de características, la validación cruzada o GridSearch) requirieron tiempos de cómputo prolongados.

Otra limitación técnica ha sido el uso limitado de la GPU. A pesar de contar con una tarjeta gráfica dedicada (RTX 4060), muchas de las bibliotecas empleadas (como Librosa o scikit-learn) no están optimizadas para aprovecharla por defecto, lo que ha llevado a que gran parte del trabajo se ejecutara en CPU. Además, en procesos largos, se detectaron momentos de alto consumo de memoria RAM, lo que ralentizaba la ejecución o la visualización de resultados.

Por último, los audios de prueba grabados de forma manual (para validar predicciones sobre nuevas muestras) se realizaron con un micrófono doméstico, lo que pudo introducir ruido o artefactos no controlados que afectaran a la calidad espectral de las grabaciones.

## Alcance metodológico

Por razones de tiempo y enfoque, este trabajo no ha explorado otras vías que podrían complementar o mejorar los resultados actuales, como el uso de modelos preentrenados, el aprendizaje por transferencia etc. Tampoco se ha realizado una validación con usuarios reales en escenarios prácticos, lo que habría permitido evaluar el sistema desde una perspectiva de experiencia de usuario.

Estas limitaciones marcan el final del alcance del presente proyecto, pero también abren oportunidades claras para seguir avanzando, especialmente si en el futuro se cuenta con más recursos, datos más diversos o un entorno de desarrollo más especializado.

## 8.8 Aplicaciones prácticas del sistema

Aunque este proyecto se ha desarrollado en un entorno experimental, su utilidad va más allá de un entorno de pruebas. La idea de detectar emociones a partir de la voz puede tener muchas aplicaciones reales, siempre que se implemente con cuidado y se adapte a cada contexto.

### Atención al cliente

Una de las áreas más evidentes es la atención al cliente. En una conversación telefónica, por ejemplo, saber el estado emocional de una persona podría ayudar a cambiar el enfoque de la respuesta o incluso a escalar la llamada a alguien más preparado para gestionar ese tipo de situaciones.

### Educación online

En educación online, el sistema podría servir para detectar desmotivación o aburrimiento durante una clase. Aunque no sustituye al criterio de un docente, sí puede ofrecer pistas sobre cómo está funcionando una sesión, sobre todo en entornos donde hay poco contacto directo con el alumnado.

### Bienestar emocional

También hay posibilidades en el ámbito del bienestar emocional. Imagina una aplicación que, sin invadir la privacidad del usuario, pueda ir recogiendo señales vocales a lo largo del día y detectar si hay cambios importantes en el tono emocional. No se trataría de hacer diagnósticos, pero sí de dar apoyo o acompañamiento, especialmente en procesos terapéuticos o programas de seguimiento.

### Vehículos

En el caso de los vehículos, podría utilizarse para detectar cansancio o nerviosismo en el conductor. Si el sistema nota, por ejemplo, que la voz suena más tensa o acelerada, podría activar una alerta suave o recomendar un descanso.

### Asistentes de voz

Y por supuesto, los asistentes de voz, tanto en casa como en el móvil, ganarían mucho si fueran capaces de captar no solo lo que decimos, sino cómo lo decimos. A veces no es lo mismo pedir algo con tono neutro que hacerlo con voz cansada o enfadada, y esa diferencia puede marcar la calidad de la interacción.

## 8.9 Seguridad de los datos, privacidad y cumplimiento normativo en sistemas de IA

El sistema desarrollado en este proyecto trabaja con grabaciones de voz para identificar emociones a partir de características acústicas. Aunque durante todo el desarrollo se ha utilizado únicamente material procedente de datasets públicos, anonimizados y bien documentados (como RAVDESS, SAVEE y TESS), es inevitable plantearse una pregunta clave: ¿qué implicaciones tendría todo esto si el sistema se aplicara con datos reales, es decir, con grabaciones de personas?

