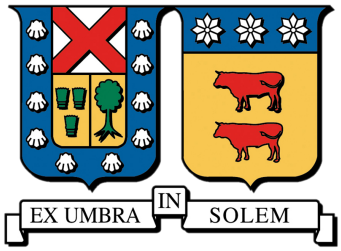
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE




# "CLASIFICACIÓN AUTOMÁTICA DE DISARTRIA COMO PREDIAGNÓSTICO. CASO *REFRACTED SPEECH*"

DIEGO SOTO CASTILLO

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN INFORMÁTICA

PROFESORA GUÍA: CAROLINA SAAVEDRA

JUNIO - 2025

# CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

## 1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

**Tipo de monografía (marcar una opción):** ☒ Memoria o trabajo de título; ☒ Tesis de Postgrado;

**Título del trabajo:** Clasificación Automática de Disartria como Prediagnóstico. Caso Refracted Speech

**Nombre del candidato(a):** Diego Nicolás Soto Castillo

**Carrera / Grado:** Ingeniería Civil Informática

**Campus:** Casa Central Valparaiso **;** **Departamento**: Departamento de Informática

## 2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Dra. Carolina Saavedra Ruiz, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

## 3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

## 4.- FIRMAS

**Profesor(a) guía o director(a) de memoria o tesis:**

**Fecha: 21/08/25** ; **Firma:** Saavedra

**Estudiante o Candidato(a):**

**Fecha:** 21/08/25 ; **Firma:** Diego

*Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.*

## Resumen

La disartria es un trastorno del habla provocado por afecciones neurológicas que compromete la comunicación verbal, dificultando la calidad de vida de las personas. Su detección temprana es fundamental para iniciar un tratamiento adecuado, pero el diagnóstico suele requerir evaluaciones clínicas especializadas. Este trabajo presenta un modelo de predicción de disartria basado en grabaciones de voz, como apoyo a un sistema de prediagnóstico automatizado. El objetivo es desarrollar un clasificador eficaz que se integre en la aplicación *Refracted Speech*, pensada para asistir en el seguimiento terapéutico. Para ello, se aplicaron técnicas de aprendizaje automático y extracción de características acústicas (MFCC, jitter, shimmer), evaluando distintos clasificadores. Los resultados obtenidos muestran que el modelo alcanza un rendimiento competitivo en métricas como precisión y sensibilidad, evidenciando su potencial como herramienta complementaria en contextos clínicos.

**Palabras clave:** disartria, aprendizaje automático, prediagnóstico

## Abstract

Dysarthria is a speech disorder caused by neurological conditions that compromise verbal communication, hindering people's quality of life. Early detection is essential for initiating appropriate treatment, but diagnosis often requires specialized clinical evaluations. This work presents a dysarthria prediction model based on voice recordings, as support for an automated prediagnosis system. The objective is to develop an effective classifier that is integrated into the *Refracted Speech* application, designed to assist in therapeutic follow-up. To this end, machine learning and acoustic feature extraction techniques (MFCC, jitter, shimmer) are applied, evaluating different classifiers. The results obtained show that the model achieves competitive performance in metrics such as precision and recall, demonstrating its potential as a complementary tool in clinical contexts.

**Keywords:** dysarthria, machine learning, prediagnosis

# Índice

## Introducción

La comunicación es un aspecto fundamental en la vida de las personas, ya que permite la interacción entre ellas y el intercambio de información, ideas y sentimientos. En este sentido, la comunicación verbal es una de las formas más comunes de comunicación, ya que es rápida e inmediata, también permite una mayor expresividad y claridad en el mensaje y los sentimientos que se quieren transmitir. Es por esto que las patologías del habla son un grave problema para las personas que las padecen, ya que afectan negativamente la capacidad para comunicarse verbalmente de forma efectiva.

Existen diferentes tipos de patologías del habla, como la dislalia, la disglosia, la disartria, entre otras. En este contexto, la disartria es una de las patologías del habla más comunes, de acuerdo con un estudio del Hospital Clínico de la Universidad de Chile [1], abarca un 54 % de los casos de trastornos de comunicación en Chile.

Un diagnóstico certero es clave para aplicar el tratamiento más adecuado según la condición del paciente, con el fin de recuperar, en la medida de lo posible, las capacidades comunicativas afectadas por el trastorno del habla.

Este trabajo se centra en el diseño e implementación de un modelo de predicción de la disartria, que utilice como datos de entrada grabaciones de voz, para poder ser incorporado en la aplicación móvil de *Refracted Speech* y que sirva como prediagnóstico y como acompañamiento al seguimiento terapéutico de esta aplicación.

Inicialmente se abordará una definición del problema a solucionar en este trabajo. Luego, se expondrá el marco conceptual asociado a la disartria y cómo se evalúa en la actualidad. Posteriormente, se detallará la propuesta y el diseño de la solución, tras lo cual se llevará a cabo su implementación. Finalmente, se evaluará el desempeño de dicha implementación.

## 1. Definición del problema

La disartria es un trastorno del habla que resulta de la debilidad o disfunción de los músculos involucrados en la producción de sonidos y palabras. Esta condición puede manifestarse por diversas causas, como las alteraciones en el sistema nervioso central o las afecciones que dañan los músculos faciales o de la garganta, como la parálisis facial o ciertas enfermedades neurológicas. Además, se caracteriza por afectar la fluidez y la claridad del habla. Los síntomas incluyen articulación dificultosa, lentitud del habla, cambios en el volumen y entonación de la voz, y falta de control sobre la pronunciación de ciertos sonidos. En muchos casos, estos síntomas pueden dar lugar a un habla entrecortada o "balbuceo", lo que dificulta la comunicación efectiva y puede generar frustración tanto en la persona afectada como en su entorno. Es por lo que el tratamiento y la terapia adecuados pueden ayudar a mejorar la calidad de vida de las personas afectadas y a facilitar su participación en actividades cotidianas y sociales.

La detección y el diagnóstico tempranos de la disartria son cruciales para iniciar un tratamiento adecuado y mejorar la calidad de vida de los afectados. Sin embargo, el diagnóstico tradicional basado en la evaluación clínica realizada por especialistas en fonoaudiología demanda tiempo y puede ser bastante subjetivo, ya que depende del criterio y la experiencia del especialista.

Por otro lado, a veces las personas evitan acudir a un especialista por diferentes razones, como la falta de tiempo, la lejanía con el centro de salud, falta de recursos económicos, entre otros. Según

un estudio hecho en Estados Unidos [2], las principales causas de no acudir a un médico son:

- Baja percepción de necesidad de atención médica (12.2%).
- Esperar a que sus síntomas mejoren con el tiempo (4%).
- Alto costo de la atención médica (24.1%).
- Falta de seguro médico (8.3%).
- Falta de tiempo (15.6%).

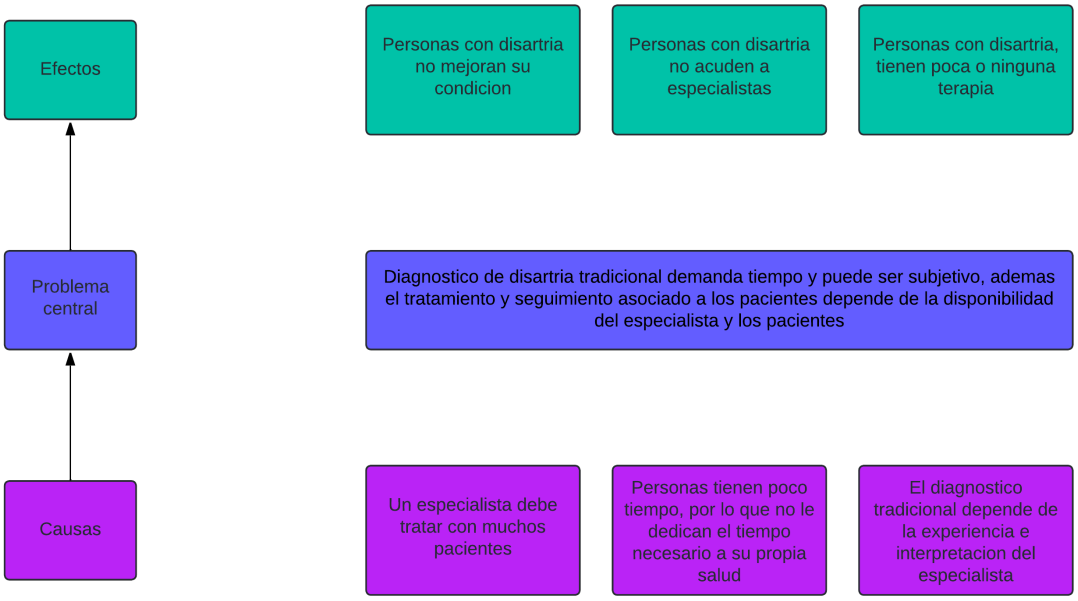



Figura 1. Árbol del problema

Como se sintetiza en la Figura 1, existen barreras de distinta índole que limitan el acceso de los pacientes a una recuperación oportuna. Frente a este contexto, el desarrollo de una herramienta digital accesible, como lo es una aplicación móvil, podría reducir estos obstáculos. Además, no solo se estaría incentivando a los usuarios a buscar una evaluación profesional posterior, sino que también podría integrarse como un apoyo al trabajo de los profesionales. Por esto, es de suma importancia que el sistema alcance un nivel de precisión alto, lo que reforzaría su utilidad para los pacientes y para los especialistas.

### 1.1. Objetivo General

Desarrollar un modelo de predicción de disartria a partir de grabaciones de voz, orientado a su uso como herramienta de apoyo al diagnóstico.

### 1.2. Objetivos específicos

- Implementar algoritmos de procesamiento y análisis de señales de audio para extraer información relevante utilizada en la clasificación de disartria.

- Comparar y seleccionar las características extraídas de las señales de audio más adecuadas para optimizar la entrada del modelo de predicción.
- Diseñar, entrenar y validar un modelo de predicción de disartria, verificando su desempeño para su uso como diagnóstico dentro del proyecto *Refracted Speech*.

## 2. Marco Conceptual

### 2.1. Disartria

La disartria se refiere a un grupo de trastornos motores del habla que resultan de una alteración en el control neuromuscular que afectan a la respiración, la fonación, la resonancia, la articulación y la prosodia [3].

1. **Respiración:** Es el proceso mediante el cual se proporciona el flujo de aire necesario para la producción del habla. Una persona que no puede controlar bien su respiración tendrá problemas para inhalar el aire necesario y/o para exhalarlo al momento de generar un sonido.
2. **Fonación:** Es la producción de sonido a través de la vibración de las cuerdas vocales en la laringe.
3. **Resonancia:** Se refiere a la amplificación y modificación del sonido producido por la laringe, a medida que pasa por las cavidades de la faringe, la boca y la nariz.
4. **Articulación:** Es el proceso por el cual la lengua, los labios, los dientes y el paladar modifican el sonido generado para producir los fonemas del lenguaje.
5. **Prosodia:** Consiste en los aspectos melódicos y rítmicos del habla, como la entonación, el acento, el ritmo y la velocidad.

Estos impedimentos del habla pueden provenir de daño en el sistema nervioso central o periférico, lo que produce debilidad, lentitud, incoordinación, alteración en el tono e inexactitud de los movimientos orales y vocales, lo que resulta en un habla con características anormales en calidad y una reducción en la inteligibilidad. La disartria en general está asociada a trastornos en el desarrollo, debido a daño cerebral, que puede ser causado por ejemplo por accidente cerebro vascular, una lesión en la cabeza o una enfermedad neurológica progresiva.

