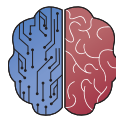




UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería de la Salud



INGENIERÍA  
DE LA SALUD

**TFG del Grado en Ingeniería de la  
Salud**

**Análisis de la voz para la  
detección de la Enfermedad  
de Parkinson  
Documentación Técnica**

Presentado por Maider Murugarren Ilundain  
en Universidad de Burgos

4 de julio de 2023

Tutores: José Francisco Díez Pasto – Esther Cubo  
Delgado



## Índice general

I

<b>Apéndice G Estudio experimental</b>	<b>39</b>
G.1. Cuaderno de trabajo. . . . .	39
G.2. Detalle de resultados. . . . .	51
<b>Bibliografía</b>	<b>61</b>

---

# Índice de figuras

---

A.1. Diagrama de Gantt . . . . .	3
B.1. Instalación Pyhton . . . . .	10
B.2. Instalación Pyhton . . . . .	10
B.3. Instalación Pyhton . . . . .	11
B.4. Instalación Jupyter Notebook . . . . .	12
B.5. Contenidos necesarios en el directorio de ejecución. . . . .	14
B.6. URL creado al iniciar la puesta en marcha de la interfaz. . . . .	15
B.7. Formato del prototipo . . . . .	16
B.8. Ejemplo respuesta del prototipo. . . . .	17
B.9. Diagrama de flujo. . . . .	18
D.1. Ejemplo conjunto de datos extraídos con Spicy . . . . .	31
D.2. Ejemplo conjunto de datos extraídos con Pytorch . . . . .	32
E.1. Diagrama de secuencias. . . . .	34
E.2. Diagrama de clases. . . . .	35
F.1. Diagrama de casos de uso. . . . .	38
G.1. Introducción de los datos paciente ejemplo. . . . .	40
G.2. Resultado paciente ejemplo. . . . .	40
G.3. Introducción de los datos paciente ejemplo. . . . .	41
G.4. Resultado paciente ejemplo. . . . .	42
G.5. Introducción de los datos paciente ejemplo. . . . .	43
G.6. Resultado paciente ejemplo 3. . . . .	44
G.7. Introducción de los datos paciente ejemplo. . . . .	45
G.8. Resultado paciente ejemplo. . . . .	46
G.9. Introducción de los datos paciente ejemplo. . . . .	47

G.10.Resultado paciente ejemplo. . . . .	48
G.11.Introducción de los datos paciente ejemplo 3. . . . .	49
G.12.Resultado paciente ejemplo 3. . . . .	50
G.13.Audio tipo 1 (vocal A), árbol de decisiones. . . . .	51
G.14.Audio tipo 2 (vocal I), árbol de decisiones. . . . .	52
G.15.Audio tipo 3 (vocal U), árbol de decisiones. . . . .	52
G.16.Audio tipo 4 (palabra campana), árbol de decisiones. . . . .	53
G.17.Audio tipo 5 (palabra gato), árbol de decisiones. . . . .	53
G.18.Audio tipo 6 (palabra petaca), árbol de decisiones. . . . .	54
G.19.Audio tipo 1 (vocal A), KNN. . . . .	54
G.20.Audio tipo 2 (vocal I), KNN. . . . .	55
G.21.Audio tipo 3 (vocal U), KNN. . . . .	55
G.22.Audio tipo 4 (palabra campana), KNN. . . . .	56
G.23.Audio tipo 5 (palabra gato), KNN. . . . .	56
G.24.Audio tipo 6 (palabra petaca), KNN. . . . .	57
G.25.Audio tipo 1 (vocal A), Random Forest. . . . .	57
G.26.Audio tipo 2 (vocal I), Random Forest. . . . .	58
G.27.Audio tipo 3 (vocal U), Random Forest. . . . .	58
G.28.Audio tipo 4 (palabra campana), Random Forest. . . . .	59
G.29.Audio tipo 5 (palabra gato), Random Forest. . . . .	59
G.30.Audio tipo 6 (palabra petaca), Random Forest. . . . .	60

## *Apéndice A*

---

# **Plan de Proyecto Software**

---

### **A.1. Introducción**

Organizar tanto temporalmente como la relevancia de todos los procesos necesarios para completar un proyecto con una magnitud como este, es de vital importancia. Para ello, hay que coordinar, relacionar y asignar un nivel de relevancia todas las tareas que se realizan. Con un objetivo principal de cumplir con lo planteado y obtener unos resultados completos respecto al estudio realizado.