La voz, además de ser un canal de comunicación, es también un dato biométrico. Como se menciona en el Reglamento General de Protección de Datos, en el artículo 4.14: *"«datos biométricos»: datos personales obtenidos a partir de un tratamiento técnico específico, relativos a las características físicas, fisiológicas o conductuales de una persona física que permitan o confirmen la identificación única de dicha persona, como imágenes faciales o datos dactiloscópicos;"* [77] . Esto significa que cualquier sistema que trabaje con grabaciones de voz reales (aunque no almacene nombres ni apellidos) puede estar manejando datos especialmente protegidos.

La voz humana puede revelar mucho más que palabras: edad aproximada, origen geográfico, nivel de estrés, rasgos emocionales e incluso indicadores de salud mental. Por eso, el tratamiento de grabaciones de voz debe hacerse con especial precaución, sobre todo si existe la posibilidad de identificar al hablante o inferir información personal sin su conocimiento.

Además, no basta con pedir permiso. El GDPR establece una serie de principios que deben cumplirse siempre que se trabajen con datos personales. Estos principios, mostrados en la Tabla 8.1 y recogidos en su artículo 5, son el marco que debería guiar el diseño de cualquier sistema que procese información como la voz:

| Principio | Aplicado al sistema SER |
|---|---|
| Licitud, lealtad y transparencia | Informar al usuario sobre qué se graba, para qué, cómo se usa y quién lo gestiona. |
| Limitación de la finalidad | Usar las grabaciones solo para reconocimiento emocional, sin desvíos de propósito. |
| Minimización de datos | Captar únicamente lo necesario para cumplir la función del sistema. |
| Exactitud | Asegurar que el modelo analiza correctamente la señal sin sesgos sistemáticos. |
| Limitación del plazo de conservación | No guardar audios más tiempo del estrictamente necesario. |
| Integridad y confidencialidad | Proteger los datos frente a accesos no autorizados o usos indebidos. |
| Responsabilidad proactiva | Ser capaz de demostrar que se cumplen todos estos principios en la práctica. |

*Tabla 8.1-Principios del GDPR de tratamiento de datos personales*

Pero esto no se queda ahí. A todo lo anterior se suma la nueva Ley de Inteligencia Artificial (*AI Act*), actualmente en proceso de aprobación por parte de la Unión Europea [23]. Esta normativa propone una clasificación de los sistemas de IA en función del riesgo que representan. Y sí, los sistemas de reconocimiento emocional aparecen dentro de las categorías de riesgo medio y alto, especialmente si se usan en sectores como educación, sanidad, vigilancia o recursos humanos.

El *AI Act* exigirá medidas adicionales como auditorías, transparencia en los algoritmos, documentación detallada y posibilidad de intervención humana. En otras palabras: no se trata solo de que el sistema funcione, sino de que lo haga con garantías éticas y legales.

Aunque este TFG no ha trabajado con audios personales ni ha recopilado información identificable, todo el diseño del sistema ha seguido una lógica alineada con estas normativas. Se han utilizado datasets legales, se ha evitado cualquier tipo de almacenamiento innecesario de datos, y el usuario conserva el control total sobre si quiere o no añadir audios externos para realizar pruebas.

Si en el futuro se decidiera ampliar el sistema y aplicarlo a grabaciones reales, sería necesario implementar medidas adicionales como:

- Solicitar consentimiento informado y documentado de cada participante.
- Anonimizar la señal de voz, si no es necesaria la identificación.

- Garantizar el derecho de revocación y borrado de datos por parte del usuario.
- Evitar el uso de la información recogida para otros fines no previstos.
- Aplicar técnicas de protección de datos por diseño y por defecto, tal y como recomienda el GDPR.

Instituciones como el *Future of Life Institute* han subrayado la necesidad de que los sistemas de IA sean transparentes, auditables y justos, especialmente cuando influyen en la percepción emocional o el bienestar de las personas [78].

En resumen, aunque este sistema no ha entrado en conflicto con ninguna norma, es fundamental entender que trabajar con voz no es trivial. Y si un sistema es capaz de saber cómo te sientes, lo mínimo que puede hacer es tratar esa información con cuidado.

# Bibliografía

[1] Juanes Mayfield, B. (2024). *Análisis Emocional con Integración de IA en Proyectos de Cambio Organizacional en Empresas Familiares*. Trabajo Fin de Máster. Universidad de Valladolid.

