GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DETECCIÓN TEMPRANA DE DEPRESIÓN MEDIANTE SEÑALES DE VOZ

Autor: Diego de los Santos González

Directora: Dido Carrero Muñiz

España, Madrid

Declaración de originalidad

Declaro bajo mi responsabilidad que el Proyecto presentado con el título **DETECCIÓN TEMPRANA DE DEPRESIÓN MEDIANTE SEÑALES DE VOZ** e la ETS de Ingeniería – ICAI de la Universidad Pontificia Comillas en el curso académico 2025-2026 es de mi autoría y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Uso de Inteligencia Artificial[1]

Declaro bajo mi responsabilidad que (indicar la opción correcta):

☐ No he utilizado Inteligencia Artificial en la elaboración del presente documento.

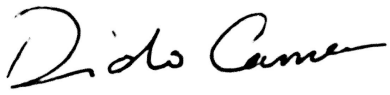
X He utilizado Inteligencia Artificial en la elaboración del presente documento y/o del Anexo B siempre en las condiciones permitidas por la Universidad Pontificia Comillas, es decir, aplicando el Nivel 2 de la Escala de Evaluación de Perkins et al. (2024): "*La IA puede utilizarse para actividades previas a la tarea, como la lluvia de ideas, la descripción y la investigación inicial. Este nivel se centra en el uso de la IA para la planificación, las síntesis y la generación de ideas, pero las evaluaciones deben hacer hincapié en la capacidad de desarrollar y refinar estas ideas de forma independiente*". En concreto, las Inteligencia Artificial ha sido empleada para:

Se ha hecho uso de Inteligencia Artificial aplicando el Nivel 2 de la Escala de Evaluación proporcionada. Principalmente con el desarrollo de ideas e investigaciónes iniciales del tema a tartar además de la ayuda para la estructuración de la redacción y el planteamiento global del proyecto.


Firmado (alumno): Diego de los Santos González
Fecha: 9 de Abril de 2026

[1] Esta declaración se refiere al uso de la Inteligencia Artificial generativa para realizar los documentos del Proyecto (Anexo B y Memoria). No aplica a Proyectos donde, por su naturaleza, deban emplear inteligencia artificial como parte de los mismos (aplicación de técnicas de aprendizaje automático, redes neuronales, análisis de datos…)

Autorización para la entrega del Proyecto

La Directora del Proyecto	El co-Director del Proyecto (si aplica)
	
Fdo: Dido Carrero Muñiz	Fdo:
Fecha: 28 de mayo de 2026	Fecha:

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DETECCIÓN TEMPRANA DE DEPRESIÓN MEDIANTE SEÑALES DE VOZ

Autor: Diego de los Santos González

Directora: Dido Carrero Muñiz

España, Madrid

Agradecimientos

A mis compañeros de clase, por hacer que incluso un lunes a las ocho de la mañana sea no solo tolerable, si no divertido.

A toda mi familia, por apoyarme en todo momento.

En especial a Papá y a mi abuelo “Bovo”, por estar siempre ahí y darme apoyo incondicional.

DETECCIÓN TEMPRANA DE DEPRESIÓN MEDIANTE SEÑALES DE VOZ

Autor: de los Santos González, Diego.
Director: Carrero Muñiz, Dido.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

1. Introducción

La depresión es uno de los trastornos de salud mental más estudiados debido al impacto que puede tener en la vida de una persona. Aun así, detectar este tipo de problemas no siempre es sencillo. En muchos casos el diagnóstico depende de entrevistas, cuestionarios y valoraciones realizadas por especialistas, por lo que sigue existiendo una parte subjetiva dentro del proceso.

Con el avance del aprendizaje automático y del procesamiento digital de señales han aparecido nuevas formas de analizar información biomédica de manera más objetiva. Una de las líneas que más interés ha generado en los últimos años es el análisis de la voz. Aunque normalmente se asocia únicamente al contenido de lo que una persona dice, la voz también contiene información relacionada con la forma en la que se expresa, el ritmo al hablar o incluso ciertos cambios en el tono y la energía de la señal.

A partir de esta idea, en este trabajo se estudia la posibilidad de utilizar grabaciones de voz para intentar encontrar patrones asociados a distintos estados depresivos. Para ello se extraen características acústicas directamente del audio y posteriormente se utilizan modelos de clasificación entrenados con un conjunto de datos etiquetado.

Durante el desarrollo del proyecto se probaron distintos enfoques y modelos hasta encontrar aquellos que ofrecían un comportamiento más estable frente a datos no vistos previamente. Más allá de obtener una métrica concreta, el interés principal del trabajo está en comprobar hasta qué punto este tipo de técnicas pueden servir como herramienta de apoyo dentro de contextos de evaluación más amplios.

2. Definición del proyecto

Este proyecto se centra en el desarrollo de un sistema capaz de analizar grabaciones de voz para clasificar distintos estados relacionados con la depresión. En concreto, se trabaja con tres categorías diferentes: ausencia de depresión, depresión de grado 1 y depresión de grado 2.

El desarrollo del trabajo parte directamente de grabaciones de audio en bruto. A partir de ellas se extraen distintas características acústicas que permiten representar la señal de voz de forma numérica y construir posteriormente una base de datos preparada para el entrenamiento de modelos supervisados. Una vez generado el conjunto de datos, se comparan distintos modelos de clasificación con el objetivo de analizar cuál es capaz de diferenciar mejor entre los tres grupos planteados.

Además de la parte de clasificación, también se utilizan algunas técnicas de agrupamiento y reducción de dimensionalidad para estudiar cómo se distribuyen internamente los datos y poder visualizar de forma más clara posibles separaciones entre categorías.

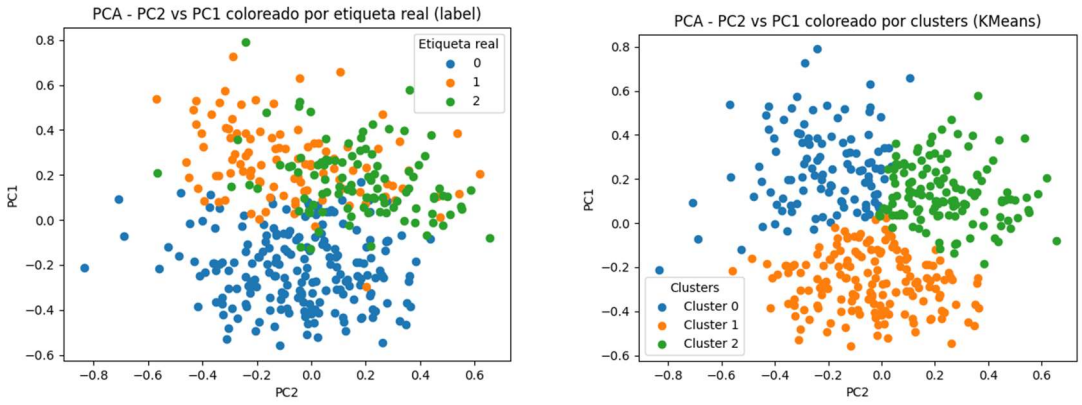


Figura 1 – Izquierda: Clúster originales. Derecha: Clúster asignados por el modelo.

Finalmente, el modelo seleccionado será sometido a un proceso de validación tanto con datos internos como con datos externos, con el propósito de evaluar su capacidad de generalización y garantizar su correcto funcionamiento fuera del conjunto de entrenamiento.

El alcance del trabajo se centra en el análisis técnico y la evaluación computacional del sistema desarrollado, sin pretender sustituir el diagnóstico clínico, sino explorar la viabilidad del uso de patrones vocales como herramienta de apoyo en la detección automatizada.

3. Descripción del modelo/sistema/herramienta

El sistema desarrollado en el presente trabajo se basa en un enfoque experimental orientado al análisis y evaluación de modelos de aprendizaje supervisado aplicados a características acústicas extraídas directamente de grabaciones de voz en bruto.

En una primera etapa, se parte de señales de audio sin procesar correspondientes a pacientes clasificados en tres categorías clínicas: ausencia de depresión, depresión grado 1 y depresión grado 2. Sobre estas señales se realiza la extracción de un conjunto de características acústicas que permiten representar propiedades relevantes de la voz. Entre estas propiedades se incluyen los siguientes parámetros: *pitch mean, pitch std, energy mean, energy std, pause ratio, MFCC*. Cada grabación queda representada mediante un vector numérico de cada uno de los atributos que representa la información contenida en la señal original.

Las características extraídas se organizan en un conjunto de datos estructurado, donde cada muestra se asocia a su etiqueta correspondiente. Este conjunto de datos será la base sobre la cual se desarrollan los experimentos de modelado.

Sobre este espacio de características se entrenan y comparan distintos modelos de aprendizaje supervisado con el objetivo de evaluar su capacidad para diferenciar entre las tres clases consideradas. El proceso de entrenamiento se realiza mediante Stratified Cross-Validation, garantizando que la proporción de clases se mantenga constante en las distintas particiones y reduciendo el riesgo de overfitting. Asimismo, se lleva a cabo la selección del modelo con mejor rendimiento en función de métricas de evaluación adecuadas al problema de clasificación multiclase.

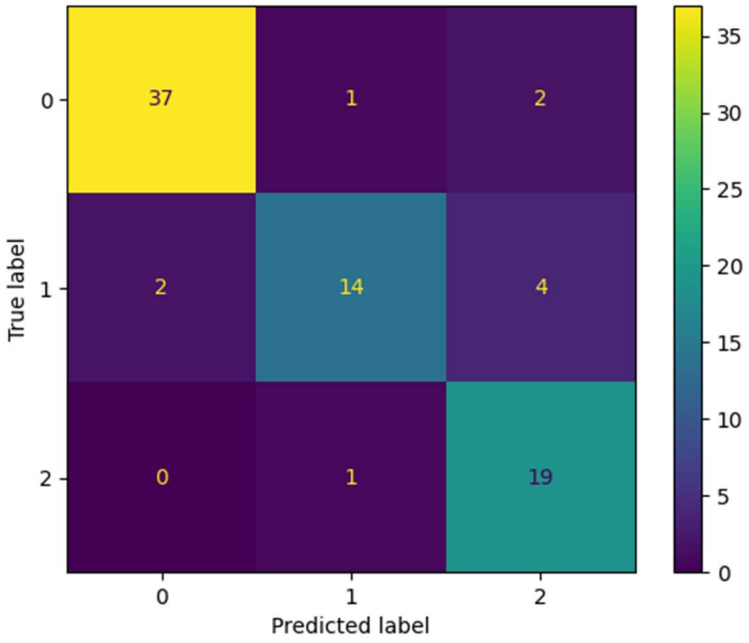


Figura 2 – Matriz de confusión de nuestro modelo de clasificación.

De forma complementaria, se aplican técnicas de agrupamiento no supervisado con el propósito de analizar la estructura interna del espacio de características y estudiar la posible separación natural entre las muestras pertenecientes a distintas clases. A la vez, se utilizan técnicas de *Kernel Density Estimation (KDE)* para modelar la distribución probabilística de las características dentro de cada clase y evaluar el grado de solapamiento entre ellas.

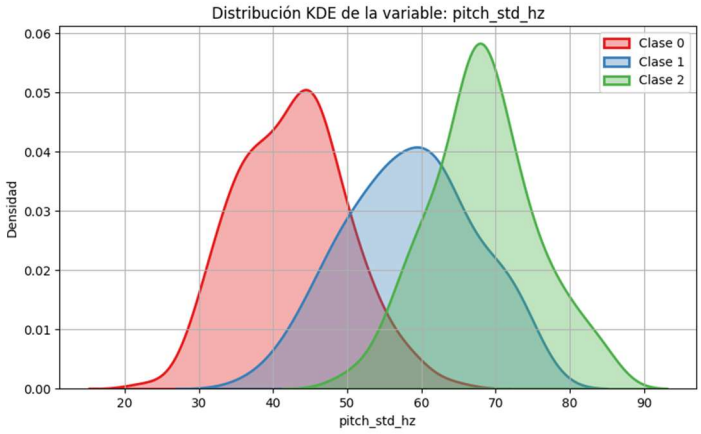


Figura 3 – Ejemplo de KDE, específicamente el de la variable Pitch_Std_Hz.

4. Resultados

El modelo seleccionado se somete a un proceso de validación adicional utilizando tanto datos internos como datos externos al conjunto empleado durante el entrenamiento, con el fin de evaluar su capacidad de generalización y analizar su comportamiento ante nuevas muestras no vistas previamente.

```
Class probabilities (decimals): ['1.0000', '0.0000', '0.0000']
Predicted label: 0
```

Figura 4 – Resultado de predicción con datos internos sin depresión

```
Class probabilities (decimals): ['0.0009', '0.9887', '0.0104']
Predicted label: 1
```

Figura 5 – Resultado de predicción con datos internos con depresión grado 1

```
Class probabilities (decimals): ['0.0000', '0.0002', '0.9998']
Predicted label: 2
```

Figura 6 – Resultado de predicción con datos internos con depresión grado 2

Tras confirmar el correcto funcionamiento del modelo, pasamos a la prueba final. Se ha optado por escoger un fragmento de audio de una serie animada para el proceso de validación externa. Este fragmento se analizará tanto en inglés como en español, pudiendo comprobar la continuidad y firmeza de nuestro modelo, analizando qué tan bueno es el doblaje en ambos idiomas y viendo si logran transmitir lo mismo.

```
Class probabilities (decimals): ['0.0006', '0.9903', '0.0091']
Predicted label: 1
```

Figura 7 – Resultados de predicción con datos externos en inglés.

```
Class probabilities (decimals): ['0.0003', '0.9989', '0.0008']
Predicted label: 1
```

Figura 8 – Resultados de predicción con datos externos en español.

Podemos observar que el modelo predice con una exactitud casi perfecta ambos casos, confirmando así el correcto funcionamiento de nuestro modelo para datos externos.

5. Conclusiones

El análisis realizado a lo largo del trabajo muestra que las características acústicas extraídas de las grabaciones de voz sí contienen información útil para diferenciar distintos grados de depresión mediante modelos de aprendizaje supervisado.

Los resultados obtenidos durante las pruebas presentan un comportamiento bastante consistente, especialmente al validar el modelo con datos que no habían sido utilizados durante el entrenamiento. El uso de técnicas de agrupamiento y estimación de densidad ayudó a visualizar mejor cómo se distribuían las distintas clases dentro del espacio de características y hasta qué punto existía separación entre ellas.

Sin embargo, en algunos casos sigue apareciendo un leve solapamiento entre categorías similares, algo esperable teniendo en cuenta la complejidad del problema y la propia variabilidad de la voz humana. Pese a ello, los resultados obtenidos tanto con datos internos como externos permiten considerar viable el uso del análisis de voz como herramienta de apoyo dentro del estudio de estados depresivos.

6. Referencias

[1] s3programmerlead, Multimodal Dataset for Depression Analysis, Kaggle, 2023.

Available: https://www.kaggle.com/datasets/s3programmerlead/multimodal-dataset-for-depression-analysis

[2] J. Namkung et al., “Novel Deep Learning-Based Vocal Biomarkers for Stress Detection in Koreans” Psychiatry Investigation, vol. 21, no. 11, pp. 1228–1237, 2024. doi: 10.30773/pi.2024.0131.

[3] A. Kumar, M. A. Shaun, and B. K. Chaurasia, “Identification of psychological stress from speech signal using deep learning algorithm” e-Prime – Advances in Electrical Engineering, Electronics and Energy, vol. 9, 2024, Art. no. 100707, doi: 10.1016/j.prime.2024.100707.

EARLY DETECTION OF DEPRESSION THROUGH VOICE SIGNALS

Author: de los Santos González, Diego.
Supervisor: Carrero Muñiz, Dido.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

1. Introducción

Mental health is one of the main health and social challenges today, with depression being one of the most prevalent disorders and one with the greatest impact on people's quality of life. Early detection is key to improving prognosis and facilitating appropriate intervention. However, traditional diagnosis methods are mainly based on clinical interviews and questionnaires, which introduce a degree of subjectivity into the diagnostic process.

In recent years, computational analysis of biomedical signals has opened up new ways for the study of different health conditions. In this context, the human voice has attracted growing interest as a potential source of information, as it conveys not only linguistic content but also acoustic characteristics related to the speaker's emotional state.

Various studies have pointed out that parameters such as energy, tone of voice, or temporal variability when speaking can present different patterns in people with depressive symptoms. This has led to the development of automatic methods capable of analysing voice recordings and extracting valuable information from them.

Within this framework, this study is situated at the intersection of digital signal processing and machine learning, exploring the potential of voice as a support tool in the automated analysis of depressive states.

2. Definición del proyecto

The objective of this project is the development and implementation of a voice analysis system aimed at classifying depressive states into three categories: absence of depression, grade 1 depression, and grade 2 depression.

The work develops the entire technical process, starting with the processing of raw audio recordings. From these recordings, a set of acoustic features is extracted that allows for the numerical definition of the voice signal and generates a database that will serve as input for supervised learning models. The objective is to analyze the ability of these models to distinguish between the three defined groups and select the model with the best performance when evaluatng with internal and external data entires.

Likewise, unsupervised clustering techniques are incorporated in order to study the internal structure of the data and facilitate the graphical interpretation of the results. Finally, the selected model will undergo a validation process with both internal and external data, with the purpose of evaluating its generalization capability and ensuring its proper functioning outside the training set.

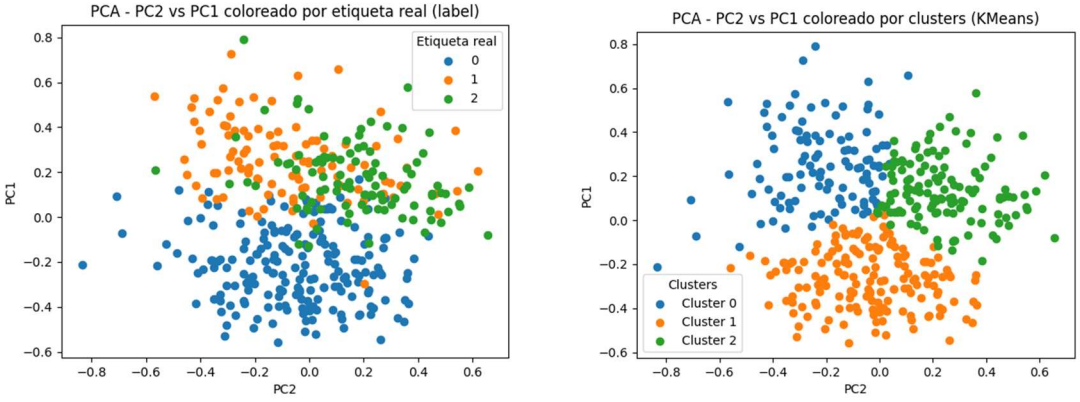


Figure 1 – Left: Dataset actual clusters. Right: Dataset predicted clusters.

The scope of the work focuses on the technical analysis and computational evaluation of the developed system, without intending to replace clinical diagnosis, but rather to explore the feasibility of using vocal patterns as a support tool in automated detection.

3. Descripción del modelo/sistema/herramienta

The system developed in this work is based on an experimental approach aimed at the analysis and evaluation of supervised learning models applied to acoustic features directly extracted from raw voice recordings.

In the first stage, unprocessed audio signals corresponding to patients classified into three clinical categories are considered: absence of depression, depression grade 1, and depression grade 2. From these signals, a set of acoustic features is extracted to provide a quantitative representation of relevant voice properties. These properties include parameters such as pitch mean, pitch standard deviation, energy mean, energy standard deviation, pause ratio, and MFCC coefficients. Each recording is represented by a numerical feature vector that summarizes the information contained in the original signal.

The extracted features are organized into a structured dataset, where each sample is associated with its corresponding label. This dataset constitutes the foundation for the subsequent modeling experiments.

Within this feature space, several supervised learning models are trained and compared to evaluate their ability to discriminate among the three classes. The training process is carried out using stratified cross-validation, ensuring that class proportions remain constant across partitions and reducing the risk of overfitting. The model selection process is based on evaluation metrics appropriate for multiclass classification.

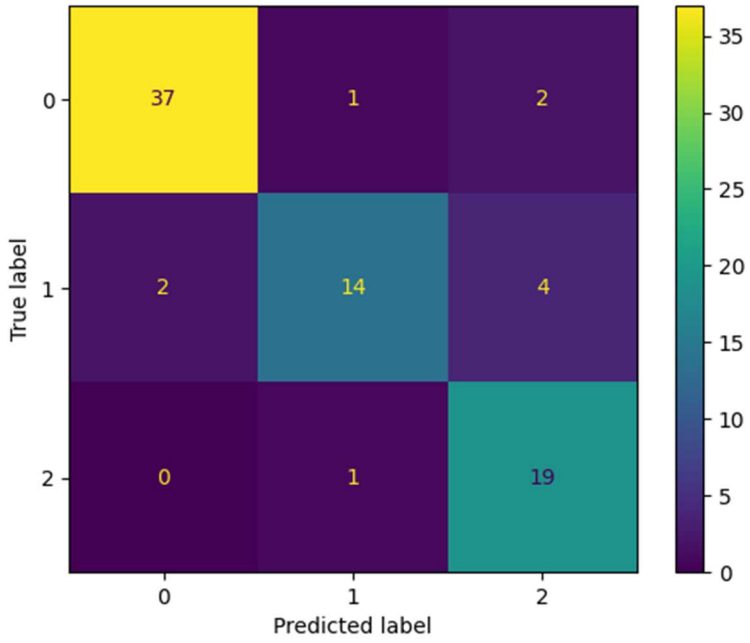


Figure 2 – Confussion Matrix of our clasification model.

In addition, unsupervised clustering techniques are applied to analyze the internal structure of the feature space and to study the possible natural separation between samples belonging to different classes. Kernel Density Estimation is also employed to model the probabilistic distribution of features within each class and to assess the degree of overlap between them.

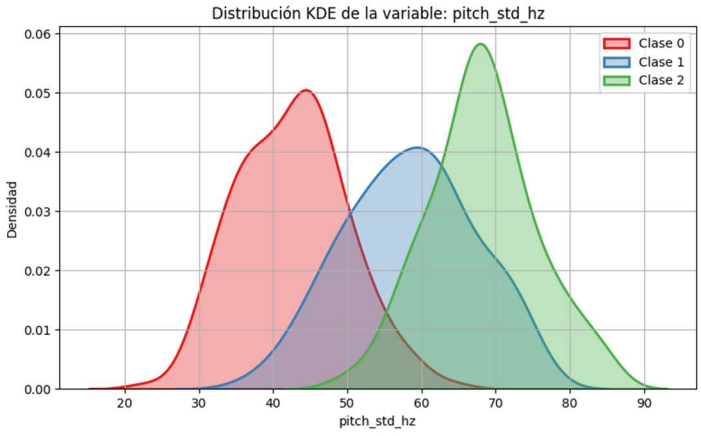


Figure 3 –KDE example of Pitch_Std_Hz

4. Resultados

The selected model undergoes an additional validation process using both internal and external datasets in order to evaluate its generalization capability and analyze its behavior on previously unseen samples.

```
Class probabilities (decimals): ['1.0000', '0.0000', '0.0000']
Predicted label: 0
```

Figure 4 – Prediction result from internal data with depression class = 0.

```
Class probabilities (decimals): ['0.0009', '0.9887', '0.0104']
Predicted label: 1
```

Figure 5 – Prediction result from internal data with depression class = 1.

```
Class probabilities (decimals): ['0.0000', '0.0002', '0.9998']
Predicted label: 2
```

Figure 6 – Prediction result from internal data with depression class = 2.

After confirming that our model Works correctly, we advence to the final test. Our model Will be tested with external audio from an animates series. This fragment Will be analysed in english and spanish, therefore analysing if the voice over transmit the same emotion without depending on the language its voiced over.

```
Class probabilities (decimals): ['0.0006', '0.9903', '0.0091']
Predicted label: 1
```

Figure 7 – Prediction result from english external data.

```
Class probabilities (decimals): ['0.0003', '0.9989', '0.0008']
Predicted label: 1
```

Figure 8 – Prediction result from spanish external data.

We can observe that the model predicts with an almost perfect accuracy in both cases, therefore confirming that our model also works with external data.

5. Conclusiones

The analysis conducted confirms that the acoustic features extracted from voice recordings contain relevant information for discriminating between different degrees of depression using supervised learning models.

The results show that it is possible to achieve consistent performance with adequate generalization capability. The complementary use of clustering techniques and density estimation has strengthened the interpretation of class separation within the feature space.

Although some overlap is observed between clinically adjacent categories, the adopted approach and the validation results obtained using both internal and external data demonstrate the technical feasibility of voice signal analysis as a support tool in the study of depression.

6. Referencias

[1] s3programmerlead, Multimodal Dataset for Depression Analysis, Kaggle, 2023. Available: https://www.kaggle.com/datasets/s3programmerlead/multimodal-dataset-for-depression-analysis

[2] J. Namkung et al., “Novel Deep Learning-Based Vocal Biomarkers for Stress Detection in Koreans” Psychiatry Investigation, vol. 21, no. 11, pp. 1228–1237, 2024. doi: 10.30773/pi.2024.0131.

[3] A. Kumar, M. A. Shaun, and B. K. Chaurasia, “Identification of psychological stress from speech signal using deep learning algorithm” e-Prime – Advances in Electrical Engineering, Electronics and Energy, vol. 9, 2024, Art. no. 100707, doi: 10.1016/j.prime.2024.100707.

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

La depresión se ha convertido en uno de los problemas de salud mental con mayor impacto durante los últimos años. Millones de personas conviven diariamente con síntomas relacionados con alteraciones emocionales, pérdida de motivación, cambios en el comportamiento o dificultades cognitivas que afectan directamente a su vida personal, social y laboral. En muchos casos el problema no aparece de forma repentina ni completamente evidente. Los síntomas pueden desarrollarse poco a poco y eso hace que detectar ciertos estados intermedios o fases tempranas no siempre resulte sencillo.

Gran parte del diagnóstico continúa dependiendo de entrevistas clínicas, observación profesional y evaluación psicológica. Todo esto sigue siendo fundamental dentro del ámbito médico, aunque existen situaciones donde determinadas señales pasan más desapercibidas o donde pequeños cambios emocionales no llegan a identificarse fácilmente en fases iniciales. Ahí es donde empieza a cobrar fuerza la idea de utilizar herramientas tecnológicas capaces de analizar información biomédica de forma complementaria.

La voz ha recibido mucha atención dentro de esta línea de investigación. Aunque muchas veces se percibe únicamente como un medio de comunicación, realmente contiene una enorme cantidad de información relacionada con el estado emocional y psicológico de una persona. Cambios en la entonación, la estabilidad vocal, la energía, la velocidad del habla o las pausas pueden variar de forma clara dependiendo del estado anímico del individuo.

Durante los últimos años distintos trabajos han mostrado que ciertos patrones acústicos tienden a repetirse en pacientes con síntomas depresivos. Algunas personas presentan voces más planas, menor variabilidad tonal, pausas más largas o cambios en el ritmo del habla. Evidentemente no existe una única “voz depresiva” ni una frontera

completamente definida entre pacientes, aunque sí empiezan a aparecer tendencias comunes cuando se analizan grandes cantidades de grabaciones.

El crecimiento reciente de la inteligencia artificial ha hecho que este tipo de análisis gane todavía más presencia dentro del ámbito biomédico. Los algoritmos de aprendizaje automático son capaces de encontrar relaciones complejas dentro de enormes volúmenes de datos y eso ha abierto muchas posibilidades en problemas donde las diferencias no siempre son fáciles de detectar visualmente o mediante análisis tradicionales.

Muchas variaciones acústicas son pequeñas y aparecen mezcladas entre sí, haciendo difícil separar manualmente qué partes contienen información realmente útil y cuáles corresponden simplemente a variaciones normales entre personas. Ahí los modelos de aprendizaje automático empiezan a ganar importancia porque pueden analizar simultáneamente múltiples características acústicas y estudiar cómo evolucionan conjuntamente.

Aun así, sigue siendo un problema complicado. Las emociones no funcionan como bloques completamente separados y muchas veces distintas personas pueden expresar estados psicológicos parecidos utilizando patrones vocales muy diferentes. El contexto de grabación, la edad, el idioma, el entorno acústico o incluso la propia personalidad terminan introduciendo una variabilidad enorme dentro de las grabaciones de voz.

Precisamente por eso gran parte de la investigación actual ya no se centra únicamente en conseguir modelos con una accuracy alta. Entender qué características contienen información útil, cómo se mezclan las distintas regiones acústicas o hasta qué punto los clasificadores consiguen generalizar fuera de condiciones controladas ha pasado a ocupar una parte muy importante dentro de este tipo de trabajos.

El análisis automático de voz sigue creciendo como una de las líneas con más proyección dentro del apoyo tecnológico aplicado a salud mental. La posibilidad de detectar ciertos patrones emocionales de forma temprana utilizando únicamente grabaciones de voz abre escenarios muy interesantes relacionados con monitorización emocional, herramientas de apoyo clínico y seguimiento no invasivo de pacientes.

1.1 MOTIVACIÓN DEL PROYECTO

La motivación principal del proyecto nace tanto del interés por la ingeniería aplicada al ámbito de la salud como de la propia complejidad que presenta el análisis de voz dentro de problemas relacionados con salud mental. Gran parte de los trabajos existentes se centran únicamente en el rendimiento final de los clasificadores, aunque muchas veces dejan más de lado todo el proceso previo relacionado con tratamiento de audio, selección de características o interpretación real de los resultados obtenidos.

Aquí la idea no consistía simplemente en entrenar un clasificador capaz de diferenciar varias etiquetas. Desde el inicio tenía bastante peso construir una metodología clara y reproducible que permitiese seguir todo el flujo de trabajo desde las grabaciones originales hasta la obtención de predicciones interpretables. Eso incluía trabajar directamente sobre audio crudo, analizar cómo evolucionaban las características acústicas entre clases y observar qué tipo de relaciones terminaban apareciendo realmente dentro del espacio de variables.

La voz lleva tiempo ganando importancia como posible biomarcador dentro del análisis emocional y psicológico. Frente a métodos tradicionales basados únicamente en cuestionarios o entrevistas clínicas, las señales de voz permiten trabajar sobre información cuantificable que puede procesarse automáticamente y analizarse desde un punto de vista mucho más objetivo. Pequeños cambios en entonación, energía, pausas o estabilidad vocal pueden reflejar alteraciones emocionales que muchas veces no resultan tan fáciles de identificar mediante observación convencional.

Precisamente ahí es donde las técnicas de aprendizaje automático empiezan a tener mucho sentido. Muchas diferencias acústicas son pequeñas, se mezclan entre sí y no siempre pueden interpretarse fácilmente observando una única variable aislada. Analizar simultáneamente decenas de características permite encontrar relaciones que visualmente pasarían mucho más desapercibidas, sobre todo en regiones donde distintas clases comparten rasgos vocales similares.

Otro aspecto importante dentro del proyecto era evitar desarrollar una solución demasiado cerrada o dependiente de un único escenario concreto. La intención desde el principio iba más orientada a construir una base que pudiese ampliarse y reutilizarse posteriormente en futuras investigaciones, nuevas bases de datos o problemas relacionados con análisis automático de voz.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

En este proyecto se utilizarán distintas herramientas relacionadas con procesamiento de señal y aprendizaje automático para construir un flujo completo de análisis y clasificación de señales de voz.

Python será el lenguaje principal del desarrollo debido a la gran cantidad de librerías disponibles dentro del ámbito de análisis de datos, *machine learning* y procesamiento de audio. Todo el código se desarrollará en *Visual Studio Code*, un entorno flexible que facilita la organización de *scripts* y la estructuración general del proyecto.

La manipulación de datos se realizará principalmente mediante *pandas* y *numpy*. pandas permitirá trabajar con estructuras tipo tabla mientras que *numpy* se utilizará durante operaciones numéricas y tratamiento de *arrays*. Para generar representaciones visuales se empleará *matplotlib*, una librería orientada a la creación de gráficas que ayudará a analizar tanto las variables acústicas como el rendimiento de los clasificadores.

Las grabaciones de voz serán procesadas utilizando *librosa*. A partir de los audios originales se extraerán características acústicas relacionadas con *pitch*, *energy*, MFCC y distintos parámetros espectrales que posteriormente formarán parte del *dataset*. Transformar señales de audio en variables numéricas permitirá trabajar sobre información cuantificable relacionada con la voz y facilitará el entrenamiento posterior de los clasificadores.

scikit-learn proporcionará gran parte de las herramientas necesarias para entrenamiento, validación y evaluación de clasificadores. La separación entre entrenamiento y *test* se realizará mediante *train_test_split* utilizando además el parámetro *stratify* para conservar la proporción original entre clases dentro de cada subconjunto.

Esto ayudará a obtener una evaluación más representativa del rendimiento y reducirá el impacto del ligero desbalanceo presente en el *dataset*.

En lugar de depender únicamente de una única división de los datos, la validación cruzada se apoyará sobre *StratifiedKFold*. Esta técnica divide el *dataset* en distintos *folds* y permite entrenar el clasificador utilizando múltiples combinaciones de entrenamiento y validación. Mantener la proporción original entre clases dentro de cada subconjunto permitirá obtener estimaciones más robustas y reducirá la dependencia de una sola partición concreta.