Por ello en este anexo se hablará de la planificación temporal dada al proyecto en conjunto.

### **A.2. Planificación temporal**

Por un lado, se ha hecho uso de una herramienta para visualizar la gestión de los proyectos denominada diagrama de Gantt. De esta manera, se muestran las diferentes actividades o tareas que se realizarán a lo largo del proyecto y se representan con un orden temporal teórico a seguir.

Este planteamiento fue realizado al principio de todo, para hacer una previsión de los plazos a cumplir, con el objetivo de llegar a los plazos de entrega determinados.

El diagrama tiene en total 9 pasos marcados temporalmente en el mes y semanas. Puesto que el planteamiento se hizo al inicio se ha intentado cumplir con ellos a pesar de que ha habido alguna variación como era de prever.

Por otro lado, se ha hecho uso de la plataforma GitHub, creando un repositorio para este proyecto, para estructura, almacenar y organizar en él todo lo requerido. En este caso, se ha creado un repositorio público con el nombre, Análisis de voz para la detección de enfermedades neurodegenerativas, que almacenará el proyecto completo además de organizar mediante distintas tareas los pasos indicados en el diagrama de Gantt.

Al cual se podrá acceder a través del siguiente link: [Repositorio GitHub del proyecto](#)

En total se han dividido las tareas en 6 pasos generales y fundamentales. Cada uno de ellos tendrá diferentes subapartados, los cuales se verán reflejados en el repositorio de GitHub. Donde se dará una continuidad en el tiempo más precisa de todos los pasos dados. Pero para tener una idea general y una aproximación temporal planteada desde el principio, se han ordenado los pasos en el tiempo mediante un diagrama de Gantt. Este viene adjuntado a continuación. Indicando de manera fácil y muy visual cual será el camino en el tiempo que tomará este proyecto.



TAREAS	FEBRERO			MARZO			ABRIL			MAYO			JUNIO	
	1ªSem	2ªSem	3ªSem	4ªSem	1ªSem	2ªSem	3ªSem	4ªSem	1ªSem	2ªSem	3ªSem	4ªSem		
0. Obtención y recopilación de la base de datos														
0. Toma de decisiones sobre la dirección del proyecto (modelos a utilizar)														
0. Documentación														
1. Ordenar la base de datos														
2. Carga de los datos														
3. Pre-procesado de los datos														
4. Extraer características de los datos con los modelos														
5. Crear un nuevo conjunto de datos con el que trabajar														
6. Entrenar el clasificador con el nuevo conjunto de datos														
7. Conclusiones de los resultados obtenidos														
8. Construcción de una página Web														
9. Terminar de redactar el proceso en la memoria de prácticas														

Figura A.1: Diagrama de Gantt

Fuente: Elaboración propia

Después, se hace una explicación detallada de lo realizado en cada tarea indicada en el repositorio.

## Sprint 0

Empezando por la documentación pertinente acerca de diferentes áreas relacionadas con el proyecto en cuestión. Esto es, haciendo una investigación exhaustiva sobre la enfermedad de Parkinson, sobre la detección de enfermedades neurodegenerativas mediante audios de voz y una lectura extensa sobre los trabajos e investigaciones publicadas en la red que hablen de algún avance en el área de la detección de enfermedades neurodegenerativas mediante el análisis de audios de voz.

El siguiente paso consistió en hacer una aproximación temporal de los pasos que se iban a seguir, así como un planteamiento de la cronología que se pretendía seguir para cumplir con los plazos precisos. En este paso además de crear el repositorio de *Github* para ir publicando los avances también se creó un diagrama de Gantt para poder ver de manera más visual la cronología que se iba seguir.

Tras haber hecho la investigación pertinente se empezó con el trabajo de campo. En este caso consistió en la recogida y grabación de los audios, para completar la base de datos con la que se pretendía trabajar. El procedimiento para la recogida de los audios viene extensamente explicado en el apartado 5. Aspectos Relevantes de la memoria.