[2] Wang, Y., & Yin, H. (2023). *Advancements and challenges in speech emotion recognition: A comprehensive review*. Neurocomputing. https://doi.org/10.1016/j.neucom.2022.11.020

[3] Niaxus. (2024). *Metodología Scrum: Fases, Ejemplos, Características, Ventajas y Desventajas*. https://niaxus.com/2024/12/14/metodologia-scrum-fases-ejemplos-caracteristicas-ventajas-desventajas/

[4] PMI (2024). *Practice Standard for Project Risk Management*. https://pmbok11.blogspot.com/2025/01/gestion-de-riesgos.html

[5] NotasTransformacionDigital (2024). *Matriz de probabilidad e impacto*. https://www.notastransformaciondigital.com/pmbok/Matriz-de-probabilidad-e-impacto.php

[6] ScienceDirect (2024). *Speech Emotion Recognition - an overview*. https://www.sciencedirect.com/topics/computer-science/speech-emotion-recognition

[7] Wikipedia contributors. (2025). *Random forest*. https://en.wikipedia.org/wiki/Random_forest

[8] Wikipedia contributors. (2025). *XGBoost* https://en.wikipedia.org/wiki/XGBoost

[9] Scikit-learn developers. (2025). *HistGradientBoostingClassifier*. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.HistGradientBoostingClassifier.html

[10] Wikipedia contributors. (2025). *Support vector machine*. https://en.wikipedia.org/wiki/Support_vector_machine

[11] Wikipedia contributors. (2025). *Multilayer perceptron*. https://en.wikipedia.org/wiki/Multilayer_perceptron

[12] IJCRT (2024). *Speech Emotion Recognition*. International Journal of Creative Research Thoughts. https://ijcrt.org/papers/IJCRT2401221.pdf

[13] Sivateja, C., Sarma, K., & Kumar, S. (2023). *Unveiling the challenges of speech recognition in noisy environments: A comprehensive review of issues and solutions*. https://www.taylorfrancis.com/reader/download/21f025e5-573a-45a3-a581-c89e5ca448ba/chapter/pdf?context=ubx

[14] Gómez-Zaragozá, L., del Amor, R., et al. (2024). *EMOVOME: A Dataset for Emotion Recognition in Spontaneous Real-Life Speech*. arXiv.
https://arxiv.org/pdf/2403.02167

[15] Chatziagapi, A., Paraskevopoulos, G., Sgouropoulos, D., Pantazopoulos, G., Nikandrou, M., Giannakopoulos, T., Katsamanis, A., & Potamianos, A. (2019).
*Data Augmentation using GANs for Speech Emotion Recognition*. Proceedings of INTERSPEECH 2019
https://slp-ntua.github.io/potam/preprints/conf/2019_INTERSPEECH_data_augmentation.pdf

[16] Wired. (2018). *This Call May Be Monitored for Tone and Emotion*.
https://www.wired.com/story/this-call-may-be-monitored-for-tone-and-emotion

[17] Wikipedia (2024). *Affective Computing*.
https://en.wikipedia.org/wiki/Affective_computing

[18] Dhuheir, M. et al. (2021). *Emotion Recognition for Healthcare Surveillance Systems Using Neural Networks: A Survey*.
https://arxiv.org/pdf/2107.05989

[19] Wikipedia (2024). *Cerence*.
https://en.wikipedia.org/wiki/Cerence

[20] El Ayadi, M. et al. (2023). *A review on speech emotion recognition: recent advances and challenges*. Neurocomputing.
https://www.sciencedirect.com/science/article/pii/S0925231223011384

[21] Sandeep Jumar Pandey (2023). *Multi-cultural speech emotion recognition using language and speaker cues*
https://www.sciencedirect.com/science/article/pii/S174680942300112X

[22] Sun, Z., Yu, Z., & Busso, C. (2024). *Iterative Prototype Refinement for Ambiguous Speech Emotion Recognition*. Proceedings of Interspeech 2024. International Speech Communication Association (ISCA).
https://www.isca-archive.org/interspeech_2024/sun24e_interspeech.pdf

[23] Wikipedia (2024). Artificial Intelligence Act
https://en.wikipedia.org/wiki/Artificial_Intelligence_Act