Categorical Boosting (CatBoost) será el clasificador principal utilizado durante el proyecto. Este algoritmo *boosting* basado en árboles de decisión construye el aprendizaje de forma secuencial utilizando los errores generados durante iteraciones anteriores para mejorar progresivamente la capacidad de predicción. *CatBoost* incorpora además mecanismos de regularización y estrategias específicas orientadas a controlar el sobreajuste y estabilizar el entrenamiento.

La búsqueda de hiperparámetros se realizará mediante *RandomizedSearchCV*. Este método permite recorrer configuraciones aleatorias dentro de rangos previamente definidos sin necesidad de evaluar todas las combinaciones posibles, reduciendo así el coste computacional. Variables como profundidad de árboles, número de iteraciones o *learning rate* podrán ajustarse utilizando validación cruzada para localizar configuraciones adecuadas para el problema planteado.

La evaluación del rendimiento no se apoyará únicamente sobre *accuracy*. ROC AUC permitirá estudiar la capacidad de separación entre clases mientras que las matrices de confusión facilitarán observar aciertos y errores dentro de cada categoría de forma mucho más detallada. Utilizar distintas métricas ayudará a obtener una visión más completa del funcionamiento general del clasificador.

El análisis exploratorio de características se realizará mediante KDE, una técnica de estimación de densidad orientada a visualizar cómo se distribuyen las distintas variables acústicas entre clases. Esto facilitará detectar posibles solapamientos y estudiar qué características contienen una mayor capacidad de separación dentro del espacio de variables.

PCA se utilizará como herramienta de reducción de dimensionalidad y visualización. Mediante combinaciones lineales entre variables originales será posible representar el dataset dentro de un espacio reducido conservando gran parte de la información original. Los *loadings* asociados a cada componente ayudarán además a interpretar qué variables acústicas tienen mayor peso dentro de la estructura general de los datos.

Sobre los datos transformados mediante PCA se aplicará *K-Means*, un algoritmo de *clustering* que agrupa muestras según similitud dentro del espacio de características. Para seleccionar el número de clústeres se utilizarán tanto *Elbow Method* como *Silhouette Score*. El primero permitirá analizar la evolución de la inercia mientras que el segundo evaluará la calidad de separación entre grupos.

.

Capítulo 3. ESTADO DE LA CUESTIÓN

El análisis de señales de voz aplicado a salud mental ha ido ganando relevancia durante los últimos años debido a la posibilidad de detectar alteraciones emocionales y psicológicas mediante características acústicas obtenidas directamente del habla. Distintos estudios han mostrado que variables relacionadas con la frecuencia fundamental de la voz, la energía, las pausas o determinadas representaciones espectrales contienen información útil sobre el estado emocional y cognitivo de una persona [2][3]. En este contexto, la voz empieza a considerarse un biomarcador digital capaz de aportar información objetiva sin necesidad de procedimientos invasivos.

Uno de los enfoques más estudiados dentro de este campo es la utilización de características acústicas tradicionales junto con modelos de clasificación supervisada. Namkung et al. desarrollaron un sistema basado en biomarcadores vocales para detección de estrés utilizando grabaciones de voz obtenidas en condiciones relajadas y bajo estrés inducido. El trabajo analiza cómo el aumento de estrés modifica distintos componentes acústicos de la señal y utiliza representaciones *Mel Spectrogram* junto con arquitecturas de *deep learning* para diferenciar ambos estados [2]. Dentro del estudio también se destaca que variables relacionadas con entonación, variabilidad tonal o componentes espectrales ya habían demostrado utilidad anteriormente para analizar alteraciones emocionales mediante voz [2].

La utilización de representaciones espectrales aparece de forma muy recurrente dentro de la literatura debido a que permiten representar información frecuencial relevante del habla de una forma compacta y relativamente robusta frente a pequeñas variaciones de grabación. En el trabajo de Namkung et al. se utilizan modelos basados en ECAPA-TDNN para capturar relaciones complejas dentro de las señales de voz, alcanzando resultados competitivos sobre muestras independientes [2]. Los autores remarcan también que los modelos *deep learning* consiguen adaptarse mejor a la

variabilidad real del habla frente a enfoques estadísticos tradicionales, especialmente cuando existen diferencias complejas entre muestras [2].

Otros trabajos han explorado directamente la identificación de alteraciones psicológicas mediante técnicas de aprendizaje profundo aplicadas sobre señales de voz. El estudio *"Identification of psychological stress from speech signal using deep learning algorithm"* plantea el uso de modelos basados en *deep learning* para detectar estrés psicológico a partir del habla, reforzando la idea de que determinadas modificaciones acústicas pueden utilizarse como indicadores relacionados con estados emocionales y mentales [3]. Este tipo de investigaciones muestran una tendencia clara hacia sistemas automáticos capaces de extraer información relevante directamente desde la señal de voz sin depender únicamente de cuestionarios subjetivos.

También resulta importante mencionar que gran parte de los trabajos existentes se centran en estrés, ansiedad o emociones concretas más que en depresión clasificada por niveles. Aun así, muchos de los cambios vocales estudiados en esos problemas aparecen relacionados con fenómenos similares: alteraciones en ritmo del habla, estabilidad vocal, pausas más prolongadas o cambios en intensidad. Precisamente por eso, varios enfoques utilizados previamente para estrés o reconocimiento emocional pueden trasladarse parcialmente al análisis de estados depresivos [2][3].

Otro aspecto que aparece repetidamente dentro de la literatura es la dificultad para separar categorías intermedias cuando las diferencias acústicas no son completamente evidentes. Incluso utilizando modelos avanzados siguen apareciendo regiones de solapamiento entre estados emocionales distintos, algo mencionado también dentro del trabajo de Namkung et al. [2]. Esto resulta especialmente relevante dentro del contexto de clasificación multiclase, donde las fronteras entre categorías pueden quedar parcialmente mezcladas.

La evolución reciente de este campo apunta hacia sistemas cada vez más orientados a monitorización remota y biomarcadores digitales. Algunos estudios plantean aplicaciones capaces de analizar la voz del usuario para estimar niveles de estrés o alteraciones emocionales utilizando únicamente grabaciones de audio y dispositivos cotidianos [2]. La posibilidad de realizar análisis no invasivos mediante voz convierte este enfoque en una línea especialmente interesante dentro del ámbito biomédico y de salud digital.

A pesar del avance observado durante los últimos años, la literatura sigue mostrando limitaciones importantes. Muchos trabajos utilizan datasets relativamente pequeños, condiciones de grabación muy controladas o problemas binarios simplificados, dificultando la generalización de los modelos a escenarios reales más complejos [2][3]. Precisamente por eso continúa existiendo interés en desarrollar sistemas capaces de mantener estabilidad incluso cuando aparecen diferencias acústicas sutiles y cierto solapamiento entre categorías.

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

El análisis del estado de la cuestión deja bastante claro que la detección de depresión mediante voz lleva varios años creciendo como línea de investigación y que ya existen trabajos capaces de obtener resultados prometedores utilizando características acústicas y aprendizaje automático. Muchas propuestas consiguen separar distintos estados emocionales con niveles de precisión elevados dentro de escenarios controlados, aunque gran parte de esos trabajos siguen quedándose muy ligados al entorno experimental donde fueron desarrollados.

Ahí empieza a aparecer uno de los principales problemas de este tipo de enfoques. En muchos casos los modelos funcionan correctamente sobre *datasets* concretos, aunque luego resulta más difícil comprobar cómo reaccionan fuera de esas condiciones iniciales o hasta qué punto consiguen generalizar frente a nuevas grabaciones, distintos pacientes o situaciones menos controladas. Parte de la investigación actual ya no gira únicamente alrededor de aumentar *accuracy*, sino de entender qué características de la voz contienen realmente información útil y cómo construir herramientas que puedan trasladarse más fácilmente a escenarios reales.

Dentro de esa idea se plantea este proyecto, el objetivo no iba orientado a construir una solución extremadamente compleja ni a depender de múltiples fuentes biomédicas simultáneamente. Toda la propuesta se centra únicamente en el análisis de la señal de audio y en estudiar hasta dónde puede llegar la propia voz como fuente de información relacionada con depresión. Reducir complejidad permitía trabajar sobre una estructura mucho más clara, interpretar mejor las relaciones entre variables acústicas y facilitar posibles ampliaciones futuras sobre nuevas bases de datos o entornos distintos.

La voz tiene además una ventaja importante frente a otros tipos de señales biomédicas: puede obtenerse de forma muy natural. No requiere equipamiento médico especializado ni procedimientos invasivos y aparece continuamente en situaciones cotidianas mediante teléfonos móviles, grabaciones o conversaciones normales. Precisamente por eso el interés alrededor de herramientas relacionadas con *e-health* y monitorización emocional ha crecido tanto durante los últimos años.

Gran parte del atractivo de este tipo de enfoques aparece ahí. Poder analizar cambios emocionales o posibles alteraciones psicológicas utilizando únicamente grabaciones de voz abre escenarios muy amplios relacionados con seguimiento continuo, apoyo clínico o detección temprana. Evidentemente siguen existiendo muchas limitaciones y las fronteras entre estados emocionales nunca quedan completamente definidas, aunque precisamente esa complejidad es una de las razones por las que este campo continúa creciendo tan rápido dentro del ámbito de inteligencia artificial aplicada a salud.

4.1.1.1 - NECESIDAD DE HERRAMIENTAS OBJETIVAS Y ACCESIBLES PARA LA DETECCIÓN DE DEPRESIÓN

Uno de los mayores problemas alrededor del diagnóstico de depresión sigue estando relacionado con la subjetividad. Gran parte de las evaluaciones continúan dependiendo de entrevistas clínicas, cuestionarios o autoinformes donde la interpretación del profesional y la capacidad del propio paciente para expresar su estado emocional terminan teniendo mucho peso. Aunque existen criterios clínicos bien definidos, como los establecidos dentro del DSM-5, siguen apareciendo diferencias entre evaluaciones y dificultades para detectar ciertos estados en fases tempranas.

Esto se vuelve todavía más complicado en situaciones donde los síntomas no aparecen de forma completamente evidente o donde los cambios emocionales son

progresivos. Muchas veces pequeñas alteraciones en el estado psicológico pueden pasar desapercibidas durante bastante tiempo, sobre todo en etapas intermedias o leves donde las fronteras entre normalidad y afectación emocional no quedan tan claras.

Ahí es donde empiezan a ganar importancia herramientas capaces de aportar información más objetiva y cuantificable. La voz contiene una gran cantidad de información relacionada con el estado emocional de una persona y permite trabajar sobre parámetros medibles de forma automática. Cambios en entonación, estabilidad vocal, energía, pausas o ritmo del habla pueden analizarse directamente a partir de grabaciones de audio sin depender únicamente de interpretación subjetiva.

Otra ventaja importante aparece en la propia accesibilidad de la señal de voz ya que capturar audio no requiere equipamiento médico complejo ni procedimientos invasivos y puede realizarse fácilmente mediante dispositivos cotidianos como teléfonos móviles, ordenadores o grabadoras convencionales. Eso abre escenarios relacionados con seguimiento remoto, monitorización continua o análisis periódico del estado emocional sin necesidad de realizar evaluaciones clínicas presenciales de forma constante.

Gran parte del interés actual alrededor de este tipo de herramientas aparece precisamente ahí. Poder analizar posibles alteraciones emocionales utilizando únicamente grabaciones de voz permitiría construir soluciones mucho más accesibles y fáciles de integrar dentro de entornos reales de salud digital, especialmente en situaciones donde una detección temprana o un seguimiento continuo pueden tener un impacto importante sobre la evolución del paciente.

4.1.2 - AUSENCIA DE SOLUCIONES PRÁCTICAS BASADAS EXCLUSIVAMENTE EN CARACTERÍSTICAS ACÚSTICAS

Muchos trabajos relacionados con detección de depresión mediante voz consiguen resultados prometedores dentro de datasets concretos y escenarios controlados. El problema aparece después, cuando gran parte de esas propuestas dependen de combinar múltiples fuentes de datos al mismo tiempo, como texto, expresiones faciales o señales fisiológicas, o utilizan metodologías demasiado complejas para trasladarlas fácilmente a entornos reales.

La voz ya ha demostrado contener información útil relacionada con estados emocionales y alteraciones psicológicas. Precisamente por eso sigue creciendo el interés alrededor de enfoques centrados únicamente en la señal de audio y en las características acústicas extraídas directamente de las grabaciones. Reducir la dependencia de otras fuentes biomédicas ayuda a simplificar mucho el proceso completo de análisis y facilita posibles aplicaciones futuras en contextos reales.

Gran parte de los trabajos anteriores continúan utilizando clasificaciones binarias donde únicamente se diferencia entre presencia o ausencia de depresión. Ese planteamiento simplifica el problema, aunque deja fuera una parte importante de la complejidad asociada a la evolución del trastorno y a los distintos grados de afectación emocional que pueden aparecer entre pacientes.

Trabajar con una clasificación multiclase permite representar de forma más precisa distintos niveles relacionados con depresión y se aproxima mejor a situaciones clínicas reales donde los estados emocionales no funcionan simplemente como dos bloques completamente separados.

Otro punto importante aparece en la interpretabilidad. Utilizar parámetros acústicos bien definidos hace posible analizar qué variables están influyendo realmente

durante clasificación y facilita mucho más la validación del proceso completo. En aplicaciones relacionadas con salud esto tiene bastante peso porque no solo importa obtener buenas predicciones, sino entender también qué información está utilizando el clasificador para tomar decisiones.

4.1.3 - POTENCIAL DE ESCALABILIDAD Y APLICACIÓN EN ENTORNOS REALES

Trabajar únicamente con señales de voz y características acústicas hace que el proyecto pueda adaptarse con facilidad a muchos entornos distintos sin depender de infraestructuras especialmente complejas. La posibilidad de capturar audio mediante teléfonos móviles, ordenadores o dispositivos cotidianos abre escenarios relacionados con aplicaciones móviles, plataformas de telemedicina o herramientas orientadas a monitorización remota del estado emocional.

La metodología planteada deja además margen para futuras ampliaciones desde distintos puntos de vista. Una de las posibilidades pasa por aumentar el número de clases utilizadas durante clasificación para representar con mayor precisión distintos grados relacionados con depresión. Eso permitiría construir evaluaciones más detalladas del estado emocional del paciente en lugar de limitar el análisis únicamente a escenarios muy simplificados.

Las mismas técnicas podrían extenderse además a otros problemas relacionados con análisis emocional mediante voz. Estados asociados a ansiedad, estrés o fatiga comparten muchas similitudes dentro del ámbito de procesamiento acústico y podrían incorporarse posteriormente dentro de una herramienta más amplia orientada al análisis automático de estados emocionales.

Otra línea importante aparece alrededor de los propios datos utilizados durante entrenamiento. Incorporar *datasets* más amplios y variados permitiría trabajar sobre una

mayor diversidad de voces, condiciones de grabación, idiomas o acentos, algo especialmente importante en problemas donde la variabilidad entre pacientes puede llegar a ser muy alta.

El crecimiento que están teniendo las plataformas de salud digital y monitorización remota hace que este tipo de herramientas empiece a ganar cada vez más presencia dentro del mercado tecnológico relacionado con salud. Poder integrar análisis de voz dentro de dispositivos que ya forman parte del día a día supone una ventaja importante porque evita depender de hardware específico y facilita mucho más la implementación práctica de este tipo de soluciones.

Todo esto hace que el proyecto no quede limitado únicamente a un caso concreto o a un único escenario experimental. La estructura planteada permite seguir ampliando funcionalidades, incorporar nuevas líneas de análisis y adaptar el enfoque a distintos contextos relacionados con salud digital y análisis emocional mediante voz.

4.2 OBJETIVOS

El proyecto tiene como objetivo principal desarrollar un sistema capaz de analizar señales de voz y predecir el estado depresivo de un individuo utilizando características acústicas extraídas directamente del audio. La idea no se centra únicamente en entrenar un clasificador funcional, sino en construir un flujo completo de análisis capaz de trabajar desde las grabaciones originales hasta la obtención de predicciones sobre datos externos a la base utilizada durante entrenamiento.

A partir de este objetivo general se plantean varios objetivos específicos:

• Preparación y organización del dataset.

Una de las primeras metas del proyecto consiste en crear, preprocesar y trabajar sobre una base de datos de audio correctamente estructurada y etiquetada según el estado depresivo o nivel de severidad correspondiente. La calidad de las grabaciones y la representatividad de las muestras tienen un impacto directo sobre el rendimiento posterior del clasificador, por lo que esta parte resulta fundamental dentro de todo el flujo de trabajo.

• Extracción de características acústicas.

Otro de los objetivos principales pasa por transformar las grabaciones de voz en variables que puedan utilizarse posteriormente durante clasificación. Para ello se plantea extraer parámetros relacionados con la energía, tono, MFCC y distintos elementos espectrales asociados al comportamiento acústico de la voz. Este proceso permite representar señales de audio complejas mediante características numéricas más fáciles de analizar.

• Desarrollo del clasificador multiclase.

El proyecto busca implementar un clasificador capaz de distinguir entre distintos estados depresivos utilizando exclusivamente información acústica procedente de la voz. El enfoque planteado no se limita únicamente a detectar presencia o ausencia de

depresión, sino que propone una clasificación multiclase orientada a representar distintos niveles de severidad.

• Evaluación de capacidad de generalización.

Otro de los puntos importantes consiste en comprobar hasta qué punto el clasificador es capaz de ofrecer predicciones fiables sobre datos no utilizados durante entrenamiento. Evaluar el comportamiento sobre grabaciones externas permitirá analizar si el enfoque mantiene capacidad de generalización fuera de la base de datos original.

• Interpretación y análisis de resultados.

Además del rendimiento final, el proyecto plantea analizar el comportamiento general del clasificador, estudiar las limitaciones del enfoque y observar qué características acústicas contienen mayor relevancia dentro del proceso de clasificación. Todo esto permitirá valorar la viabilidad real del sistema y dejar una base preparada para posibles ampliaciones futuras.

4.3 METODOLOGÍA

La metodología planteada para el proyecto organiza todas las etapas necesarias desde la obtención de las grabaciones de voz hasta la validación final del clasificador. Todo el flujo de trabajo gira alrededor del análisis de señales de audio, su transformación en variables numéricas y el desarrollo de un sistema capaz de relacionar determinadas características acústicas con distintos estados depresivos.

El primer paso consiste en localizar una base de datos adecuada que contenga grabaciones etiquetadas según el estado depresivo correspondiente. A partir de ahí se realizará un análisis exploratorio inicial orientado a comprender cómo se distribuyen las muestras dentro del *dataset* y detectar posibles limitaciones relacionadas con desbalanceo entre clases, calidad de las grabaciones o diferencias entre audios.

Las grabaciones de voz pasarán por distintas fases de limpieza y preprocesamiento antes de transformarse en información numérica utilizable durante clasificación. A partir de cada audio se obtendrán distintas representaciones acústicas relacionadas con la forma en la que evoluciona la voz, permitiendo describir aspectos vocales que posteriormente servirán para entrenar los clasificadores.

Toda esa información se organizará después dentro de una base de datos estructurada donde se aplicarán procesos de limpieza y validación orientados a asegurar consistencia entre variables y etiquetas. Sobre ese conjunto de datos se realizará un análisis exploratorio de características utilizando técnicas relacionadas con estimación de densidad y análisis de importancia para estudiar qué variables contienen mayor capacidad de separación dentro del problema planteado.

Una vez completada la fase de análisis de datos se desarrollarán distintos clasificadores con el objetivo de comparar enfoques y seleccionar la alternativa más adecuada en términos de rendimiento y capacidad de generalización. Las predicciones generadas por el clasificador seleccionado se estudiarán posteriormente mediante distintas representaciones visuales orientadas a interpretar mejor el funcionamiento general del sistema.

La última parte del flujo metodológico se centrará en la validación utilizando tanto datos internos como grabaciones externas al conjunto de entrenamiento. Esto permitirá comprobar hasta qué punto el clasificador mantiene capacidad de generalización frente a nuevos audios y evaluar su posible aplicación fuera de las condiciones iniciales del dataset.

4.4 PLANIFICACIÓN TEMPORAL Y ESTIMACIÓN ECONÓMICA

4.4.1 – PLANIFICACIÓN TEMPORAL

La planificación temporal del proyecto se planteó inicialmente al comienzo del curso académico, con el objetivo de ordenar las distintas fases del trabajo y establecer una previsión razonable de dedicación para cada una de ellas. Al tratarse de un proyecto con una parte importante de análisis experimental, la planificación inicial servía como guía general, aunque era previsible que algunas actividades pudieran modificarse a medida que avanzase el desarrollo.

En un primer momento, el trabajo se estructuró siguiendo una secuencia lineal. Las primeras semanas se reservaron para la investigación inicial y la elección definitiva del tema. Esta fase incluía la revisión de trabajos relacionados con el análisis de señales de voz, la detección de depresión y el uso de modelos de aprendizaje supervisado en problemas biomédicos. Una vez definido el enfoque general, se planificó la búsqueda de un conjunto de datos compatible con los objetivos del proyecto, prestando atención a que incluyera grabaciones de voz etiquetadas y que permitiera plantear un problema de clasificación en varios grados de depresión.

Después de esta fase inicial, la planificación contemplaba el preprocesado de los datos y la extracción de características acústicas. Esta parte era especialmente importante, ya que el trabajo dependía de transformar grabaciones de voz en bruto en variables numéricas útiles para los modelos. A partir de ahí, se preveía dedicar varias semanas a la evaluación de modelos de clasificación, comparando distintas alternativas y seleccionando aquella que ofreciera mejores resultados. También se incluyeron en la planificación inicial técnicas de análisis complementario, como *Kernel Density Estimation* y modelos de *clustering*, con el fin de estudiar la distribución interna de los datos y visualizar posibles separaciones entre clases.

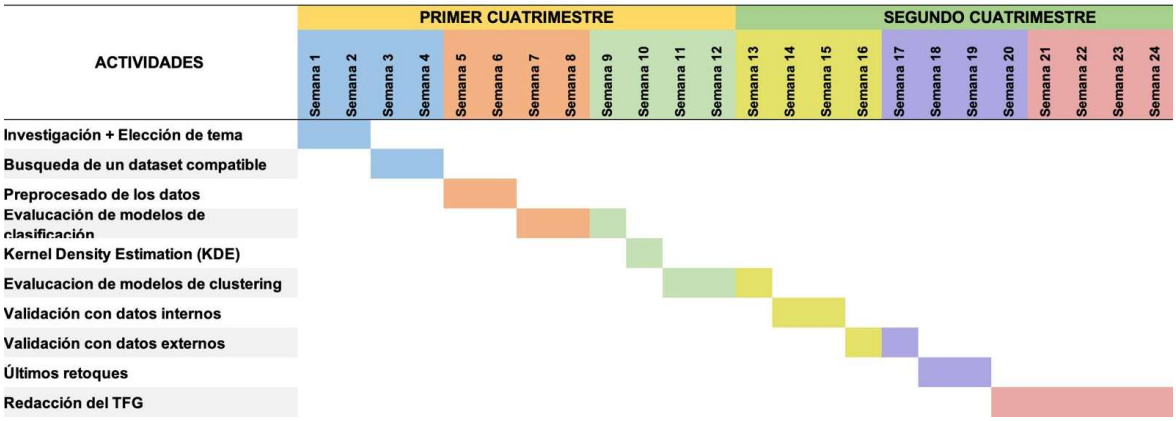


Figura 1 - Planificación temporal inicial del proyecto.

Sin embargo, a medida que avanzó el desarrollo experimental del proyecto, la planificación inicial tuvo que modificarse parcialmente debido a distintos problemas que fueron apareciendo durante el análisis de los datos y la evaluación de los modelos. Aunque la estructura general del trabajo se mantuvo, varias etapas necesitaron replantearse y repetirse, especialmente durante la fase relacionada con la selección de características acústicas y el entrenamiento de modelos de clasificación.

En las primeras pruebas realizadas se observó que algunas de las variables escogidas inicialmente no aportaban la capacidad de separación esperada, generando resultados inconsistentes entre distintos entrenamientos y dificultando la generalización de los modelos. Esto incitó a revisar nuevamente las características extraídas a partir de las grabaciones de voz y volver sobre parte del trabajo ya realizado para estudiar qué parámetros resultaban realmente útiles dentro del problema planteado.

A esta situación se sumaron ciertas inconsistencias encontradas dentro del propio conjunto de datos. Algunas diferencias entre grabaciones, junto con determinados comportamientos irregulares observados en varias muestras, hacían que los resultados cambiasen de forma considerable dependiendo de las variables utilizadas o de la

configuración del entrenamiento. Debido a ello, parte del desarrollo experimental terminó convirtiéndose en un proceso iterativo en el que fue necesario probar distintas combinaciones de características, repetir entrenamientos y comparar resultados hasta encontrar configuraciones más estables y con mejor accuracy.

Como consecuencia, varias de las fases que inicialmente estaban planteadas de manera separada acabaron solapándose durante gran parte del proyecto. El análisis de características, el entrenamiento de modelos y la validación de resultados dejaron de funcionar como bloques independientes y pasaron a retroalimentarse continuamente. En muchos casos, los resultados obtenidos durante una etapa obligaban a volver atrás para modificar decisiones tomadas anteriormente, ya fuera cambiando variables, ajustando parámetros o replanteando parte del conjunto de datos utilizado.

Esto hizo que la planificación real terminara siendo más dinámica que la prevista inicialmente. Algunas tareas necesitaron más tiempo del esperado, mientras que otras se fueron desarrollando de forma paralela conforme avanzaban las pruebas experimentales. De hecho, técnicas que en un principio tenían un papel más secundario, como *Kernel Density Estimation*, *PCA* o *clustering*, acabaron utilizándose de forma más frecuente para intentar entender mejor la distribución de las muestras y analizar visualmente por qué ciertas características producían mejores resultados que otras.

La planificación final refleja precisamente esa evolución del trabajo a lo largo del curso. Más que seguir una secuencia completamente cerrada, el desarrollo del proyecto estuvo marcado por un proceso continuo de prueba, análisis y reajuste, algo bastante habitual en trabajos relacionados con aprendizaje supervisado y análisis de datos, donde pequeñas modificaciones en las características utilizadas pueden afectar de forma importante al comportamiento de los modelos.

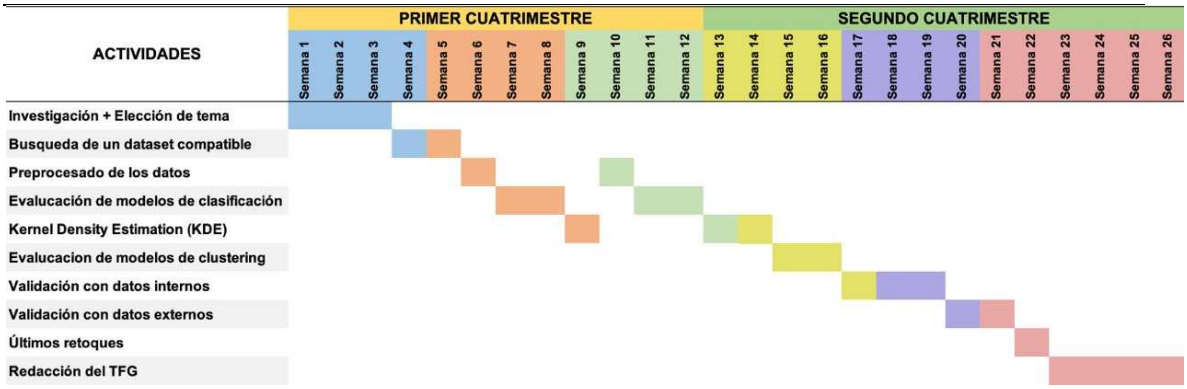


Figura 2 - Planificación temporal final del proyecto.

4.4.2 – ESTIMACIÓN ECONÓMICA

Desde el inicio del proyecto se planteó que el desarrollo se realizaría utilizando recursos locales y herramientas de software de acceso gratuito, evitando la necesidad de infraestructuras externas o servicios de computación especializados. La mayor parte del trabajo experimental se llevó a cabo utilizando un ordenador portátil *MacBook M3 Pro*, empleado tanto para el procesamiento de audio como para el entrenamiento y evaluación de los modelos de aprendizaje supervisado.

Todo el desarrollo se realizó en entorno local, utilizando principalmente *Python* junto con distintas librerías orientadas al análisis de datos, procesamiento de señales y aprendizaje automático. Al tratarse de herramientas *open source* y de uso gratuito, no fue necesario asumir costes asociados a licencias de software. En cuanto a los datos utilizados durante el proyecto, las grabaciones de voz proceden de bases de datos públicas y gratuitas, por lo que tampoco existieron costes relacionados con adquisición de datasets o acceso a plataformas de almacenamiento externas. Del mismo modo, el entrenamiento de modelos se realizó utilizando únicamente CPU, sin recurrir a servicios *cloud* ni a recursos de computación distribuida, lo que redujo considerablemente los costes asociados al desarrollo.

A pesar de ello, la realización del proyecto sí implica un coste ligado al tiempo de desarrollo. Aunque se trata de un Trabajo Fin de Grado realizado en entorno académico, resulta posible realizar una estimación aproximada del coste humano considerando las horas de trabajo invertidas a lo largo del curso. Teniendo en cuenta las tareas de investigación bibliográfica, selección y análisis de datasets, extracción de características acústicas, entrenamiento de modelos, validación experimental y redacción de la memoria, se estima una dedicación aproximada de unas 400 horas de trabajo.

Para realizar la estimación económica se considera una valoración orientativa de 15€ por hora de trabajo técnico, cifra razonable dentro de un entorno académico. Bajo esta aproximación, el coste asociado al desarrollo del proyecto constituye la mayor parte del presupuesto total.

Además del coste humano, también puede considerarse una estimación aproximada del coste proporcional del equipo utilizado y del consumo energético asociado al desarrollo experimental. Aunque el ordenador empleado no ha sido adquirido específicamente para el proyecto, se incluye una amortización parcial correspondiente al periodo de utilización durante el desarrollo del TFG.

Concepto	Descripción	Coste estimado
Equipo informático	Amortización parcial de MacBook M3 Pro utilizado durante el proyecto	400 €
Software	-	0 €
Dataset	-	0 €
Consumo eléctrico	-	50 €
Desarrollo técnico	-	6.000 €
Total estimado	-	6.450 €

Tabla 1 - Estimación económica del desarrollo del proyecto.

Capítulo 5. SISTEMA/MODELO DESARROLLADO

5.1 BÚSQUEDA Y SELECCIÓN DEL CONJUNTO DE DATOS

La primera parte del proyecto estuvo centrada en encontrar un conjunto de datos que permitiera trabajar de forma realista con el problema planteado. Aunque existen numerosos datasets relacionados con análisis de voz, no todos resultan útiles para tareas de clasificación de depresión, especialmente cuando se busca diferenciar entre varios grados del trastorno y no únicamente realizar una clasificación binaria.

Durante la fase inicial se revisaron diferentes repositorios públicos, publicaciones científicas y plataformas especializadas, analizando datasets orientados tanto a reconocimiento emocional como a detección de trastornos mentales mediante señales de voz. En muchos casos aparecían problemas relacionados con el número de muestras disponibles, la ausencia de etiquetas claras o la dificultad para acceder directamente a las grabaciones originales. Algunos conjuntos de datos incluían muy pocos ejemplos por clase, mientras que otros presentaban formatos poco prácticos para realizar posteriormente la extracción de características acústicas.

Uno de los aspectos que más peso tuvo durante la selección fue la necesidad de trabajar con un conjunto de datos relativamente equilibrado. Se buscaba que cada clase tuviese al menos 100 muestras por clase para evitar que el entrenamiento quedara demasiado condicionado por desbalances severos entre categorías. Esto era especialmente importante teniendo en cuenta que posteriormente se pretendía entrenar modelos multiclase capaces de distinguir entre ausencia de depresión, depresión grado 1 y depresión grado 2.

Otro punto importante tenía que ver con el formato y la calidad de los audios es que algunas bases de datos revisadas contenían grabaciones excesivamente cortas, niveles

elevados de ruido o inconsistencias entre archivos que podían dificultar el análisis posterior. En este caso se priorizó trabajar con un conjunto de datos que permitiera extraer características de forma relativamente consistente y que no obligara a realizar un preprocesado excesivamente complejo antes de comenzar la fase experimental.

Finalmente, el dataset seleccionado fue "Multimodal Dataset for Depression Analysis", publicado en Kaggle por s3programmerlead en 2023. El conjunto de datos presentaba una estructura suficientemente organizada y contenía grabaciones etiquetadas según distintos estados depresivos, algo fundamental para el enfoque planteado en el proyecto.

5.2 ANÁLISIS INICIAL DEL CONJUNTO DE DATOS

Una vez descargado el dataset, una de las primeras cosas que se hizo fue revisar cómo estaban distribuidas las muestras y qué tipo de grabaciones contenía realmente el conjunto de datos. Aunque durante la fase de búsqueda ya se habían revisado varios aspectos generales, hasta no trabajar directamente con los audios resulta difícil hacerse una idea clara de la calidad real del dataset y de las limitaciones que puede presentar.