Existen métodos estandarizados para evaluar la capacidad del habla con el fin de diagnosticar la disartria. Los dos más utilizados son el *Frenchay Dysarthria Assessment (FDA)* [4] y el *Assessment of Intelligibility of Dysarthric Speech (AIDS)* [5]. *Frenchay Dysarthria Assessment* consiste en una tabla que contiene 4 categorías: reflejo, respiración, labios y mandíbula, donde además cada una de ellas contiene subcategorías específicas. Basándose en esta tabla (Figura 2), el especialista debe calificar cada subcategoría según su propia percepción del paciente, utilizando una escala que va desde la letra *a* (el mejor puntaje) hasta *e* (el peor). Por otro lado, *Assessment of Intelligibility of Dysarthric Speech* se centra en evaluar la inteligibilidad del habla del paciente.

Aunque existen estos métodos, junto con otros que pueden variar según el especialista o la institución médica donde se realiza la evaluación, todos dependen de la experiencia y habilidad del fonoaudiólogo o profesional a cargo, lo que introduce un alto grado de subjetividad en el proceso. Esta limitación, sumada a la falta de recursos, disponibilidad y tiempo por parte de algunos pacientes para someterse a una evaluación de este tipo, hace necesaria una solución que les brinde

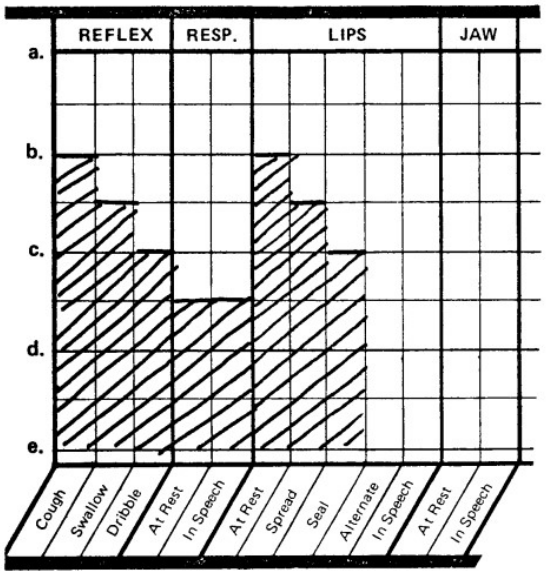



Figura 2. *Frenchay Dysarthria Assessment (FDA)* [4]

una preevaluación rápida y efectiva. Dicha herramienta podría incentivar a los pacientes a buscar un examen profesional más exhaustivo y adecuado.

En este trabajo se expone a *Refracted Speech*, una aplicación que permite detectar la disartria en pacientes rápida y efectivamente. La aplicación utiliza un modelo de aprendizaje automático que analiza la voz del paciente y determina si este padece de disartria, además ofrece una serie de ejercicios terapéuticos para mejorar la calidad del habla del paciente, permitiendo un seguimiento y tratamiento eficaz y donde el paciente puede realizar los ejercicios en el lugar y momento que más le acomode, apoyando al paciente y al especialista en la realización de la terapia.

Actualmente, existen soluciones (Tabla 1) que ofrecen algunas de estas características:

- **"Rehabla"** [6]: Esta es una aplicación que implementa detección y apoyo en forma de ejercicios, pero la página del producto no está operativa así que no se saben más detalles.
- **"Voice Online Lab"** [7]: Ofrece un diagnóstico biomecánico de la voz, para ello se tiene que enviar una grabación de audio, la cual es redirigida a un laboratorio donde es analizada por profesionales, para ello se requiere de una suscripción anual de 120 € o 650 € para la suscripción institucional para clínicas, hospitales, etc.
- **"Stutters Stars"** [8]: Esta aplicación es un juego enfocado a niños con tartamudeo, aunque no está enfocada en la disartria, esta es la App más completa que se encontró con ejercicios de ayuda para personas con problemas del habla. La suscripción cuesta $20 USD mensuales y tiene ofertas por 3 meses o un año.

### 2.2. Señales de Audio

Una señal de audio se refiere a la conversión de lo que sería una onda de presión (o sonido), a una señal eléctrica usando un micrófono. Las frecuencias en las que se miden estas ondas suelen estar

Tabla 1. Comparación de aplicaciones

| **Característica** | **Rehabla** | **Voice Online Lab** | **Stutters Stars** | ***Refracted Speech*** |
|---|---|---|---|---|
| Detección automática de disartria | No se sabe | Si (Requiere respuesta de laboratorio) | No | Si |
| Ejercicios terapéuticos | Si | No | Si (para tartamudeo) | Si |
| Disponibilidad | No | App móvil (requiere micrófono autorizado) | App móvil | App móvil |
| Seguimiento y reevaluación | Si, pero requiere de un especialista | No | Si | Si |

acotadas por el rango de frecuencias audibles por el oído de los seres humanos, entre los 20 Hz y los 20000 Hz. A pesar de esto, una señal de audio que represente sonido generado por voz humana no suele tener información relevante en frecuencias por encima de 10000 Hz. En la Figura 3 se puede apreciar una señal de audio perteneciente a la base de datos TORGO [9].

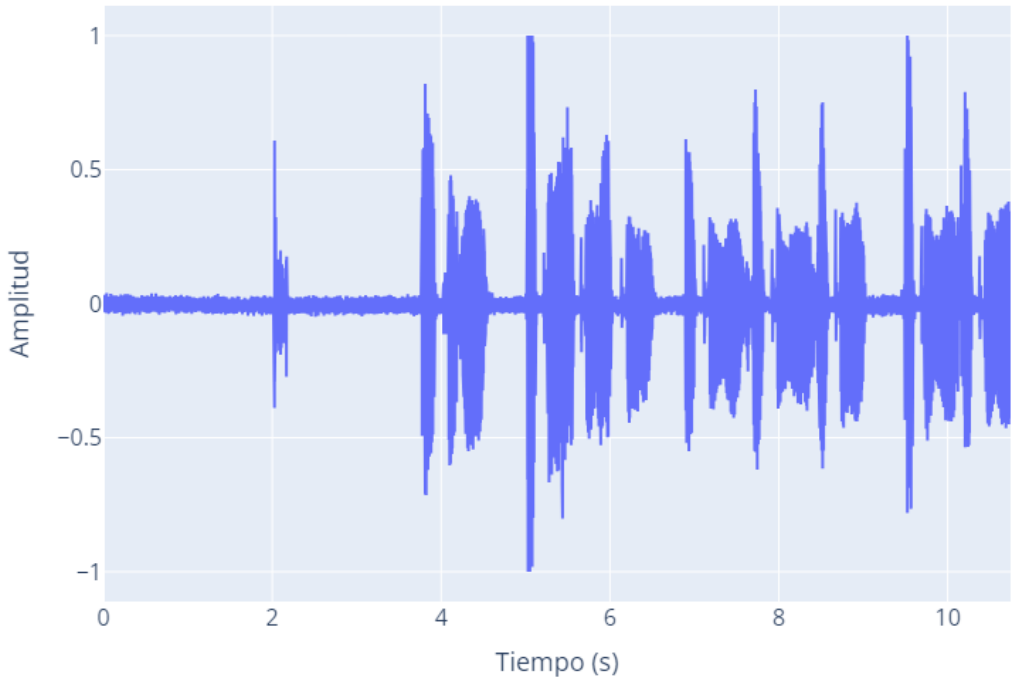



Figura 3. Representación visual de señal de audio. Ejemplo de *TORGO* [9], generada en *Python*.

### 2.2.1. Teorema de Muestreo de Nyquist-Shannon

Algo a tener en consideración al momento de trabajar con cualquier tipo de señal, es el Teorema de Muestreo de Nyquist-Shannon, el cual fue propuesto por Nyquist en 1928 [10] y demostrado por Shannon en 1949 [11]. Este teorema establece que para la representación precisa de una señal $s(t)$ a través muestras temporales $s(nT)$, se deben cumplir dos condiciones:

- La señal $s(t)$ debe tener un ancho de banda límite, o en otras palabras debe estar acotado a tener frecuencias que no sobrepasen la frecuencia máxima $f_{max}$.
- La tasa de muestreo $f_s$ de la señal $s(t)$ debe ser al menos el doble de la máxima frecuencia $f_{max}$, es decir $f_s \geq 2f_{max}$. Por ejemplo, si se quiere medir una señal de audio hasta la máxima frecuencia $f_{max}$ que el humano puede oír (20000 Hz), entonces nuestra frecuencia de muestreo $f_s$ debe ser de al menos 40000 Hz.

Este teorema es especialmente importante al trabajar con señales de audio, pues si sabemos la frecuencia de muestreo de cualquier señal con la que se trabaje, entonces sabemos cuál es la frecuencia máxima con la que podemos trabajar, ya sea para transformar, procesar o extraer información de la señal.

### 2.2.2. Transformada de Fourier

La *Transformada de Fourier* fue introducida por Joseph Fourier en 1822 [12], es empleada para transformar señales entre el dominio del tiempo y el dominio de las frecuencias, tiene numerosas aplicaciones en distintas áreas como la matemática, la física y la ingeniería. Además, es reversible, por lo que podemos pasar del dominio del tiempo al dominio de la frecuencia y viceversa.

Si $x(t)$ es una señal estacionaria y $\omega = 2\pi f$, su *Transformada de Fourier* está definida por:

$$F\{x(t)\} = \mathcal{F}_x(\omega) = \langle x(t), e^{-i\omega t} \rangle = \int_{-\infty}^{\infty} x(t) e^{-i\omega t} dt \tag{1}$$

Luego la señal $x(t)$ puede ser recuperada mediante la inversa de la *Transformada de Fourier*, dada por:

$$x(t) = \int_{-\infty}^{\infty} \mathcal{F}_x(w) e^{i\omega t} d\omega \tag{2}$$

Como esta definición de la transformada está definida solo para señales estacionarias, o en otras palabras, señales donde la frecuencia es la misma para cualquier instante $t$, esta transformación tiene 2 grandes desventajas:

1. Ausencia de información sobre la evolución de las frecuencias en señales no estacionarias.
2. No sirve para describir señales que no son continuas

La necesidad de una transformación que conservara las ventajas de la *Transformada de Fourier*, pero sin sus desventajas, motivó al desarrollo de una versión modificada. Esta modificación dio origen a *Short Time Fourier Transform (STFT)*, si $s(t)$ representa una señal arbitraria y $w(\tau)$ una función ventana, entonces su *STFT* se define como:

$$STFT\{s(t)\}(\tau,\omega) = \int_{-\infty}^{\infty} s(t)w(t-\tau)e^{-i\omega t}dt \tag{3}$$

La función ventana se "mueve" a lo largo del eje del tiempo, por lo que en simples palabras es como si se calculara una *Transformada de Fourier* por cada sección que toma la función ventana. También se dice en simples palabras que la *STFT* es esencialmente una *Transformada de Fourier* multiplicada por la función ventana, por eso también se llama *Windowed Fourier Transform*.