[24] Ferreira, A. I. (2025). *Enhancing Speech Emotion Recognition with Graph-Based Multimodal Fusion and Prosodic Features for the Speech Emotion Recognition in Naturalistic Conditions Challenge at Interspeech 2025*.
https://arxiv.org/pdf/2506.02088

[25] Kaggle (2024). *Surrey Audio-Visual Expressed Emotion (SAVEE) Dataset*.
https://www.kaggle.com/datasets/ejlok1/surrey-audiovisual-expressed-emotion-savee

[26] Kaggle (2024). *Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)*. https://www.kaggle.com/datasets/uwrfkaggler/ravdess-emotional-speech-audio

[27] Kaggle (2024). *Toronto Emotional Speech Set (TESS)*. https://www.kaggle.com/datasets/ejlok1/toronto-emotional-speech-set-tess

[28] Schuller, B. W. (2018). *Speech Emotion Recognition: Two Decades in a Nutshell, Benchmarks, and Ongoing Trends*. Communications of the ACM, 61(5), https://dl.acm.org/doi/pdf/10.1145/3129340

[29] Baklouti, I., Ben Ahmed, O., & Fernandez-Maloigne, C. (2024). *Cross-Lingual Low-Resources Speech Emotion Recognition with Domain Adaptive Transfer Learning*. https://www.scitepress.org/Papers/2024/127881/127881.pdf

[30] Parry, R., Schuller, B., Cummins, N., & Cowie, R. (2019). *Analysis of Deep Learning Architectures for Cross-Corpus Speech Emotion Recognition*. https://www.isca-archive.org/interspeech_2019/parry19_interspeech.pdf

[31] Kaggle (2024). *Crowd-sourced Emotional Multimodal Actors Dataset (CREMA-D)*. https://www.kaggle.com/datasets/ejlok1/cremad

[32] Kaggle (2024). *IEMOCAP Emotion Speech Database*. https://www.kaggle.com/datasets/samuelsamsudinng/iemocap-emotion-speech-database

[33] Kaggle (2024). *Berlin Database of Emotional Speech (Emo-DB)*. https://www.kaggle.com/datasets/piyushagni5/berlin-database-of-emotional-speech-emodb

[34] GitHub (2024). *ASED_V1: Arabic Speech Emotion Dataset*. https://github.com/Ethio2021/ASED_V1

[35] Wikipedia. (2024). *Mel-frequency cepstrum*. https://en.wikipedia.org/wiki/Mel-frequency_cepstrum

[36] Amrutha K., Sunanda Panigrahi, Rohit M., Rama S. (2021). *Speech Emotion Recognition using Acoustic Features* https://www.irjet.net/archives/V8/i4/IRJET-V8I4431.pdf

[37] Maelfabien. (2023). *Sound Feature Extraction with Python and Librosa*. GitHub. https://maelfabien.github.io/machinelearning/Speech9/#6-mel-frequency-cepstral-coefficients-mfcc

[38] Wikipedia. *Zero-crossing rate* https://en.wikipedia.org/wiki/Zero-crossing_rate

[39] Fouad Al-Qurashi and Majed al-Nakhli (2023). *A comparison of feature extraction techniques for speech emotion identification* https://www.computersciencejournals.com/ijccn/article/65/5-1-7-696.pdf

[40] Feature extraction. *Librosa*
https://librosa.org/doc/latest/feature.html

[41] Python Software Foundation. (2024). *About Python*
https://www.python.org/about/

[42] Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). *Array programming with NumPy*.
https://numpy.org/

[43] da Costa-Luis, C. (2022). *tqdm: A Fast, Extensible Progress Bar for Python and CLI*.
https://tqdm.github.io/

[44] Scikit-learn: Machine learning in Python.
https://scikit-learn.org/stable/

[45] Friedman, J. H. (2001). *Greedy Function Approximation: A Gradient Boosting Machine*.
https://doi.org/10.1214/aos/1013203451

[46] Optuna Developers. (2024). *Optuna: A hyperparameter optimization framework*.
https://optuna.org/

[47] Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
https://dl.acm.org/doi/pdf/10.1145/2939672.2939785