El conjunto de datos utilizado está formado por 800 grabaciones de voz relativamente cortas, normalmente entre 4 y 10 segundos. Esto terminó siendo bastante útil para el tipo de análisis planteado en el proyecto. Audios más largos habrían aumentado mucho el tiempo de procesamiento y probablemente habrían introducido información redundante dentro de las características acústicas extraídas pudiendo provocar lo conocido como *overfitting*.

En general, la calidad de las grabaciones era aceptable. Tras realizar una valoración cualitativa, se apreciaron algunas diferencias entre unas muestras y otras,

sobre todo relacionadas con volumen, pequeñas variaciones de ruido o la propia forma de hablar de cada persona, pero nada especialmente problemático como para impedir trabajar con ellas. De hecho, parte del interés de utilizar un conjunto de datos de este tipo estaba precisamente en no trabajar con señales completamente artificiales o excesivamente limpias, ya que en un escenario real siempre existe cierta variabilidad entre grabaciones.

Uno de los aspectos que más llamó la atención desde el principio fue la distribución de las clases. Como puede observarse en la Figura 3, la cantidad de muestras no se encuentra completamente equilibrada entre las distintas categorías utilizadas en el proyecto.

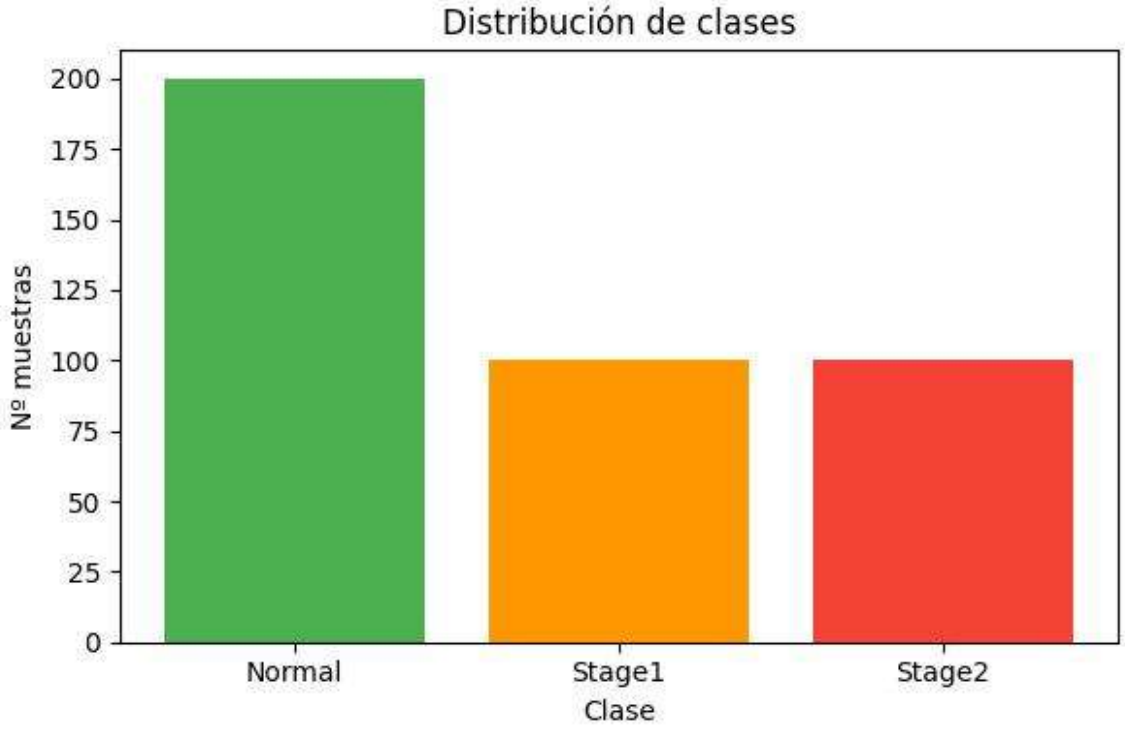


Figura 3 - Distribución de muestras por clase dentro del conjunto de datos utilizado.

La clase correspondiente a pacientes sin depresión cuenta con 400 muestras, mientras que las categorías asociadas a depresión grado 1 y grado 2 tienen 200 muestras cada una. Aunque el desbalanceo no es extremo, sí existe una diferencia clara hacia la

clase "Normal", que termina teniendo el doble de ejemplos que cada una de las otras dos categorías lo cual podía convertirse en un problema importante durante el entrenamiento.

En modelos de clasificación supervisada es habitual que las clases con mayor representación terminen teniendo más peso dentro del aprendizaje, simplemente porque el modelo ve más ejemplos de ese tipo durante el entrenamiento. En este caso existía el riesgo de que el clasificador aprendiera con más facilidad patrones relacionados con pacientes sin depresión y acabara mostrando cierta tendencia a favorecer esa categoría durante las predicciones.

El caso de la clase Stage1 era probablemente el más delicado. Muchas veces las diferencias acústicas entre pacientes sin depresión y pacientes con síntomas leves no son especialmente marcadas, por lo que parte de las muestras terminan compartiendo características relativamente parecidas. Al existir menos ejemplos de Stage1 dentro del dataset, aparecía la posibilidad de que algunas de estas muestras quedaran "absorbidas" por la clase "Normal" durante la clasificación, especialmente en aquellas zonas donde las características acústicas presentaban más solapamiento.

Aun así, después de revisar varios datasets distintos, este seguía siendo la mejor opción disponible para el tipo de trabajo que se quería realizar. El número total de muestras era razonable, las etiquetas estaban bien organizadas y las grabaciones permitían trabajar de forma cómoda en la extracción de características acústicas. Las primeras pruebas exploratorias daban resultados prometedores, por lo que finalmente se decidió continuar trabajando con este conjunto de datos pese al desbalanceo existente entre clases.

5.3 EXTRACCIÓN DE CARACTERÍSTICAS ACÚSTICAS

Una vez revisada la estructura general del dataset y comprobada la distribución de las distintas clases, el siguiente paso del proyecto consistió en trabajar directamente sobre las grabaciones de voz para transformar cada audio en información numérica utilizable dentro del análisis. Esta fase terminó siendo una de las más importantes de todo el

desarrollo, principalmente porque el funcionamiento posterior de los modelos dependía casi por completo de la calidad y del tipo de características acústicas seleccionadas para desarrollar una base de datos reproducible y en la que poder trabajar una vez depurada y preprocesada.

Al comienzo del trabajo se probaron distintas variables relacionadas con tono, energía, pausas y patrón espectral de la voz. No todas daban buenos resultados ni todas aportaban información útil de cara a la clasificación. Algunas presentaban distribuciones demasiado parecidas entre clases, mientras que otras generaban resultados bastante inestables dependiendo de la grabación analizada. Por eso, más que intentar extraer una gran cantidad de características, el objetivo terminó siendo encontrar un conjunto relativamente reducido pero que consiguiera representar distintos aspectos relevantes del habla sin introducir demasiada redundancia entre variables. Se escogió encontrar las mejores 7 características para este ámbito; el número 7 es una métrica perfecta ya que es el balance deseado entre coste computacional y efectividad a la hora de buscar el *accuracy* de un modelo.

Después de varias pruebas y comparaciones, las características que terminaron utilizándose de forma definitiva fueron las siguientes: *pitch mean, pitch standard deviation, energy mean, energy standard deviation, pause ratio, pause mean* y *MFCC*. Entre todas ellas permitían describir distintos comportamientos de la señal vocal relacionados con el tono, la intensidad, la fluidez del habla y la estructura espectral del audio.

Las primeras variables seleccionadas estuvieron relacionadas con el tono de la señal. El *pitch mean* representa la frecuencia fundamental media de la voz y, en términos prácticos, sirve para aproximar el tono habitual con el que habla una persona durante la grabación. Aunque esta característica por sí sola no genera una separación especialmente marcada entre clases, sí aporta información útil porque determinados estados depresivos suelen asociarse a una menor expresividad vocal y a patrones de habla más planos o

menos dinámicos. Además, el tono medio puede variar entre individuos, algo que hacía importante no analizar esta variable de forma aislada sino junto con el resto de las características extraídas.

Más relevante terminó siendo el *pitch standard deviation*, que mide cuánto varía el tono de la voz a lo largo del audio. Aquí ya no interesa tanto si una voz es más grave o aguda, sino el nivel de variabilidad tonal que presenta durante el habla. En las primeras pruebas realizadas se observó que esta característica sí mostraba diferencias bastante visibles entre clases, especialmente en comparación con otras variables más simples. Una voz excesivamente monótona o con muy poca modulación puede reflejar menor expresividad emocional, algo estudiado dentro del análisis de voz asociado a depresión.

La energía de la señal también terminó con peso dentro del conjunto final de características. Para representarla se utilizaron dos medidas distintas calculadas a partir del valor RMS del audio. La primera fue *energy mean*, que describe la intensidad media de la grabación y permite estimar aproximadamente con qué fuerza habla la persona, aunque el volumen puede verse afectado por factores externos, como el micrófono utilizado o la distancia de grabación, seguía siendo una variable relevante porque muchas muestras asociadas a estados depresivos presentaban niveles de energía más bajos o una forma de hablar menos intensa. Junto a ella se analizó también la *energy standard deviation*, que se usó para medir cómo cambia esa intensidad a lo largo de la grabación. Dos personas pueden tener valores medios relativamente parecidos y, aun así, comportamientos completamente distintos durante el habla. Algunas voces mantienen prácticamente el mismo nivel de intensidad durante todo el audio, mientras que otras presentan cambios constantes de volumen y mayor dinamismo, precisamente por eso tenía sentido mantener ambas variables.

También se decidió incorporar dos características relacionadas con las pausas y los silencios dentro del habla. Durante la revisión inicial de las grabaciones ya se había observado que algunos audios presentaban ritmos bastante distintos entre sí,

especialmente en la frecuencia y duración de las pausas. En algunos casos el habla resultaba mucho más continua, mientras que en otros aparecían silencios más frecuentes o fragmentos donde el discurso parecía más entrecortado.

La primera medida utilizada fue *pause ratio*, que representa la proporción total de silencio respecto a la duración completa del audio. Esta característica permitía cuantificar cuánto tiempo permanecía la señal en silencio dentro de cada grabación. Su utilidad aparecía sobre todo al comparar muestras donde el ritmo del habla era claramente diferente.

Sin embargo, únicamente con esa medida no era suficiente. Dos grabaciones podían tener proporciones de silencio parecidas y aun así comportarse de forma completamente distinta: una podía contener muchas pausas pequeñas repartidas durante el audio y otra presentar pocos silencios, pero más largos. Por eso se añadió también *pause mean*, que mide la duración media de las pausas detectadas y complementa relativamente bien la información proporcionada por pause ratio.

La última característica incorporada fue MFCC, probablemente una de las variables más importantes de todo el conjunto final. Los Mel-Frequency Cepstral Coefficients son una representación ampliamente utilizada en procesamiento de voz porque permiten resumir gran parte de la información espectral de una señal en un conjunto relativamente compacto de coeficientes. A diferencia de otras variables más simples, los MFCC no describen únicamente tono o intensidad, sino la forma general del espectro del audio y determinadas propiedades relacionadas con el timbre y la estructura acústica del habla.

Su cálculo se basa en la escala Mel (Figura 4), una transformación de frecuencias diseñada para aproximar cómo percibe el oído humano el sonido. Esto es importante porque la percepción humana de las frecuencias no es lineal. Somos mucho más sensibles a pequeñas diferencias en frecuencias bajas que en frecuencias altas, por lo que trabajar

directamente sobre el espectro original no siempre resulta la mejor opción cuando se analizan señales de voz.

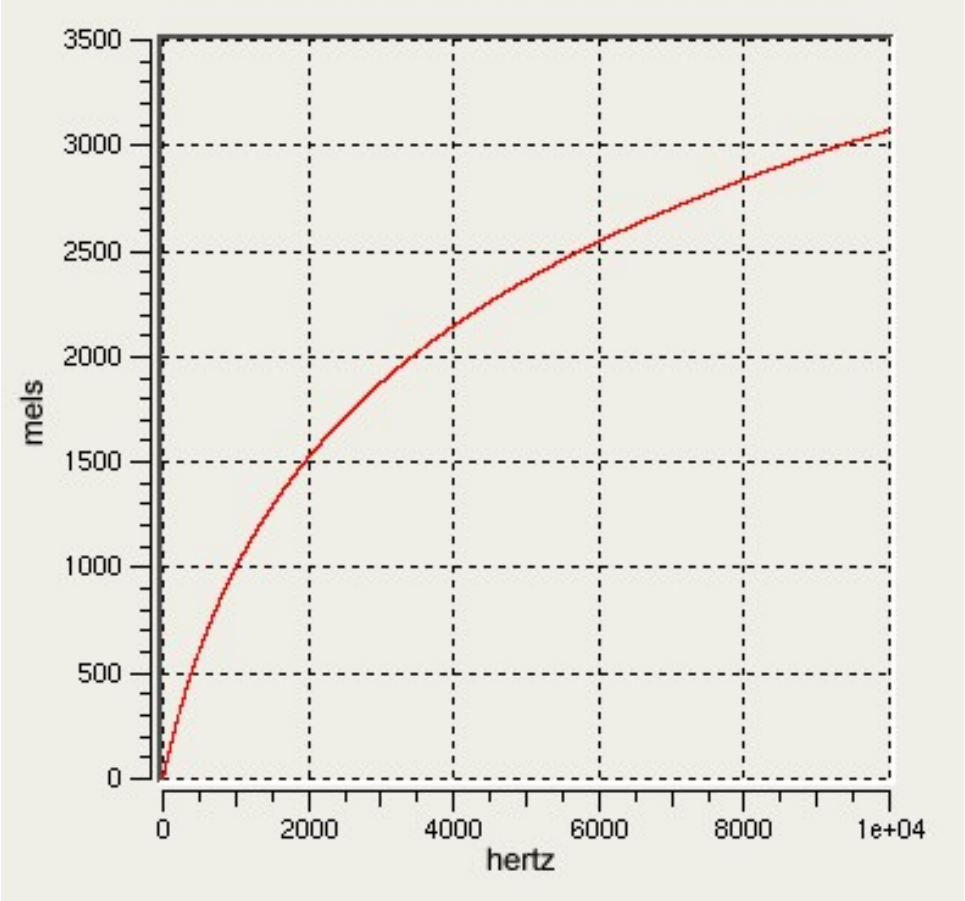


Figura 4 - Escala Mel

El cálculo de los MFCC se realiza en varias etapas. Primero, la señal de audio se divide en pequeñas ventanas temporales para poder analizar fragmentos muy cortos de la grabación de forma independiente. Sobre cada una de estas ventanas se obtiene el espectro de frecuencias utilizando la transformada de Fourier y, a continuación, se aplica un conjunto de filtros distribuidos según la escala Mel, diseñada para aproximar la forma en la que el oído humano percibe las frecuencias del sonido. Después se calcula la energía asociada a cada filtro y se aplica una transformación matemática que permite resumir toda

esa información espectral en un conjunto reducido de valores numéricos capaces de describir características relevantes del habla y del timbre de la voz.

La expresión utilizada habitualmente para calcular estos coeficientes es la siguiente:

$$MFCC_n = \sum_{k=1}^{K} \left(\log(E_k) \cdot \cos \frac{\pi n (k - 0.5)}{K} \right)$$

donde E_k representa la energía asociada al filtro Mel número k, K corresponde al número total de filtros utilizados y n indica el coeficiente calculado. La aplicación de la transformada discreta del coseno sobre las energías obtenidas en escala Mel permite resumir gran parte de la información espectral de la señal en un conjunto reducido de valores numéricos, manteniendo las características acústicas más relevantes del habla.

En este proyecto, los MFCC terminaron siendo especialmente útiles porque conseguían capturar patrones acústicos mucho más complejos que otras variables individuales. Mientras características como el tono o la energía describían propiedades concretas de la voz, los MFCC resumían información relacionada con el comportamiento espectral global del habla, algo que posteriormente fue de gran ayuda durante el entrenamiento de los modelos de clasificación.

Una vez definidas las siete características finales, se realizó su extracción sobre las 800 grabaciones del dataset. Cada audio quedó representado mediante un conjunto de valores numéricos asociados a las variables seleccionadas. De esta manera, las señales de voz originales pasaron a transformarse en una estructura tabular mucho más manejable para el análisis posterior con los modelos de clasificación, *clustering* y KDE.

Durante este proceso también fue necesario revisar manualmente distintos problemas relacionados con inconsistencias en los valores obtenidos. Algunas

grabaciones generaban resultados anómalos en variables relacionadas con pausas o energía, especialmente cuando aparecían silencios demasiado prolongados o diferencias importantes de intensidad dentro del audio. En lugar de eliminar automáticamente cualquier valor atípico, primero se analizaron los casos más extremos para comprobar si correspondían realmente a errores de extracción o simplemente a comportamientos válidos dentro de la variabilidad natural de las muestras.

Para analizar visualmente el comportamiento de las características extraídas se utilizaron diagramas de caja o *boxplots*. Este tipo de representación resulta especialmente útil cuando se trabaja con variables numéricas y varias clases diferentes, ya que permite comparar de forma rápida cómo se distribuyen los valores de cada característica dentro de cada grupo.

Los *boxplots* muestran distintos aspectos estadísticos de una variable. La línea central representa la mediana de los datos, mientras que la caja contiene el rango intercuartílico, es decir, la zona donde se concentra aproximadamente el 50% central de las muestras. Los extremos superiores e inferiores permiten observar la dispersión general de los datos y los puntos situados fuera de ese rango aparecen representados como posibles valores atípicos.

En este proyecto, este tipo de gráficas resultó especialmente útil porque permitía comprobar si determinadas características acústicas presentaban diferencias visibles entre las clases Normal, Stage1 y Stage2 antes incluso de entrenar los modelos de clasificación. Si las distribuciones de una variable aparecen prácticamente superpuestas entre clases, esa característica probablemente tendrá poca capacidad discriminativa por sí sola. Por el contrario, cuando las medianas se desplazan claramente o las distribuciones muestran comportamientos distintos, puede interpretarse que esa variable contiene información potencialmente útil para separar categorías.

Además de analizar la separación entre clases, los *boxplots* también permitían detectar rápidamente comportamientos anómalos dentro de algunas variables. En características relacionadas con pausas o energía aparecían ciertas muestras alejadas de la dinámica general del conjunto de datos, algo importante porque podía indicar desde diferencias naturales entre locutores hasta posibles errores durante la extracción de características.

En la siguiente ilustración pueden apreciarse diferencias bastante claras entre algunas clases, especialmente en variables relacionadas con la variabilidad tonal y con las pausas.

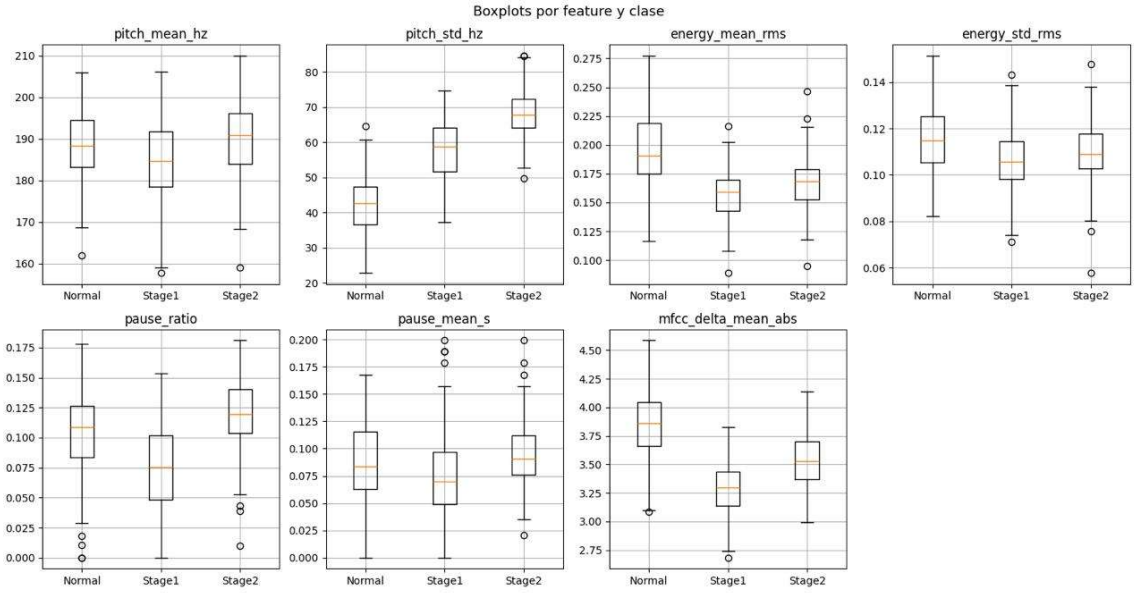


Figura 5 - Distribución de las características acústicas seleccionadas mediante diagramas de caja para las clases Normal, Stage1 y Stage2.

El caso de *pitch_std_hz* resulta significativo, ya que las medianas de las distintas categorías aparecen relativamente separadas y la distribución cambia de forma apreciable entre Normal, Stage1 y Stage2. Algo parecido ocurre con *pause_ratio*, donde las

grabaciones asociadas a Stage2 presentan, en general, una mayor proporción de silencio dentro del audio.

No todas las variables muestran una separación tan evidente. Algunas presentan solapamiento entre clases, algo completamente normal en un problema de este tipo. De hecho, una de las conclusiones más importantes de esta fase fue comprobar que ninguna característica individual era suficiente para separar completamente las categorías por sí sola. La información útil aparecía realmente cuando todas las variables se analizaban conjuntamente.

Tras estudiar la distribución individual de cada característica, también se analizó la relación existente entre ellas mediante una matriz de correlación.

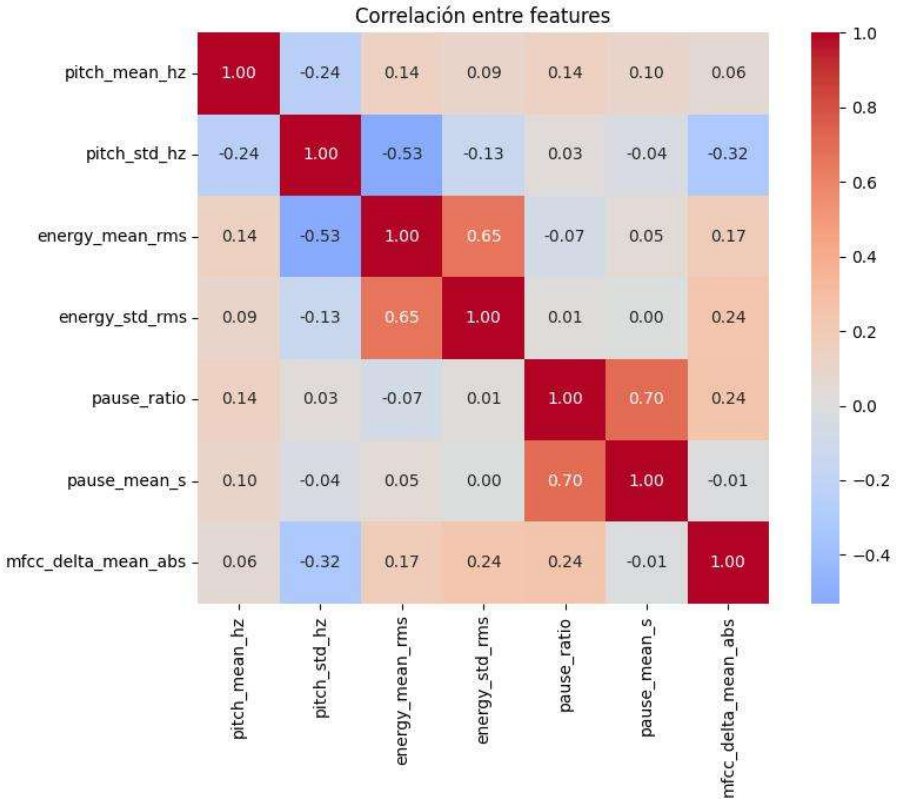


Figura 6 - Matriz de correlación entre las características acústicas seleccionadas.

Esta representación permitió comprobar qué variables estaban aportando información similar y cuáles describían comportamientos relativamente independientes dentro de la señal. Algunas relaciones eran esperables. Por ejemplo, pause_ratio y *pause_mean* muestran una correlación positiva relativamente alta, ya que ambas están relacionadas con el comportamiento temporal de las pausas. Aun así, no representan exactamente lo mismo y por eso se decidió mantener ambas dentro del conjunto final.

También puede observarse cierta correlación entre *energy_mean* y *energy_std* ya que grabaciones con mayor intensidad media tienden en algunos casos a presentar también mayor variabilidad energética. Sin embargo, el nivel de correlación no era lo suficientemente alto como para considerar ambas variables completamente redundantes.

En general, la matriz mostraba una tendencia bastante positiva para el tipo de análisis que se quería realizar. Aunque algunas variables presentaban relaciones moderadas entre sí, el conjunto final seguía manteniendo suficiente diversidad como para describir distintos aspectos de la voz desde perspectivas complementarias: tono, energía, pausas y características espectrales, que era precisamente lo que se buscaba al seleccionar las variables finales del proyecto.

5.4 ANÁLISIS DE CARACTERÍSTICAS MEDIANTE KDE

Una vez extraídas y revisadas las características escogidas, se realizó un análisis adicional utilizando la técnica de *Kernel Density Estimation (KDE)* con el objetivo de estudiar de forma más detallada cómo se distribuían las distintas clases dentro de cada variable. Este análisis permitió obtener una representación mucho más suave y continua del comportamiento de los datos, facilitando la comparación visual entre las categorías Normal, Stage1 y Stage2.

Antes de realizar esta parte del trabajo fue necesario revisar distintos ejemplos y referencias sobre cómo interpretar correctamente una gráfica KDE y qué debía

considerarse un resultado útil dentro del contexto del proyecto. Esto era importante porque una gráfica KDE puede resultar visualmente "bonita" y aun así no aportar información relevante para clasificación. En general, lo que se buscaba era encontrar distribuciones cuyos picos aparecieran razonablemente separados entre clases y donde el solapamiento entre curvas no fuese excesivo. Cuanto mayor es la separación entre distribuciones, mayor suele ser la capacidad de una característica para diferenciar categorías dentro del espacio de datos. Por el contrario, un KDE donde todas las curvas aparecen completamente superpuestas suele indicar que la variable apenas aporta capacidad discriminativa, ya que las distintas clases presentan comportamientos muy similares dentro de esa característica.

En un problema real de clasificación biomédica resulta muy difícil obtener separaciones perfectas entre grupos, especialmente trabajando con señales de voz y estados clínicos relativamente próximos entre sí. Aun así, este análisis permitía hacerse una idea clara de qué variables parecían más prometedoras antes incluso de entrenar los modelos definitivos.

El KDE resulta especialmente útil porque a diferencia de histogramas más tradicionales no depende tanto de divisiones discretas o intervalos fijos. En lugar de representar simplemente cuántas muestras caen dentro de determinados rangos, genera una estimación continua de la densidad probabilística de los datos. Esto hace que las distribuciones sean mucho más fáciles de comparar visualmente y permite apreciar mejor desplazamientos, solapamientos o diferencias de funcionamiento entre clases.

En la siguiente serie de ilustraciones se muestran las distribuciones KDE obtenidas para las distintas características acústicas seleccionadas durante el proyecto.

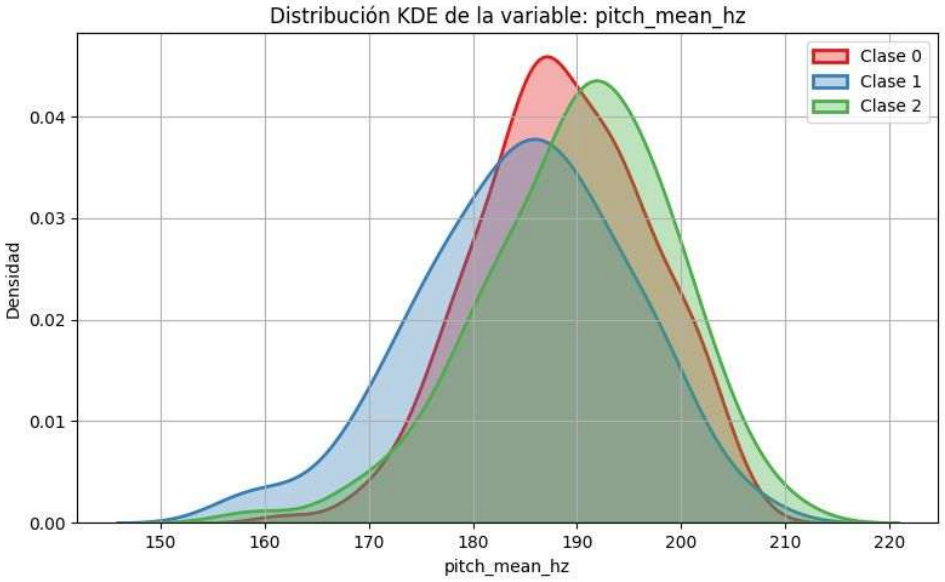


Figura 7 - KDE de Pitch_Mean_Hz

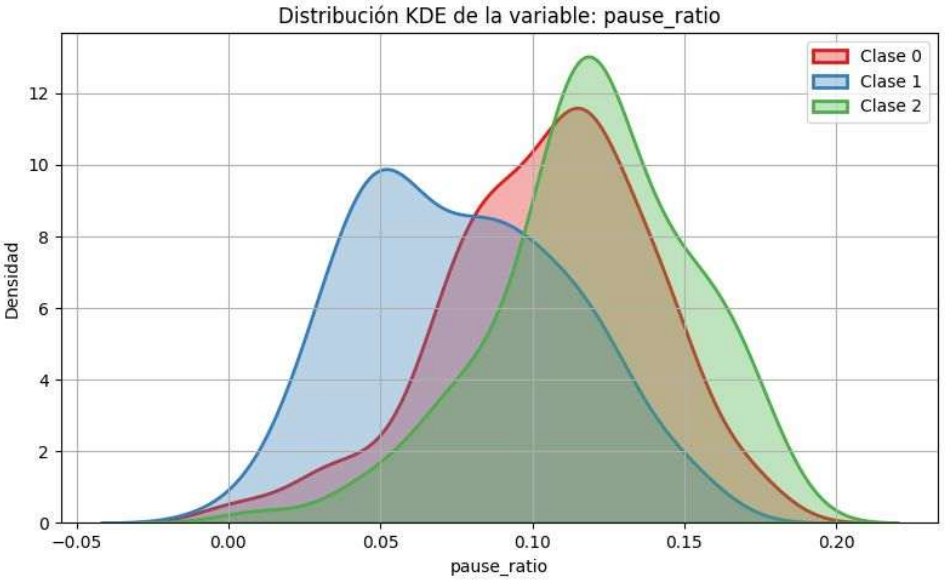


Figura 8 - KDE de Pause_Ratio

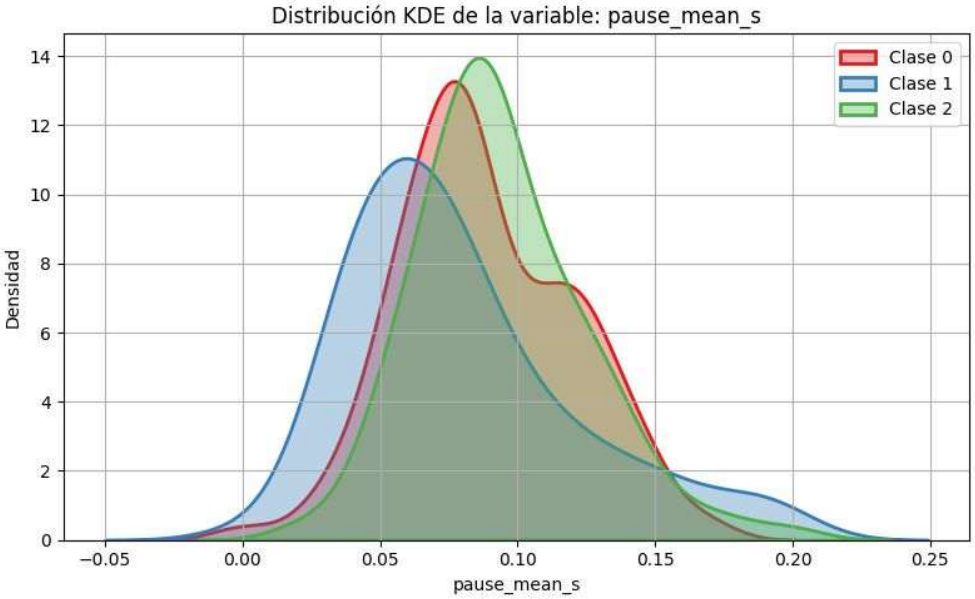


Figura 9 - KDE de Pause_Mean_S

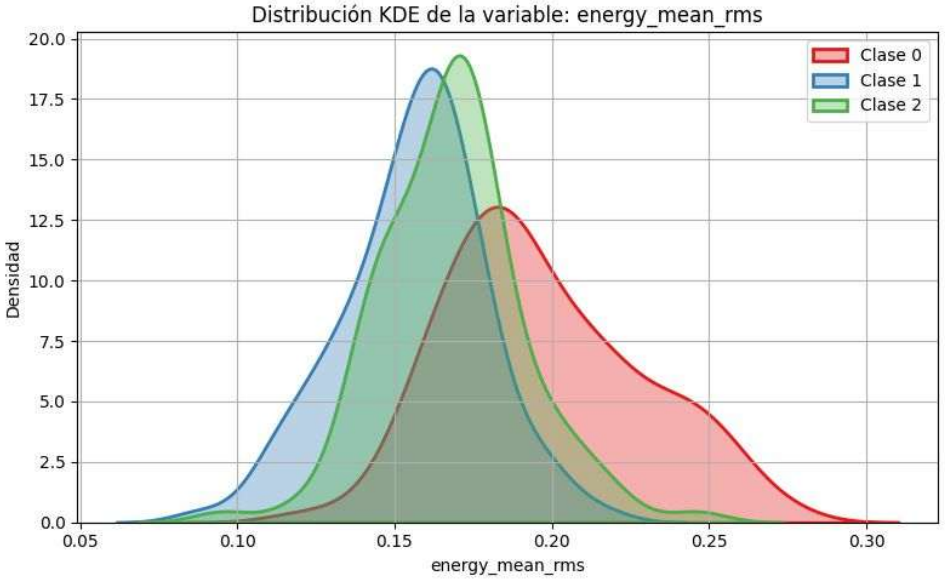


Figura 10 - KDE de Energy_Mean_RMS

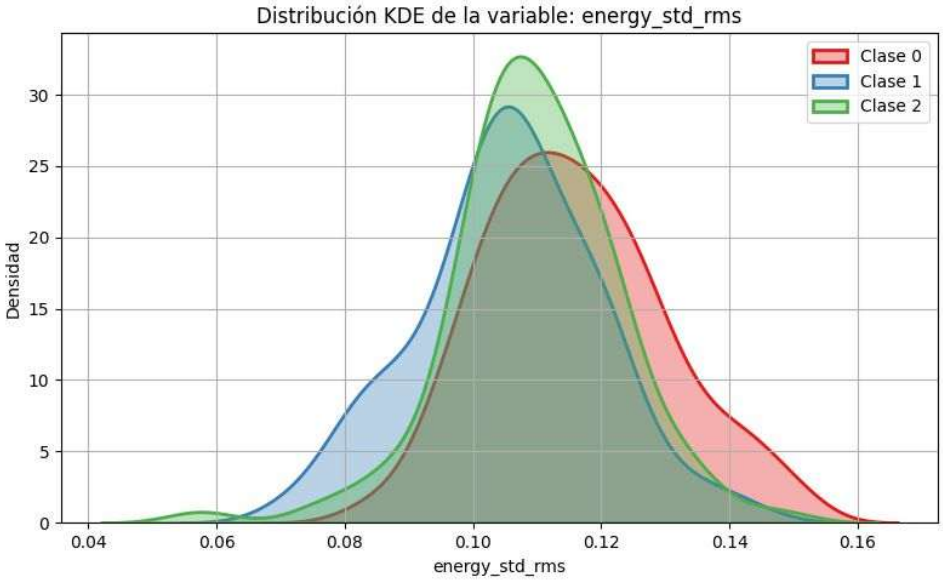


Figura 11 - KDE de Energy_Std_RMS

En general, los resultados obtenidos durante este análisis fueron bastante positivos, aunque es cierto que ninguna de estas 5 variables consigue separar completamente las tres clases, algo que ya se intuía durante las fases anteriores, sin embargo, sí que empiezan a observarse desplazamientos relativamente claros entre algunas distribuciones.