Para el caso donde el tiempo no es continuo, sino que discreto, como lo es el caso de una señal obtenida digitalmente donde la cantidad de muestras que se tienen por segundo está limitada por la frecuencia de muestreo utilizada en la medición. Existe una versión discreta de *Short Time Fourier Transform (STFT)* dada por:

$$STFT\{s[n]\}(m,\omega) = \sum_{n=0}^{N-1} s[n]w[n-m]e^{-i\omega n} \tag{4}$$

En este caso, $m$ es discreta y $\omega$ es continua. Esta forma de la $STFT$ es la más utilizada hoy en día, especialmente cuando se habla de cálculos computacionales. Esta transformación es la más utilizada en el procesamiento de señales, teniendo usos en áreas como análisis espectral de la señal, extracción de características, reducción de ruido, entre otros.

## 2.3. Características de Audio

Las características que pueden ser extraídas de una señal de audio pueden ser clasificadas en dos tipos: características de audio y de imagen. En el caso de las características de imagen, cualquier información extraída que tenga dos o más dimensiones (como *espectrogramas* o representaciones tiempo-frecuencia), cae en esta categoría. Por otro lado, las características de audio se refieren en general a parámetros unidimensionales derivados directamente de la señal en el dominio temporal o espectral, como el *pitch*. Las características de audio permiten capturar propiedades físicas o perceptibles de la señal.

### 2.3.1. *Mel-Frecuency Cepstral Coefficients (MFCC)*

Los *Mel-Frecuency Cepstral Coefficients (MFCC)* son ampliamente utilizados en el análisis de señales de voz, fueron introducidos por Davis y Mermelstein en 1980 [13], y han sido utilizados desde entonces en el área de *Automatic Speech Recognition (ASR)*. Su principal ventaja es que para obtenerlos es necesario transformar las frecuencias de la señal a la Escala de Mel, la cual representa mejor las frecuencias más bajas que son las que más varían en el habla humana.

En [14] se utilizan los *MFCC* para entrenar un clasificador binario, donde el valor 0 quiere decir que el paciente tiene disartria con severidad baja-media (0-50 %) y el valor 1 quiere decir disartria de severidad media-alta (51-100 %). Utilizan una metodología no convencional, que consiste en tomar los datos ya etiquetados y realizar un segundo etiquetado "débil" pasando las señales por una serie de reglas basadas en la energía de la señal, luego entrenan 2 clasificadores, uno con las etiquetas reales y otro con las etiquetas "débiles", para finalmente combinar los resultados de ambos clasificadores y obtener una salida final. Los resultados son bastantes buenos, obteniendo hasta un 99 % de accuracy en la tarea de diferenciar entre severidad de disartria de 0-50 % o 51-100 %.

Otro trabajo donde se utilizan los *MFCC* para entrenar un clasificador automático de disartria es el caso de [15], aquí se utilizan los coeficientes junto a un conjunto de características adicionales, algunas de estas características son la media, la mediana, el mínimo, el máximo, la desviación estándar, el primer y el tercer cuartil de la *Frecuencia Fundamental (F0)*, número de sílabas por segundo, número de sílabas sin pausas por segundo, entre otros. El clasificador utilizado no es del tipo binario, sino que clasifica los ejemplos en distintos niveles de severidad de disartria (sano, leve, moderada y severa). El mejor resultado obtenido usando una Red Neuronal para la clasificación, alcanzando un *F1-Score* de 86.23 %, 82.11 %, 57.14 % y 68.02 % para los casos sanos, con disartria leve, moderada y severa respectivamente.

Los *MFCC* se calculan de la siguiente forma:

1. **Aplicar *Short Time Fourier Transform (STFT)* discreta**: Se calcula $STFT\{s[n]\}$, este proceso puede ser dividido en los siguientes pasos:
   *a*) **Dividir la señal en cuadros de tiempo**: Se divide la señal en cuadros de tiempo de largo $l$ (usualmente 25 ms) con desplazamientos de largo $u$ (usualmente de 10 ms). La señal original en el dominio del tiempo será $s(n)$, una vez dividida en cuadros $s_i(n)$ donde $n$ se mueve en el número de muestras por cada cuadro e $i$ se mueve por el número de cuadros.
   *b*) **Aplicar la Transformada de Fourier**: Se aplica la Transformada de Fourier Discreta (DFT) a cada cuadro:

$$S_i(k) = \sum_{n=1}^{N} s_i(n)h(n)e^{-2j\pi kn/N} \qquad 1 \leq k \leq N \tag{5}$$

   Donde, $h(n)$ es la ventana aplicada al cuadro, $k$ es el largo de la DFT y $N$ es el número de muestras por cuadro. Normalmente se utiliza la ventana de Hamming para $h(n)$ y se aplica una Fast Fourier Transform (FFT) con 512 puntos ($N$), pero solo usamos los primeros 257 puntos, ya que el espectro es simétrico.
2. **Calcular la Potencia Espectral**: Se obtiene la estimación espectral de potencia basada en periodograma por cada $s_i(n)$, de la siguiente forma:

$$P_i(k) = \frac{1}{N}|S_i(k)|^2 \tag{6}$$

3. **Aplicar el Mel-Spaced Filterbank**: Esto es un conjunto de filtros triangulares de tamaño $r$ (usualmente entre 20 o 40 filtros) aplicados al espectro de potencia basada en periodograma, los filtros vienen en forma de $r$ vectores de largo 257, cada uno de estos vectores se multiplica con los espectros de potencia y se suman los coeficientes obteniendo $r$ energías por cuadro. La fórmula para obtener los coeficientes es la siguiente:
   *a*) Definir una límite inferior y superior para las frecuencias de los filtros, normalmente se elige 300 Hz y 8000 Hz considerando una señal de 16 kHZ, se convierten a la escala de Mel con la siguiente fórmula:

$$M(f) = 1125\ln(1 + \frac{f}{700}) \tag{7}$$

*b*) Obtener puntos equidistantes en la escala de Mel entre el límite inferior y superior, necesitamos $r+2$ puntos para definir los $r$ filtros, llamaremos a estos puntos $m(i)$.

*c*) Convertir los puntos de la escala de Mel a la escala de frecuencia con la siguiente fórmula:

$$f = 700(\exp(m/1125) - 1) \tag{8}$$

*d*) Como no tenemos la resolución de frecuencia exacta para los puntos obtenidos $h(i)$, se redondean a los puntos más cercanos en la FFT:

$$f(i) = \left\lfloor \frac{513 \times h(i)}{16\text{kHz}} \right\rfloor \tag{9}$$

*e*) Ahora creamos nuestros $r$ filtros triangulares (Figura 4) con la siguiente fórmula:

$$H_m(k) = \begin{cases} 0 & \text{if } k < f(m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & \text{if } f(m-1) \leq k \leq f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & \text{if } f(m) \leq k \leq f(m+1) \\ 0 & \text{if } k > f(m+1) \end{cases} \tag{10}$$

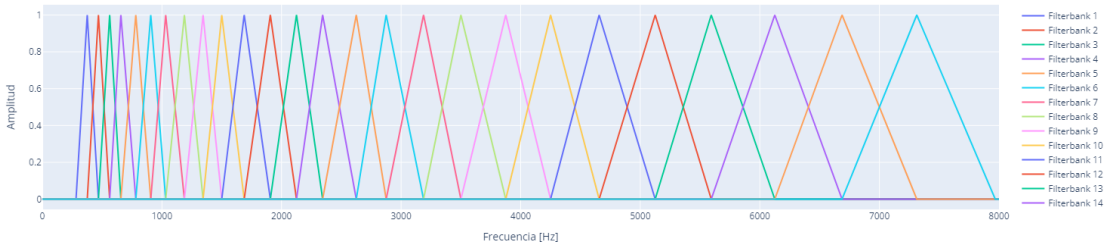



Figura 4. 26 Filtros Mel-Espaciados, Generado en *Python*

*f*) Aplicamos los filtros a los espectros de potencia:

$$E_i(m) = \sum_{k=1}^{257} H_m(k) \times P_i(k) \tag{11}$$

Donde, $E_i(m)$ es la energía del cuadro $i$ en el filtro $m$.

4. **Aplicar el Logaritmo**: Se aplica el logaritmo a las energías obtenidas en el paso anterior.
5. **Aplicar la Transformada Discreta de Coseno (DCT)**: Se aplica la DCT a los logaritmos de las energías, luego se escoge un número de coeficientes a considerar, para *Automatic Speech Recognition (ASR)*, se utilizan los primeros 12 coeficientes de la DCT, estos son los MFCC.
6. **Calcular los Deltas y Delta-Deltas**: Los deltas y delta-deltas son las derivadas de primer y segundo orden de los MFCC, también son conocidos como velocidades y aceleraciones de

los MFCC y nos dan información sobre la dinámica de los MFCC. Para calcular los deltas se utiliza la siguiente fórmula:

$$d_t = \frac{\sum_{n=1}^{N} n \times (c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2} \tag{12}$$

Donde, $d_t$ es el coeficiente delta en tiempo $t$ calculado en términos de los coeficientes estáticos $c_{t+n}$ y $c_{t-n}$, un valor típico para $N$ es 2. Los delta-deltas se calculan de la misma forma pero usando los coeficientes delta en vez de los coeficientes estáticos.

El resultado final corresponde a un vector con 36 coeficientes, compuesto por:

- 12 coeficientes *MFCC* (que capturan las características espectrales estáticas).
- 12 coeficientes delta (que describen la variación temporal de los *MFCC*).
- 12 coeficientes delta-delta (que reflejan la aceleración de dicha variación).

En la Figura 5, estos coeficientes se visualizan mediante un mapa de calor, donde la intensidad del color corresponde a la magnitud de cada coeficiente a lo largo del tiempo. Si se habla de los *MFCC* como tal, valores altamente positivos indican una alta correlación con el patrón del filtro correspondiente, y por el contrario, valores altamente negativos indican una baja correlación, esto es debido al uso del logaritmo en la obtención de estos coeficientes.

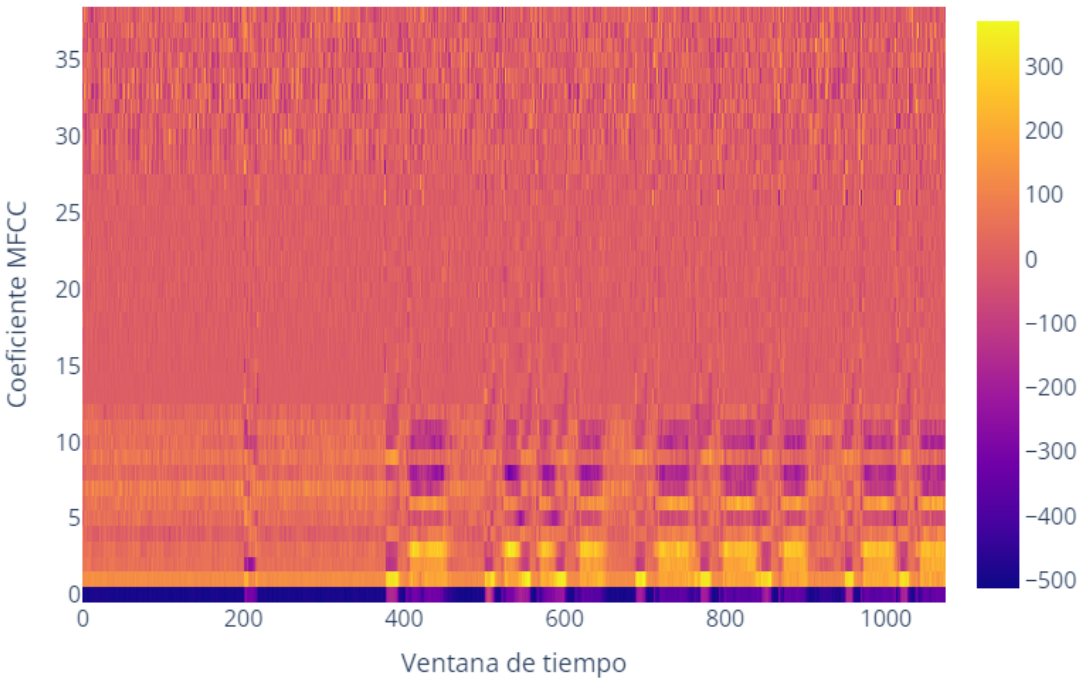



Figura 5. Mapa de calor de los 36 coeficientes por cada ventana de tiempo, correspondiente a la señal de la Figura 3. Generado en *Python*

### 2.3.2. *Frecuencia Fundamental (F0)*

La *Frecuencia Fundamental (F0)* [15, 16, 17] representa la tasa de vibración de las cuerdas vocales y se mide en Hertz (Hz). En términos musicales, corresponde al tono que se percibe de una nota. También corresponde al inverso del periodo entre dos cierres consecutivos de las cuerdas vocales:

$$F0_i = \frac{1}{Ti} \tag{13}$$

Teniendo así para una señal de audio una lista de $F0$ de largo $N$ que corresponde al número de periodos. Es común calcular diferentes estadísticos con esta lista, como el caso de [15], que como se mencionó anteriormente en 2.3.1, se hizo el cálculo de la media, la mediana, desviación estándar y otras medidas de los valores de $F0$.