## Sprint 1

En el segundo paso, se hizo tanto el proceso de reordenar y hacer la base de datos con una estructura adecuada para trabajar con ella. Como empezar a tratar con herramientas que permitieran trabajar con el análisis y carga de audios en Python.

Creando además los ficheros “.csv” con los datos clínicos pertinentes obtenidos en el trabajo de campo al recoger los audios.

Por último, en este paso, se aseguró que los datos recogidos con los audios cumplieran con la anonimización pertinente. Ya que se trabajará con datos de pacientes reales y la base de datos tiene que cumplir con los criterios legales de privacidad y seguridad de los pacientes voluntarios participantes en la investigación.

## Sprint 2

Para este tercer paso, ya se procedió a empezar a tratar con la base de datos formada por los audios. Para ello, se empezó con la carga de los mimos en formato “.wav”. Por otro lado, se preprocesaron, para intentar evitar problemas o interferencias no deseadas a la hora de procesar los datos y trabajar con ellos.

Para este paso de preprocesado, se quiso eliminar tanto el silencio que contenían la mayoría de los audios al principio y al final, eliminar todos los ruidos de fondo no deseados y por último normalizar el volumen de todos ellos.

## Sprint 3

A estas alturas ya se pretendía conformar la base de datos completa, esto es, la que contendría las características de interés de cada audio, uniéndolas a los datos clínicos ya recogidos.

Para ello, se decidió extraer las características de dos formas distintas haciendo uso de dos librerías y conformando así con conjuntos de datos diversos, de la misma procedencia de audios, los cuales contenían diferentes características que resultarían de interés para la posterior clasificación de los audios en base a ellos. Uniendo los archivos que contenían las características mediante un *join* con los archivos con los datos clínicos.

Tras obtener los dos conjuntos, se hará una comparación de los resultados obtenidos con cada uno de ellos para cada tipo de audios. De tal manera, que finalmente se creará un prototipo de detector de la enfermedad de Parkinson con el extractor de características más eficiente en cada caso.

## Sprint 4

En esta fase del proyecto con un conjunto de datos completo con el que trabajar, se seleccionaron tres modelos de entrenamiento apropiados para este caso de datos, el modelo de clasificación árboles de decisión, modelo de clasificación de Random Forest y el modelo de clasificación de KNN.

De tal manera que usando los dos conjuntos de cada tipo de audios se entrenaron mediante el proceso de validación cruzada. Y en consecuencia se obtuvieron los valores de las métricas que evaluarían el rendimiento de cada modelo en cada caso y se acumularon en archivos de formato “.csv”. Pudiendo así recuperarlos después y analizarlos para determinar cuál de ellos

actuaba de manera más eficiente para clasificar o identificar a los pacientes que padecen de Parkinson.

Tras la comparación del rendimiento de cada uno, explicado extensamente en la memoria en el apartado de discusión. Se ha determinado cuales serán los modelos que compondrán el prototipo final.

### **Sprint 5**

Antes de realizar la interfaz, se realizó en un notebook llamado *Prototipo*, el programa completo. En el cual se hace uso de las funciones realizadas durante todo el proyecto y se va llamando una a una partiendo de archivos en formato *“.py*. Y finalmente imprimiendo los resultados de la predicción.

Tras la búsqueda e investigación de las posibles opciones a la hora de desarrollar un prototipo, se concluyó que el uso de la librería Streamlit para Python, podía ser una opción adecuada para este tipo de proyecto realizado con notebooks en Python.

Para ello, se ha modificado el prototipo anterior a lo requerido de una ejecución adecuada con la librería mencionada. Logrando finalmente un archivo en formato *“.py*. El cual se ejecutará en la consola del propio ordenador y se abrirá la interfaz en un navegador.

Para terminar con lo anterior se han hecho mejoras estéticas para que se trate de una interfaz intuitiva y de fácil uso para el usuario.

### **Sprint 6**

Como último paso se tiene la redacción de la memoria a entregar. En la cual se analizará el proceso realizado por etapas, así como los análisis realizados, los resultados obtenidos y las conclusiones del proyecto. Planteando también posibles líneas futuras y mejoras del proyecto.