[48] Breiman, L. (2001). *Random Forests*. *Machine Learning*.
https://doi.org/10.1023/A:1010933404324

[49] Cortes, C., & Vapnik, V. (1995). *Support-vector networks*. *Machine Learning*
https://doi.org/10.1007/BF00994018

[50] Scikit-learn documentation. (2024). *SVM: Maximizing the margin*.
https://scikit-learn.org/stable/modules/svm.html

[51] Scikit-learn Developers. (2025). *Multi-layer Perceptron-MLPClassifier*.
https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[52] Kingma, D. P., & Ba, J. (2015). *Adam: A Method for Stochastic Optimization*. International Conference on Learning Representations.
https://arxiv.org/pdf/1412.6980

[53] Wikipedia contributors. (2024). *Precision and recall*. Wikipedia.
https://en.wikipedia.org/wiki/Precision_and_recall

[54] Wikipedia contributors. (2024). *F1 score*. Wikipedia.
https://en.wikipedia.org/wiki/F1_score

[55] Wikipedia contributors. (2024). *Accuracy and precision*. Wikipedia. https://en.wikipedia.org/wiki/Accuracy_and_precision

[56] Ajitesh Kumar. (2023). *Micro-average, Macro-average, Weighting: Precision, Recall, F1-Score*. *Vitalflux*. https://vitalflux.com/micro-average-macro-average-scoring-metrics-multi-class-classification-python/

[57] Wikipedia contributors. (2024). *Confusion matrix*. Wikipedia. https://en.wikipedia.org/wiki/Confusion_matrix

[58] Brownlee, J. (2020). *A Gentle Introduction to k-Fold Cross-Validation*. Machine Learning Mastery. https://machinelearningmastery.com/k-fold-cross-validation/

[59] Optuna Developers. (2025). *optuna.samplers.TPESampler*. Optuna 4.4.0 documentation. https://optuna.readthedocs.io/en/stable/reference/samplers/generated/optuna.samplers.TPESampler.html

[60] Rogozhkin, D. (2024). *How to Optimize Hyperparameter Search Using Bayesian Optimization and Optuna*. Neptune.ai Blog. https://neptune.ai/blog/how-to-optimize-hyperparameter-search

[61] Mol, N. (2023). *Python Optuna: A Guide to Hyperparameter Optimization*. Datagy. https://datagy.io/python-optuna/

[62] Molnar, C. (2022). *Interpretable Machine Learning*. Leanpub. https://christophm.github.io/interpretable-ml-book/feature-importance.html

[63] Braunschweiler, N., Doddipatla, R., Keizer, S., & Stoyanchev, S. (2022). *A study on cross-corpus speech emotion recognition and data augmentation*. https://arxiv.org/pdf/2201.03511

[64] Chatziagapi, A., Paraskevopoulos, G., Pantazopoulos, G., Sgouropoulos, D., Katsamanis, A., Potamianos, A. (2019). *Data Augmentation using GANs for Speech Emotion Recognition*. In INTERSPEECH 2019. https://www.isca-archive.org/interspeech_2019/chatziagapi19_interspeech.pdf

[65] Mukhlas, A., & Zahra, A. (2023). *Enhancing speech emotion recognition with deep learning using multi-feature stacking and data augmentation*. https://www.beei.org/index.php/EEI/article/view/6049/3727

[66] Adigwe, A., Tits, N., El Haddad, K., Ostadabbas, S., & Dutoit, T. (2018). The Emotional Voices Database (EmoV-DB) https://openslr.org/115/

[67] James, J., Tian, L., & Watson, C. (2018). JL-Corpus: Emotional Speech Corpus with Primary and Secondary Emotions [Dataset]. Kaggle.
https://www.kaggle.com/datasets/tli725/jl-corpus

[68] Tientcheu, D. L., He, Q., & Xie, W. (2020). ASVP-ESD: Speech & Non-Speech Emotional Sound Dataset [Dataset]. Kaggle.
https://www.kaggle.com/datasets/dejolilandry/asvpesdspeech-nonspeech-emotional-utterances?resource=download