En varias de las gráficas puede apreciarse cómo las curvas correspondientes a cada clase presentan máximos en zonas diferentes, aunque exista cierto solapamiento entre ellas. Este comportamiento era importante porque indicaba que las variables seleccionadas sí contenían información relacionada con el estado depresivo de los pacientes. También, el hecho de que las distribuciones no fueran completamente caóticas ni totalmente superpuestas reforzaba la idea de que el conjunto de características escogido tenía sentido desde el punto de vista acústico.

Algunas variables muestran resultados más discretos que otras. Por ejemplo, en *pitch_mean_hz* las distribuciones aparecen relativamente próximas y existe solapamiento entre categorías. Esto era esperable, ya que el tono medio de la voz puede variar mucho

entre personas por motivos completamente ajenos al estado depresivo, como sexo, edad o características fisiológicas individuales. Aun así, aunque esta variable por sí sola no pareciera especialmente potente, seguía aportando cierta información útil al combinarse con el resto de las características, así que decidimos seguir con ella en vez de eliminarla por la posibilidad de ahorrar coste computacional.

En variables relacionadas con pausas y energía también empiezan a verse diferencias interesantes. Las curvas correspondientes a Grado 2 tienden en algunos casos a desplazarse ligeramente hacia regiones asociadas a una mayor presencia de silencios o determinados patrones energéticos, algo coherente con la hipótesis inicial del proyecto. Evidentemente, sigue existiendo bastante variabilidad entre muestras individuales, pero el comportamiento general de las distribuciones resultaba suficientemente consistente como para considerar útiles estas características.

De todas las variables analizadas, una de las que mostró un desempeño más sorprendente fue.

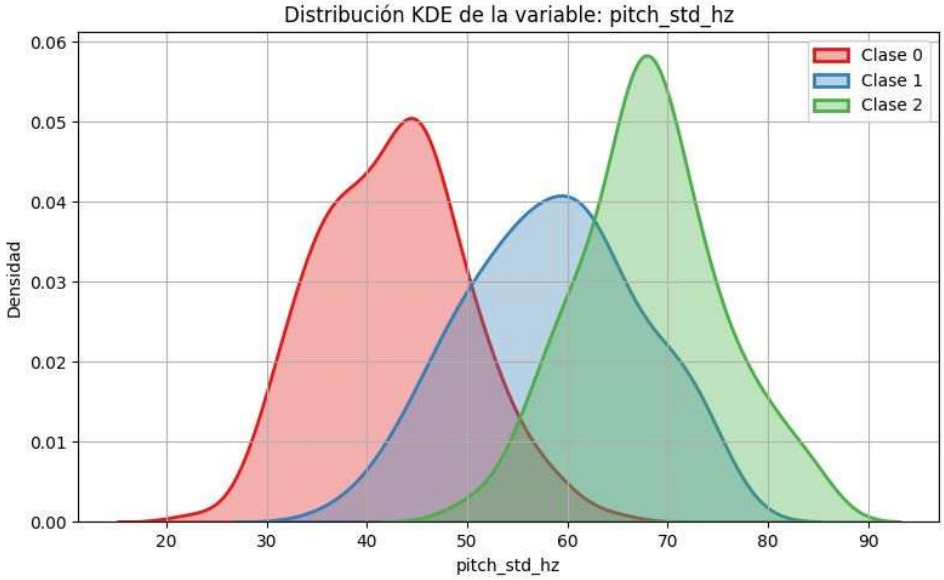


Figura 12 - KDE de Pitch_Std_Hz

En esta ilustración la separación entre clases aparece mucho más clara que en otras variables. Los máximos de densidad de cada categoría se encuentran desplazados y el solapamiento entre curvas es considerablemente menor, especialmente entre las clases Normal y Stage2. Este resultado fue importante dentro del desarrollo del proyecto porque indicaba que la variabilidad tonal de la voz podía estar relacionada de forma relativamente consistente con el estado depresivo de los pacientes.

Además, las curvas presentan formas relativamente estables y bien definidas, sin comportamientos excesivamente erráticos ni distribuciones completamente mezcladas. Eso hacía pensar que esta característica estaba capturando información útil para el proceso de clasificación. De hecho, durante las primeras pruebas experimentales ya se observó que *pitch_std_hz* aparecía frecuentemente entre las variables con mayor capacidad discriminativa.

Otra de las gráficas que más llamó la atención durante el análisis fue la correspondiente a *mfcc_delta_mean_abs*.

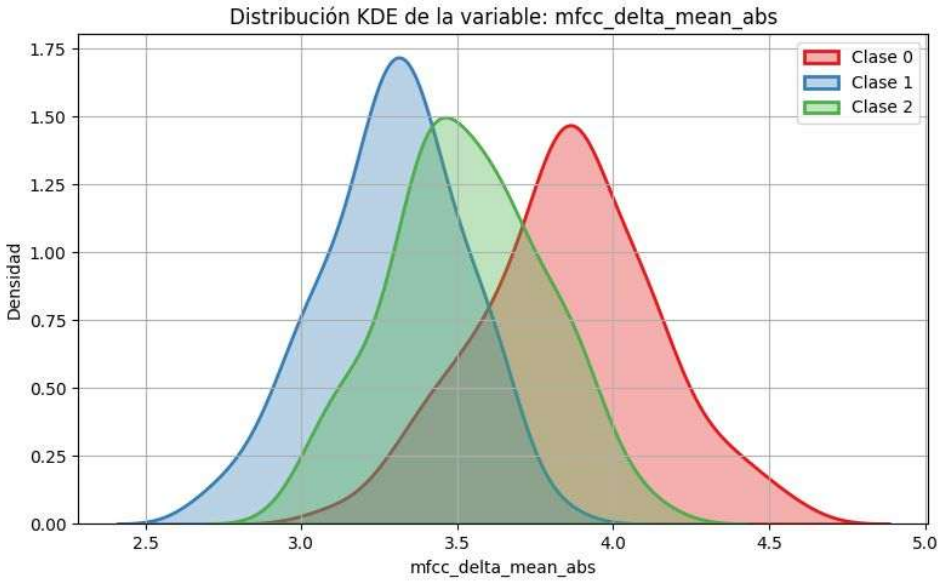


Figura 13 - KDE de MFCC_delta_mean_abs

En esta gráfica también pueden apreciarse ciertas diferencias entre clases, aunque el comportamiento es distinto al observado anteriormente en *pitch_std_hz*. Aquí las distribuciones siguen teniendo bastante solapamiento, pero las curvas ya no coinciden completamente y empiezan a concentrarse en rangos diferentes de valores.

Llamaba la atención cómo la clase Stage1 quedaba más separada de la clase Normal de lo que ocurría en otras variables analizadas anteriormente. Esto resultaba en especial interesante porque precisamente una de las mayores dificultades del proyecto era distinguir correctamente los casos intermedios o más leves de depresión. En muchas características anteriores las muestras de Stage1 tendían a mezclarse con la clase Normal, mientras que aquí empezaba a apreciarse una diferencia algo más clara, lo cual ayudaría y sería muy enriquecedor para la realización del proyecto tener una característica que separa tan claramente lo que las demás tienen problemas en separar.

También daba cierta confianza ver que las distribuciones mantenían formas relativamente coherentes y no aparecían excesivamente deformadas o dispersas. Aunque la separación seguía estando lejos de ser perfecta, este tipo de resultados ayudaba a confirmar que las variables relacionadas con MFCC estaban aportando información útil dentro del conjunto final de características y que tenía sentido seguir trabajando con ellas en las siguientes fases del proyecto.

5.5 ANÁLISIS DE IMPORTANCIA DE LAS CARACTERÍSTICAS

Después de analizar el funcionamiento individual de las variables mediante *boxplots* y distribuciones KDE, se realizó un estudio adicional orientado a medir la influencia real de cada característica sobre el comportamiento del modelo de clasificación. El objetivo de esta etapa no era únicamente comprobar qué variables parecían más útiles visualmente, sino entender cuáles estaban teniendo realmente un mayor impacto durante las predicciones realizadas por el clasificador.

Para ello se utilizaron técnicas de interpretabilidad basadas en SHAP (*Shapley Additive explanations*), una metodología ampliamente utilizada en aprendizaje automático para analizar cómo influye cada característica sobre las decisiones tomadas por un modelo. Este tipo de análisis resulta prometedor en problemas como el planteado en este proyecto, donde no solo importa obtener buenas métricas de clasificación, sino también entender qué patrones acústicos están siendo utilizados realmente durante el proceso de decisión.

SHAP permite analizar hasta qué punto cada característica está ayudando al modelo a tomar una decisión durante la clasificación. En lugar de limitarse únicamente a decir qué variables son "importantes" de forma general, este método permite observar cómo afecta cada característica a las predicciones realizadas sobre cada muestra concreta del dataset. De forma simplificada, SHAP estudia cuánto cambia el resultado del modelo cuando una determinada variable participa en la predicción. Si al utilizar una característica la decisión del clasificador cambia mucho, significa que esa variable está teniendo mucha influencia sobre el resultado final. Por el contrario, si una característica apenas modifica la predicción, su importancia dentro del modelo será menor. Gracias a esto, es posible identificar qué variables están siendo realmente más útiles durante el proceso de clasificación y comprobar si el modelo está tomando decisiones coherentes con la evolución observada anteriormente en las gráficas y análisis acústicos realizados.

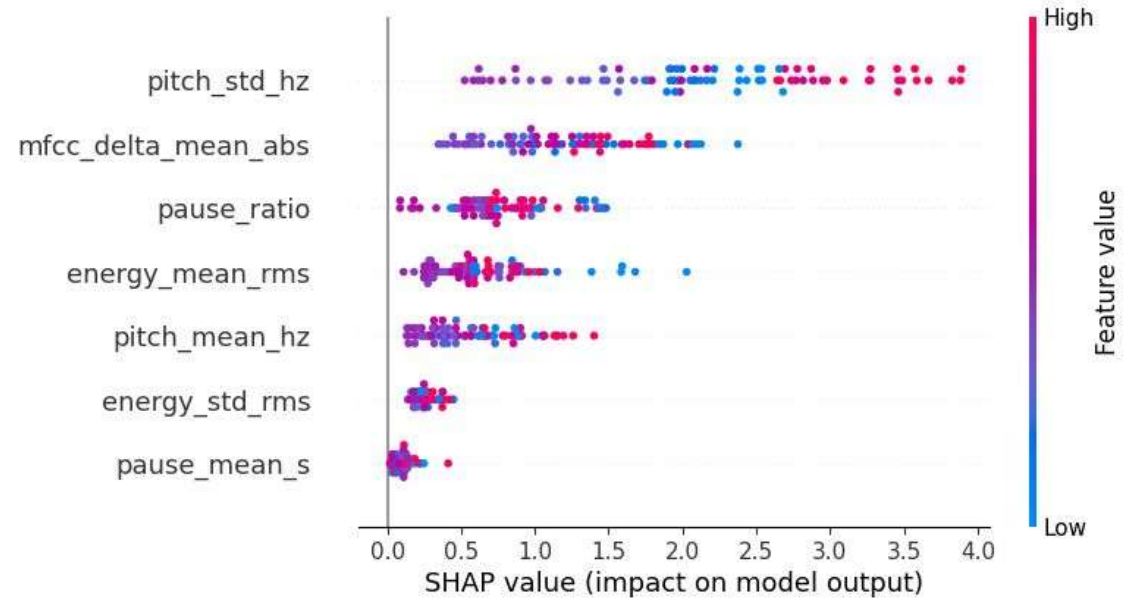


Figura 14 - Análisis SHAP de importancia e impacto de las características acústicas sobre un modelo de clasificación genérico.

La gráfica organiza las características según la influencia que están teniendo sobre las predicciones del modelo. Las variables situadas en la parte superior son aquellas que más modifican el comportamiento del clasificador cuando cambian sus valores, mientras que las inferiores tienen un efecto menor. Cada punto representa una muestra concreta del dataset y su posición horizontal indica hasta qué punto esa característica está empujando la predicción hacia una clase u otra. El color permite distinguir si el valor de la variable es alto o bajo dentro de esa muestra concreta, algo útil para entender no solo qué variables son importantes, sino también cómo se están comportando realmente dentro del modelo.

Desde el principio llama la atención el desempeño de *pitch_std_hz*. La dispersión horizontal de los puntos es mucho mayor que en el resto de las variables y ocupa prácticamente todo el rango visible de la representación, lo que indicaba que esta característica estaba influyendo mucho más que otras sobre las decisiones del clasificador. No solo aparecía como la variable más importante a nivel global, sino que su impacto era notablemente consistente entre muchas muestras distintas del dataset.

También resulta interesante observar cómo los valores altos de *pitch_std_hz* aparecen concentrados principalmente en zonas donde el impacto sobre el modelo es mayor. Esto encajaba bien con lo que ya se había visto anteriormente durante el análisis KDE, donde precisamente esta característica mostraba una separación clara entre clases. Al final, lo que empezaba a verse aquí era que el modelo realmente estaba utilizando esa información y no simplemente ignorándola durante el entrenamiento.

En los coeficientes MFCC el funcionamiento es distinto. La separación no es tan fuerte como ocurría con *pitch_std_hz* y sigue existiendo mezcla entre clases, aunque aun así la variable mantiene una dispersión considerable y aparece claramente situada entre las características con más peso dentro del modelo. Esto era importante porque reforzaba la idea de que las variables derivadas de MFCC estaban capturando información útil relacionada con la estructura espectral de la voz y no únicamente pequeñas fluctuaciones aleatorias dentro de las grabaciones. De hecho, una de las cosas más importantes de esta representación era precisamente comprobar que el modelo no parecía depender de una sola variable dominante. Aunque *pitch_std_hz* destacaba claramente sobre el resto, otras características como *pause_ratio, mfcc_delta_mean_abs* o *energy_mean_rms* seguían mostrando impactos relativamente amplios y alejados de cero en muchas muestras.

Empezaban a verse diferencias claras entre variables más “estables” y otras mucho más dependientes de cada muestra individual. Algunas características presentan distribuciones muy compactas alrededor de valores pequeños, mientras que otras muestran puntos mucho más dispersos horizontalmente. En la práctica, esto sugería que ciertas variables afectan de forma muy distinta dependiendo del paciente concreto analizado. Y, honestamente, tenía bastante sentido que ocurriera algo así en un problema relacionado con señales biomédicas y comportamiento humano, donde no todas las personas manifiestan los mismos patrones vocales de la misma manera.

Las variables relacionadas con pausas y energía, aunque menos dominantes que *pitch_std_hz*, seguían aportando información útil ya que sus distribuciones no aparecían completamente comprimidas alrededor de cero y mantenían cierto nivel de impacto sobre las predicciones lo cual resultaba importante porque ayudaba a confirmar que el conjunto de características seleccionado seguía teniendo mucho equilibrio entre distintos tipos de información acústica: tono, variabilidad, pausas y comportamiento espectral.

Durante esta fase también se planteó la posibilidad de eliminar algunas variables menos relevantes para simplificar ligeramente el modelo, sin embargo, después de revisar los resultados obtenidos se decidió mantener las siete características originales. El número total de variables seguía siendo relativamente pequeño y el coste computacional del entrenamiento no representaba realmente un problema importante dentro del proyecto. Además, aunque algunas características tenían menos peso individual que otras, ninguna parecía completamente irrelevante ni empeoraba el comportamiento del clasificador, por lo que tenía más sentido conservar toda la información disponible y continuar trabajando con el conjunto completo de variables.

5.6 MARCO TEÓRICO DE LOS MODELOS DE CLASIFICACIÓN

Una vez completadas las fases relacionadas con extracción y análisis de características acústicas, la siguiente etapa del proyecto se centrará en estudiar distintos modelos de clasificación supervisada orientados a diferenciar entre las tres clases planteadas dentro del dataset: pacientes sin depresión, depresión grado 1 y depresión grado 2.

Dentro de problemas de clasificación supervisada el rendimiento no suele evaluarse utilizando una única métrica aislada. Existen medidas relacionadas con precisión, sensibilidad, *recall* o *F1-score* que permiten analizar distintos aspectos del clasificador dependiendo del tipo de problema planteado. En este proyecto la métrica principal utilizada será *accuracy*, ya que permite observar de forma directa qué porcentaje

total de muestras consigue clasificarse correctamente sobre el conjunto completo de datos evaluados.

Desde las primeras fases del trabajo se tomó como referencia considerar competitivo cualquier clasificador capaz de superar aproximadamente el 85% de *accuracy*. Ese valor no surgía de forma arbitraria, sino teniendo en cuenta la dificultad asociada al propio problema. Las diferencias acústicas entre estados depresivos no aparecen completamente separadas y muchas grabaciones comparten regiones similares dentro del espacio de variables, especialmente alrededor de estados intermedios.

En problemas biomédicos relacionados con voz suele existir una variabilidad elevada incluso entre pacientes pertenecientes a una misma clase. Factores relacionados con edad, ritmo del habla, entonación, condiciones de grabación o diferencias naturales entre locutores introducen cambios importantes dentro de las señales acústicas. Precisamente por eso alcanzar accuracies elevadas en este tipo de escenarios no resulta tan sencillo como podría parecer inicialmente.

Superar ese umbral aproximado del 85% permitía establecer una referencia razonable para comparar distintos enfoques y analizar qué clasificadores conseguían adaptarse mejor al problema planteado. El objetivo no consistía únicamente en obtener un valor alto dentro de una métrica concreta, sino estudiar cómo reaccionaba cada algoritmo frente al solapamiento existente entre clases y comprobar qué modelos conseguían mantener una separación más estable entre categorías.

En lugar de trabajar directamente sobre un único clasificador elegido únicamente desde un punto de vista teórico, se decidió estudiar distintos enfoques de aprendizaje supervisado para comparar cómo respondía cada uno utilizando exactamente el mismo conjunto de características acústicas. La intención no era incorporar varios modelos simultáneamente dentro del sistema final, sino analizar qué alternativa se ajustaba mejor

al comportamiento real del dataset antes de seleccionar una solución definitiva para fases posteriores del proyecto.

Cada algoritmo aborda el problema de clasificación de una forma distinta. Algunos clasificadores funcionan mejor cuando las fronteras entre clases quedan relativamente claras, mientras que otros consiguen adaptarse mejor a relaciones no lineales o regiones donde distintas muestras comparten características similares. Teniendo en cuenta el tipo de señales utilizadas durante el proyecto, comparar enfoques distintos permitía entender mejor qué tipo de clasificador podía aprovechar con mayor eficacia la información acústica extraída de las grabaciones.

Uno de los primeros algoritmos estudiados fue *K-Nearest Neighbors* (KNN), un método basado en proximidad entre muestras dentro del espacio de características. El funcionamiento del algoritmo consiste en observar cuáles son las muestras más cercanas alrededor de un nuevo punto y asignar la clase predominante entre esos vecinos próximos. La decisión final depende directamente de cómo quedan distribuidas las muestras dentro del espacio de variables y del número de vecinos utilizados durante clasificación.

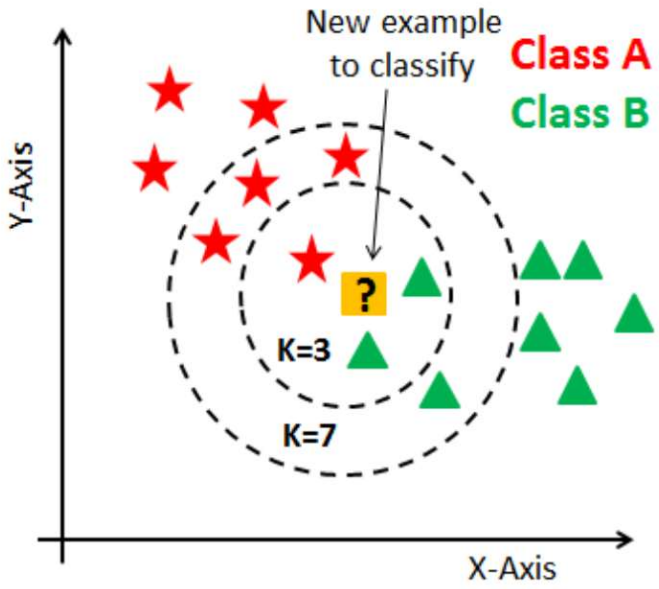


Figura 15 - Ejemplo simplificado del funcionamiento del algoritmo K-Nearest Neighbors (KNN).

Cuando se utiliza un valor pequeño de vecinos el clasificador puede reaccionar de forma muy sensible frente a pequeñas variaciones entre muestras. Valores más altos generan fronteras más suaves y menos dependientes de ejemplos concretos. Esa simplicidad hacía que KNN fuese interesante dentro de la comparativa inicial, especialmente para comprobar hasta qué punto las características acústicas generaban agrupaciones relativamente claras entre pacientes.

Los modelos basados en árboles de decisión fueron ganando importancia durante el desarrollo experimental debido a su capacidad para adaptarse bien a relaciones complejas entre variables. Uno de los primeros enfoques estudiados dentro de esta familia fue *Random Forest*, un algoritmo que construye múltiples árboles de decisión utilizando subconjuntos aleatorios de datos y variables para posteriormente combinar todas las predicciones dentro de una decisión final más estable.

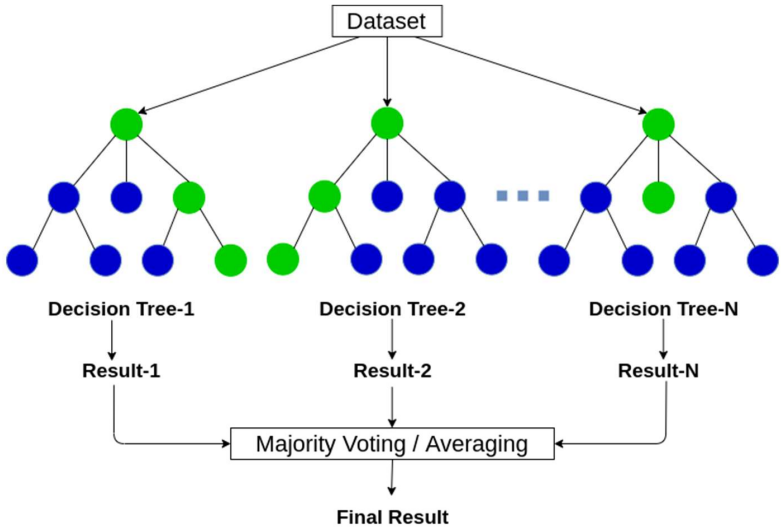


Figura 16 - Funcionamiento básico del modelo Random Forest.

Este tipo de enfoques suele adaptarse bien a escenarios donde existen relaciones no lineales y cierta variabilidad dentro de las características utilizadas durante

entrenamiento. Al trabajar mediante múltiples árboles el clasificador reduce dependencia respecto a una única estructura de decisión y consigue generar predicciones más robustas frente a ruido o pequeñas variaciones presentes dentro del dataset.

A partir de esta misma familia aparecieron posteriormente los modelos *boosting*. A diferencia de *Random Forest*, donde los árboles funcionan de forma más independiente, los algoritmos *boosting* construyen el aprendizaje secuencialmente intentando corregir en cada iteración los errores generados anteriormente.

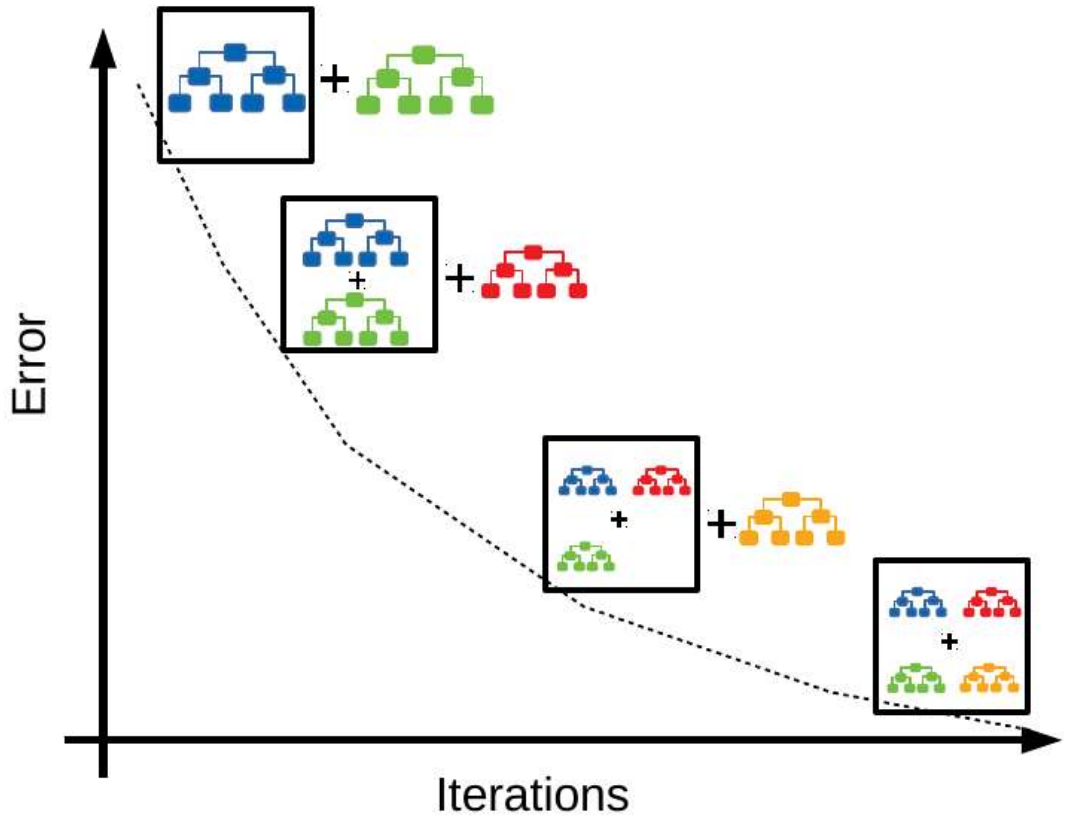


Figura 17 - Proceso iterativo de un modelo boosting.

El primero de los algoritmos estudiados dentro de esta categoría fue *Gradient Boosting*. Este enfoque genera nuevos árboles progresivamente y cada uno intenta mejorar las predicciones realizadas por el conjunto anterior. La estructura secuencial del

aprendizaje hace que estos modelos puedan adaptarse bien a relaciones complejas entre variables y capturar fronteras difíciles de representar mediante enfoques más simples.

En problemas relacionados con señales acústicas esto resulta especialmente útil porque gran parte de la información relevante no suele depender de una única variable aislada, sino de combinaciones más complejas entre distintas regiones del espacio de características. Los modelos *boosting* suelen adaptarse bien precisamente a este tipo de situaciones donde las fronteras entre clases no aparecen completamente definidas.

A partir de *Gradient Boosting* se decidió estudiar *Extreme Gradient Boosting* (XGBoost), una versión optimizada del *boosting* tradicional orientada a mejorar eficiencia computacional y control del sobreajuste. XGBoost incorpora distintos mecanismos internos de regularización y suele ofrecer buenos resultados en problemas donde existen relaciones complejas entre variables y regiones parcialmente solapadas dentro del dataset.

Finalmente se incluyó *Categorical Boosting* (CatBoost), otro algoritmo basado en boosting diseñado para mejorar algunas limitaciones presentes en modelos *boosting* más tradicionales. Aunque comparte gran parte de la filosofía general de Gradient Boosting y XGBoost, introduce distintos mecanismos internos orientados a estabilizar el entrenamiento y reducir problemas relacionados con sobreajuste.

Este tipo de enfoques resultaba especialmente interesante dentro del proyecto porque el dataset utilizado no era extremadamente grande y existía un solapamiento visible entre distintas clases, especialmente alrededor de estados depresivos intermedios. Situaciones así suelen requerir clasificadores capaces de capturar diferencias acústicas relativamente sutiles sin terminar ajustándose de forma excesiva al ruido presente dentro de los datos.

Otro aspecto importante relacionado con CatBoost tiene que ver con su capacidad para trabajar correctamente incluso cuando las relaciones entre variables son complejas o poco lineales. Eso encajaba bien con las características acústicas utilizadas durante el proyecto y con la propia naturaleza del problema planteado.

Modelo	***Tipo de dataset recomendado***	***Relaciones no lineales***	***Riesgo de overfitting***	***Coste computacional***
KNN	*Datasets pequeños y con clases relativamente separadas*	*Limitado*	*Medio*	*Bajo*
Random Forest	*Datasets con ruido y variables heterogéneas*	*Bueno*	*Bajo*	*Medio*
Gradient Boosting	*Datasets con patrones complejos*	*Muy bueno*	*Medio-Alto*	*Medio*
XGBoost	*Datasets complejos y con gran cantidad de variables*	*Muy bueno*	*Bajo-Medio*	*Alto*
CatBoost	*Datasets complejos con relaciones no lineales y cierto solapamiento entre clases*	*Muy bueno*	*Bajo*	*Medio-Alto*

Tabla 2 - Comparativa general de los modelos de clasificación explorados durante el proyecto.

5.7 DESARROLLO DE MODELOS DE CLASIFICACIÓN

Para analizar la respuesta real de cada clasificador se entrenaron los distintos modelos. A partir de aquí empezaban a verse diferencias claras entre algoritmos, no solo en accuracy, sino también en la forma en la que cada uno confundía las distintas clases del problema.