## **Planificación económica**

En este apartado se hablará del coste teórico que tendría la realización de un proyecto como este.

Donde a nivel informático se ha hecho uso de un único portátil personal. El cual tubo un coste de 1000€ el día de su compra. Y de momento se estima una amortización del mismo a unos 5 años. Por lo que el presupuesto de uso en 6 meses es el siguiente:

$$1000/(12 \times 5) = 16,66\text{€}$$

$$16,66 \times 6 = 99,96\text{€}$$

Además, se han utilizado hasta 2 dispositivos móviles, como grabadoras de audios con un coste aproximado de 300€ cada uno, amortizados a 2 años de uso. Por lo que el presupuesto de uso en 6 meses es el siguiente:

$$(300 + 300)/(12 \times 2) = 25\text{€}$$

$$25 \times 6 = 150\text{€}$$

En cambio, los programas y sistemas de herramientas utilizadas han sido siempre de acceso y uso gratuito al público.

Por último, a nivel de personas involucradas, habría que tener en cuenta tanto el salario de los profesores involucrados en el durante 6 meses. Como el salario de la investigadora y doctora que también ha sido partícipe. Los cuales son datos de los que no se tienen disposición.

Por ello, el presupuesto final tendrá en cuenta únicamente, los dispositivos físicos utilizados para su desarrollo.

A continuación se detallan los ítems presupuestarios:

Dispositivos	Cantidad	Precio
Portátil	1	99,96€
Móvil	2	150€
<b>Total</b>		<b>249,96€</b>

Pudiendo concluir de esta manera que el presupuesto final para la realización de este proyecto es de 249,96€

## Viabilidad legal

En este apartado se hablará de la viabilidad legal del proyecto. Abordando así la importancia de garantizar el cumplimiento de las normativas legales y éticas, así como el manejo y uso de los datos de los pacientes que han participado en el mismo.

Se empieza destacando que todos los participantes en las grabaciones de los audios han firmado un consentimiento informado en el cual se indica

de manera clara y precisa el objetivo de la investigación en la que están participando, el procedimiento que se seguirá en él, y que han leído, entendido y podido preguntar todo lo necesario antes de firmar.

De esta manera, han firmado un consentimiento donde indican que están conformes con la participación en el estudio donde se hará uso de sus audios de voz grabados, para el proyecto de investigación titulado, Estudio de la voz como biomarcador diferencial de la enfermedad de Parkinson.

Después, se ha asegurado de cumplir todas las leyes y regulaciones necesarias en el uso de audios de voz de los pacientes. Cumpliendo por ende con la ley de protección de datos personales y privacidad.

Para cumplir con el punto mencionado anteriormente, se ha hecho una anonimización de los datos recogidos. Para ello, se ha cambiado el nombre completo de cada paciente participante por un identificativo aleatorio con el que hacer referencia a los datos relacionados con el mismo. Con el objetivo de proteger la privacidad e identidad además de los datos sensibles de los participantes en todo momento.

También se dispone de una evaluación ética de la investigación en sí. Donde el comité ético de la investigación con medicamentos ha aprobado el estudio titulado, “Estudio de la voz como biomarcador diferencial de la enfermedad de Parkinson”.

Tanto el documento de confidencialidad como el documento del comité ético se encuentran adjuntados en el directorio *Base de datos*, en la subcarpeta *Documentos\_grabaciones*.

Por último, se reconoce la importancia asumida en la responsabilidad legal del manejo de todos los datos de pacientes, por parte de todos los participantes de este proyecto.

## *Apéndice B*

---

# Documentación de usuario

---

En este anexo se proporcionará la información necesaria sobre la documentación que tendrá que seguir el usuario, para la correcta implementación y uso del prototipo creado.

### **B.1. Requisitos software y hardware para ejecutar el proyecto.**

Comenzamos con los requisitos de software, que serán necesarias para la correcta ejecución de todo el proyecto. Siendo el entorno de ejecución y las herramientas utilizadas indispensables en este caso.

Python será el lenguaje con el que se desarrollará toda la implementación del código. Se trata de un lenguaje de programación interpretado, que fue creado el 1990 por Guido van Rossum. Se trata de uno de los lenguajes de programación más popular de la actualidad, destacando sobre el resto su sintaxis sencilla y intuitiva, además de las amplias áreas de aplicación, como son, por ejemplo, las ciencias de datos, inteligencia artificial o desarrollo web[17].