### 2.3.3. *Jitter*

El Jitter [15, 16, 17] corresponde a una medida de la variación de la *Frecuencia Fundamental*, es también llamada "Perturbación de la Frecuencia". Este valor es sensible a los cambios de frecuencia y periodo, una voz normal tendrá un bajo nivel de inestabilidad, lo que se traduce en bajos valores de *Jitter*. En la práctica existen 4 parámetros relacionados con estas perturbaciones, *Absolute Jitter*, *Local Jitter*, *Five-Point Period Perturbation (PPQ5)* y *Relative Average Perturbation (RAP)*.

Si $T_i$ es la duración del i-ésimo periodo y $N$ el número total de periodos, tenemos las siguientes definiciones:

- ***Absolute Jitter:*** Representa el promedio de la diferencia absoluta entre periodos consecutivos.

$$Absolute\ Jitter = \frac{1}{N-1}\sum_{i=1}^{N-1}|T_i - T_{i-1}| \tag{14}$$

- ***Local Jitter:*** Representa el promedio de la diferencia absoluta entre 2 periodos consecutivos dividida por el periodo promedio, en forma de porcentaje.

$$Local\ Jitter = \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}|T_i - T_{i-1}|}{\frac{1}{N}\sum_{i=1}^{N}T_i} \times 100 \tag{15}$$

- ***Five-Point Period Perturbation (PPQ5):*** Corresponde a la razón de perturbación dentro de 5 periodos.

$$PPQ5 = \frac{\frac{1}{N-1}\sum_{i=2}^{N-2}\left|T_i - \left(\frac{1}{5}\sum_{n=i-2}^{i+2}T_n\right)\right|}{\frac{1}{N}\sum_{i=1}^{N}T_i} \times 100 \tag{16}$$

- ***Relative Average Perturbation (RAP):*** Es igual al promedio de la perturbación, es decir, la diferencia absoluta de promedio de un periodo y el promedio del periodo con sus 2 vecinos, dividido por el periodo promedio.

$$RAP = \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}\left|T_i - \left(\frac{1}{3}\sum_{n=i-1}^{i+1}T_n\right)\right|}{\frac{1}{N}\sum_{i=1}^{N}T_i} \times 100 \tag{17}$$

### 2.3.4. *Shimmer*

El *Shimmer* [15, 16, 17] es una medida similar al *Jitter*, pero con la diferencia de que en vez de considerar los periodos, *Shimmer* toma en cuenta las máximas amplitudes de estos periodos. Existen 4 parámetros que miden las perturbaciones, en este caso de las amplitudes, de distinta forma, *Absolute Shimmer (dB)*, *Local Shimmer*, *Three-Point Amplitude Perturbation Quotient (APQ3)*, *Five-Point Amplitude Perturbation Quotient (APQ5)*.

- ***Absolute Shimmer (dB):*** Representa el promedio de la diferencia absoluta del logaritmo del cociente de dos amplitudes de periodos consecutivos.

$$\textit{Absolute Shimmer (dB)} = \frac{1}{N-1}\sum_{i=1}^{N-1}\left|20\times\log\left(\frac{A_{i+1}}{A_i}\right)\right| \tag{18}$$

- ***Local Shimmer:*** Representa el promedio de la diferencia absoluta entre las amplitudes de periodos consecutivos, dividida por la amplitud promedio.

$$\textit{Local Shimmer} = \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}|A_i - A_{i+1}|}{\frac{1}{N}\sum_{i=1}^{N}A_i}\times 100 \tag{19}$$

- ***Three-Point Amplitude Perturbation Quotient (APQ3):*** Representa el cociente de la perturbación de la amplitud dentro de 3 periodos.

$$APQ3 = \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}\left|A_i - \left(\frac{1}{3}\sum_{n=i-1}^{i+1}A_n\right)\right|}{\frac{1}{N}}\times 100 \tag{20}$$

- ***Five-Point Amplitude Perturbation Quotient (APQ5):*** Representa la razón de la perturbación de la amplitud dentro de 5 periodos.

$$APQ5 = \frac{\frac{1}{N-1}\sum_{i=2}^{N-2}\left|A_i - \left(\frac{1}{5}\sum_{n=i-2}^{i+2}A_n\right)\right|}{\frac{1}{N}\sum_{i=1}^{N}A_i}\times 100 \tag{21}$$

### 2.3.5. *Harmonic-To-Noise Ratio (HNR)*

El *Harmonic-To-Noise Ratio (HNR)* [16, 18] es una medida que compara la energía armónica (componente periódica) con la energía del ruido (componente aperiódica) presente en una señal de voz. Esta relación se expresa en decibelios (dB) y refleja la eficiencia con la que el aire expulsado por los pulmones es transformado en vibraciones periódicas por las cuerdas vocales.

Un valor alto de *HNR* está asociado a una voz más armónica y sonora, mientras que un valor bajo indica la presencia de ruido glotal (ruido generado al generar sonidos con las cuerdas vocales), lo que puede evidenciar una voz disfónica. Generalmente, valores menores a 7 dB pueden considerarse indicativos de una voz patológica [18].

El cálculo de *HNR* se basa en la función de autocorrelación de la señal de voz. Si $ACV(0)$ representa el valor máximo de autocorrelación en el retardo cero y $ACV(T)$ el valor en el retardo

correspondiente al periodo fundamental $T$, entonces el valor de $HNR$ se obtiene mediante la siguiente expresión:

$$HNR = 10 \cdot \log_{10} \left( \frac{ACV(T)}{ACV(0) - ACV(T)} \right) \tag{22}$$

### 2.4. Modelos de Clasificación Automática

En el ámbito de la detección de patologías de la voz, existen numerosos modelos de clasificación que han sido ampliamente utilizados debido a su capacidad para identificar patrones en las señales acústicas. Entre estos destacan enfoques tradicionales, como *Support Vector Machine (SVM)*, *Random Forest* o *K-Nearest Neighboors*, así como técnicas más avanzadas basadas en aprendizaje profundo (Deep Learning), las cuales han demostrado un alto desempeño en tareas de discriminación entre voces sanas y patológicas. La elección de modelos a utilizar en este trabajo está basada en la comparación hecha en [19], aquí se realiza una comparación del desempeño de distintos modelos en trabajos previos.

#### 2.4.1. *Support Vector Machine (SVM)*

El *Support Vector Machine (SVM)* es un algoritmo de aprendizaje supervisado utilizado tanto para clasificación como para regresión. Su principio fundamental consiste en encontrar un hiperplano óptimo en un espacio de características que maximice el margen de separación entre clases diferentes. *SVM* es particularmente efectivo en problemas de clasificación binaria y ha demostrado ser robusto en escenarios con alta dimensionalidad, lo que lo hace especialmente útil para el análisis de señales acústicas.

Una de las ventajas clave de *SVM* es su capacidad para manejar datos no linealmente separables mediante el uso de funciones kernel, que transforman los datos originales a un espacio de mayor dimensionalidad donde sí pueden ser separados linealmente. Kernels comunes incluyen el lineal, polinomial y de base radial (RBF). Además, *SVM* es menos propenso al sobreajuste en comparación con otros algoritmos, gracias a su enfoque de maximización del margen.

En [15] uno de los modelos usados es *Support Vector Machine (SVM)*, aunque como se mencionó anteriormente en 2.3.1, este no fue el que obtuvo mejores resultados, alcanzando un *accuracy* de 71 % en promedio para todos los niveles de severidad.

Una metodología distinta es usada en [20], en este caso la tarea propuesta es clasificar los ejemplos en severidad baja-media o media-alta, para luego estimar el nivel de inteligibilidad a través de una regresión lineal. La tarea de clasificación usando una *SVM* alcanzó un accuracy del 75.75 %.

Otro ejemplo de clasificación binaria utilizando *SVM* es el caso de [21], donde para la tarea de detectar disartria, se obtuvo un *F1-Score* de 82.9 % y un *Accuracy* de 82.3 %, los cuales son bastante altos considerando que la media de *accuracy* obtenida usando *SVM* es alrededor de 80 % [19] en los trabajos previos de clasificación automática de disartria.

#### 2.4.2. *Random Forest*

El *Random Forest* (Bosque Aleatorio) es un algoritmo de aprendizaje supervisado ampliamente utilizado para clasificación y regresión. Su principio fundamental se basa en la construcción de

múltiples árboles de decisión durante el entrenamiento, combinando sus resultados para mejorar la precisión y robustez del modelo. *Random Forest* es especialmente efectivo en problemas con alta dimensionalidad y relaciones no lineales entre variables, lo que lo hace ideal para el análisis de señales acústicas y otros dominios complejos. Además, el algoritmo es inherentemente resistente al sobreajuste, ya que la agregación de múltiples árboles (mediante promediado o votación) reduce la varianza del modelo.

En [15] *Random Forest* fue uno de los modelos entrenados para clasificar la disartria en distintos niveles de severidad, utilizando una combinación de *MFCC* con otras características, se obtiene un *accuracy* de 70.1 % en promedio para los distintos niveles de seguridad.

Otro ejemplo es en [22], donde usando una gran variedad de características, desde las que son ampliamente utilizadas como *MFCC*, *F0* o *Jitter*, a otras que no son mencionadas en muchos estudios como *Spectral Entropy*, *Band Energy*, *Flux*, entre otras. En este caso, *Random Forest* fue el que obtuvo el mejor resultado, con un 95.8 % de *accuracy*.

### 2.4.3. *XGBoost (eXtreme Gradient Boosting)*

*XGBoost* es un algoritmo de aprendizaje supervisado basado en gradient boosting, es ampliamente usado por su alto rendimiento en problemas de clasificación y regresión. A diferencia de métodos que entrenan modelos en paralelo como *Random Forest*, *XGBoost* construye árboles de decisión secuencialmente, donde cada nuevo árbol intenta corregir los errores del árbol anterior. Al igual que *Random Forest* es efectivo en problemas con alta dimensionalidad y con relaciones no lineales entre las variables.