El primer modelo probado fue KNN. Su rendimiento resultaba clave porque permitía comprobar hasta qué punto las muestras quedaban separadas dentro del espacio de características utilizando un enfoque relativamente simple basado únicamente en proximidad entre vecinos.

```
Test Accuracy: 0.8
Test ROC AUC: 0.8741319444444443

Classification report:
              precision    recall  f1-score   support

           0       0.86      0.93      0.89        40
           1       0.67      0.40      0.50        20
           2       0.76      0.95      0.84        20

    accuracy                           0.80        80
   macro avg       0.76      0.76      0.75        80
weighted avg       0.79      0.80      0.78        80
```

Figura 18 - Métricas obtenidas con KNN.

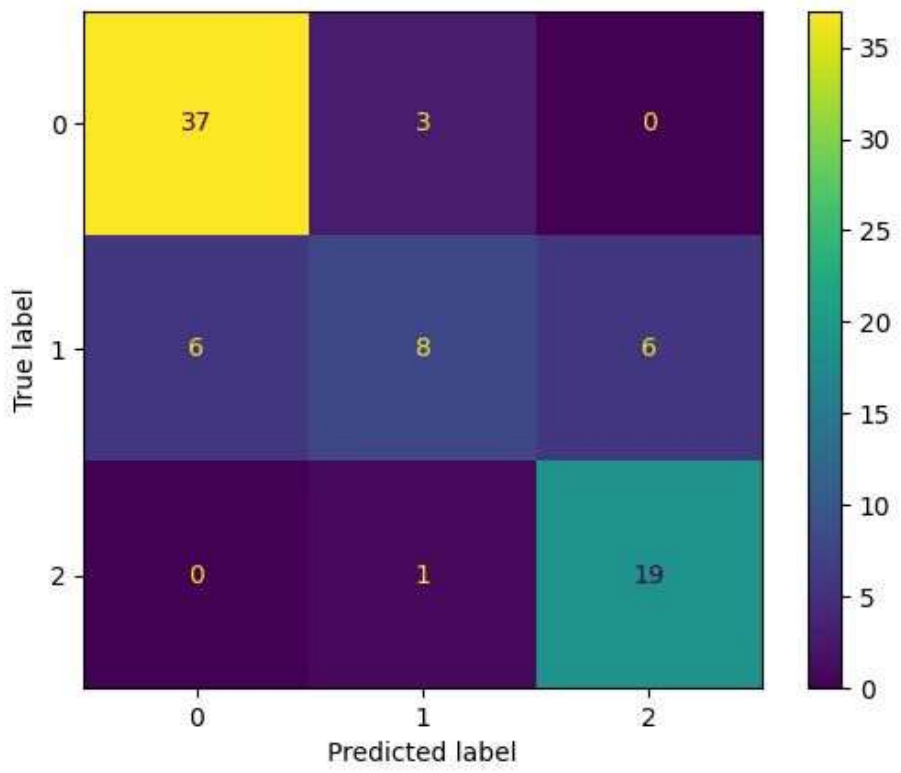


Ilustración 19 - Matriz de confusión de KNN.

El resultado de KNN fue el más limitado de la comparativa. La *accuracy* se quedó en 0.80, por debajo del umbral del 85% que se había establecido como referencia mínima para considerar un modelo suficientemente competitivo dentro del proyecto. La clase Normal y la clase Stage2 se comportaron bien, con 37 aciertos sobre 40 en la clase 0 y 19 sobre 20 en la clase 2.

El problema se encuentra a la hora de clasificar Stage 1, donde la matriz de confusión muestra que de las 20 muestras reales de Stage1 solo se clasificaron correctamente 8. El resto se repartió casi por igual entre Normal y Stage2, con 6 muestras confundidas con cada una de esas clases. Esto no se trata de un fallo pequeño sino de una incapacidad clara para colocar bien la clase intermedia. KNN depende mucho de la cercanía entre muestras y, cuando las clases se mezclan dentro del espacio de

características, termina tomando decisiones muy sensibles a los vecinos más próximos. En este caso se ve bastante bien que Stage1 queda atrapada entre las otras dos categorías, algo que ya se intuía al analizar el solapamiento entre variables.

El *recall* de Stage1, es decir, la capacidad del modelo para detectar correctamente las muestras reales pertenecientes a esa clase cae hasta 0.40, un valor demasiado bajo para un clasificador que tendría que distinguir entre distintos grados de depresión. Aunque la accuracy global pueda parecer aceptable a primera vista, el desglose por clase deja claro que el modelo no estaba funcionando de forma equilibrada.

Random Forest mejoró claramente el comportamiento observado con KNN. La accuracy alcanzó 0.8875, superando el umbral fijado del 85%, y el ROC AUC obtuvo un valor de 0.97. La matriz también se ve más ordenada, con una diagonal principal mucho más marcada y errores menos dispersos.

```
Test Accuracy: 0.8875
Test ROC AUC: 0.9700000000000001

Classification report:
              precision    recall  f1-score   support

           0       0.95      0.93      0.94        40
           1       0.88      0.75      0.81        20
           2       0.79      0.95      0.86        20

    accuracy                           0.89        80
   macro avg       0.87      0.88      0.87        80
weighted avg       0.89      0.89      0.89        80
```

Figura 20 - Métricas obtenidas con Random Forest.

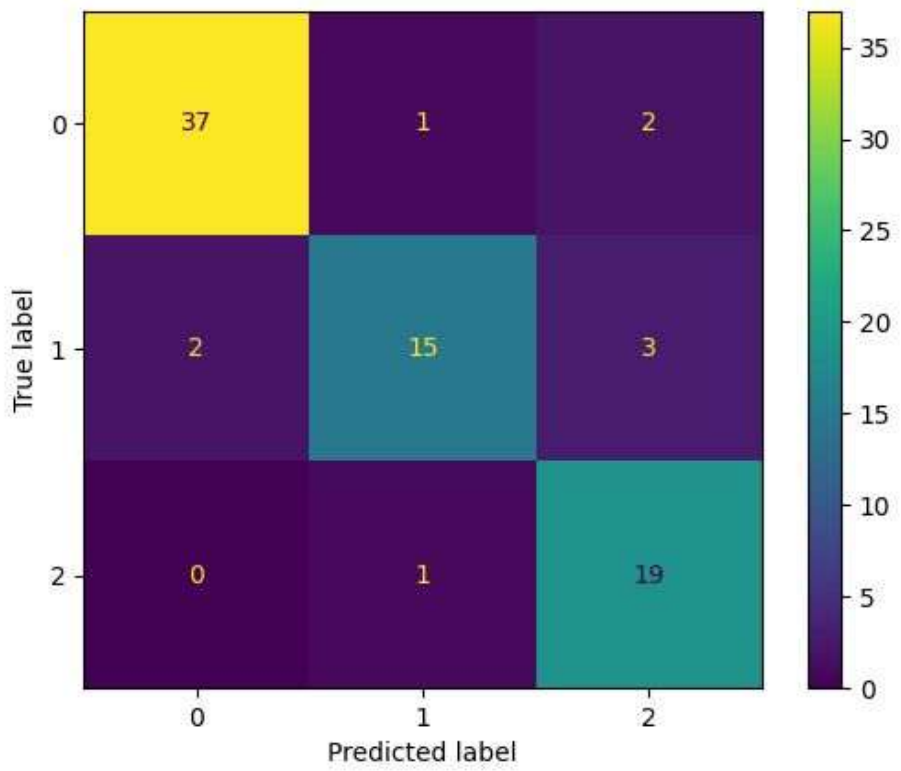


Figura 21 - Matriz de confusión de Random Forest.

La clase Normal vuelve a clasificarse muy bien, con 37 aciertos sobre 40. Stage2 también mantiene un comportamiento fuerte, alcanzando 19 aciertos sobre 20. El punto más débil sigue estando en Stage1, aunque aquí el problema ya no es tan grave como ocurría en KNN. De las 20 muestras reales pertenecientes a esta categoría, el modelo clasifica correctamente 15, mientras que 2 terminan siendo identificadas como Normal y 3 como Stage2.

La diferencia respecto a KNN empieza a notarse rápido al observar la matriz de confusión. Random Forest ya no parece "perder" completamente la clase intermedia ni repartir las muestras de Stage1 de forma casi aleatoria entre las otras dos categorías. Siguen existiendo errores evidentemente, pero el comportamiento general resulta mucho más consistente y estable. El recall de Stage1 sube hasta 0.75, algo importante porque

significa que el modelo ya es capaz de detectar correctamente la mayoría de las muestras reales pertenecientes a esa clase. La precisión de Stage1 también alcanza 0.88, un valor sólido teniendo en cuenta el nivel de solapamiento existente entre categorías. Esto indicaba que cuando el modelo predecía una muestra como Stage1 normalmente lo hacía con fiabilidad y no generaba una cantidad excesiva de falsos positivos dentro de esta clase.

Random Forest era uno de los modelos donde más sentido tenía esperar una mejora respecto a KNN en un problema como este, donde muchas características acústicas presentan cierto solapamiento y donde las fronteras entre clases no son completamente claras, ese tipo de enfoque resultaba más adecuado.

También ayudaba el hecho de que Random Forest suele ser relativamente robusto frente al ruido y frente a pequeñas variaciones dentro de los datos. Esto era especialmente importante en el proyecto porque las grabaciones de voz introducían variabilidad natural relacionada con cada locutor, el ritmo del habla o incluso las propias condiciones de grabación. El modelo parecía adaptarse relativamente bien a esa situación y no daba la sensación de estar sobre ajustándose de forma agresiva al conjunto de entrenamiento.

Aun así, la matriz de confusión seguía dejando claro que el problema principal del proyecto continuaba siendo Stage1. Aunque el modelo mejora claramente respecto a KNN, todavía aparecen muestras intermedias que terminan desplazándose hacia las clases vecinas. Esto tenía sentido viendo los análisis realizados anteriormente sobre KDE y *boxplots*, donde precisamente Stage1 aparecía como la categoría con mayor mezcla respecto al resto.

La respuesta sobre Stage2 también resultaba relevante. El recall de 0.95 mostraba que el modelo prácticamente no estaba dejando escapar muestras correspondientes a los casos más severos de depresión, algo positivo desde el punto de vista del problema planteado. La precisión bajaba ligeramente debido a que algunas muestras de Stage1

terminaban entrando dentro de esta categoría, aunque el rendimiento general seguía siendo sólido.

Después se evaluó Gradient Boosting, otro modelo basado en árboles, pero con una forma de aprendizaje más secuencial.

```
Test Accuracy: 0.8875
Test ROC AUC: 0.9833333333333334

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.93      0.95        40
           1       0.88      0.75      0.81        20
           2       0.76      0.95      0.84        20

    accuracy                           0.89        80
   macro avg       0.87      0.88      0.87        80
weighted avg       0.90      0.89      0.89        80
```

Figura 22 - Métricas obtenidas con Gradient Boosting.

Gradient Boosting obtuvo una accuracy de 0.8875 y un ROC AUC cercano a 0.983, uno de los valores más altos observados hasta este punto del análisis. Aunque la *accuracy* prácticamente no cambia respecto a Random Forest, el comportamiento interno del modelo sí resulta interesante al observar tanto la matriz de confusión como el *classification report*.

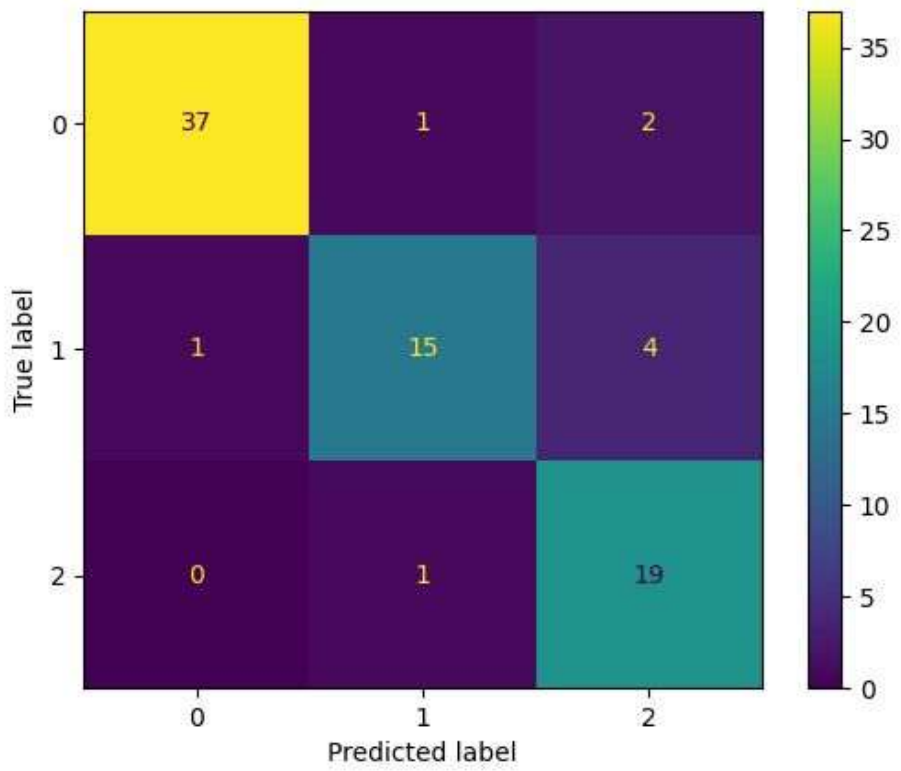


Figura 23 - Matriz de confusión de Gradient Boosting.

La diagonal principal vuelve a aparecer limpia y eso ya deja ver rápidamente que el modelo consigue mantener una clasificación estable en las tres categorías principales. La clase Normal mantiene 37 muestras correctamente clasificadas sobre 40 y Stage2 vuelve a alcanzar 19 aciertos sobre 20, algo que empieza a repetirse mucho entre los modelos más fuertes y que parecía indicar que las clases extremas del problema estaban relativamente bien definidas dentro del espacio de características acústicas.

Stage1 seguía siendo el punto complicado, aunque aquí el comportamiento del modelo resultaba algo más interesante de analizar. Las 15 muestras correctamente clasificadas muestran que Gradient Boosting sí consigue capturar una parte importante de los patrones asociados a esta categoría, pero siguen apareciendo errores desplazados tanto hacia Normal como hacia Stage2. No parecía existir una tendencia exagerada hacia una

sola de las clases vecinas, algo que sí habría sido más preocupante porque podría indicar un sesgo fuerte del modelo hacia una categoría concreta.

El recall de Stage1 volvía a quedarse en 0.75, prácticamente igual que en Random Forest. La diferencia aparecía en la precisión, que descendía ligeramente hasta 0.79 porque algunas muestras pertenecientes a otras clases acababan clasificándose dentro de Stage1. Eso hacía que la matriz se viese algo más dispersa alrededor de la clase intermedia, aunque el modelo seguía manteniendo una separación relativamente estable y sin errores excesivamente descontrolados.

En el caso de Gradient Boosting daba la sensación de que el modelo conseguía adaptarse mejor a pequeños patrones complejos presentes dentro de las variables acústicas. Al ir corrigiendo de forma iterativa los errores de árboles anteriores, terminaba afinando más las fronteras entre clases, sobre todo en regiones donde la separación no era tan evidente. Esto se apreciaba bastante bien en el valor de ROC AUC, que era especialmente alto y reforzaba la idea de que, a nivel probabilístico, el clasificador estaba diferenciando correctamente las categorías. Aun así, seguían apareciendo ciertas dificultades alrededor de Stage1, que volvía a actuar como la clase más ambigua del problema.

También resultaba curioso que los errores siguieran apareciendo concentrados en regiones concretas del problema en lugar de repartirse aleatoriamente por toda la matriz. Las confusiones entre Stage1 y las clases vecinas empezaban a parecer más una limitación dada del propio dataset y del solapamiento entre características acústicas que un fallo específico del modelo. Viendo las gráficas KDE y los análisis anteriores, tenía lógica que precisamente la categoría intermedia siguiese siendo la más difícil de separar.

XGBoost fue el primer modelo capaz de alcanzar una accuracy de 0.90 dentro de toda la comparativa, algo que ya empezaba a marcar cierta diferencia respecto a los clasificadores analizados anteriormente. El ROC AUC se mantuvo también en valores

muy altos, alrededor de 0.973, y la matriz de confusión transmitía una sensación bastante limpia visualmente, sobre todo al observar cómo se concentraban la mayoría de las muestras alrededor de la diagonal principal.

```
Test Accuracy: 0.9
Test ROC AUC: 0.9729861111111111

Classification report:
              precision    recall  f1-score   support

           0       0.95      0.95      0.95        40
           1       0.88      0.75      0.81        20
           2       0.83      0.95      0.88        20

    accuracy                           0.90        80
   macro avg       0.89      0.88      0.88        80
weighted avg       0.90      0.90      0.90        80
```

Figura 24 - Métricas obtenidas con Extreme Gradient Boosting.

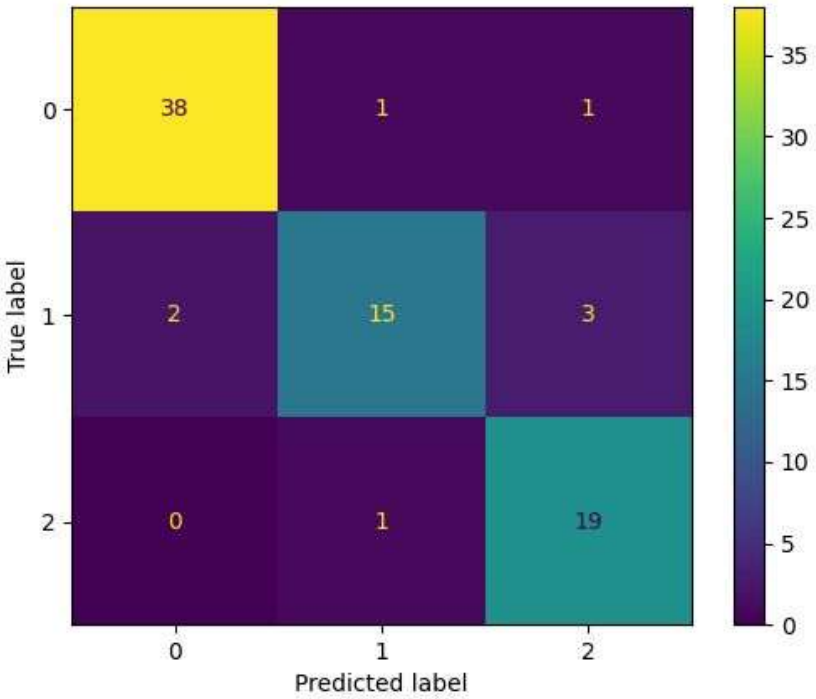


Figura 25 - Matriz de confusión de Extreme Gradient Boosting.

La clase Normal prácticamente no presenta problemas en este modelo. 38 de las 40 muestras terminan correctamente clasificadas y tanto la precisión como el *recall* alcanzan valores de 0.95, algo que ya tendía a indicar un comportamiento en gran medida estable en la detección de pacientes sin depresión.

Stage2 vuelve a mantener resultados muy altos, con 19 aciertos sobre 20. La mayoría de los errores siguen apareciendo alrededor de Stage1 y eso se aprecia muy bien en la matriz, como en todas las anteriores, Stage1 era el principal causante de los problemas, de hecho, prácticamente toda la "actividad" fuera de la diagonal principal vuelve a concentrarse en esa categoría intermedia.

Lo interesante es que esos errores empiezan a verse más controlados. No aparece una tendencia exagerada hacia una única clase ni una pérdida fuerte de sensibilidad en

ninguna categoría concreta. La matriz mantiene el equilibrio y eso era importante porque uno de los riesgos al trabajar con un dataset ligeramente desbalanceado era que el modelo terminara favoreciendo excesivamente la clase Normal durante la clasificación.

El comportamiento de XGBoost encajaba muy bien con lo que se esperaba teóricamente de este tipo de modelos. Su estructura basada en boosting secuencial le permite ir corrigiendo progresivamente errores anteriores y ajustar relaciones relativamente complejas entre variables, algo especialmente útil en un problema donde muchas diferencias entre clases no aparecen de forma completamente evidente cuando se analiza cada característica por separado.

También empezaba a notarse cierta mejora en la forma en la que el modelo gestionaba el solapamiento entre categorías. Stage1 seguía siendo claramente la zona más complicada del problema, aunque aquí el clasificador parecía manejar algo mejor esa transición entre pacientes sin depresión y casos más severos. Las confusiones seguían existiendo porque el propio dataset presentaba mezcla en determinadas variables acústicas, pero ya no daba la sensación de que el modelo estuviese tomando decisiones especialmente inestables o inconsistentes.

El valor de ROC AUC reforzaba bastante esta idea. Aunque la accuracy final es la métrica más visible, un ROC AUC tan elevado indicaba que el modelo estaba separando bien las clases a nivel probabilístico y que internamente las predicciones mantenían coherencia incluso en muestras más ambiguas.

El último modelo evaluado fue CatBoost.

```
Test Accuracy: 0.875
Test ROC AUC: 0.9808333333333333

Classification report:
              precision    recall  f1-score   support

           0       0.95      0.93      0.94        40
           1       0.88      0.70      0.78        20
           2       0.76      0.95      0.84        20

    accuracy                           0.88        80
   macro avg       0.86      0.86      0.85        80
weighted avg       0.88      0.88      0.87        80
```

Figura 26 - Métricas obtenidas con Categorical Boosting.

CatBoost obtuvo una accuracy de 0.875, también por encima del umbral del 85%, con un ROC AUC de aproximadamente 0.981. Aunque la accuracy queda ligeramente por debajo de XGBoost y de los modelos que alcanzaron 0.8875, el valor de ROC AUC sigue siendo muy alto, lo que indicaba una buena capacidad de separación general entre clases.

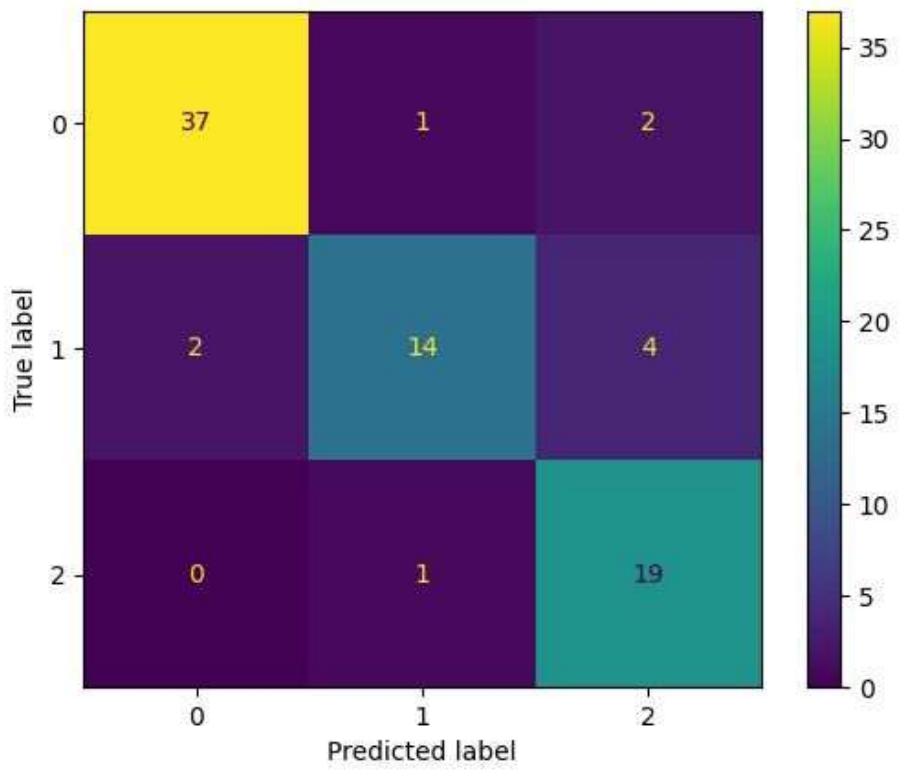


Figura 27 - Matriz de confusión de Categorical Boosting.

La matriz de confusión mantiene un patrón bastante parecido al observado en los otros modelos fuertes. La clase Normal se clasifica correctamente en 37 de las 40 muestras, Stage2 vuelve a quedar muy bien representada con 19 aciertos sobre 20, y Stage1 aparece otra vez como la clase más complicada, con 14 aciertos sobre 20. Los errores de Stage1 se reparten entre Normal y Stage2, aunque hay una ligera tendencia a confundirla más con Stage2.

Esto vuelve a reforzar una idea que se repitió durante toda la evaluación: el problema principal no estaba tanto en detectar los extremos, sino en separar correctamente la categoría intermedia. CatBoost no rompe esa tendencia, pero tampoco muestra un comportamiento inestable o incoherente. Sus errores siguen apareciendo en zonas

esperables del problema y no se observa una pérdida fuerte de capacidad sobre las clases principales.

El recall de Stage1 baja a 0.70, algo que conviene tener en cuenta, pero la precisión se mantiene en 0.88. Esto significa que el modelo no predice Stage1 de forma excesiva o descontrolada; cuando asigna una muestra a esa clase suele hacerlo con acierto, aunque deja algunas muestras reales de Stage1 fuera. En cambio, Stage2 alcanza un recall de 0.95, mostrando que los casos más severos siguen siendo detectados con consistencia.

CatBoost quedaba, así como un modelo con resultados competitivos, especialmente por su ROC AUC elevado y por mantener una matriz de confusión bastante razonable. No era un modelo perfecto, y Stage1 seguía siendo el punto más débil, pero su comportamiento general seguía siendo suficientemente sólido como para mantenerlo dentro del grupo de modelos fructíferos para el análisis posterior.

5.8 SELECCIÓN DEL MODELO FINAL Y VISUALIZACIÓN DE RESULTADOS

Después de analizar individualmente los distintos clasificadores empezó a verse un patrón repetido entre prácticamente todos los modelos. Las clases correspondientes a pacientes sin depresión y a los casos más severos tendían a separarse relativamente bien, mientras que Stage1 aparecía constantemente como la región más problemática del problema. Independientemente del algoritmo utilizado, gran parte de los errores terminaban concentrándose alrededor de esa categoría intermedia.

A partir de ahí se llegó a plantear la posibilidad de simplificar completamente el problema y convertirlo en una clasificación binaria, agrupando las muestras únicamente en “Depresión” y “No depresión”. La idea tenía sentido desde el punto de vista del rendimiento, ya que eliminando Stage1 desaparecía gran parte del solapamiento entre

clases que estaba dificultando el trabajo de los modelos. Durante estas pruebas se realizaron distintos ajustes orientados específicamente a ese escenario binario incluyendo modificaciones sobre el conjunto de características acústicas y cambios en la configuración de algunos clasificadores para intentar maximizar todavía más la accuracy. Los resultados mejoraban ligeramente en determinadas métricas, algo relativamente esperable al reducir el número de categorías del problema, aunque la diferencia obtenida no terminaba compensando la pérdida de información que suponía eliminar completamente una de las clases.

Reducir el problema a dos categorías hacía desaparecer precisamente una de las partes más enriquecedoras del proyecto: la capacidad de distinguir distintos niveles de severidad dentro del ámbito de la depresión. La mejora en accuracy existía, pero resultaba demasiado pequeña para justificar la eliminación completa de Stage1 y simplificar tanto el sistema de clasificación. Por ese motivo se decidió continuar trabajando con el enfoque multiclase original.

Dentro de todos los modelos evaluados, CatBoost empezó a posicionarse como uno de los clasificadores más interesantes para continuar el desarrollo del proyecto. Aunque XGBoost alcanzó una accuracy ligeramente superior, las diferencias reales entre ambos modelos eran relativamente pequeñas y CatBoost mantenía un comportamiento bastante estable tanto en métricas como en matriz de confusión y capacidad de separación entre clases.

El ROC AUC obtenido por CatBoost seguía moviéndose en valores muy altos, algo bastante relevante porque indicaba que el modelo era capaz de separar correctamente las categorías incluso cuando las muestras resultaban más ambiguas, la matriz de confusión tampoco reflejaba errores descontrolados ni clases claramente perjudicadas durante la clasificación. La mayoría de los fallos seguían concentrándose alrededor de Stage1. Viendo el comportamiento general del modelo, daba más la impresión de que esa

dificultad venía provocada por el propio solapamiento natural entre estados intermedios de depresión que por una limitación concreta del clasificador.

Uno de los aspectos que más peso terminó teniendo durante esta decisión fue precisamente el solapamiento entre clases ya que tanto las gráficas KDE como las matrices de confusión ya habían mostrado anteriormente que las fronteras entre categorías no estaban completamente definidas y que muchas muestras compartían características acústicas relativamente parecidas, especialmente alrededor de los casos intermedios. CatBoost suele comportarse bastante bien precisamente en este tipo de situaciones donde existen relaciones complejas entre variables y donde las clases no aparecen perfectamente separadas dentro del espacio de características.

Esto empezaba a resultar todavía más importante pensando no únicamente en el dataset utilizado durante el proyecto, sino también en una posible ampliación futura del sistema. A mayor escala probablemente aparecería todavía más variabilidad entre pacientes, diferentes condiciones de grabación, distintos micrófonos, variaciones en edad, ritmo del habla o intensidad vocal, haciendo que el solapamiento entre categorías pudiera ser incluso mayor que el observado durante estas pruebas. Precisamente por eso interesaba trabajar con un modelo que siguiese manteniendo un comportamiento relativamente robusto incluso cuando las fronteras entre clases empezaran a difuminarse más.

También se analizó el comportamiento interno de CatBoost variando distintos hiperparámetros del modelo para comprobar cómo afectaban al rendimiento general y hasta qué punto el clasificador mantenía estabilidad frente a cambios en complejidad y configuración.

La optimización de hiperparámetros se realizó *mediante RandomizedSearchCV* utilizando validación cruzada estratificada con 7 folds (*StratifiedKFold*). Durante esta fase se evaluaron 250 combinaciones distintas de hiperparámetros sobre variables

relacionadas con profundidad de árboles, número de iteraciones, learning rate, regularización y estrategias internas de muestreo del modelo. Todo el proceso se optimizó utilizando accuracy como métrica principal.

La mejor configuración obtenida seleccionó una profundidad de 5 niveles, un *learning rate de* 0.15 y 1000 iteraciones, combinando además distintos parámetros internos orientados a controlar el sobreajuste y estabilizar el entrenamiento del clasificador. La validación cruzada alcanzó una *Best CV accuracy* de 0.928, reflejando que el modelo mantenía un rendimiento elevado incluso utilizando distintas particiones internas del dataset.

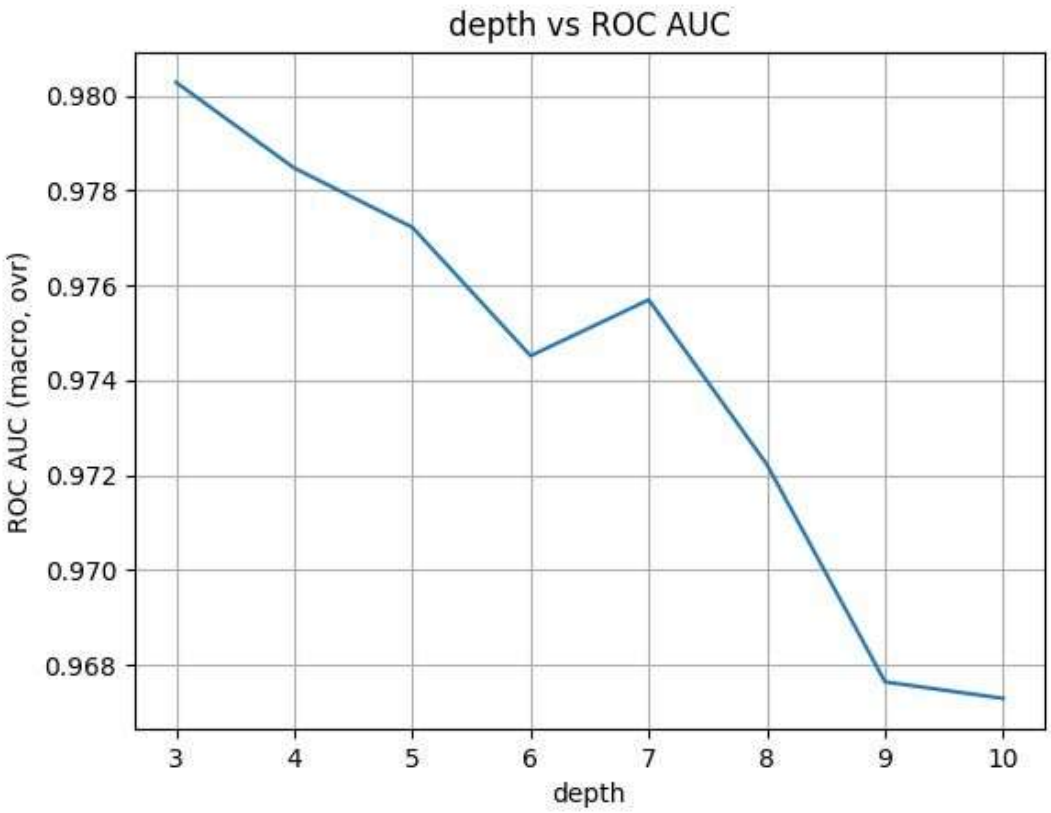


Figura 28 - Relación entre profundidad y ROC AUC de Categorical Boosting.