La instalación del mismo será también necesaria pero sencilla de hacer. Comenzamos descargando desde la propia pagina oficial de pyhton ([Descargar Python](#)) la versión que se quiera para el sistema operativo del ordenador con el que vaya a utilizar. Y se ejecuta, abriéndose la siguiente ventana emergente donde habrá que seleccionar las siguientes opciones dadas.

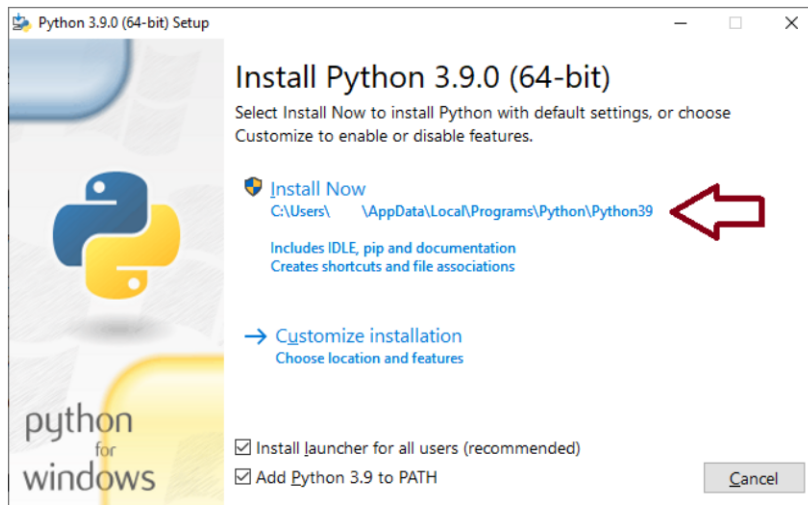


Figura B.1: Instalación Pyhton

[19]

De esta manera, comenzará la instalación.

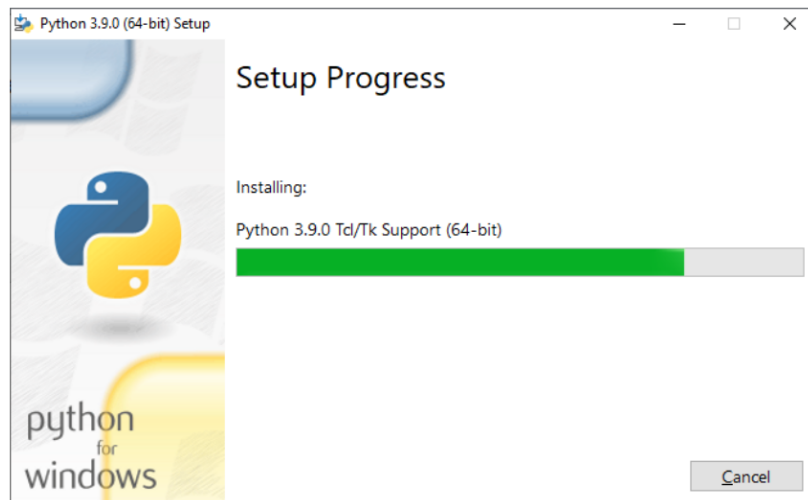


Figura B.2: Instalación Pyhton

[19]



Y finalmente tendremos Python instalado correctamente.

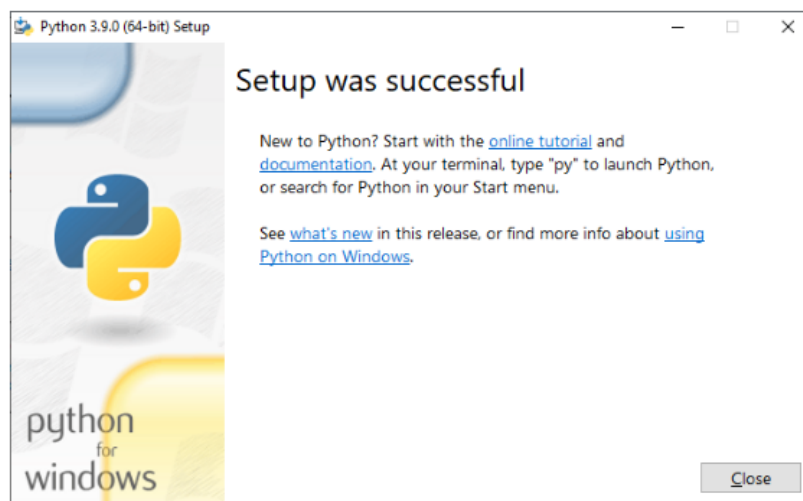


Figura B.3: Instalación Pyhton

[19]