En [23] *XGBoost* se compara con un algoritmo de multi-objetivo propuesto en el estudio, donde también las características usadas en ambos modelos, fueron diseñadas en el mismo trabajo. La *accuracy* obtenida por *XGBoost* utilizando estas características es algo baja (56 %), aunque supera al algoritmo multi-objetivo propuesto (50 %) usando estas características, esto es porque este algoritmo fue pensado para aceptar como entrada la señal de audio pura, la *accuracy* obtenida por este último algoritmo es de 66 %. También hay que mencionar que el conjunto de datos usado en este caso tiene el nombre de *Qolt* [24] (una base de datos grabada en coreano), este conjunto tiene valores de *accuracy* más bajos comparados a *TORGO* y *UA-Speech*, obteniendo un máximo de 70 % [19].

## 3. Propuesta y Diseño de Solución

Refracted Speech impactaría positivamente a personas con disartria, al ofrecerles una herramienta que pueda mejorar la inteligibilidad de su habla y facilitar su participación en actividades cotidianas y sociales. Todo esto gracias a la detección temprana y los ejercicios terapéuticos que ofrece la aplicación, estos ejercicios están respaldados por Viviana García, profesora y doctora de la Facultad de Medicina de la Universidad de Valparaíso.

Este trabajo se enfoca en el desarrollo de un modelo de predicción de disartria a partir de grabaciones de voz, con el objetivo de alcanzar el mayor desempeño posible.

### 3.1. Conjuntos de Datos

Los conjuntos de datos a utilizar en este trabajo son *TORGO* [9] y *UA Speech* [25]. Aunque hay que mencionar, que durante el desarrollo de *Refracted Speech* el único conjunto disponible de los mencionados por la literatura era *TORGO*, pero con el fin de aumentar los ejemplos utilizados, para este trabajo se utilizará también *UA Speech*, que está disponible para uso público desde noviembre de 2024.

#### 3.1.1. *TORGO*

Este conjunto de datos fue creado por el Departamento de Computer Science de la Universidad de Toronto, El Departamento de Speech-Language Pathology de la Universidad de Toronto y el Holland-Bloorview Kids Rehabilitation Hospital. La base de datos provee 4 subconjuntos sin costo y para uso académico:

- 3 participantes femeninas con disartria.
- 3 participantes femeninas sin disartria.
- 5 participantes masculinos con disartria.
- 4 participantes masculinos sin disartria.

El dataset contiene grabaciones de voz en formato `WAV`, sus respectivas transcripciones a texto en formato `TXT`, los resultados de un *Frenchay Dysarthria Assessment* en formato `CSV` para cada sujeto y un archivo en formato `XLS` con las grabaciones con errores. Adicionalmente, contiene información de los movimientos de la lengua de la persona en cada grabación, obtenida por un dispositivo *3D AG500 Electro-Magnetic Articulograph (EMA)*, lo cual no será utilizado en este trabajo

Existen 2 tipos de grabaciones, la primera corresponde a una hecha con 8 micrófonos puestos alrededor del sujeto de prueba, que da información posicional de la grabación de audio. El segundo tipo corresponde a grabaciones hechas por un solo micrófono montado en la cabeza de la persona a 16 kHz. En este trabajo se utilizarán solo las grabaciones del segundo tipo, ya que se asemejan más a lo que nos puede brindar un micrófono en un teléfono móvil, en total el conjunto contiene 8216 grabaciones de este tipo (contando las que tienen errores).

#### 3.1.2. *UA Speech*

Este conjunto de datos fue creado por el University of Illinois Review Board, la base de datos original contaba con grabaciones de 19 personas con disartria, pero actualmente tiene 15, ya que con el pasar de los años algunos quitaron su permiso y/o los datos fueron corruptos. Los conjuntos disponibles son:

- 4 participantes femeninas con disartria.
- 4 participantes femeninas sin disartria.
- 11 participantes masculinos con disartria.
- 9 participantes masculinos sin disartria.

Al igual que *TORGO*, las grabaciones se hicieron con 8 micrófonos puestos alrededor del sujeto de prueba, y además se cuenta con una versión de las grabaciones con un solo canal de audio, que son las que se usarán para este trabajo. La diferencia radica en que en este caso las grabaciones se hicieron a 48 kHz, por lo que es necesario un procesamiento previo para poder trabajar simultáneamente con ambos conjuntos sin que esta diferencia afecte a los resultados. En total, el conjunto contiene 143290 grabaciones de voz.

### 3.2. Reducción de Ruido

Con el fin de reducir el ruido de fondo de las grabaciones, el cual es fácilmente perceptible a oído humano, se usó un algoritmo [26] que tiene la ventaja de mantener la mayoría de la información espectral de la señal. Utiliza una técnica llamada "*espectral gating*", donde se calcula el espectrograma de una señal y se estima el nivel de ruido por cada banda de frecuencias en este. En el trabajo original, se utiliza este algoritmo para reducir el ruido de grabaciones de animales, para luego detectar y visualizar estructuras de distintas especies, y además está disponible gratuitamente como paquete [27] de *Python*.

### 3.3. Metodología

Para desarrollar un modelo de predicción de disartria que sea rápido y preciso, aprovechando al máximo los conjuntos de datos, es fundamental emplear las técnicas que han tenido mejores resultados. Existen 2 tipos de *features* o *características* que pueden ser extraídas de una grabación de voz, las basadas en audio y las basadas en imagen. Las *características* basadas en audio son las que se obtienen directamente a partir de la señal de audio, algunos ejemplos son la *Frecuencia Fundamental (F0)*, *Harmonics-to-Noise Ratio (HNR)*, *Mel-Frecuency Cepstral Coefficients (MFCC)*, etc. En cambio, las *características* basadas en imagen son las que se obtienen a partir de una transformación de la señal de audio, obteniendo una representación bi-dimensional de la señal, principalmente espectrogramas.

Como base, ya se tiene un modelo entrenado y evaluado, el cual fue usado durante la presentación de *Refracted Speech* durante la Feria de Software (FESW) USM 2024. Este modelo está basado en características de audio, la idea de la solución propuesta es desarrollar modelos adicionales a este, un grupo basado en características de audio, que incluya características de audio, modelos y datos adicionales al modelo base, y otro grupo basado en características de imagen. Luego se elegirá al mejor modelo basado en características de audio y al mejor modelo basado en características de imagen, para compararlos con el modelo base y así lograr la máxima precisión en el prediagnóstico.

### 3.4. Modelo Base (Usado en FESW 2024)

Este modelo fue entrenado y evaluado solamente usando la base de datos *TORGO* [9] que era la que estaba disponible en la fecha. Se extrajo la siguiente configuración de características de cada señal de audio:

- Media, desviación estándar, mediana, máximo y mínimo de los 12 primeros *MFCC*.
- Media, desviación estándar, mediana, máximo y mínimo de los deltas de los 12 primeros *MFCC*.

- Media, desviación estándar, mediana, máximo y mínimo de los delta-deltas de los 12 primeros *MFCC*.
- Media y desviación estándar de la *Frecuencia Fundamental (F0)*.
- *Local Jitter*.
- *Absolute Shimmer (dB)*.

Teniendo un total de 184 de características para el entrenamiento y evaluación de los modelos. Para este proceso, se empleó la API de *scikit-learn* [28], que proporciona herramientas para el aprendizaje automático, desde el preprocesamiento de datos hasta validación de modelos.

El modelo escogido fue *SVM* con kernel linear, esto por su simpleza y su alto nivel de optimización, ya que *scikit-learn* provee dos clases para clasificadores de *SVM*: `SVC` (*Support Vector Classiffier*) y `LinearSVC`, el primero es una implementación general de *SVM* que permite escoger kernel y una gran variedad de parámetros, el segundo solo permite el kernel linear, pero es mucho más eficiente, en especial con datasets de gran tamaño, y por eso fue escogido inicialmente para el modelo base.

Ya habiendo escogido las características a utilizar y el clasificador a entrenar, el proceso para obtener los modelos, consta de las siguientes etapas:

### 3.4.1. Preprocesamiento de Datos

Esta etapa incluye la reducción de ruido de las grabaciones de audio y una preselección de las grabaciones que son útiles para el entrenamiento de un clasificador. En primer lugar, el conjunto utilizado para este modelo base fue *TORGO*, el cual tiene información de cuáles de sus grabaciones tuvieron errores al momento de tomarlas, estas 59 grabaciones no se tomaron en cuenta. Además, el rango de duración de las grabaciones es bastante elevado, desde fracciones de segundo hasta aproximadamente 194 segundos, por un lado grabaciones muy cortas tendrán muy poca información, en especial para variaciones de frecuencias y amplitudes, y por otro lado grabaciones muy largas tendrán mucha variación de frecuencia, amplitud y también muchos silencios si es que hay frases de por medio. Ambos casos pueden introducir ruido al entrenamiento, en la Figura 6 se puede apreciar en la distribución de la duración, que la mayoría de las grabaciones están entre 1 y 10 segundos.

Luego de eliminar las 59 grabaciones con errores y las 260 grabaciones con menos de 1 segundo o más de 10 segundos de duración, se tienen 7897 grabaciones para entrenamiento y evaluación de las 8216 que había inicialmente.

Por otro lado, los datos muestran un leve desbalance de clases, 5142 (65.1%) casos negativos y 2755 (34.9%) casos positivos. Por esta razón, es importante utilizar métricas que evalúen correctamente el desempeño del modelo en ambas clases, como *precision*, *recall* y *F1-Score*.

### 3.4.2. Extracción de Características

El siguiente paso consiste en obtener un arreglo de 184 características para cada una de las 7897 grabaciones de audio. Para este proceso, se utilizó la biblioteca *Librosa* [29], encargada de la lectura de los archivos de audio y el cálculo de los *MFCC*, así como la biblioteca *Parselmouth* [30], utilizada para extraer el resto de las características acústicas.

Como resultado, se obtuvo una matriz bidimensional de tamaño $7897 \times 184$, donde cada fila representa una grabación y cada columna una característica. Esta matriz será utilizada como

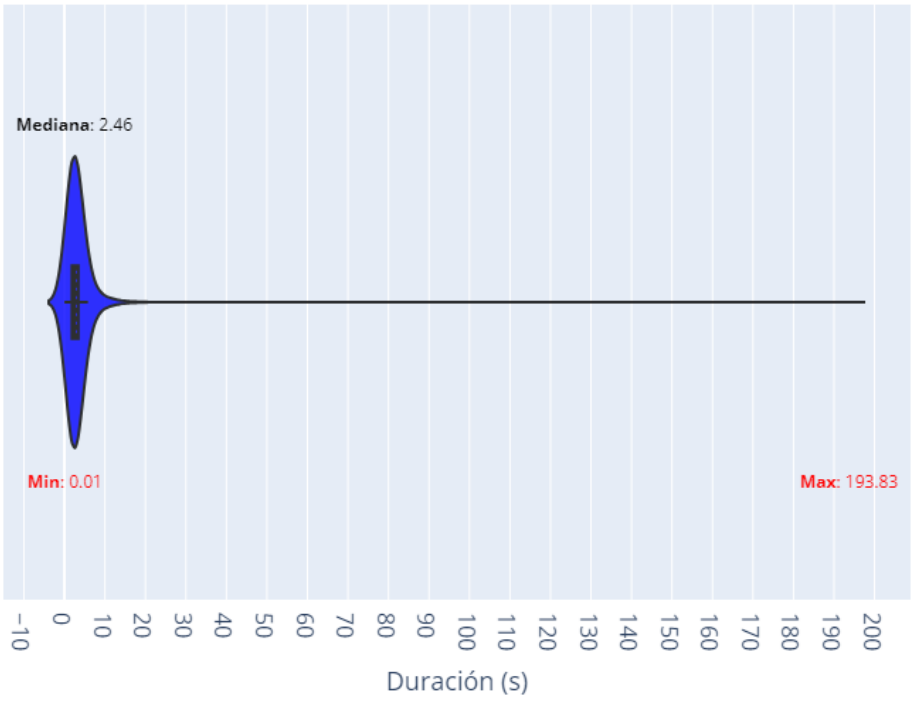



Figura 6. Distribución de la duración de las grabaciones de *TORGO*

conjunto de entrada para el entrenamiento y la evaluación del clasificador de disartria.