La profundidad de los árboles fue una de las variables donde más claramente empezó a verse el equilibrio entre complejidad y capacidad de generalización. Los mejores resultados aparecían utilizando profundidades relativamente moderadas y a partir de ciertos valores el ROC AUC comenzaba a disminuir progresivamente. Esto era de esperar porque árboles demasiado profundos terminaban ajustándose excesivamente a patrones concretos del dataset y reducían la capacidad del modelo para generalizar correctamente sobre muestras nuevas.

El valor finalmente seleccionado para *depth* fue 5, precisamente una configuración intermedia que mantenía muy buen rendimiento sin necesidad de generar estructuras excesivamente complejas. Viendo el tamaño moderado del dataset y el riesgo de sobreajuste asociado al problema, resultaba lógico que el mejor comportamiento apareciese lejos de profundidades extremas.

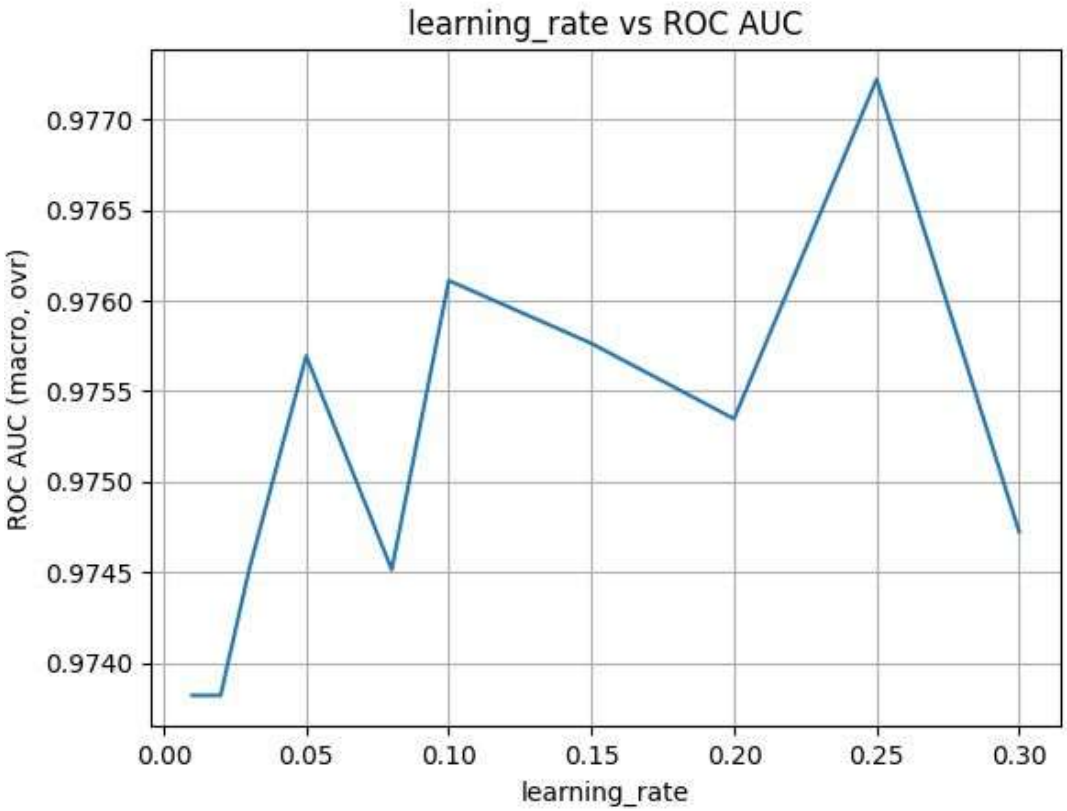


Figura 29 - Relación entre Learning rate y ROC AUC de Categorical Boosting.

La variación del learning rate no producía diferencias especialmente agresivas en el rendimiento del modelo. La gráfica se mantiene compacta y el ROC AUC oscila dentro de un rango relativamente pequeño incluso probando configuraciones distintas entre sí. Eso resultaba curioso porque daba la sensación de que el comportamiento de CatBoost seguía siendo bastante consistente sin necesidad de depender de un valor extremadamente concreto para funcionar correctamente.

Los mejores resultados aparecían alrededor de valores intermedios, mientras que configuraciones demasiado bajas ralentizaban el aprendizaje y valores más altos empezaban a volver el entrenamiento algo menos estable. Finalmente se utilizó un learning rate de 0.15, una configuración que mantenía un equilibrio bastante razonable

entre velocidad de aprendizaje y capacidad de generalización sin forzar excesivamente el ajuste del modelo sobre el dataset.

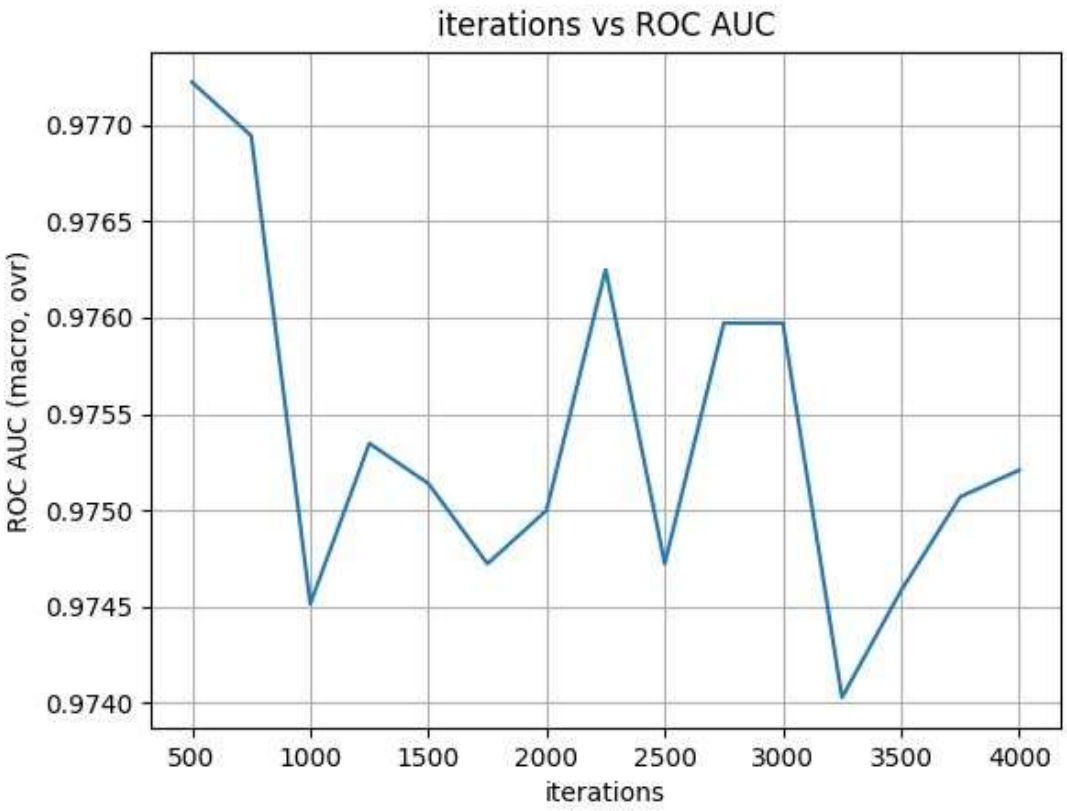


Figura 30 - Relación entre iteraciones y ROC AUC de Categorical Boosting.

La evolución respecto al número de iteraciones también aportaba bastante información sobre el comportamiento real del modelo. Al observar la gráfica se observaba que el rendimiento alcanzaba valores altos relativamente pronto y que aumentar continuamente el número de árboles no producía mejoras proporcionales en el ROC AUC. De hecho, después de cierto punto las diferencias entre configuraciones distintas pasaban a ser bastante pequeñas e incluso aparecían ligeras caídas en algunas zonas.

Esto resultaba llamativo porque daba la sensación de que CatBoost conseguía estabilizar rápido el aprendizaje sin necesidad de crecer excesivamente en complejidad. El modelo parecía capturar los patrones importantes del dataset utilizando una configuración relativamente contenida y no necesitaba miles de iteraciones adicionales

para mantener un rendimiento competitivo. Finalmente se utilizaron 1000 iteraciones, un valor que seguía situándose dentro de una de las regiones más sólidas de la gráfica y que mantenía un equilibrio razonable entre rendimiento y capacidad de generalización.

También empezó a resultar bastante evidente que los mejores resultados no aparecían utilizando configuraciones extremadamente agresivas. Las pruebas terminaban apuntando hacia un modelo relativamente equilibrado, con árboles de profundidad moderada, aprendizaje estable y mecanismos de regularización suficientes para evitar que el clasificador empezara a ajustarse demasiado al conjunto de entrenamiento causando un posible *overfitting*. Viendo el tamaño del dataset y el nivel de solapamiento existente entre clases, esto tenía bastante sentido, forzar demasiado la complejidad probablemente habría terminado empeorando la capacidad de generalización del sistema en lugar de mejorarla.

El modelo alcanzó un *Best Cross Validation accuracy* cercano a 0.928, un resultado especialmente excepcional porque ya no dependía únicamente de una única partición concreta de entrenamiento y test, el rendimiento seguía manteniéndose alto incluso cambiando las divisiones internas del dataset, algo que reforzaba la sensación de estabilidad general del clasificador y daba cierta confianza de que el comportamiento observado no estaba apareciendo simplemente por casualidad en un único *split* concreto.

Precisamente esa estabilidad terminó teniendo peso durante la selección final del modelo. A lo largo de todas las pruebas empezó a verse que el verdadero problema del proyecto no era detectar las clases extremas, sino manejar correctamente el solapamiento existente alrededor de Stage1. Las matrices de confusión, las gráficas KDE y el propio análisis de características ya mostraban desde fases anteriores que muchas muestras compartían patrones acústicos relativamente parecidos y que las fronteras entre categorías no estaban completamente definidas.

CatBoost fue uno de los modelos que mejor parecía adaptarse a esa situación. Aunque seguían existiendo errores en la clase intermedia, el comportamiento del

clasificador se mantenía coherente incluso en las zonas más ambiguas del dataset. Los errores no aparecían dispersos aleatoriamente ni mostraban un sesgo exagerado hacia una única categoría, algo importante porque indicaba que el modelo seguía siendo capaz de separar razonablemente bien las clases incluso cuando las diferencias acústicas entre pacientes empezaban a ser mucho más sutiles.

También empezó a resultar evidente que el objetivo del proyecto no podía limitarse únicamente a obtener la accuracy más alta posible sobre este dataset concreto. Si el sistema evolucionaba a futuro hacia escenarios más grandes y menos controlados, probablemente aparecería todavía más variabilidad entre pacientes, diferencias en condiciones de grabación y un solapamiento aún mayor entre categorías. Precisamente por eso interesaba trabajar con un modelo que no dependiera de fronteras extremadamente claras entre clases para mantener un comportamiento sólido.

Los resultados obtenidos durante las pruebas daban la sensación de que CatBoost conseguía mantener mejor ese equilibrio entre rendimiento, estabilidad y capacidad de generalización. El modelo no necesitaba configuraciones especialmente agresivas para obtener métricas competitivas, mantenía resultados consistentes durante validación cruzada y seguía comportándose de forma relativamente robusta incluso en las regiones donde el propio dataset mostraba mayor ambigüedad acústica. Todo esto terminó haciendo que CatBoost fuese el modelo seleccionado para continuar el desarrollo final del proyecto.

5.9 ANÁLISIS MEDIANTE PCA + CLUSTERING

Con el objetivo de interpretar mejor la estructura interna del dataset y analizar cómo se distribuían las muestras dentro del espacio de características acústicas, se aplicaron distintas técnicas de reducción de dimensionalidad y clustering sobre las variables seleccionadas durante las fases anteriores del proyecto. Este análisis permitía observar visualmente hasta qué punto las clases presentaban agrupaciones coherentes y

cómo se relacionaban entre sí dentro del espacio generado por las características acústicas extraídas a partir de las grabaciones de voz.

Para ello se utilizó *Principal Component Analysis* (PCA), una técnica de reducción de dimensionalidad que transforma las variables originales en nuevas componentes principales capaces de concentrar la mayor parte de la variabilidad presente en los datos. De esta forma resultaba posible representar el comportamiento global del dataset utilizando únicamente unas pocas dimensiones, facilitando la visualización y el análisis de relaciones entre muestras.

La gráfica de varianza explicada mostraba que las primeras componentes principales concentraban una parte importante de la información total del dataset. La primera componente (PC1) explicaba aproximadamente un 30% de la varianza y el segundo alrededor de un 27%, haciendo que entre ambas tuvieran cerca de un 58% de la información global de los datos. A partir de la tercera componente el crecimiento empezaba a ser más gradual y la varianza añadida por cada nueva dimensión se reducía progresivamente. Con las cuatro primeras componentes ya se superaba aproximadamente el 80% de la varianza acumulada, algo que hacía bastante razonable utilizar representaciones en dos y tres dimensiones para visualizar el comportamiento general de las muestras sin perder una cantidad excesiva de información respecto al espacio original.

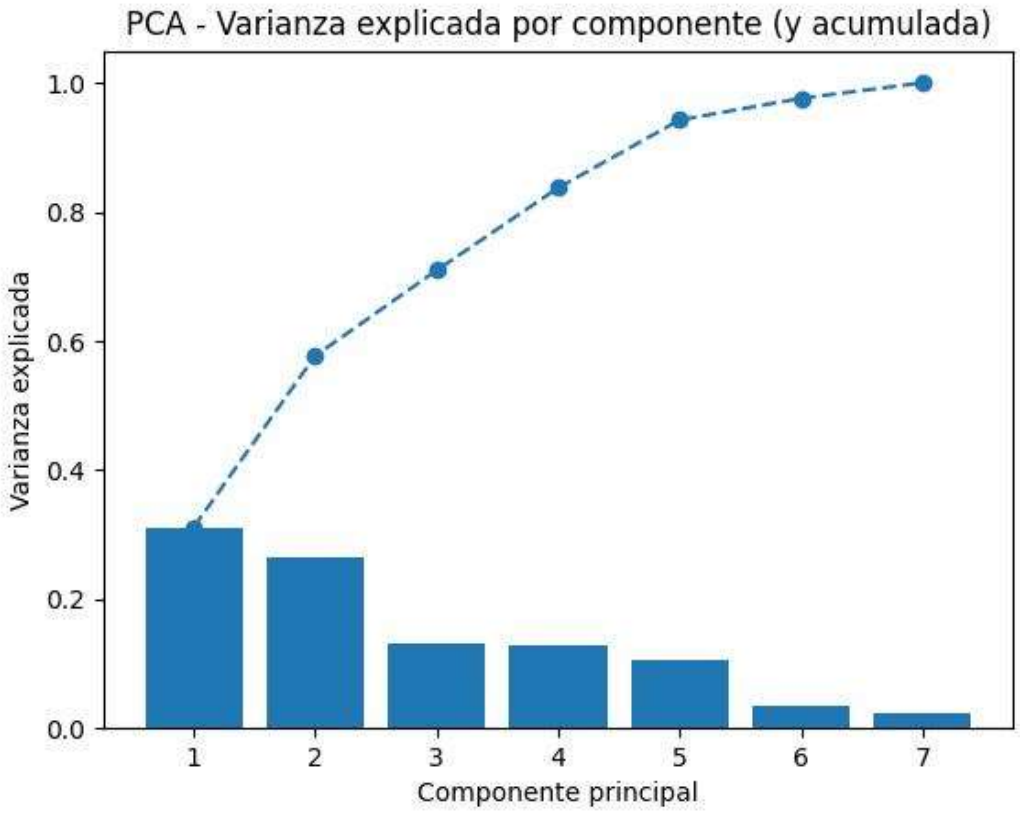


Figura 31 - Varianza explicada por las componentes principales del PCA.

Una vez proyectados los datos sobre las componentes principales, también resultaba interesante analizar qué variables acústicas estaban teniendo más peso dentro de cada eje generado por PCA. Para ello se calcularon los loadings de cada característica, que permiten observar cuánto contribuye cada variable original a las distintas componentes principales y ayudan a interpretar qué tipo de información acústica está organizando realmente el espacio PCA.

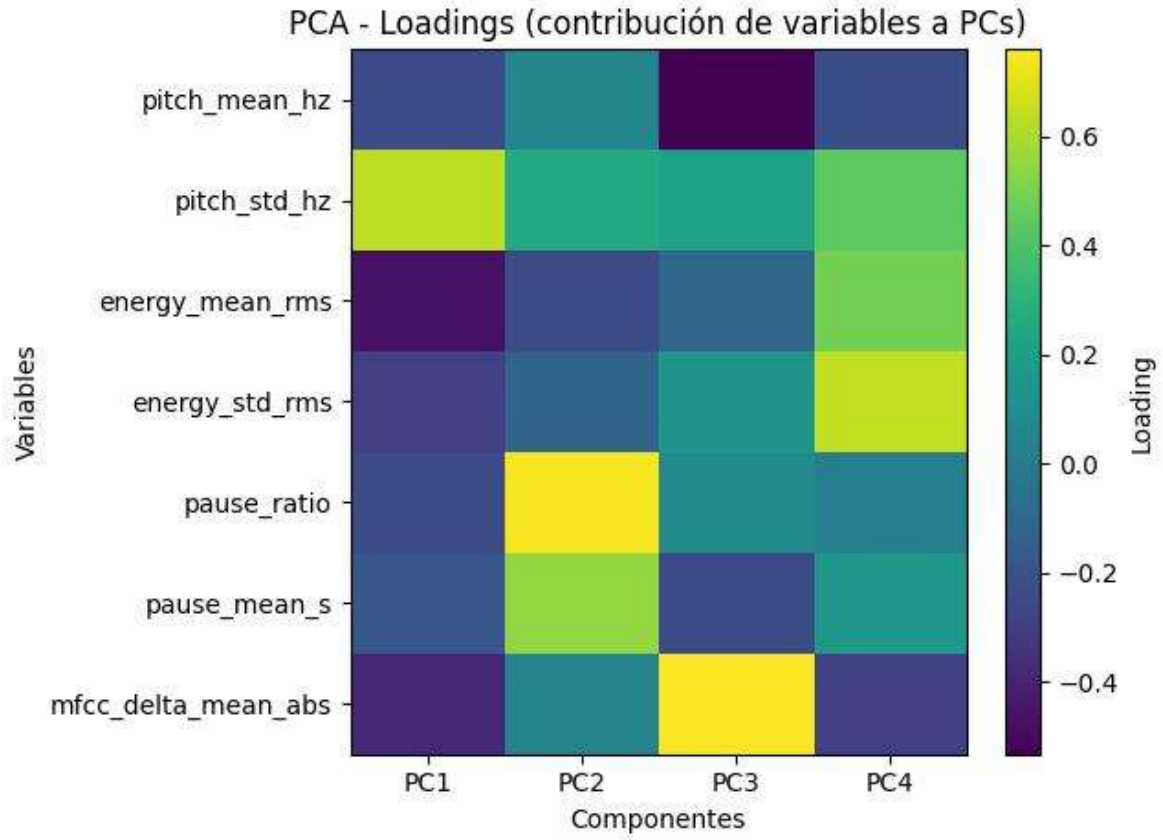


Figura 32 - Variables con mayor contribución absoluta en las componentes principales.

Las distintas componentes empezaban a mostrar comportamientos coherentes con lo observado anteriormente durante el análisis SHAP y las gráficas KDE. En PC1 aparecía dominando claramente *pitch_std_hz*, seguida por *energy_mean_rms* y *mfcc_delta_mean_abs*, algo que merece atención porque volvía a colocar la variabilidad tonal como una de las partes más influyentes dentro del problema. De hecho, esta misma variable ya había aparecido anteriormente como una de las características más relevantes durante el entrenamiento de los clasificadores y aquí volvía a reaparecer como uno de los principales ejes de separación del espacio PCA.

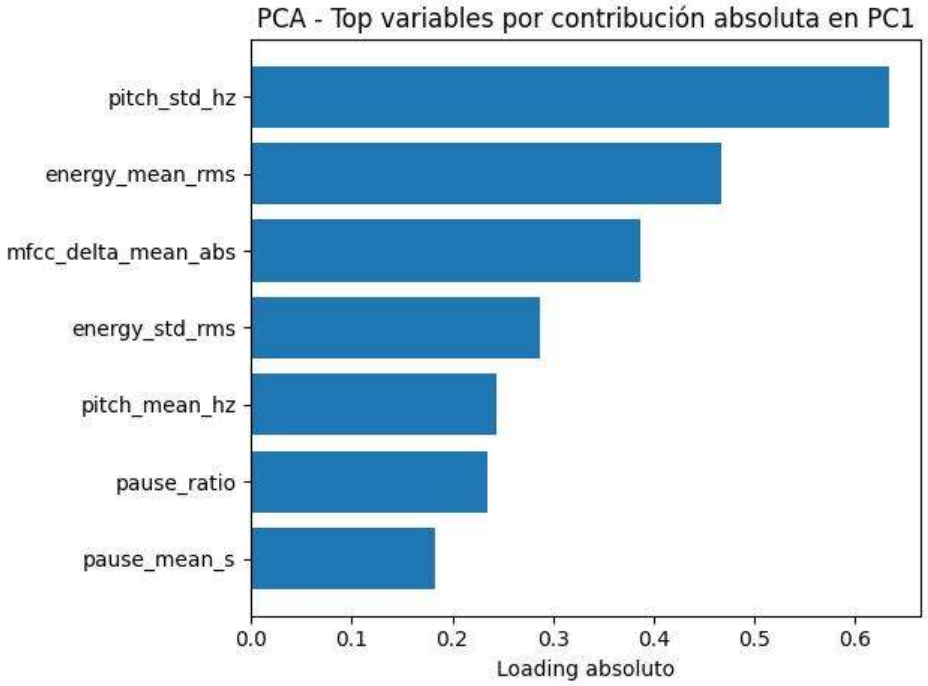


Figura 33 - Variables con mayor contribución en PC1

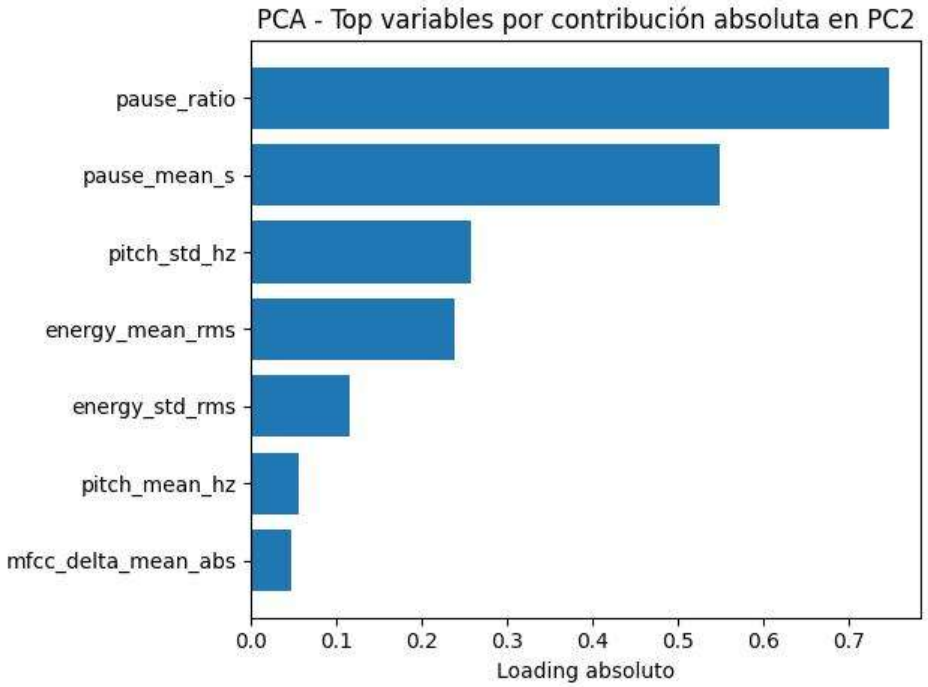


Figura 34 - Variables con mayor contribución en PC2

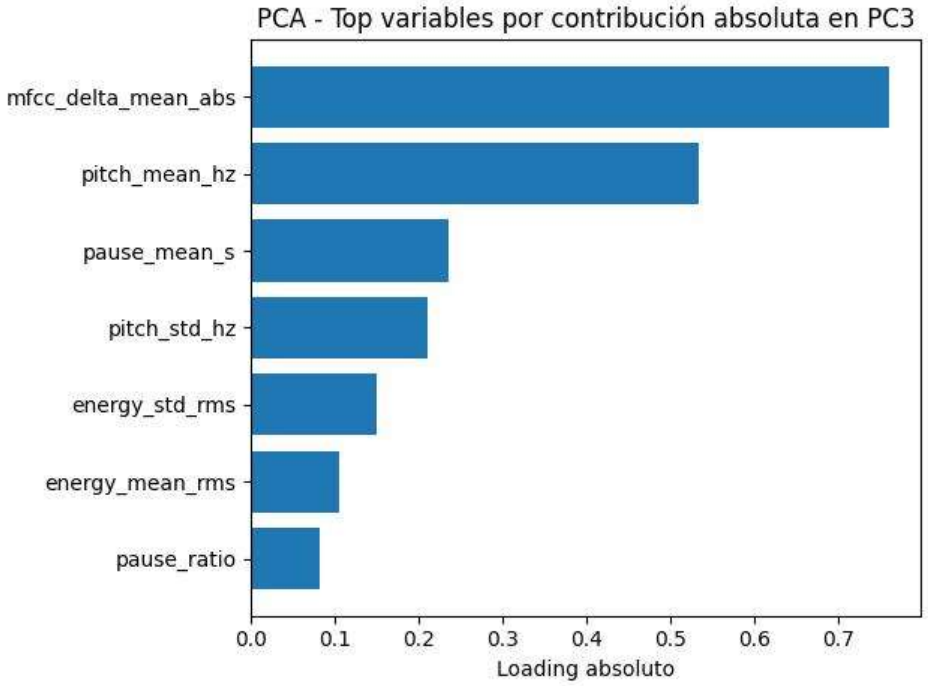


Figura 35 - Variables con mayor contribución en PC3

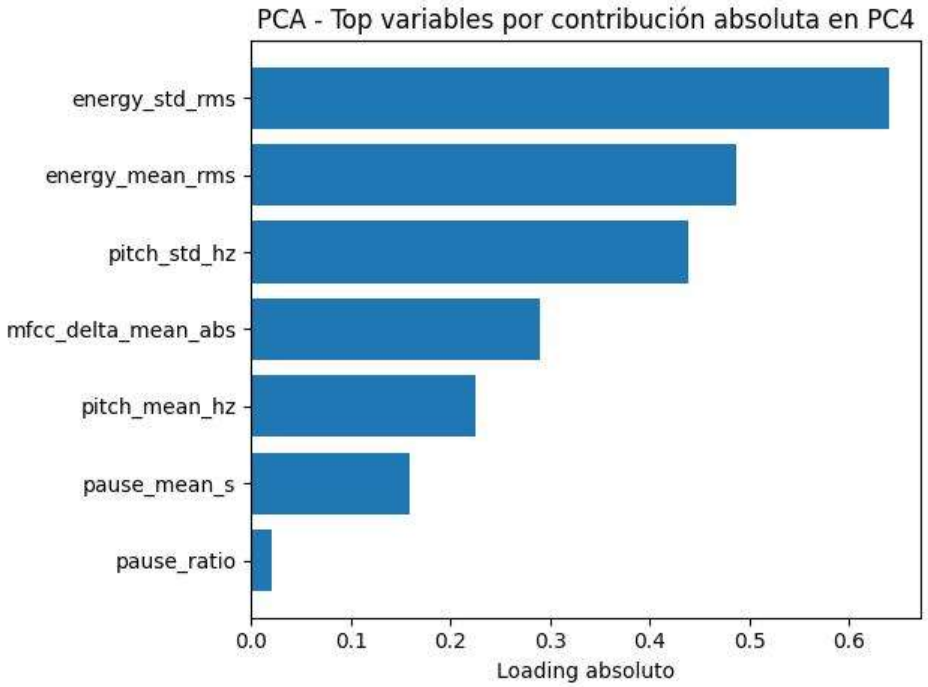


Figura 36 - Variables con mayor contribución en PC4

En PC2 empezaban a ganar mucho peso las variables relacionadas con pausas y ritmo del habla, especialmente *pause_ratio* y *pause_mean_s*. Esto resultaba importante porque mostraba que PCA no estaba concentrando toda la información únicamente alrededor del tono o la energía de la voz, sino que distintas componentes empezaban a capturar tipos diferentes de comportamiento acústico.

PC3 mostraba un comportamiento bastante curioso alrededor de *mfcc_delta_mean_abs*, que pasaba a convertirse en la variable claramente dominante de esa componente. Esto encajaba bastante bien con lo observado anteriormente durante SHAP, donde las variables derivadas de MFCC también mostraban relevancia dentro de las decisiones del modelo.

En PC4 volvía a ganar peso la información relacionada con energía y variabilidad de amplitud, especialmente *energy_std_rms y energy_mean_rms*. Viendo todas las componentes de forma conjunta daba la sensación de que PCA conseguía redistribuir bien distintos tipos de información acústica dentro de ejes relativamente diferenciados: tono, pausas, energía y características espectrales aparecían repartidas entre varias componentes principales en lugar de quedar completamente concentradas en una única dirección dominante.

Después de analizar la estructura PCA se aplicó el algoritmo *K-Means* para estudiar si las muestras tendían a agruparse de forma relativamente coherente dentro del nuevo espacio generado por las componentes principales. Antes de realizar el *clustering* fue necesario estimar cuántos grupos podían resultar razonables dentro del dataset, para lo cual se utilizaron tanto el *Elbow Method* como el *Silhouette Score*.

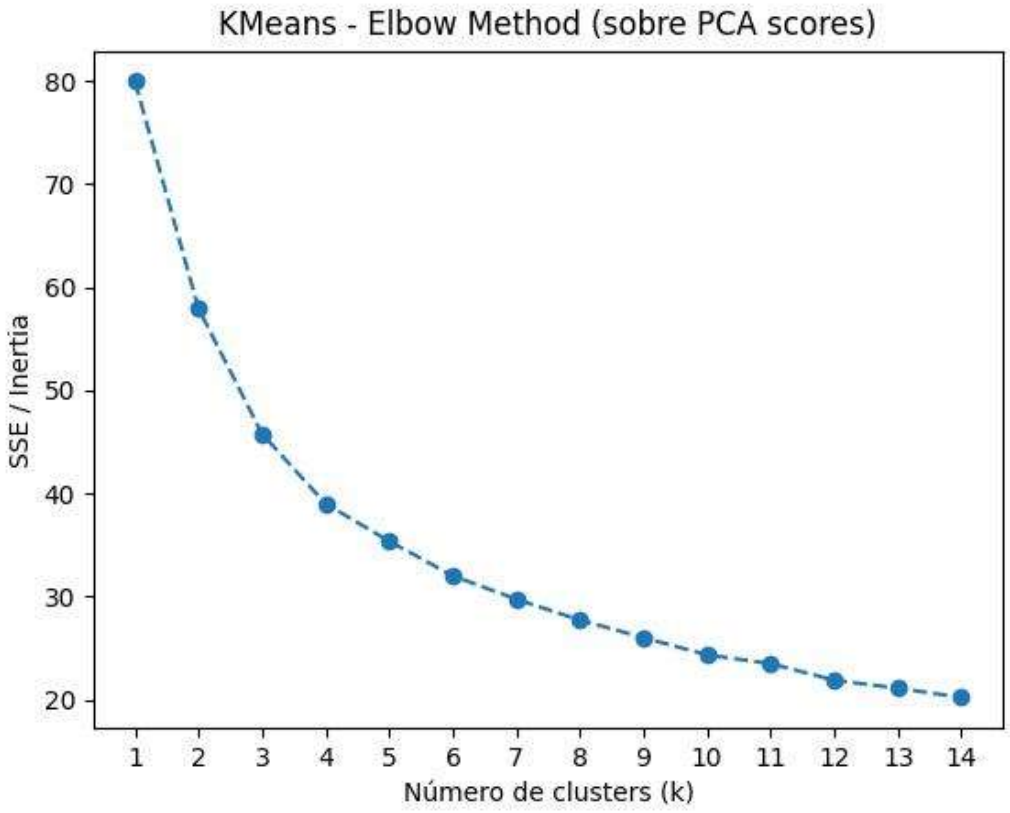


Figura 37 - Elbow method aplicado sobre los datos transformados mediante PCA.