Pudiendo abrir la aplicación directamente en nuestra computadora. Para hacer uso de ella y poder programar lo que necesitemos [19].

Jupyter Notebook será el entorno de programación utilizado para el desarrollo de este proyecto. Se trata de una aplicación web de código abierto y en este caso se usará de manera local. Hace uso de archivos notebook para crear y compartir el código creado en ellos en diferentes lenguajes de programación. Es un entorno intuitivo y fácil para desarrollar y visualizar el código a tiempo real.

Se basa en archivos notebooks compuestos por diferentes celdas, donde puede contener tanto código ejecutable como texto escrito en formato mark-down. Permitiendo así a los programadores mezclar tanto la documentación escrita con el código como los resultados de salida de dichas celdas a tiempo real [7].

Para instalar el entorno de programación simplemente hay que tener Python instalado como anteriormente se ha descrito, de tal manera que se tendrá agregado al PATH del sistema operativo. Por lo que únicamente, habrá que abrir la consola del sistema e introducir el siguiente comando `!pip install jupyter`.

Después de la descarga para acceder a ello habrá que escribir la siguiente instrucción “jupyter notebook” en la terminal y se abrirá en una pestaña del navegador de la siguiente manera.



Figura B.4: Instalación Jupyter Notebook

Fuente: Elaboración propia

En el caso de la instalación de las herramientas utilizadas, han sido instaladas una a una en los propios notebooks que se han requerido. Estas han sido las herramientas instaladas:

- **“os”**: la biblioteca “os” tiene funciones para interactuar con el sistema operativo, como en este caso para hacer uso de rutas de archivo o creación de archivos [5].
- **“numpy”**: es una biblioteca utilizada para la manipulación científica computacional en Python. Proporcionando funciones matemáticas, operadores algebraicos o trabajar con estructuras multidimensionales [4].
- **“pandas”**: biblioteca comúnmente usada para el análisis y manipulación de datos, mediante estructuras eficientes y herramientas para la ordenación de estos [6].
- **“scikit-learn”**: biblioteca de aprendizaje automáticos, que dispone de diversos algoritmos y herramientas para trabajar con la minería de datos y el análisis de datos [9].

- **“pydub.AudioSegment”**: es una biblioteca de pyhub con funciones como para cargar, manipular y exportar audios en diversos formatos [8].
- **“shutil”** : la biblioteca proporciona funciones de manipulación de archivos y directorios en Python [11].
- **“csv”**: biblioteca con funciones como para leer y escribir de manera sencilla en archivos con formato .csv [1].
- **“librosa”**: es una biblioteca de Python utilizada para el análisis de señales de audio, proporcionando una amplia variedad de funcionalidades de carga, procesamiento, analizadores y visualizadores [3].
- **“spicy”**: biblioteca que proporciona funcionalidades avanzadas del área de las matemáticas, ciencia e ingeniería y en este caso utilizada para el análisis de audios de voz mediante diferentes subbibliotecas [10].
- **“torchaudio”**: biblioteca de Pytorch con funcionalidades para el procesamiento de señales de audio [13].
- **“matplotlib”**: es una biblioteca para visualizar los datos con herramientas como gráficos, diagramas, histogramas, gráficos de barras... facilitando así la interpretación de estos [16].
- **“streamlit”**: una herramienta de código abierto que permite crear aplicaciones web con Python. Siendo una manera útil de crear prototipos de manera sencilla e intuitiva [12].

Respecto a los requisitos de hardware para ejecutar este proyecto únicamente es necesario una computadora. En este caso se ha hecho uso de un Portátil Asus VivoBook, con un procesador 11th Gen Intel(R) Core(TM) i7, con una memoria RAM de 8GB y un sistema operativo de Windows 11 Home.