### 3.4.3. Selección de Hiperparámetros

Para obtener el mejor rendimiento de un modelo, en este caso de `LinearSVC`, es importante encontrar la combinación de hiperparámetros que optimice los resultados. Para este proceso se hizo una búsqueda entre distintas combinaciones de hiperparámetros, de la cual para cada combinación se realizó un *5-Fold Cross-Validation* con el objetivo de darle una robustez adicional a los resultados. Los valores de los parámetros se pueden ver en la Tabla 2.

Tabla 2. Valores de hiperparámetros evaluados en el modelo base

| **Hiperparámetro** | Valores |
|---|---|
| **C** | 0.001, 0.01, 0.1, 1, 10, 100, 1000 |
| **Penalty** | l1, l2 |
| **Loss** | hinge, squared_hinge |

- **C:** Es el parámetro de regularización. Valores pequeños implican una mayor penalización a los errores del modelo (mayor regularización), lo que puede evitar el sobreajuste. Valores

grandes permiten que el modelo se ajuste más a los datos de entrenamiento, lo que puede resultar en un menor sesgo pero mayor varianza.

- **Penalty:** Determina la norma utilizada en la regularización. "l1" promueve la generación de modelos más dispersos (es decir, con menos características no nulas), mientras que "l2" tiende a distribuir el peso entre todas las características de manera más uniforme.
- **Loss:** Define la función de pérdida a optimizar. "hinge" corresponde a la pérdida tradicional de una máquina de vectores de soporte, mientras que "squared hinge" es una versión cuadrática que penaliza más fuertemente los errores grandes.

### 3.5. Modelo Propuesto

En esta versión, además de usar la base de datos *TORGO*, también se utilizó *UA Speech*. Pero no solo se añadieron datos, también se amplió el número de características extraídas de cada señal de audio y los modelos de aprendizaje automático, de los cuales se hizo una selección de hiperparámetros para comparar los modelos con mejor desempeño y obtener la configuración del modelo que aproveche mejor la información de los datos. Los clasificadores evaluados son: *SVM*, *Random Forest* y *XGBoost*.

Con respecto a las características de audio usadas, se utilizaron todas las usadas en el modelo base, y además se añadieron las siguientes:

- Rango (diferencia entre máximo y mínimo) de la *Frecuencia Fundamental (F0)*.
- *Absolute Jitter*.
- *Five-Point Period Perturbation (PPQ5)*.
- *Relative Average Perturbation (RAP)*.
- *Local Shimmer*.
- *Three-Point Amplitude Perturbation Quotient (APQ3)*.
- *Five-Point Amplitude Perturbation Quotient (APQ5)*.
- *Harmonic-To-Noise Ratio (HNR)*.

Teniendo un total de 192 características de audio para el entrenamiento y evaluación de los modelos. En la Tabla 3 se puede ver un resumen de la configuración de características usada para el modelo propuesto y el modelo base.

#### 3.5.1. Preprocesamiento de Datos

El procedimiento de esta etapa tiene algunos cambios comparados al del modelo base. Primero, para poder usar los dos conjuntos de datos, se tienen que igualar los formatos en los que están representadas las señales de audio. Ambos conjuntos se encuentran en archivos de formato `WAV`, con muestras de 16 bits, pero tienen distintas frecuencias de muestreo, con 16 kHz versus 48 kHz para *TORGO* y *UA Speech* respectivamente. La mejor opción es realizar un downsample a *UA Speech* para que ambos conjuntos tengan una frecuencia de muestreo de 16 kHz, esto también ofrece una ventaja en tiempo computacional, ya que se estaría trabajando con señales de un tercio del tamaño original, lo cual afectaría tanto al procesamiento de la señal como al envío de esta desde el cliente al servidor en el caso de la aplicación de *Refracted Speech*.

Tabla 3. Comparación de características de audio entre modelos (resumen).

| **Característica** | **Modelo Base** | **Modelo Propuesto** |
|---|---|---|
| *MFCC (12 coeficientes)* | | |
| Media | ✓ | ✓ |
| Desviación estándar | ✓ | ✓ |
| Mediana | ✓ | ✓ |
| Máximo | ✓ | ✓ |
| Mínimo | ✓ | ✓ |
| *Deltas de MFCC* | | |
| Media | ✓ | ✓ |
| Desviación estándar | ✓ | ✓ |
| Mediana | ✓ | ✓ |
| Máximo | ✓ | ✓ |
| Mínimo | ✓ | ✓ |
| *Delta-Deltas de MFCC* | | |
| Media | ✓ | ✓ |
| Desviación estándar | ✓ | ✓ |
| Mediana | ✓ | ✓ |
| Máximo | ✓ | ✓ |
| Mínimo | ✓ | ✓ |
| *Frecuencia Fundamental (F0)* | | |
| Media | ✓ | ✓ |
| Desviación estándar | ✓ | ✓ |
| Rango | × | ✓ |
| *Otras* | | |
| *Absolute Jitter* | × | ✓ |
| *Local Jitter* | ✓ | ✓ |
| *PPQ5* | × | ✓ |
| *RAP* | × | ✓ |
| *Local Shimmer* | × | ✓ |
| *Local Shimmer (dB)* | ✓ | ✓ |
| *APQ3* | × | ✓ |
| *APQ5* | × | ✓ |
| *HNR* | × | ✓ |

El siguiente paso es la reducción de ruido a ambos conjuntos y la preselección de las grabaciones de audio que nos sirven para el entrenamiento de un clasificador automático. La diferencia es que *UA Speech* no posee información de grabaciones con errores, por lo que quedaría analizar la duración de estas, que al igual que en *TORGO*, existe un rango de duración muy elevado, existiendo señales que van desde fracciones de segundo hasta aproximadamente 56 segundos. En la Figura 7 se puede apreciar que la mayoría de las grabaciones están entre 1 y 10 segundos.

Luego de eliminar las 1119 grabaciones de menos de 1 segundo y más de 10 segundos que están en *UA Speech*, quedan 142171 grabaciones en este conjunto. Finalmente al combinar ambos

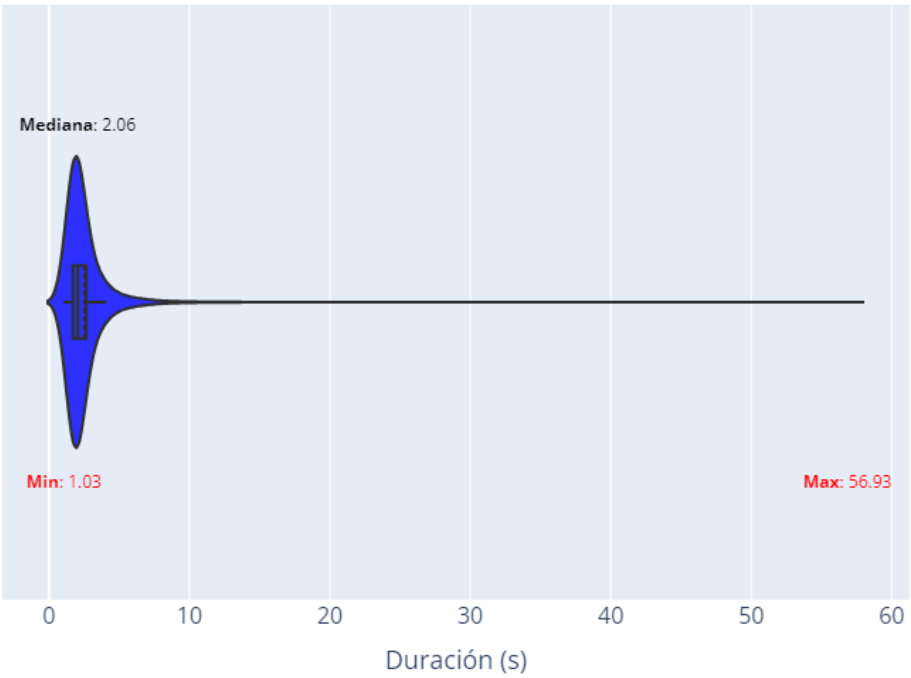



Figura 7. Distribución de la duración de las grabaciones de *UA Speech*

conjuntos se tiene un total de 150068 señales de audio para utilizar en el entrenamiento y evaluación de los modelos.

A diferencia del modelo base, en este caso el conjunto resultante está bastante balanceado (49.8 % de casos negativos y 50.2 % de casos positivos), por lo que no hay ningún problema a tener en cuenta en este ámbito.

### 3.5.2. Extracción de Características

Lo que sigue es obtener el arreglo de 192 características por cada una de las 150068 señales de audio. Al igual que para el modelo base, se utilizó la biblioteca *Librosa* [29] para la lectura de los archivos de audio y la extracción de los *MFCC*, así como *Parselmouth* [30] para extraer el resto de características acústicas de las grabaciones.

En este caso el resultado es una matriz bidimensional de tamaño 150068x192, la cual será utilizada como conjunto de entrada para el entrenamiento y evaluación del clasificador.

### 3.5.3. Selección de Hiperparámetros

Como se mencionó previamente, para encontrar un modelo que pueda aprovechar al máximo la cantidad de datos y características extraídas de estos, se eligió un grupo de distintos tipos de

modelos de clasificación automática, para todos ellos es importante encontrar la combinación de hiperparámetros que optimice sus resultados. De igual manera, para cada configuración se realizó un *5-Fold Cross-Validation* obteniendo una robustez adicional en los resultados y así poder compararlos y decidir cuál fue el que tuvo mejor resultado.

- ***Support Vector Classifier (SVC):*** Los valores de los hiperparámetros evaluados están en la Tabla 4 y tienen las siguientes definiciones:

  - **Kernel:** Función núcleo que determina la transformación del espacio de características. Se usa "linear" para separación lineal, "rbf" para transformación no lineal gaussiana, "poly" para polinomial y "sigmoid" para función sigmoide.

Tabla 4. Valores de hiperparámetros evaluados en `SVC` en modelo propuesto

| **Hiperparámetro** | Valores |
|---|---|
| **Kernel** | "linear", "rbf", "poly", "sigmoid" |
| **C** | 0.001, 0.01, 0.1, 1, 10, 100, 1000 |

- ***Random Forest (RF):*** Los valores de los hiperparámetros evaluados están en la Tabla 5 y tienen las siguientes definiciones:

  - **N° Estimators:** Número de árboles en el bosque.
  - **Max Depth:** Profundidad máxima de los árboles. "None" permite que los nodos se expandan hasta que todas las hojas sean puras. Valores menores (10-20) previenen sobreajuste limitando la complejidad.
  - **Min Samples Split:** Número mínimo de muestras requeridas para dividir un nodo interno. Valores más altos (5) limitan el crecimiento del árbol, evitando sobreajuste para datos pequeños.