El *Elbow Method* analiza cómo evoluciona la inercia interna del modelo a medida que aumenta el número de clústeres. La idea consiste en observar a partir de qué punto añadir más grupos deja de producir mejoras significativas en el agrupamiento interno de las muestras. En este caso la caída más pronunciada aparecía durante los primeros valores de k y empezaba a suavizarse progresivamente alrededor de 3-4 clústeres, algo que ya sugería que utilizar un número excesivamente alto de grupos no aportaría muchísima información adicional, por lo que estaba bien cortar en esos 3 o 4 clústeres.

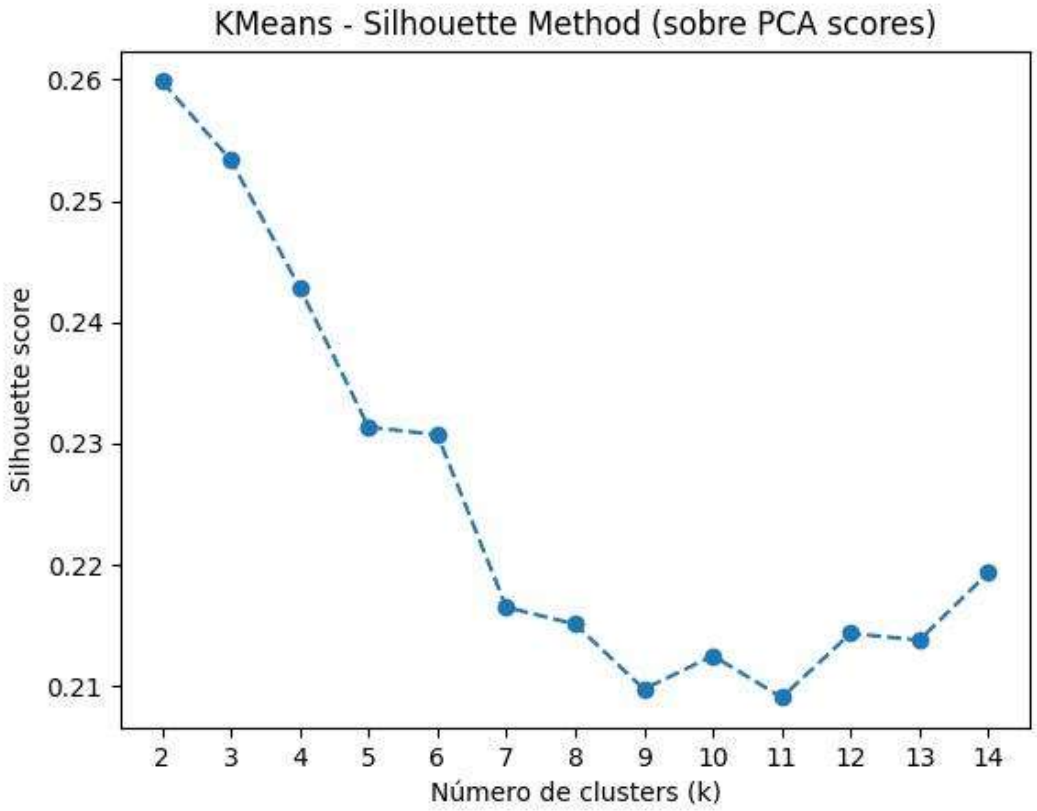


Figura 38 - Evolución del silhouette score según el número de clústeres.

El *Silhouette Score* permitía complementar este análisis desde otra perspectiva distinta, mientras el *Elbow method* se centra en la agrupación interna de los clústeres, el *silhouette* evalúa simultáneamente la cohesión dentro de cada grupo y la separación respecto al resto de clústeres. Utilizar ambos métodos conjuntamente resultaba importante porque un clustering podía presentar buena agrupación interna y aun así mostrar solapamiento entre grupos.

Los valores obtenidos en silhouette no eran especialmente altos, aunque sí mostraban cierta estructura dentro del dataset. Las clases nunca aparecían completamente separadas entre sí y tanto las matrices de confusión como las gráficas KDE ya habían mostrado mezcla alrededor de las categorías intermedias. Aun así, los resultados seguían

indicando que las muestras no se distribuían de forma completamente aleatoria dentro del espacio de características.

Una vez seleccionado el número de clústeres se representaron las muestras dentro del espacio PCA utilizando las dos primeras componentes principales. Primero se visualizaron los grupos generados automáticamente por K-Means para observar cómo estaba organizando internamente el algoritmo las grabaciones según similitud acústica.

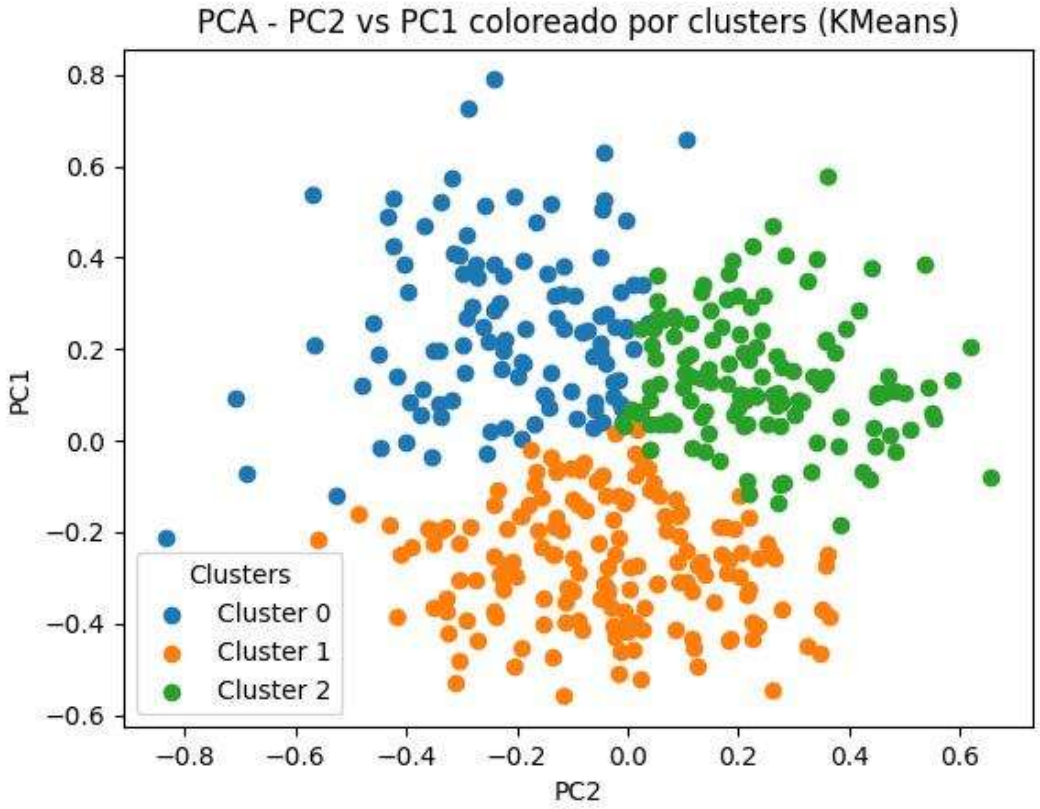


Figura 39 - Distribución de clústeres sobre el espacio PCA de dos dimensiones.

La representación mostraba una estructura interna bastante clara dentro del dataset. El clúster azul aparecía concentrado principalmente en la zona superior izquierda de la gráfica, mientras que el naranja ocupaba regiones inferiores y el verde se desplazaba hacia la parte derecha del espacio PCA. K-Means estaba generando agrupaciones reconocibles sin utilizar en ningún momento las etiquetas reales de los pacientes, algo

que ya empezaba a indicar que las características acústicas contenían cierta organización interna coherente.

Las fronteras entre grupos, aun así, no aparecían completamente separadas. Existían regiones centrales donde muchos puntos compartían espacio entre varios clústeres y donde las divisiones empezaban a difuminarse visualmente. Esto coincidía bastante con lo observado anteriormente durante el análisis KDE y las matrices de confusión. Las categorías nunca llegaban a separarse de forma completamente limpia y aquí volvía a aparecer esa misma idea representada directamente sobre el espacio PCA.

También podía verse que la mezcla entre grupos no se distribuía de forma totalmente aleatoria. Algunas regiones mantenían concentraciones muy marcadas de un mismo clúster mientras que otras actuaban casi como zonas de transición entre categorías. La distribución general daba la sensación de que el dataset sí poseía estructura, aunque manteniendo fronteras acústicas parcialmente compartidas entre pacientes.

Después de visualizar los clústeres generados automáticamente por K-Means se representaron las etiquetas reales originales sobre exactamente el mismo espacio PCA para comparar ambas distribuciones.

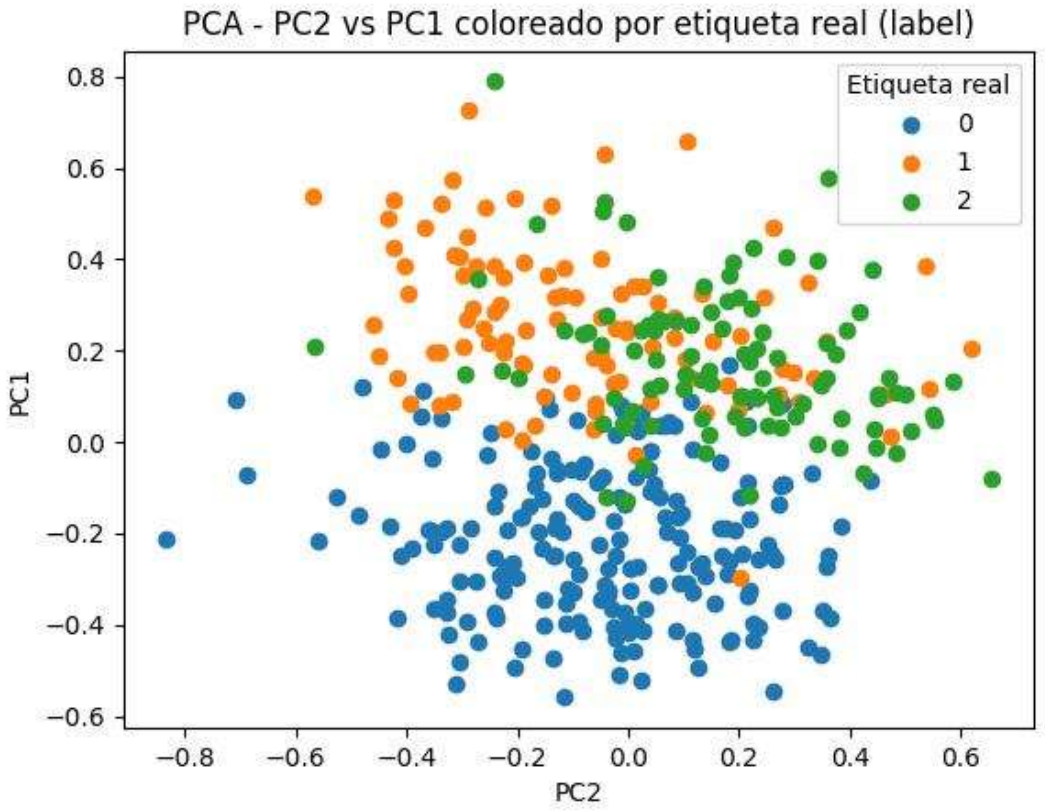


Figura 40 - Distribución de las etiquetas reales sobre el espacio PCA en dos dimensiones.

Aquí la relación con los resultados obtenidos durante clasificación aparecía de forma mucho más clara. Las muestras correspondientes a la clase 0 tendían a ocupar sobre todo regiones inferiores del gráfico, mientras que la clase 2 aparecía desplazada hacia zonas más altas y hacia la derecha. La clase 1 quedaba repartida entre ambas regiones y compartía espacio con las otras dos categorías en numerosas zonas de la representación.

La distribución ayudaba a entender por qué Stage1 había generado tantos errores durante la clasificación supervisada. Muchas grabaciones intermedias compartían características acústicas similares tanto con pacientes sin depresión como con casos más severos, haciendo que las fronteras entre categorías apareciesen mucho menos definidas en esa región concreta del espacio PCA. Al comparar esta gráfica con las matrices de confusión obtenidas anteriormente aparecía una conexión directa entre ambas partes del

proyecto. Los errores de clasificación no surgían únicamente por limitaciones de los modelos, sino porque determinadas regiones del dataset presentaban una separación acústica ambigua.

También podía apreciarse que las clases extremas mantenían distribuciones más compactas visualmente, muchos pacientes pertenecientes a las categorías 0 y 2 aparecían agrupados en zonas reconocibles, mientras que Stage1 mostraba una dispersión mucho mayor.

Para complementar la visualización en dos dimensiones también se realizó una representación utilizando las tres primeras componentes principales con el objetivo de observar la distribución espacial de las grabaciones desde una perspectiva más completa.

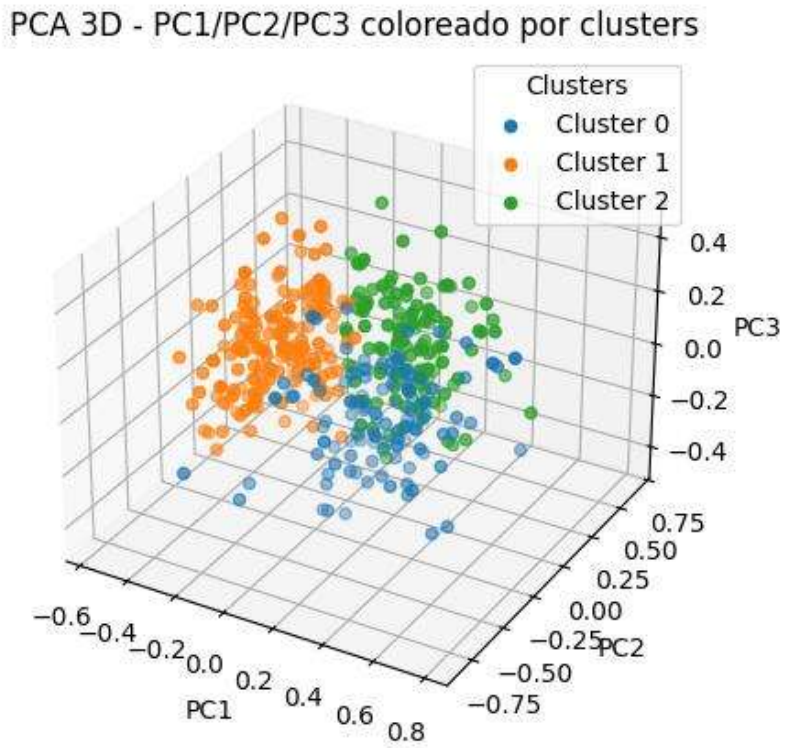


Figura 41 - Representación tridimensional de los clústeres sobre las componentes principales.

La representación tridimensional mostraba con mayor claridad algunas separaciones que en dos dimensiones quedaban ocultas por la proyección PCA. Dependiendo del ángulo de visualización ciertos grupos aparecían más compactos y algunas zonas mostraban distancias mayores entre clústeres. La mezcla entre categorías seguía apareciendo en distintas regiones del espacio tridimensional, sobre todo alrededor de áreas centrales donde distintos grupos compartían posiciones cercanas.

Las grabaciones terminaban agrupándose alrededor de ciertas regiones del espacio PCA y eso volvía a coincidir con gran parte de lo observado anteriormente durante el desarrollo experimental. Las categorías extremas ocupaban regiones más compactas mientras que las muestras intermedias aparecían mucho más repartidas y mezcladas entre ambos extremos.

Capítulo 6. ANÁLISIS DE RESULTADOS

6.1 VALIDACIÓN CON DATOS INTERNOS

Una vez finalizado el entrenamiento del modelo se utilizó el clasificador *CatBoost* para predecir a qué categoría pertenecerían distintas grabaciones de voz pertenecientes al propio dataset empleado durante el proyecto. Para ello se seleccionó un audio representativo de cada una de las tres etiquetas utilizadas durante la clasificación: pacientes sin depresión, depresión grado 1 y depresión grado 2.

El objetivo de esta parte no consistía únicamente en comprobar si el modelo acertaba la clase final. También interesaba observar cómo repartía internamente la probabilidad entre las distintas categorías, ya que eso permitía ver con qué nivel de seguridad estaba realizando cada predicción y cómo reaccionaba frente a muestras más claras o ambiguas.

```
Class probabilities (decimals): ['1.0000', '0.0000', '0.0000']

Predicted label: 0
```

Figura 42 - Resultado de predicción con datos internos sin depresión.

En la primera predicción, correspondiente a un paciente sin depresión, toda la probabilidad queda prácticamente concentrada en la clase 0. Stage1 y Stage2 aparecen reducidas a valores mínimos y el modelo no muestra ninguna duda visible durante la clasificación. Esto tiene sentido viendo lo que ya aparecía anteriormente en PCA y clustering, donde gran parte de las muestras pertenecientes a esta categoría ocupaban zonas más compactas y separadas del resto.

```
Class probabilities (decimals): ['0.0009', '0.9887', '0.0104']

Predicted label: 1
```

Figura 43 - Resultado de predicción con datos internos con depresión grado 1.

La predicción asociada a Stage1 mostraba una situación distinta. La clase correcta seguía dominando claramente la salida del modelo con una probabilidad cercana a 0.99, aunque aquí ya aparecía una pequeña parte de la distribución desplazándose hacia Stage2. Viendo las matrices de confusión y las representaciones PCA esto no resultaba extraño porque Stage1 había sido la región donde más mezcla aparecía entre categorías y muchas grabaciones compartían características acústicas similares con casos más severos.

La distribución de probabilidades seguía manteniendo una dirección clara hacia la clase correcta, aunque aquí ya podía verse algo que había aparecido durante prácticamente todo el proyecto: las muestras intermedias no quedaban tan separadas acústicamente como las clases extremas.

```
Class probabilities (decimals): ['0.0000', '0.0002', '0.9998']

Predicted label: 2
```

Figura 44 - Resultado de predicción con datos internos con depresión grado 2.

En la muestra correspondiente a depresión grado 2 volvía a aparecer una predicción completamente definida. La práctica totalidad de la probabilidad se concentra sobre la categoría correcta y las otras clases quedan reducidas a valores residuales. Igual que ocurría con la clase 0, la separación acústica aquí resultaba mucho más clara que en Stage1.

Las tres predicciones terminaban siguiendo la misma lógica observada durante toda la fase experimental. Los audios pertenecientes a las categorías extremas quedaban clasificados con una seguridad muy alta mientras que las grabaciones intermedias concentraban gran parte de la ambigüedad entre clases. Aquí volvía a verse esa transición acústica alrededor de Stage1 que ya había aparecido anteriormente en SHAP, KDE, PCA y matrices de confusión.

6.2 VALIDACIÓN CON DATOS EXTERNOS

Después de comprobar el funcionamiento del modelo utilizando grabaciones pertenecientes al propio dataset, se realizaron predicciones sobre audios externos no utilizados durante ninguna fase del entrenamiento. El objetivo aquí ya no consistía únicamente en verificar si el clasificador acertaba una categoría concreta, sino observar hasta qué punto era capaz de mantener un funcionamiento coherente fuera de las condiciones originales del dataset.

Para esta validación se utilizó un fragmento de la serie animada *"Bojack Horseman"*, seleccionada principalmente por la gran carga emocional presente en muchas de sus escenas y por la intensidad expresiva que suelen transmitir los personajes durante los diálogos. Esto hacía que el audio utilizado resultase útil para comprobar cómo reaccionaba el modelo frente a una grabación mucho menos controlada y más cercana a una situación real de conversación.

Otro punto importante de esta prueba es que el mismo fragmento se utilizó tanto en inglés como en español. La intención no era comprobar diferencias entre idiomas, sino analizar si el modelo dependía realmente del contenido lingüístico o si estaba trabajando principalmente sobre las características acústicas seleccionadas durante el proyecto.

También existía una diferencia importante respecto a los audios originales utilizados durante el entrenamiento ya que gran parte de las grabaciones del dataset tenían una duración relativamente corta, de apenas unos segundos, en raras ocasiones sobrepasando los 10 segundos, mientras que el fragmento seleccionado para esta validación alcanzaba aproximadamente 1 minuto y 25 segundos. Esto permitía comprobar si el sistema conseguía mantener predicciones razonables incluso trabajando sobre señales mucho más largas y con una variabilidad vocal considerablemente mayor dentro del mismo audio.

```
Class probabilities (decimals): ['0.0006', '0.9903', '0.0091']

Predicted label: 1
```

Figura 45 - Predicción externa utilizando el fragmento original en inglés.

La predicción obtenida sobre el audio original en inglés mostraba una distribución muy concentrada hacia una única categoría, con una probabilidad dominante claramente superior al resto. El modelo no parecía reaccionar al idioma como un factor principal y seguía manteniendo una clasificación bastante definida utilizando únicamente la información acústica presente en la señal de voz.

Esto era importante porque durante el entrenamiento nunca se trabajó con contenido semántico ni procesamiento lingüístico del texto hablado. El clasificador únicamente recibía variables acústicas derivadas de la señal de audio, por lo que una respuesta consistente entre idiomas empezaba a reforzar la idea de que el sistema realmente estaba capturando patrones vocales relacionados con tono, pausas, energía y dinámica de la voz, no palabras concretas o estructuras del lenguaje.

```
Class probabilities (decimals): ['0.0003', '0.9989', '0.0008']

Predicted label: 1
```

Figura 46 - Predicción externa utilizando el fragmento doblado al español.

La predicción sobre la versión doblada al español seguía una línea muy parecida. La categoría dominante volvía a concentrar gran parte de la probabilidad total y la distribución entre clases mantenía una estructura similar a la observada en el audio original. Aunque podían existir pequeñas diferencias entre ambas salidas debido a cambios naturales introducidos por el doblaje como una entonación distinta, ritmo del habla diferente o variaciones en intensidad vocal, por ello esa muy leve variación en la proporción de las predicciones.

Aquí también empezaba a verse otro aspecto importante relacionado con la duración del audio. A pesar de trabajar sobre una grabación mucho más extensa que las utilizadas originalmente durante el entrenamiento, el clasificador seguía generando predicciones estables y no aparecían distribuciones caóticas entre múltiples categorías. Esto sugería que las características acústicas extraídas seguían conservando suficiente información representativa incluso cuando el audio incluía cambios emocionales, pausas prolongadas y variaciones vocales más complejas dentro de la misma escena.

La validación externa terminaba mostrando un escenario bastante distinto respecto a las pruebas internas del dataset. Aquí el modelo ya no trabajaba sobre grabaciones conocidas ni sobre condiciones similares a las utilizadas durante el entrenamiento, aun así, las predicciones mantenían una lógica coherente y continuaban siguiendo patrones muy parecidos a los observados anteriormente durante el desarrollo experimental.

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

7.1 CONCLUSIONES

El proyecto ha terminado construyendo una herramienta capaz de analizar grabaciones de voz y asociarlas a distintos niveles relacionados con depresión utilizando aprendizaje automático. Todo el trabajo partía únicamente de señales acústicas y desde el inicio existía cierta duda sobre hasta qué punto variables simples podían contener información suficiente como para separar estados emocionales donde las fronteras nunca aparecen completamente definidas.

Conforme avanzaba el análisis iban repitiéndose varias ideas entre casi todos los apartados. Las variables relacionadas con variabilidad tonal, pausas, energía y MFCC volvían constantemente a ocupar posiciones importantes dentro de SHAP, PCA o KDE, en algunas gráficas las diferencias entre clases podían distinguirse con claridad mientras que en otras zonas la mezcla aumentaba mucho, sobre todo alrededor de Stage1. Gran parte de los errores acababan concentrándose ahí y eso dejaba claro que el problema no consistía únicamente en encontrar un clasificador más potente, si no en la propia naturaleza de los datos. Había grabaciones que compartían rasgos acústicos con ambos extremos y esa transición iba apareciendo una y otra vez en matrices de confusión, clustering, probabilidades y validaciones posteriores.

Durante las primeras pruebas ya podía verse que ciertos algoritmos sufrían bastante cuando las fronteras comenzaban a difuminarse. KNN dependía demasiado de distancias locales y rápidamente empezaba a confundirse en regiones intermedias. Los enfoques basados en árboles reaccionaban mejor a relaciones complejas entre variables acústicas y los métodos *boosting* conseguían aprovechar mejor la combinación entre

características. CatBoost fue ganando peso poco a poco por eso mismo. No existían diferencias enormes frente a otros clasificadores en accuracy pura, aunque transmitía una sensación de mayor estabilidad cuando las muestras se complicaban y las fronteras entre clases dejaban de estar claras.

Las probabilidades generadas durante validación seguían una lógica muy parecida a la observada anteriormente en PCA y *clustering*. Los audios pertenecientes a las clases extremas quedaban clasificados con una seguridad muy alta mientras que las grabaciones intermedias desplazaban parte de la probabilidad hacia regiones vecinas. Aquí volvía a aparecer algo que llevaba presente desde gran parte del proyecto: Stage1 funcionaba casi como una transición acústica entre ambos extremos en lugar de actuar como un bloque completamente separado.

PCA y K-Means ayudaron bastante a visualizar esa idea. Las muestras no quedaban repartidas de forma aleatoria dentro del espacio de características. Existían zonas reconocibles y algunos grupos acababan concentrándose en regiones definidas, aunque seguían apareciendo áreas donde distintas clases compartían espacio entre sí. Al comparar esas representaciones con las matrices de confusión podía verse una relación muy directa entre ambas partes del análisis. Muchos errores de clasificación ya estaban reflejados visualmente dentro del propio espacio PCA incluso antes de entrenar los clasificadores.

La validación externa reforzó varias de esas ideas. Utilizar el fragmento de la serie animada *"Bojack Horseman"* en inglés y español servía para comprobar si el clasificador dependía realmente del idioma o si estaba trabajando sobre propiedades acústicas generales de la voz. Las predicciones mantenían una línea muy parecida entre ambas versiones y eso tenía sentido viendo que durante todo el proyecto nunca se utilizó contenido semántico ni procesamiento lingüístico del texto hablado.

El fragmento utilizado presentaba otra diferencia importante respecto a los audios originales del dataset: la duración. Muchas grabaciones de entrenamiento ocupaban apenas unos segundos mientras que aquí se trabajaba sobre una pista de más de un minuto con cambios emocionales, pausas largas, variaciones de intensidad y modificaciones continuas en la entonación. Incluso trabajando sobre una señal tan distinta, el clasificador seguía generando probabilidades lógicas y no comenzaba a repartir la predicción de forma caótica entre múltiples clases.

Una de las ideas que más se repite durante todo el proyecto es que la voz sí contiene información útil relacionada con estados emocionales y patrones asociados a depresión. Las fronteras nunca quedan completamente limpias y existen regiones donde varias clases comparten rasgos acústicos muy parecidos, aunque incluso en esas zonas las distintas técnicas utilizadas conseguían encontrar organización dentro de las grabaciones y generar clasificaciones coherentes en la mayoría de los escenarios analizados.

7.2 TRABAJOS FUTUROS

El dataset utilizado durante el proyecto seguía siendo relativamente pequeño y gran parte de las grabaciones procedían de condiciones bastante controladas. Uno de los siguientes pasos más claros sería ampliar mucho la variedad de pacientes, edades, acentos, micrófonos y entornos de grabación. Cuanta más variabilidad exista dentro de las muestras, más fácil será comprobar cómo reacciona el clasificador fuera de escenarios concretos de entrenamiento.

La validación externa todavía quedaba algo limitada. Las pruebas realizadas funcionaban bien como una primera aproximación, aunque ampliar esa parte ayudaría mucho a observar cómo cambian las predicciones frente a conversaciones reales, entrevistas, podcasts o grabaciones espontáneas donde el ruido y los cambios emocionales aparecen de forma mucho menos controlada.

Otra línea clara estaría relacionada con el propio tratamiento del audio. En este trabajo toda la clasificación se apoyaba sobre características acústicas extraídas previamente y resumidas mediante variables estadísticas. Existen enfoques mucho más recientes basados en redes neuronales profundas capaces de trabajar directamente sobre espectrogramas o incluso sobre la señal temporal completa. Ahí podrían aparecer relaciones acústicas mucho más complejas que las utilizadas durante este proyecto.

La estructura de las clases podría evolucionar bastante. Muchas veces Stage1 no funcionaba como una categoría completamente separada sino como una región intermedia compartida entre ambos extremos. Trabajar con escalas más progresivas o introducir más niveles ayudaría a representar mejor esa transición acústica que iba apareciendo continuamente durante todo el análisis.

Otra posibilidad interesante sería combinar voz junto a otros tipos de información. Expresión facial, velocidad de escritura, interacción verbal o análisis temporal de conversaciones podrían aportar contexto adicional en regiones donde el audio por sí solo deja demasiada ambigüedad entre clases.

El proyecto deja una base bastante amplia sobre la que seguir construyendo. Gran parte del trabajo realizado aquí ya apunta hacia escenarios donde herramientas de este tipo podrían utilizarse como apoyo durante monitorización emocional, análisis psicológico o detección temprana de alteraciones relacionadas con salud mental a partir de la voz.

Capítulo 8. BIBLIOGRAFÍA

[1] s3programmerlead, Multimodal Dataset for Depression Analysis, Kaggle, 2023. https://www.kaggle.com/datasets/s3programmerlead/multimodal- dataset-for-depression-analysis

[2] J. Namkung et al., "Novel Deep Learning-Based Vocal Biomarkers for Stress Detection in Koreans" Psychiatry Investigation, vol. 21, no. 11, pp. 1228–1237, 2024. doi: 10.30773/pi.2024.0131.

[3] A. Kumar, M. A. Shaun, and B. K. Chaurasia, "Identification of psychological stress from speech signal using deep learning algorithm" e-Prime – Advances in Electrical Engineering, Electronics and Energy, vol. 9, 2024, Art. no. 100707, doi: 10.1016/j.prime.2024.100707.

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

El presente proyecto, además de su componente técnico y académico, se alinea con varios Objetivos de Desarrollo Sostenible (ODS), al situarse en la intersección entre salud, educación, innovación y desarrollo tecnológico.

Figura 47 - Objetivos de Desarrollo Sostenible

- **ODS 3: Salud y bienestar**

El proyecto contribuye a este objetivo mediante el desarrollo de una herramienta basada en el análisis automatizado de señales de voz que puede servir como apoyo al diagnóstico en el ámbito de la salud mental. La utilización de la voz como biomarcador no invasivo abre la posibilidad de facilitar la detección temprana de trastornos del estado de ánimo, complementando los métodos clínicos tradicionales y favoreciendo intervenciones más rápidas y personalizadas.

- **ODS 4: Educación de calidad**

La realización del proyecto implica la aplicación práctica de conocimientos en inteligencia artificial, procesamiento digital de señal y programación, promoviendo un aprendizaje técnico sólido y actualizado. La construcción de un modelo estructurado y reproducible puede servir como referencia para futuros estudiantes o investigadores interesados en este campo, fomentando una formación alineada con las demandas tecnológicas actuales.

- **ODS 8: Trabajo decente y crecimiento económico**

El proyecto fomenta la adquisición de competencias en áreas estratégicas como la inteligencia artificial y el análisis de datos, sectores con alta proyección profesional. Esto contribuye al desarrollo de perfiles técnicos cualificados y alineados con el crecimiento de industrias tecnológicas emergentes.

- **ODS 9: Industria, innovación e infraestructura**

El desarrollo de soluciones tecnológicas basadas en aprendizaje automático aplicadas al análisis biomédico impulsa la innovación y la integración de nuevas herramientas digitales en el ámbito de la investigación. Este tipo de proyectos fortalece la conexión entre el entorno académico y el desarrollo tecnológico, favoreciendo la creación de infraestructuras digitales orientadas al análisis avanzado de datos.

- **ODS 10: Reducción de las desigualdades**

El uso de herramientas automatizadas y potencialmente accesibles puede contribuir a reducir barreras en el acceso a evaluaciones preliminares en salud mental, especialmente en entornos con recursos limitados o menor disponibilidad de especialistas. De este modo, se promueve una mayor equidad en el acceso a tecnologías de apoyo sanitario.