[69] Pensamiento Amplio. (s.f.). *Las emociones básicas según Daniel Goleman: ¿cuántas definió?* Pensamiento Amplio.
https://pensamientoamplio.net/crecimiento/las-emociones-basicas-segun-daniel-goleman-cuantas-definio/

[70] La Mente es Maravillosa. (s.f.). *Daniel Goleman y su teoría de la inteligencia emocional.*
https://lamenteesmaravillosa.com/daniel-goleman-teoria-la-inteligencia-emocional/

[71] Wikipedia. (2023). *Cross Industry Standard Process for Data Mining*.
https://es.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining

[72] GanttProject. (s.f.). *GanttProject: Free project scheduling and management software*.
https://www.ganttproject.biz/

[73] Scikit-learn developers. (2024). *sklearn.preprocessing.LabelEncoder*.
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html

[74] Scikit-learn developers. (2024). *sklearn.preprocessing.StandardScaler*.
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[75] Scikit-learn developers. (2024). *sklearn.pipeline.Pipeline*.
https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

[76] Scikit-learn developers. (2024). *sklearn.svm.LinearSVC - scikit-learn documentation*.
https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

[77] European Union. (2016). *Regulation (EU) 2016/679 of the European Parliament and of the Council (General Data Protection Regulation)*.
https://eur-lex.europa.eu/legal-content/ES/TXT/PDF/?uri=CELEX:32016R0679

[78] Future of Life Institute. (2021). *Ethical Guidelines for Trustworthy AI*.
https://futureoflife.org/ai-policy/

[79] Matplotlib (2024). *Matplotlib: Visualization with Python*.
https://matplotlib.org/

[80] Russell, S., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach* (4th ed.).
https://aima.cs.berkeley.edu/newchap00.pdf

[81] XGBoost Developers (2024). *XGBoost Documentation*.
https://xgboost.readthedocs.io/

[82] InfoJobs. (s.f.). *Salarios por profesión y provincia*.
https://salarios.infojobs.net

## Imágenes

[100] Essedi (s.f.). *SCRUM: El marco de trabajo para el éxito en proyectos*.
https://www.essedi.es/scrum-el-marco-de-trabajo-para-el-exito-en-proyectos/

[101] iPMO Guide (s.f.). *Fases de la metodología CRISP-DM*.
https://ipmoguide.com/fases-de-la-metodologia-crisp-dm/

[102] Wikipedia (s.f.). *Escala Mel*.
https://es.wikipedia.org/wiki/Escala_Mel

[103] Wikipedia (s.f.). *Chroma feature*.
https://en.wikipedia.org/wiki/Chroma_feature

[104] Music Information Retrieval. (s.f.). *Energy*.
https://musicinformationretrieval.com/energy.html

[105] Analytics Vidhya (2022). *Histogram Boosting Gradient Classifier*.
https://www.analyticsvidhya.com/blog/2022/01/histogram-boosting-gradient-classifier/

[106] NVIDIA (s.f.). *What is XGBoost?*
https://www.nvidia.com/en-us/glossary/xgboost/

[107] Analytics Vidhya (2021). *Understanding Random Forest Algorithm*.
https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

[108] Scikit-learn (2024). *Support Vector Machines*.
https://scikit-learn.org/stable/modules/svm.html

[109] DataCamp (2023). *Multilayer Perceptrons in Machine Learning*.
https://www.datacamp.com/tutorial/multilayer-perceptrons-in-machine-learning

[110] Avhale, K. (2023). *Understanding of Optuna: A machine learning hyperparameter optimization framework*. Medium.
https://medium.com/@kalyaniavhale7/understanding-of-optuna-a-machine-learning-hyperparameter-optimization-framework-ed31ebb335b9

[111] Datacamp (2024). *K-Fold Cross Validation – A Complete Guide*. https://www.datacamp.com/tutorial/k-fold-cross-validation

[112] Scikit-learn. (2024). *Permutation feature importance*. https://scikit-learn.org/stable/modules/permutation_importance.html

[113] Zapata, J. (s.f.). *Extracción de características*. Minería de Audio. https://joserzapata.github.io/courses/mineria-audio/extraccion_caracteristicas/#centroide-espectral-spectral-centroid