Tabla 5. Valores de hiperparámetros evaluados en `Random Forest` en modelo propuesto

| **Hiperparámetro** | Valores |
|---|---|
| **N° Estimators** | 100, 200 |
| **Max Depth** | None, 10, 20 |
| **Min Samples Split** | 2, 5 |

- ***XGBoost:*** Los valores de los hiperparámetros evaluados están en la Tabla 6 y tienen las siguientes definiciones:
  - **N° Estimators:** Número de árboles que se entrenan en el modelo.
  - **Max Depth:** Profundidad máxima de los árboles. "None" permite que los nodos se expandan hasta que todas las hojas sean puras. Valores menores (10-20) previenen sobreajuste limitando la complejidad.
  - **Learning Rate:** Tasa de aprendizaje usada en el ajuste de los árboles.

Tabla 6. Valores de hiperparámetros evaluados en `XGBoost` en modelo propuesto

| **Hiperparámetro** | Valores |
|---|---|
| **N° Estimators** | 100, 200, 300 |
| **Max Depth** | None, 10, 20 |
| **Learning Rate** | 0.01, 0.1, 0.3 |

# 4. Validación de la Solución

## 4.1. Métricas de Evaluación

La elección de las métricas para evaluar las predicciones de los modelos es de suma importancia, ya que permite medir el rendimiento de manera objetiva y comparar distintas configuraciones o algoritmos. Dependiendo del contexto y del balance entre clases, ciertas métricas pueden ofrecer una visión más adecuada del comportamiento del modelo que otras.

Antes de presentar las métricas escogidas, es importante definir los siguientes conceptos basados en una matriz de confusión:

- **TP (True Positives):** Casos positivos correctamente clasificados como positivos.
- **TN (True Negatives):** Casos negativos correctamente clasificados como negativos.
- **FP (False Positives):** Casos negativos incorrectamente clasificados como positivos.
- **FN (False Negatives):** Casos positivos incorrectamente clasificados como negativos.

Entonces, las métricas que se escogieron fueron las siguientes:

- ***Accuracy:*** Proporción de predicciones correctas sobre el total de muestras.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{23}$$

- ***Recall:*** También conocido como sensibilidad o tasa de verdaderos positivos. Indica la proporción de los casos positivos que fueron correctamente identificados.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{24}$$

- ***Precision:*** Proporción de verdaderos positivos entre todas las muestras que el modelo clasificó como positivas. Es especialmente útil cuando el costo de los falsos positivos es alto.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{25}$$

- ***F1-Score:*** Media armónica entre *precision* y *recall*. Proporciona un equilibrio entre ambas métricas, siendo útil cuando se busca un compromiso entre la precisión y la cobertura del modelo. Se usará esta métrica para elegir a los mejores modelos, ya que penaliza tanto los falsos positivos como los falsos negativos de manera equilibrada.

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{26}$$

### 4.2. Modelo Base (Usado en FESW 2024)

Como se observa en la Tabla 7, los resultados obtenidos por el modelo base fueron bastante positivos. El valor más alto de *F1-Score* alcanzado fue 0.8006, correspondiente a la mejor configuración evaluada. Esta configuración también obtuvo el mejor resultado en *recall*, lo que indica que fue la más efectiva al identificar correctamente los casos de disartria. Aunque no logró los valores más altos en *accuracy* ni en *precision*, su desempeño en estas métricas fue bastante cercano a los máximos observados.

Tabla 7. Resultados de las diferentes combinaciones de hiperparámetros para Modelo Base (Colores verde y rojo corresponde a mejor y peor resultado por columna respectivamente)

| C | loss | penalty | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| 0.001 | hinge | l2 | 0.8108 | 0.8079 | 0.7549 | 0.7432 |
| 0.001 | squared_hinge | l1 | **0.7672** | **0.7813** | **0.7111** | **0.6895** |
| 0.001 | squared_hinge | l2 | 0.8298 | 0.825 | 0.7679 | 0.7656 |
| 0.01 | hinge | l2 | 0.8438 | 0.8335 | 0.7874 | 0.7859 |
| 0.01 | squared_hinge | l1 | 0.8295 | 0.8312 | 0.7528 | 0.7585 |
| 0.01 | squared_hinge | l2 | 0.8458 | 0.831 | 0.7877 | 0.788 |
| 0.1 | hinge | l2 | 0.8511 | **0.835** | 0.7915 | 0.7941 |
| 0.1 | squared_hinge | l1 | 0.8482 | 0.8313 | 0.7918 | 0.7911 |
| 0.1 | squared_hinge | l2 | 0.8497 | 0.8307 | 0.7938 | 0.793 |
| 1 | hinge | l2 | **0.8549** | 0.8336 | 0.7983 | 0.7996 |
| 1 | squared_hinge | l1 | 0.8496 | 0.8301 | 0.7938 | 0.793 |
| 1 | squared_hinge | l2 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |
| 10 | hinge | l2 | 0.8542 | 0.8313 | 0.7986 | 0.7993 |
| 10 | squared_hinge | l1 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |
| 10 | squared_hinge | l2 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |
| 100 | hinge | l2 | 0.8542 | 0.8296 | **0.8024** | **0.8006** |
| 100 | squared_hinge | l1 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |
| 100 | squared_hinge | l2 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |
| 1000 | hinge | l2 | 0.8393 | 0.8049 | 0.7573 | 0.7656 |
| 1000 | squared_hinge | l1 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |
| 1000 | squared_hinge | l2 | 0.8501 | 0.8303 | 0.7952 | 0.7939 |

### 4.3. Modelo Propuesto

Para los cuatro algoritmos evaluados, *Linear Support Vector Classifier*, *Support Vector Classifier*, *Random Forest* y *XGBoost*, se observó que existe una combinación específica de hiperparámetros que maximiza todas las métricas consideradas. Los detalles de los resultados obtenidos para cada modelo se presentan en las Tablas 8, 9, 10 y 11, respectivamente.

En general, todos los algoritmos superaron ampliamente el rendimiento del modelo base. El peor desempeño lo obtuvo *LinearSVC*, con un *F1-Score* de 0.9089, lo que representa una mejora del 13.5 % respecto al modelo base. Por otro lado, el mejor resultado fue alcanzado por *SVC*, con un *F1-Score* de 0.9751, correspondiente a un incremento del 21.8 %.

Tabla 8. Resultados de las diferentes combinaciones de hiperparámetros para *Linear Support Vector Classifier* (Colores verde y rojo corresponde a mejor y peor resultado por columna respectivamente)

| C | penalty | loss | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| 0.001 | l1 | hinge | - | - | - | - |
| 0.001 | l1 | squared_hinge | 0.9017 | 0.9016 | 0.9018 | 0.9016 |
| 0.001 | l2 | hinge | 0.9086 | 0.9086 | 0.9084 | 0.9085 |
| 0.001 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 0.01 | l1 | hinge | - | - | - | - |
| 0.01 | l1 | squared_hinge | 0.9042 | 0.904 | 0.9041 | 0.9041 |
| 0.01 | l2 | hinge | 0.9089 | 0.9089 | 0.9088 | 0.9088 |
| 0.01 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 0.1 | l1 | hinge | - | - | - | - |
| 0.1 | l1 | squared_hinge | 0.9037 | 0.9036 | 0.9037 | 0.9037 |
| 0.1 | l2 | hinge | **0.909** | **0.909** | **0.9089** | **0.9089** |
| 0.1 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9037 | 0.9037 |
| 1 | l1 | hinge | - | - | - | - |
| 1 | l1 | squared_hinge | 0.9039 | 0.9037 | 0.9038 | 0.9038 |
| 1 | l2 | hinge | 0.9089 | 0.9089 | 0.9087 | 0.9088 |
| 1 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 10 | l1 | hinge | - | - | - | - |
| 10 | l1 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 10 | l2 | hinge | 0.9089 | 0.9089 | 0.9087 | 0.9088 |
| 10 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 100 | l1 | hinge | - | - | - | - |
| 100 | l1 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 100 | l2 | hinge | 0.9081 | 0.9081 | 0.9079 | 0.908 |
| 100 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 1000 | l1 | hinge | - | - | - | - |
| 1000 | l1 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |
| 1000 | l2 | hinge | **0.8473** | **0.8475** | **0.847** | **0.847** |
| 1000 | l2 | squared_hinge | 0.9038 | 0.9037 | 0.9038 | 0.9037 |

Tabla 9. Resultados de las diferentes combinaciones de hiperparámetros para *Support Vector Classifier* (Colores verde y rojo corresponde a mejor y peor resultado por columna respectivamente)

| kernel | C | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|
| linear | 0.01 | 0.909 | 0.9089 | 0.9088 | 0.9089 |
| linear | 0.1 | 0.9078 | 0.9078 | 0.9076 | 0.9077 |
| linear | 1 | 0.6697 | 0.6799 | 0.6643 | 0.6554 |
| linear | 10 | 0.5885 | 0.6037 | **0.5803** | **0.5562** |
| linear | 100 | **0.5879** | **0.5921** | 0.5805 | 0.5599 |
| poly | 0.01 | 0.8985 | 0.9004 | 0.8975 | 0.8981 |
| poly | 0.1 | 0.9323 | 0.9332 | 0.9316 | 0.9321 |
| poly | 1 | 0.9562 | 0.9566 | 0.9559 | 0.9561 |
| poly | 10 | 0.9586 | 0.9587 | 0.9584 | 0.9586 |
| poly | 100 | 0.9579 | 0.9579 | 0.9577 | 0.9578 |
| rbf | 0.01 | 0.9102 | 0.9102 | 0.91 | 0.9101 |
| rbf | 0.1 | 0.9466 | 0.9467 | 0.9465 | 0.9466 |
| rbf | 1 | 0.9681 | 0.9682 | 0.9681 | 0.9681 |
| rbf | 10 | **0.9751** | **0.9751** | **0.975** | **0.9751** |
| rbf | 100 | 0.9747 | 0.9747 | 0.9746 | 0.9747 |
| sigmoid | 0.01 | 0.8974 | 0.8972 | 0.8973 | 0.8973 |
| sigmoid | 0.1 | 0.8569 | 0.8567 | 0.8568 | 0.8568 |
| sigmoid | 1 | 0.8426 | 0.8424 | 0.8424 | 0.8424 |
| sigmoid | 10 | 0.8407 | 0.8405 | 0.8405 | 0.8405 |
| sigmoid | 100 | 0.8437 | 0.8436 | 0.8435 | 0.8435 |

Tabla 10. Resultados de las diferentes combinaciones de hiperparámetros para *Random Forest* (Colores verde y rojo corresponde a mejor y peor resultado por columna respectivamente)

| n_estimators | max_depth | min_samples_split | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| 100 | 10 | 2 | **0.923** | **0.9229** | **0.923** | **0.9229** |
| 100 | 10 | 5 | **0.923** | **0.9229** | **0.923** | **0.9229** |
| 100 | 20 | 2 | 0.9384 | 0.9383 | 0.9383 | 0.9383 |
| 100 | 20 | 5 | 0.9393 | 0.9392 | 0.9392 | 0.9392 |
| 100 | - | 2 | 0.939 | 0.9388 | 0.939 | 0.9389 |
| 100 | - | 5 | 0.9397 | 0.9396 | 0.9396 | 0.9396 |
| 200 | 10 | 2 | 0.924 | 0.9239 | 0.9239 | 0.9239 |
| 200 | 10 | 5 | 0.9231 | 0.923 | 0.9231 | 0.923 |
| 200 | 20 | 2 | 0.9397 | 0.9396 | 0.9396 | 0.9396 |
| 200 | 20 | 5 | 0.9389 | 0.9389 | 0.9389 | 0.9389 |
| 200 | - | 2 | **0.9402** | **0.9401** | **0.9402** | **0.9401** |
| 200 | - | 5 | 0.94 | 0.94 | 0.94 | 0.94 |