ANEXO I I: CÓDIGO DESARROLLADO

- Creación de base de datos a partir del dataset de audio, extrayendo las 7 características escogidas.

```
import os
import sys
import numpy as np
import pandas as pd
import librosa

DATASET_DIR = "Dataset"
OUTPUT_CSV = "features_top7.csv"

def safe_stat(x, fn=np.nanmean):
    x = np.asarray(x, dtype=float)
    if x.size == 0:
        return 0.0
    v = fn(x)
    return float(v) if np.isfinite(v) else 0.0

def extract_features(path):
    y, sr = librosa.load(path, sr=None, mono=True)
    if y.size == 0:
        return None

    y = librosa.util.normalize(y)

    rms = librosa.feature.rms(y=y, frame_length=2048,
hop_length=512).flatten()
    energy_mean = safe_stat(rms, np.nanmean)
    energy_std  = safe_stat(rms, np.nanstd)

    f0 = librosa.yin(y, fmin=50, fmax=300, sr=sr, frame_length=2048,
hop_length=512)

    f0_voiced = f0[np.isfinite(f0)]
    pitch_mean = safe_stat(f0_voiced, np.nanmean)
    pitch_std  = safe_stat(f0_voiced, np.nanstd)

    if rms.size > 0:
        thr = max(np.percentile(rms, 20) * 0.5, 1e-6)
    else:
        thr = 1e-6

    is_sil = rms < thr
```

```
    pause_ratio = safe_stat(is_sil.astype(float),
np.nanmean)

    hop_length = 512
    frame_time = hop_length / sr

    pause_durations = []
    if is_sil.size > 0:
        run = 0
        for v in is_sil:
            if v:
                run += 1
            else:
                if run > 0:
                    pause_durations.append(run * frame_time)
                    run = 0
        if run > 0:
            pause_durations.append(run * frame_time)

    pause_mean_s = safe_stat(pause_durations, np.nanmean)

    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13, n_fft=2048,
hop_length=512)
    mfcc_delta = librosa.feature.delta(mfcc, order=1)

    mfcc_delta_mean_abs = safe_stat(np.abs(mfcc_delta).ravel(), np.nanmean)

    feats = {
        "pitch_mean_hz": pitch_mean,
        "pitch_std_hz": pitch_std,
        "energy_mean_rms": energy_mean,
        "energy_std_rms": energy_std,
        "pause_ratio": pause_ratio,
        "pause_mean_s": pause_mean_s,
        "mfcc_delta_mean_abs": mfcc_delta_mean_abs,
    }

    for k, v in feats.items():
        if not np.isfinite(v):
            feats[k] = 0.0

    return feats

def main(dataset_dir=DATASET_DIR, out_csv=OUTPUT_CSV):
    rows = []
    for root, _, files in os.walk(dataset_dir):
        for fname in files:
            if fname.lower().endswith(".wav"):
                fpath = os.path.join(root, fname)
                try:
                    feats = extract_features(fpath)
                    if feats is None:
                        continue
```

```
                    row = {"id": fname}
                    row.update(feats)
                    rows.append(row)
                except Exception:

                    continue

    cols = [
        "id",
        "pitch_mean_hz",
        "pitch_std_hz",
        "energy_mean_rms",
        "energy_std_rms",
        "pause_ratio",
        "pause_mean_s",
        "mfcc_delta_mean_abs",
    ]
    df = pd.DataFrame(rows, columns=cols)
    df.to_csv(out_csv, index=False)

if __name__ == "__main__":
    if len(sys.argv) >= 2:
        DATASET = sys.argv[1]
    else:
        DATASET = DATASET_DIR
    if len(sys.argv) >= 3:
        OUT = sys.argv[2]
    else:
        OUT = OUTPUT_CSV
    main(DATASET, OUT)
```

- Etiquetado de ‘0’, ‘1’ o ‘2’ a cada entrada de audio:

```
import os
import pandas as pd

csv_path = "features_top7.csv"
audio_dir = "Audio_Dataset"
output_csv = "dataset_etiq.csv"

df = pd.read_csv(csv_path)

if "id" not in df.columns:
    print("❌ Error: No se encontró la columna 'id' en el CSV.")
    print("Columnas encontradas:", df.columns.tolist())
    exit()

labels = []
```

```
for file_name in df["id"]:
    label = None
    for root, dirs, files in os.walk(audio_dir):
        if file_name in files:
            if "Normal" in root:
                label = 0
            elif "Stage1" in root:
                label = 1
            elif "Stage2" in root:
                label = 2
            break

    if label is None:
        label = -1
    labels.append(label)

df["label"] = labels

df.to_csv(output_csv, index=False, sep=";")
print(f"✅ Archivo generado correctamente: {output_csv}")
```

- Análisis del dataset.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.drop(columns=["label"])
y = dataset["label"]

label_map = {0: "Normal", 1: "Stage1", 2: "Stage2"}
features = X.columns.tolist()

print("=" * 60)
print("ANÁLISIS DEL DATASET")
print("=" * 60)

print("\n--- 1) BALANCE DEL DATASET ---")
counts = y.value_counts().sort_index()
for lbl, cnt in counts.items():
    pct = cnt / len(y) * 100
    print(f"  {label_map[lbl]} (label={lbl}): {cnt} muestras ({pct:.1f}%)")
print(f"  TOTAL: {len(y)} muestras")
```

```
plt.figure(figsize=(6, 4))
plt.bar([label_map[l] for l in counts.index], counts.values, color=["#4CAF50",
"#FF9800", "#F44336"])
plt.title("Distribución de clases")
plt.xlabel("Clase")
plt.ylabel("N° muestras")
plt.tight_layout()
plt.show()

print("\n--- 2) ESTADÍSTICAS GLOBALES POR FEATURE ---")
desc = X.describe().T
desc["cv"] = desc["std"] / desc["mean"]
print(desc[["mean", "std", "min", "25%", "50%", "75%", "max",
"cv"]].to_string())

print("\n--- 3) ESTADÍSTICAS POR CLASE ---")
for lbl in sorted(y.unique()):
    print(f"\n  [{label_map[lbl]}]")
    subset = X[y == lbl]
    print(subset.describe().T[["mean", "std", "min", "max"]].to_string())

print("\n--- 5) OUTLIERS (|z-score| > 3) ---")
z_scores = np.abs(stats.zscore(X))
outlier_counts = (z_scores > 3).sum(axis=0)
for feat, cnt in zip(features, outlier_counts):
    print(f"  {feat}: {cnt} outliers")

print("\n--- 6) MATRIZ DE CORRELACIÓN ---")
corr = X.corr()
print(corr.to_string())

plt.figure(figsize=(9, 7))
sns.heatmap(corr, annot=True, fmt=".2f", cmap="coolwarm", center=0,
square=True)
plt.title("Correlación entre features")
plt.tight_layout()
plt.show()

class_means = dataset.groupby("label")[features].mean()
class_means.index = [label_map[i] for i in class_means.index]

plt.figure(figsize=(10, 4))
sns.heatmap(class_means.T, annot=True, fmt=".3f", cmap="YlOrRd")
plt.title("Media de cada feature por clase")
plt.tight_layout()
plt.show()

fig, axes = plt.subplots(2, 4, figsize=(16, 8))
axes = axes.flatten()

for i, feat in enumerate(features):
    data_by_class = [X[y == lbl][feat].values for lbl in sorted(y.unique())]
```

```
    axes[i].boxplot(data_by_class, labels=[label_map[l] for l in
sorted(y.unique())])
    axes[i].set_title(feat)
    axes[i].grid(True)

if len(features) < len(axes):
    for j in range(len(features), len(axes)):
        axes[j].set_visible(False)

plt.suptitle("Boxplots por feature y clase", fontsize=13)
plt.tight_layout()
plt.show()

print("\n✅ Análisis completado.")
```

- Generación de los diagramas de *Kernel Density Estimation.*

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.drop(columns=["label"])
y = dataset["label"]

classes = sorted(y.unique())

palette = sns.color_palette("Set1", len(classes))

for feature in X.columns:
    plt.figure(figsize=(8, 5))

    for cls, color in zip(classes, palette):
        sns.kdeplot(
            data=dataset[dataset["label"] == cls],
            x=feature,
            fill=True,
            alpha=0.35,
            linewidth=2,
            label=f"Clase {cls}",
            color=color
        )

    plt.title(f"Distribución KDE de la variable: {feature}")
    plt.xlabel(feature)
    plt.ylabel("Densidad")
    plt.legend()
    plt.grid(True)
```

```
    plt.tight_layout()
    plt.show()
```

- Generación de gráficas para el análisis de *feature importance*.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from catboost import CatBoostClassifier

import shap
import catboost as cb

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

base_cb = CatBoostClassifier(
    loss_function="MultiClass" if y.nunique() > 2 else "Logloss",
    random_seed=42,
    verbose=0,
    allow_writing_files=False
)

param_dist = {
    "iterations": np.arange(500, 4001, 250),
    "depth": np.arange(3, 11),
    "learning_rate": np.array([0.01, 0.02, 0.03, 0.05, 0.08, 0.1, 0.15]),
    "l2_leaf_reg": np.array([1, 2, 3, 5, 7, 10, 15, 20]),
    "random_strength": np.array([0, 0.5, 1, 2, 5, 10]),
    "bagging_temperature": np.array([0, 0.25, 0.5, 1, 2, 5]),
    "border_count": np.array([32, 64, 128, 254]),
    "bootstrap_type": ["Bayesian", "Bernoulli", "MVS"],
    "subsample": np.array([0.6, 0.7, 0.8, 0.9, 1.0]),
}
```

```
rand_search = RandomizedSearchCV(
    estimator=base_cb,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)
best_cb = rand_search.best_estimator_

X_test_df = X_test.copy()

model_feature_names = best_cb.feature_names_
X_shap = X_test_df[model_feature_names]

shap_values = best_cb.get_feature_importance(
    data=cb.Pool(X_shap, label=y_test),
    type="ShapValues"
)

if shap_values.ndim == 3:
    shap_core = shap_values[:, :, :-1]
    shap_for_plot = np.mean(np.abs(shap_core), axis=1)
else:
    shap_for_plot = shap_values[:, :-1]

plt.figure()
shap.summary_plot(shap_for_plot, X_shap, show=False)
plt.tight_layout()
plt.show()

plt.figure()
shap.summary_plot(shap_for_plot, X_shap, plot_type="bar", show=False)
plt.tight_layout()
plt.show()
```

- Algoritmo Gradient Boosting

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from sklearn import metrics
from sklearn.metrics import (
    classification_report,
    confusion_matrix,
    ConfusionMatrixDisplay,
    roc_auc_score
)
from sklearn.ensemble import GradientBoostingClassifier

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

base_gb = GradientBoostingClassifier(random_state=42)

param_dist = {
    "n_estimators":    np.arange(100, 1001, 50),
    "max_depth":       np.arange(3, 11),
    "learning_rate":   np.array([0.01, 0.02, 0.03, 0.05, 0.08, 0.1, 0.15]),
    "subsample":       np.array([0.6, 0.7, 0.8, 0.9, 1.0]),
    "min_samples_split": np.array([2, 5, 10, 20]),
    "min_samples_leaf":  np.array([1, 2, 4, 8]),
    "max_features":    ["sqrt", "log2", None],
}

rand_search = RandomizedSearchCV(
    estimator=base_gb,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
```

```
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)

best_gb = rand_search.best_estimator_
print("\nBest hyperparameters:", rand_search.best_params_)
print("Best CV accuracy:", rand_search.best_score_)

best_gb.fit(X_train, y_train)

y_pred  = best_gb.predict(X_test)
y_proba = best_gb.predict_proba(X_test)

accuracy = metrics.accuracy_score(y_test, y_pred)

if y.nunique() > 2:
    auc_macro_ovr = roc_auc_score(y_test, y_proba, multi_class="ovr",
average="macro")
else:
    auc_macro_ovr = roc_auc_score(y_test, y_proba[:, 1])

print("\nTest Accuracy:", accuracy)
print("Test ROC AUC:", auc_macro_ovr)
print("\nClassification report:\n", classification_report(y_test, y_pred))

cm   = confusion_matrix(y_test, y_pred, labels=best_gb.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=best_gb.classes_)
disp.plot()
plt.show()
```

- Algoritmo Extreme Gradient Boosting

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from sklearn import metrics
from sklearn.metrics import (
    classification_report,
    confusion_matrix,
    ConfusionMatrixDisplay,
    roc_auc_score
)
```

```
from xgboost import XGBClassifier

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

n_classes = y.nunique()

base_xgb = XGBClassifier(
    random_state=42,
    n_jobs=-1,
    objective="multi:softprob" if n_classes > 2 else "binary:logistic",
    num_class=n_classes if n_classes > 2 else None,
    eval_metric="mlogloss" if n_classes > 2 else "logloss",
    tree_method="hist"
)

param_dist = {
    "n_estimators":      np.arange(200, 2001, 100),
    "max_depth":         np.arange(3, 13, 1),
    "learning_rate":     np.linspace(0.01, 0.3, 30),
    "subsample":         np.linspace(0.6, 1.0, 9),
    "colsample_bytree":  np.linspace(0.6, 1.0, 9),
    "min_child_weight":  np.arange(1, 11, 1),
    "gamma":             np.linspace(0.0, 0.5, 11),
    "reg_alpha":         np.linspace(0.0, 1.0, 11),
    "reg_lambda":        np.linspace(0.5, 3.0, 11),
}

rand_search = RandomizedSearchCV(
    estimator=base_xgb,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)
```

```
best_xgb = rand_search.best_estimator_
print("\nBest hyperparameters:", rand_search.best_params_)
print("Best CV accuracy:", rand_search.best_score_)

best_xgb.fit(X_train, y_train)

y_pred  = best_xgb.predict(X_test)
y_proba = best_xgb.predict_proba(X_test)

accuracy = metrics.accuracy_score(y_test, y_pred)

if n_classes > 2:
    auc_macro_ovr = roc_auc_score(y_test, y_proba, multi_class="ovr",
average="macro")
else:
    auc_macro_ovr = roc_auc_score(y_test, y_proba[:, 1])

print("\nTest Accuracy:", accuracy)
print("Test ROC AUC:", auc_macro_ovr)
print("\nClassification report:\n", classification_report(y_test, y_pred))

cm   = confusion_matrix(y_test, y_pred, labels=best_xgb.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=best_xgb.classes_)
disp.plot()
plt.show()
```

- Algoritmo KNN

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
from sklearn.metrics import (
    classification_report,
    confusion_matrix,
    ConfusionMatrixDisplay,
    roc_auc_score
)

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

base_knn = KNeighborsClassifier()

param_dist = {
    "n_neighbors":  np.arange(1, 51),
    "weights":      ["uniform", "distance"],
    "metric":       ["euclidean", "manhattan", "minkowski", "chebyshev"],
    "p":            [1, 2, 3],
    "leaf_size":    np.arange(10, 101, 10),
    "algorithm":    ["auto", "ball_tree", "kd_tree", "brute"],
}

rand_search = RandomizedSearchCV(
    estimator=base_knn,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)

best_knn = rand_search.best_estimator_
print("\nBest hyperparameters:", rand_search.best_params_)
print("Best CV accuracy:", rand_search.best_score_)

best_knn.fit(X_train, y_train)

y_pred  = best_knn.predict(X_test)
y_proba = best_knn.predict_proba(X_test)

accuracy = metrics.accuracy_score(y_test, y_pred)

if y.nunique() > 2:
    auc_macro_ovr = roc_auc_score(y_test, y_proba, multi_class="ovr",
average="macro")
else:
    auc_macro_ovr = roc_auc_score(y_test, y_proba[:, 1])

print("\nTest Accuracy:", accuracy)
print("Test ROC AUC:", auc_macro_ovr)
```

```
print("\nClassification report:\n", classification_report(y_test, y_pred))

cm   = confusion_matrix(y_test, y_pred, labels=best_knn.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=best_knn.classes_)
disp.plot()
plt.show()
```

- Algoritmo Random Forest

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.metrics import (
    classification_report,
    confusion_matrix,
    ConfusionMatrixDisplay,
    roc_auc_score
)

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

base_rf = RandomForestClassifier(
    random_state=42,
    n_jobs=-1
)

param_dist = {
    "n_estimators":        np.arange(300, 2501, 100),
    "max_depth":           [None] + list(np.arange(3, 61, 3)),
    "max_features":        ["sqrt", "log2", None, 0.3, 0.5, 0.7],
    "min_samples_split":   np.arange(2, 31, 2),
    "min_samples_leaf":    np.arange(1, 21, 1),
```

```
    "bootstrap":            [True, False],
    "class_weight":         [None, "balanced", "balanced_subsample"],
    "min_impurity_decrease": [0.0, 1e-4, 5e-4, 1e-3],
    "ccp_alpha":            np.linspace(0.0, 0.02, 21),
    "max_samples":          [None, 0.6, 0.7, 0.8,
0.9],
}

rand_search = RandomizedSearchCV(
    estimator=base_rf,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)

best_rf = rand_search.best_estimator_
print("\nBest hyperparameters:", rand_search.best_params_)
print("Best CV accuracy:", rand_search.best_score_)

best_rf.fit(X_train, y_train)

y_pred  = best_rf.predict(X_test)
y_proba = best_rf.predict_proba(X_test)

accuracy = metrics.accuracy_score(y_test, y_pred)

if y.nunique() > 2:
    auc_macro_ovr = roc_auc_score(y_test, y_proba, multi_class="ovr",
average="macro")
else:
    auc_macro_ovr = roc_auc_score(y_test, y_proba[:, 1])

print("\nTest Accuracy:", accuracy)
print("Test ROC AUC:", auc_macro_ovr)
print("\nClassification report:\n", classification_report(y_test, y_pred))

cm   = confusion_matrix(y_test, y_pred, labels=best_rf.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=best_rf.classes_)
disp.plot()
plt.show()
```

- Algoritmo Categorical Boosting

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from sklearn import metrics
from sklearn.metrics import (
    classification_report,
    confusion_matrix,
    ConfusionMatrixDisplay,
    roc_auc_score
)

from catboost import CatBoostClassifier

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)
cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

base_cb = CatBoostClassifier(
    loss_function="MultiClass" if y.nunique() > 2 else "Logloss",
    random_seed=42,
    verbose=0,
    allow_writing_files=False
)
param_dist = {
    "iterations": np.arange(500, 4001, 250),
    "depth": np.arange(3, 11),
    "learning_rate": np.array([0.01, 0.02, 0.03, 0.05, 0.08, 0.1, 0.15]),
    "l2_leaf_reg": np.array([1, 2, 3, 5, 7, 10, 15, 20]),
    "random_strength": np.array([0, 0.5, 1, 2, 5, 10]),
    "bagging_temperature": np.array([0, 0.25, 0.5, 1, 2, 5]),
    "border_count": np.array([32, 64, 128, 254]),
    "bootstrap_type": ["Bayesian", "Bernoulli", "MVS"],

    "subsample": np.array([0.6, 0.7, 0.8, 0.9, 1.0]),
}

rand_search = RandomizedSearchCV(
```

```
    estimator=base_cb,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)

best_cb = rand_search.best_estimator_
print("\nBest hyperparameters:", rand_search.best_params_)
print("Best CV accuracy:", rand_search.best_score_)

best_cb.fit(X_train, y_train)

y_pred = best_cb.predict(X_test)
y_pred = np.asarray(y_pred).reshape(-
1)
y_proba = best_cb.predict_proba(X_test)

accuracy = metrics.accuracy_score(y_test, y_pred)

if y.nunique() > 2:
    auc_macro_ovr = roc_auc_score(y_test, y_proba, multi_class="ovr",
average="macro")
else:
    auc_macro_ovr = roc_auc_score(y_test, y_proba[:, 1])

print("\nTest Accuracy:", accuracy)
print("Test ROC AUC:", auc_macro_ovr)
print("\nClassification report:\n", classification_report(y_test, y_pred))

cm = confusion_matrix(y_test, y_pred, labels=best_cb.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=best_cb.classes_)
disp.plot()
plt.show()

def plot_hyperparameter_vs_auc(param_name, values):
    scores = []
    for val in values:
        model = CatBoostClassifier(
            loss_function="MultiClass" if y.nunique() > 2 else "Logloss",
            random_seed=42,
            verbose=0,
            allow_writing_files=False,
            **{param_name: val}
        )
        model.fit(X_train, y_train)
```

```
        y_proba_local = model.predict_proba(X_test)

        if y.nunique() > 2:
            score = roc_auc_score(y_test, y_proba_local, multi_class="ovr",
average="macro")
        else:
            score = roc_auc_score(y_test, y_proba_local[:, 1])

        scores.append(score)

    plt.figure()
    plt.plot(list(values), scores)
    plt.grid()
    plt.xlabel(param_name)
    plt.ylabel("ROC AUC (macro, ovr)" if y.nunique() > 2 else "ROC AUC")
    plt.title(f"{param_name} vs ROC AUC")
    plt.show()

plot_hyperparameter_vs_auc("depth", range(3, 11))
plot_hyperparameter_vs_auc("iterations", range(500, 4001, 250))
plot_hyperparameter_vs_auc("learning_rate", [0.01, 0.02, 0.03, 0.05, 0.08,
0.1, 0.15, 0.2, 0.25, 0.3])
```

- Algoritmo Categorical Boosting + Módulo de predicción

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split, RandomizedSearchCV,
StratifiedKFold
from sklearn import metrics
from sklearn.metrics import (
    classification_report,
    confusion_matrix,
    ConfusionMatrixDisplay,
    roc_auc_score
)

from catboost import CatBoostClassifier

AUDIO_TO_PREDICT = "carlota.mp3"

dataset = pd.read_csv("dataset_etiq.csv", sep=";")
dataset = dataset.iloc[:, 1:]

X = dataset.loc[:, dataset.columns != "label"]
y = dataset["label"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
```

```
    test_size=0.2,
    random_state=42,
    stratify=y
)

cv = StratifiedKFold(n_splits=7, shuffle=True, random_state=42)

base_cb = CatBoostClassifier(
    loss_function="MultiClass" if y.nunique() > 2 else "Logloss",
    random_seed=42,
    verbose=0,
    allow_writing_files=False
)

param_dist = {
    "iterations": np.arange(500, 4001, 250),
    "depth": np.arange(3, 11),
    "learning_rate": np.array([0.01, 0.02, 0.03, 0.05, 0.08, 0.1, 0.15]),
    "l2_leaf_reg": np.array([1, 2, 3, 5, 7, 10, 15, 20]),
    "random_strength": np.array([0, 0.5, 1, 2, 5, 10]),
    "bagging_temperature": np.array([0, 0.25, 0.5, 1, 2, 5]),
    "border_count": np.array([32, 64, 128, 254]),
    "bootstrap_type": ["Bayesian", "Bernoulli", "MVS"],
    "subsample": np.array([0.6, 0.7, 0.8, 0.9, 1.0]),
}

rand_search = RandomizedSearchCV(
    estimator=base_cb,
    param_distributions=param_dist,
    n_iter=250,
    cv=cv,
    scoring="accuracy",
    n_jobs=-1,
    random_state=42,
    verbose=1,
    refit=True
)

rand_search.fit(X_train, y_train)

best_cb = rand_search.best_estimator_
print("\nBest hyperparameters:", rand_search.best_params_)
print("Best CV accuracy:", rand_search.best_score_)

best_cb.fit(X_train, y_train)

y_pred = best_cb.predict(X_test)
y_pred = np.asarray(y_pred).reshape(-1)
y_proba = best_cb.predict_proba(X_test)

accuracy = metrics.accuracy_score(y_test, y_pred)

if y.nunique() > 2:
```

```
    auc_macro_ovr = roc_auc_score(y_test, y_proba, multi_class="ovr",
average="macro")
else:
    auc_macro_ovr = roc_auc_score(y_test, y_proba[:, 1])

print("\nTest Accuracy:", accuracy)
print("Test ROC AUC:", auc_macro_ovr)
print("\nClassification report:\n", classification_report(y_test, y_pred))

import librosa

def safe_stat(x, fn=np.nanmean):
    x = np.asarray(x, dtype=float)
    if x.size == 0:
        return 0.0
    v = fn(x)
    return float(v) if np.isfinite(v) else 0.0

def extract_features_single(path):
    y, sr = librosa.load(path, sr=None, mono=True)
    if y.size == 0:
        raise ValueError("Empty audio")

    y = librosa.util.normalize(y)

    rms = librosa.feature.rms(y=y, frame_length=2048,
hop_length=512).flatten()
    energy_mean = safe_stat(rms, np.nanmean)
    energy_std = safe_stat(rms, np.nanstd)

    f0 = librosa.yin(y, fmin=50, fmax=300, sr=sr, frame_length=2048,
hop_length=512)
    f0_voiced = f0[np.isfinite(f0)]
    pitch_mean = safe_stat(f0_voiced, np.nanmean)
    pitch_std = safe_stat(f0_voiced, np.nanstd)

    thr = max(np.percentile(rms, 20) * 0.5, 1e-6) if rms.size > 0 else 1e-6
    is_sil = rms < thr
    pause_ratio = safe_stat(is_sil.astype(float), np.nanmean)

    frame_time = 512 / sr
    pause_durations = []
    run = 0
    for v in is_sil:
        if v:
            run += 1
        else:
            if run > 0:
                pause_durations.append(run * frame_time)
                run = 0
    if run > 0:
        pause_durations.append(run * frame_time)
```

```
    pause_mean_s = safe_stat(pause_durations, np.nanmean)

    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13, n_fft=2048,
hop_length=512)
    mfcc_delta = librosa.feature.delta(mfcc, order=1)
    mfcc_delta_mean_abs = safe_stat(np.abs(mfcc_delta).ravel(), np.nanmean)

    return pd.DataFrame([{
        "pitch_mean_hz": pitch_mean,
        "pitch_std_hz": pitch_std,
        "energy_mean_rms": energy_mean,
        "energy_std_rms": energy_std,
        "pause_ratio": pause_ratio,
        "pause_mean_s": pause_mean_s,
        "mfcc_delta_mean_abs": mfcc_delta_mean_abs,
    }])

X_audio = extract_features_single(AUDIO_TO_PREDICT)

pred_label = int(best_cb.predict(X_audio).ravel()[0])
pred_proba = best_cb.predict_proba(X_audio)[0]

pred_proba_fmt = [f"{p:.4f}" for p in pred_proba]
print("Class probabilities (decimals):", pred_proba_fmt)

print("\nPredicted label:", pred_label)
```

- Implementación de PCA + Clustering.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

from mpl_toolkits.mplot3d import Axes3D

dataset_raw = pd.read_csv("dataset_etiq.csv", sep=';')

features = dataset_raw.drop(['id', 'label'], axis=1)

obj_cols = features.select_dtypes(include=['object']).columns
if len(obj_cols) > 0:
    features[obj_cols] = features[obj_cols].apply(
        lambda s: s.astype(str).str.replace('.', '',
regex=False).str.replace(',', '.', regex=False)
    )
```

```
features = features.apply(pd.to_numeric, errors='coerce')
features = features.dropna()

y = dataset_raw.loc[features.index, 'label']
y_cat = y.astype('category')
y_codes = y_cat.cat.codes
label_names = list(y_cat.cat.categories)

print("Dataset shape (valid rows):", features.shape)
print("Labels:", label_names)
print("Label counts:\n", y.value_counts())

scaler = MinMaxScaler()
X_scaled = pd.DataFrame(
    scaler.fit_transform(features),
    columns=features.columns
)

pca_full = PCA()
pca_full.fit(X_scaled)

expl_var = pca_full.explained_variance_ratio_
cum_var = np.cumsum(expl_var)

print("\nExplained variance ratio per component:")
print(expl_var)

plt.figure()
x = np.arange(1, len(expl_var) + 1)
plt.bar(x, expl_var)
plt.plot(x, cum_var, marker='o', linestyle='--')
plt.title("PCA - Varianza explicada por componente (y acumulada)")
plt.xlabel("Componente principal")
plt.ylabel("Varianza explicada")
plt.xticks(x)
plt.ylim(0, 1.05)
plt.show()

n_components = 4
pca = PCA(n_components=n_components)
scores_pca = pca.fit_transform(X_scaled)

loadings = pd.DataFrame(
    pca.components_.T,
    index=features.columns,
    columns=[f"PC{i}" for i in range(1, n_components + 1)]
)

plt.figure()
plt.imshow(loadings.values, aspect='auto')
plt.title("PCA - Loadings (contribución de variables a PCs)")
plt.xlabel("Componentes")
plt.ylabel("Variables")
```

```
plt.xticks(np.arange(n_components), loadings.columns)
plt.yticks(np.arange(len(loadings.index)), loadings.index)
plt.colorbar(label="Loading")
plt.tight_layout()
plt.show()

for pc in loadings.columns:

    contrib = loadings[pc].abs().sort_values(ascending=False)
    top = contrib.head(min(10, len(contrib)))

    plt.figure()
    plt.barh(top.index[::-1], top.values[::-1])
    plt.title(f"PCA - Top variables por contribución absoluta en {pc}")
    plt.xlabel("Loading absoluto")
    plt.tight_layout()
    plt.show()

sse = []
k_for_elbow = range(1, 15)
for k in k_for_elbow:
    km = KMeans(n_clusters=k, n_init=10, random_state=42)
    km.fit(scores_pca)
    sse.append(km.inertia_)

plt.figure()
plt.plot(list(k_for_elbow), sse, marker='o', linestyle='--')
plt.title("KMeans - Elbow Method (sobre PCA scores)")
plt.xlabel("Número de clusters (k)")
plt.ylabel("SSE / Inertia")
plt.xticks(list(k_for_elbow))
plt.show()

silhouette_scores = []
k_range = range(2, 15)
for k in k_range:
    km = KMeans(n_clusters=k, n_init=10, random_state=42)
    labels_tmp = km.fit_predict(scores_pca)
    silhouette_scores.append(silhouette_score(scores_pca, labels_tmp))

best_k = list(k_range)[int(np.argmax(silhouette_scores))]
print("\nBest k by silhouette:", best_k)

plt.figure()
plt.plot(list(k_range), silhouette_scores, marker='o', linestyle='--')
plt.title("KMeans - Silhouette Method (sobre PCA scores)")
plt.xlabel("Número de clusters (k)")
plt.ylabel("Silhouette score")
plt.xticks(list(k_range))
plt.show()

k_final = 3
kmeans = KMeans(n_clusters=k_final, n_init=10, random_state=42)
```

```
clusters = kmeans.fit_predict(scores_pca)

print("\nCluster counts:",
dict(pd.Series(clusters).value_counts().sort_index()))
print("Inertia (SSE):", kmeans.inertia_)

plt.figure()
for cluster_id in np.unique(clusters):
    plt.scatter(
        scores_pca[clusters == cluster_id, 1],
        scores_pca[clusters == cluster_id, 0],
        label=f"Cluster {cluster_id}"
    )
plt.title("PCA - PC2 vs PC1 coloreado por clusters (KMeans)")
plt.xlabel("PC2")
plt.ylabel("PC1")
plt.legend(title="Clusters")
plt.show()

plt.figure()
for code, name in enumerate(label_names):
    mask = (y_codes.values == code)
    plt.scatter(scores_pca[mask, 1], scores_pca[mask, 0], label=str(name))
plt.title("PCA - PC2 vs PC1 coloreado por etiqueta real (label)")
plt.xlabel("PC2")
plt.ylabel("PC1")
plt.legend(title="Etiqueta real")
plt.show()

if scores_pca.shape[1] >= 3:
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    for cluster_id in np.unique(clusters):
        pts = scores_pca[clusters == cluster_id]
        ax.scatter(pts[:, 0], pts[:, 1], pts[:, 2], label=f"Cluster
{cluster_id}")
    ax.set_title("PCA 3D - PC1/PC2/PC3 coloreado por clusters")
    ax.set_xlabel("PC1")
    ax.set_ylabel("PC2")
    ax.set_zlabel("PC3")
    ax.legend(title="Clusters")
    plt.show()

cont = pd.crosstab(
    pd.Series(clusters, name="Cluster"),
    pd.Series(y.values, name="Etiqueta real")
)

print("\nContingency table (Cluster vs Etiqueta real):\n", cont)

plt.figure()
plt.imshow(cont.values, aspect='auto')
plt.title("Cluster vs Etiqueta real (tabla de contingencia)")
```

```
plt.xlabel("Etiqueta real")
plt.ylabel("Cluster")
plt.xticks(np.arange(cont.shape[1]), cont.columns, rotation=45, ha='right')
plt.yticks(np.arange(cont.shape[0]), cont.index)
plt.colorbar(label="N° muestras")

for i in range(cont.shape[0]):
    for j in range(cont.shape[1]):
        plt.text(j, i, str(cont.values[i, j]), ha='center', va='center')

plt.tight_layout()
plt.show()

df_profile = features.copy()
df_profile["Cluster"] = clusters

cluster_means = df_profile.groupby("Cluster").mean(numeric_only=True)
print("\nCluster means (original features):\n", cluster_means)

for cluster_id in cluster_means.index:
    plt.figure()
    vals = cluster_means.loc[cluster_id].values
    plt.bar(cluster_means.columns, vals)
    plt.title(f"Perfil del Cluster {cluster_id} (media por variable en espacio
original)")
    plt.xlabel("Variables")
    plt.ylabel("Media")
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

pc1 = scores_pca[:, 0]
pc2 = scores_pca[:, 1]

plt.figure()
for cluster_id in np.unique(clusters):
    plt.hist(pc1[clusters == cluster_id], bins=20, alpha=0.5, label=f"Cluster
{cluster_id}")
plt.title("Distribución de PC1 por cluster")
plt.xlabel("PC1")
plt.ylabel("Frecuencia")
plt.legend(title="Clusters")
plt.show()

plt.figure()
for cluster_id in np.unique(clusters):
    plt.hist(pc2[clusters == cluster_id], bins=20, alpha=0.5, label=f"Cluster
{cluster_id}")
plt.title("Distribución de PC2 por cluster")
plt.xlabel("PC2")
plt.ylabel("Frecuencia")
plt.legend(title="Clusters")
plt.show()
```