Tabla 11. Resultados de las diferentes combinaciones de hiperparámetros para *XGBoost* (Colores verde y rojo corresponde a mejor y peor resultado por columna respectivamente)

| n_estimators | max_depth | learning_rate | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| 100 | 3 | 0.01 | **0.8794** | **0.8793** | **0.8794** | **0.8793** |
| 100 | 3 | 0.1 | 0.9282 | 0.9281 | 0.9281 | 0.9281 |
| 100 | 3 | 0.3 | 0.9466 | 0.9465 | 0.9465 | 0.9465 |
| 100 | 6 | 0.01 | 0.9112 | 0.9111 | 0.9112 | 0.9111 |
| 100 | 6 | 0.1 | 0.9537 | 0.9537 | 0.9537 | 0.9537 |
| 100 | 6 | 0.3 | 0.9629 | 0.9629 | 0.9628 | 0.9629 |
| 100 | 9 | 0.01 | 0.9316 | 0.9315 | 0.9316 | 0.9316 |
| 100 | 9 | 0.1 | 0.9619 | 0.9619 | 0.9619 | 0.9619 |
| 100 | 9 | 0.3 | 0.9672 | 0.9672 | 0.9671 | 0.9671 |
| 300 | 3 | 0.01 | 0.8962 | 0.896 | 0.8962 | 0.8961 |
| 300 | 3 | 0.1 | 0.9473 | 0.9472 | 0.9472 | 0.9472 |
| 300 | 3 | 0.3 | 0.9597 | 0.9597 | 0.9596 | 0.9596 |
| 300 | 6 | 0.01 | 0.93 | 0.9299 | 0.93 | 0.93 |
| 300 | 6 | 0.1 | 0.966 | 0.966 | 0.966 | 0.966 |
| 300 | 6 | 0.3 | 0.9706 | 0.9706 | 0.9705 | 0.9705 |
| 300 | 9 | 0.01 | 0.9473 | 0.9472 | 0.9473 | 0.9472 |
| 300 | 9 | 0.1 | 0.9696 | 0.9696 | 0.9696 | 0.9696 |
| 300 | 9 | 0.3 | **0.9711** | **0.9711** | **0.9711** | **0.9711** |

Tabla 12. Resumen de los mejores resultados de cada algoritmo evaluado (Colores verde y rojo corresponde a mejor y peor resultado por columna respectivamente)

| Modelo | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|
| *Support Vector Classifier* (Modelo Propuesto) | **0.9751** | **0.9751** | **0.975** | **0.9751** |
| *Linear Support Vector Classifier* (Modelo Propuesto) | 0.909 | 0.909 | 0.9089 | 0.9089 |
| *Random Forest* (Modelo Propuesto) | 0.9402 | 0.9401 | 0.9402 | 0.9401 |
| *XGBoost* (Modelo Propuesto) | 0.9711 | 0.9711 | 0.9711 | 0.9711 |
| Modelo Base | **0.8542** | **0.8296** | **0.8024** | **0.8006** |

La Tabla 12 resume los mejores resultados de cada clasificador, incluyendo al modelo base. El clasificador escogido para el modelo propuesto es *Support Vector Classifier*, el cuál logró el mejor rendimiento global, convirtiéndose en la opción más adecuada para el prediagnóstico automático de disartria a partir de grabaciones de voz. Estos resultados refuerzan la viabilidad del enfoque propuesto como una herramienta de apoyo clínico eficiente y confiable.

# 5. Conclusiones

## 5.1. Efectividad del modelo propuesto

El modelo desarrollado demostró un alto desempeño en la clasificación automática de la disartria, superando significativamente al modelo base utilizado inicialmente en *Refracted Speech*. El mejor resultado corresponde al clasificador *Support Vector Machine (SVM)*, alcanzando un *F1-Score* de 0.97511, lo que representa una mejora del 21.8 % respecto al modelo base. Estos resultados, respaldados por una mejora significativa en las métricas de evaluación y un conjunto de datos ampliado y diversificado, demuestran que el modelo propuesto ofrece un alto grado de confiabilidad para su uso como herramienta de prediagnóstico en la aplicación *Refracted Speech*.

## 5.2. Importancia de las diferencias clave

La inclusión de características adicionales *HNR* o los distintos tipos de *Jitter* y *Shimmer* en el modelo propuesto permitió capturar con mayor precisión las perturbaciones típicas de la disartria. Por otro lado, el aumento en la diversidad de los conjuntos de datos y clasificadores evaluados permitió obtener resultados superiores en todo sentido al modelo base usado inicialmente en *Refracted Speech*.

## 5.3. Potencial para integración en *Refracted Speech*

Los resultados respaldan la viabilidad de incorporar este modelo en la herramienta de prediagnóstico de *Refracted Speech*. Su alta precisión y eficiencia computacional lo hacen adecuado para apoyar el diagnóstico y el tratamiento de los pacientes

## 5.4. Limitaciones y trabajo futuro

Aunque el modelo mostró un alto rendimiento, su evaluación se limitó a conjuntos de datos en ingles. Futuros trabajos podrían explorar su adaptación a otros idiomas, como el español, y validarlo con grabaciones de pacientes chilenos. Además, se podría evaluar el uso de modelos basados en deep learning (por ejemplo redes neuronales convolucionales) para comparar su desempeño con los enfoques tradicionales.

## Referencias


[1] Javiera Órdenez. *¿Por qué los chilenos hablan "mal"?* [Online; Último acceso: 11-09-2024]. 2024. URL: https://www.latercera.com/que-pasa/noticia/por-que-en-chile-se-habla-mal/4DE374IKEJG3XP4XR33LJRJE2I/.

[2] Jennifer M Taber, Bryan Leyva y Alexander Persoskie. «Why do people avoid medical care? A qualitative study using national data». En: *Journal of general internal medicine* 30 (2015), págs. 290-297.

[3] Rebecca Palmer y Pamela Enderby. «Methods of speech therapy treatment for stable dysarthria: A review». En: *Advances in Speech Language Pathology* 9.2 (2007), págs. 140-153.

[4] Pamela Enderby. «Frenchay dysarthria assessment». En: *British Journal of Disorders of Communication* 15.3 (1980), págs. 165-173.

[5] Kathryn M Yorkston, David R Beukelman y Charles Traynor. *Assessment of intelligibility of dysarthric speech*. Pro-ed Austin, TX, 1984.

[6] Spri. *Inteligencia Artificial para corregir (y rehabilitar) patologías del habla o del lenguaje...* [Online; Último acceso: 11-09-2024]. 2020. URL: https://www.spri.eus/es/emprendimiento/inteligencia-artificial-para-corregir-y-rehabilitar-patologias-del-habla-o-del-lenguaje/.

[7] Voice Clinical Systems. *Voice Online Lab*. [Online; Último acceso: 11-09-2024]. 2024. URL: https://voiceclinicalsystems.com/app-onlinelab/.

[8] Say It Labs. *StutterStars*. [Online; Último acceso: 11-09-2024]. 2024. URL: https://www.sayitlabs.com/stutter-stars.

[9] Frank Rudzicz, Aravind Kumar Namasivayam y Talya Wolff. «The TORGO database of acoustic and articulatory speech from speakers with dysarthria». En: *Language resources and evaluation* 46 (2012), págs. 523-541.

[10] Harry Nyquist. «Abridgment of certain topics in telegraph transmission theory». En: *Journal of the AIEE* 47.3 (1928), págs. 214-217.

[11] Claude E Shannon. «Communication in the presence of noise». En: *Proceedings of the IRE* 37.1 (1949), págs. 10-21.

[12] Jean Baptiste Joseph baron de Fourier. *Théorie analytique de la chaleur*. Firmin Didot, 1822.

[13] Steven Davis y Paul Mermelstein. «Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences». En: *IEEE transactions on acoustics, speech, and signal processing* 28.4 (1980), págs. 357-366.

[14] Mirali Purohit et al. «Weak speech supervision: A case study of dysarthria severity classification». En: *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE. 2021, págs. 101-105.

[15] Abner Hernandez, Sunhee Kim y Minhwa Chung. «Prosody-based measures for automatic severity assessment of dysarthric speech». En: *Applied Sciences* 10.19 (2020), pág. 6999.

[16] João Paulo Teixeira, Carla Oliveira y Carla Lopes. «Vocal acoustic analysis–jitter, shimmer and hnr parameters». En: *Procedia technology* 9 (2013), págs. 1112-1122.

[17] HM Chandrashekar, Veena Karjigi y N Sreedevi. «Breathiness indices for classification of dysarthria based on type and speech intelligibility». En: *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*. IEEE. 2019, págs. 266-270.

[18] Paul Boersma et al. «Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound». En: *Proceedings of the institute of phonetic sciences*. Vol. 17. 1193. Amsterdam. 1993, págs. 97-110.

[19] Afnan Al-Ali et al. «The detection of dysarthria severity levels using AI models: A review». En: *IEEE Access* (2024).

[20] N P Narendra y Paavo Alku. «Automatic intelligibility assessment of dysarthric speech using glottal parameters». En: *Speech Communication* 123 (2020), págs. 1-9.

[21] Abner Hernandez et al. «Dysarthria Detection and Severity Assessment Using Rhythm-Based Metrics.» En: *Interspeech*. 2020, págs. 2897-2901.

[22] Bassam Ali Al-Qatab y Mumtaz Begum Mustafa. «Classification of dysarthric speech according to the severity of impairment: an analysis of acoustic features». En: *IEEE Access* 9 (2021), págs. 18183-18194.

[23] Eun Jung Yeo et al. «Automatic severity classification of dysarthric speech by using self-supervised model with multi-task learning». En: *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2023, págs. 1-5.

[24] Dae-Lim Choi et al. «Dysarthric Speech Database for Development of QoLT Software Technology.» En: *LREC*. 2012, págs. 3378-3381.

[25] Heejin Kim et al. «Dysarthric speech database for universal access research.» En: *Interspeech*. Vol. 2008. 2008, págs. 1741-1744.

[26] Tim Sainburg, Marvin Thielk y Timothy Q Gentner. «Finding, visualizing, and quantifying latent structure across diverse animal vocal repertoires». En: *PLoS computational biology* 16.10 (2020), e1008228.

[27] Tim Sainburg. *timsainb/noisereduce: v1.0*. Ver. db94fe2. Jun. de 2019. DOI: 10.5281/zenodo.3243139. URL: https://doi.org/10.5281/zenodo.3243139.

[28] F. Pedregosa et al. «Scikit-learn: Machine Learning in Python». En: *Journal of Machine Learning Research* 12 (2011), págs. 2825-2830.

[29] Brian McFee et al. *librosa/librosa: 0.11.0*. Ver. 0.11.0. Mar. de 2025. DOI: 10.5281/zenodo.15006942. URL: https://doi.org/10.5281/zenodo.15006942.

[30] Yannick Jadoul, Bill Thompson y Bart de Boer. «Introducing Parselmouth: A Python interface to Praat». En: *Journal of Phonetics* 71 (2018), págs. 1-15. DOI: https://doi.org/10.1016/j.wocn.2018.07.001.