## B.2. Instalación / Puesta en marcha

La puesta en marcha del prototipo se compone de varios pasos sencillos pero fundamentales.

Se comienza por asegurarse de que el archivo de formato “.py” está en la ubicación adecuada junto con todos los archivos necesarios para su correcta ejecución.


















 <i>características_Spicy</i>	2023/7/2 18:51	Fitxategi-karpeta	
 <i>Modelos</i>	2023/6/13 13:12	Fitxategi-karpeta	
 <i>paciente_prueba1</i>	2023/6/5 20:37	Fitxategi-karpeta	
 <i>paciente_prueba2</i>	2023/7/2 18:28	Fitxategi-karpeta	
 <i>paciente_prueba3</i>	2023/6/1 16:52	Fitxategi-karpeta	
 <i>paciente_prueba4</i>	2023/6/13 13:35	Fitxategi-karpeta	
 <i>paciente_prueba5</i>	2023/6/5 20:39	Fitxategi-karpeta	
 <i>paciente_prueba6</i>	2023/6/1 16:54	Fitxategi-karpeta	
 <i>Extraer_caract_Pytorch.py</i>	2023/5/24 23:19	PY fitxategia	4 KB
 <i>Extraer_caract_Spicy.py</i>	2023/5/24 23:18	PY fitxategia	4 KB
 <i>Preprocesado_Eliminar_silencios.py</i>	2023/6/5 18:49	PY fitxategia	2 KB
 <i>Preprocesado_Normalizar_volumen.py</i>	2023/5/24 23:52	PY fitxategia	1 KB
 <i>Prototipo.ipynb</i>	2023/7/2 18:53	IPYNB fitxategia	37 KB
 <i>streamlit_app.ipynb</i>	2023/7/2 18:53	IPYNB fitxategia	21 KB
 <i>streamlit_app.py</i>	2023/7/2 18:50	PY fitxategia	8 KB
 <i>Suma_prueba.py</i>	2023/5/25 00:42	PY fitxategia	1 KB
 <i>Union_dataset.py</i>	2023/5/28 16:29	PY fitxategia	1 KB

Figura B.5: Contenidos necesarios en el directorio de ejecución.

Fuente: Elaboración propia

En este caso tal y como podemos ver en la figura necesitará de los siguientes directorios y archivos:

- Una carpeta denominada *características\_Spicy*, la cual se encuentra vacía.
- El directorio *Modelos*, la cual contendrá los modelos que utilizará el programa a la hora de hacer las predicciones en formato *joblib*.
- El directorio con los audios del paciente a analizar. En este caso hay más de uno, pero esto es opcional.
- Y los archivos con las funciones necesarias para la ejecución del prototipo:
  - *Extraer\_caract\_Pytorch.py*
  - *Extraer\_caract\_Spicy.py*

- *Preprocesado\_Eliminar\_silencios.py*
- *Preprocesado\_Normalizar\_volumen.py*
- *Union\_dataset.py*
- El propio programa con la interfaz: *streamlit\_app.py*
- El notebook que contiene el programa interfaz: *streamlit\_app.ipynb*
- El programa: *Prototipo.ipynb*

Después, a través del uso de la terminal hay que moverse al directorio donde se encuentra el archivo con el siguiente comando: `cd C:\Ruta\Localización_del_prototipo`

Y una vez situados correctamente hay que ejecutar el siguiente comando: `streamlit run streamlit_app.py`

Abriéndose así en el navegador la interfaz. De tal manera, que se iniciará un servidor local, mostrando el URL de la ventana donde se ha abierto la interfaz. Pudiendo hacer uso de ella para diferentes pruebas a realizar.

```
>> python -m streamlit run streamlit_app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.29:8501
```

Figura B.6: URL creado al iniciar la puesta en marcha de la interfaz.

Fuente: Elaboración propia

## B.3. Manuales y/o Demostraciones prácticas

Para hacer un manual de uso del prototipo, se ha optado por explicar como es el prototipo visualmente. Donde podemos ver en la siguiente captura, que está compuesta por diferentes cuadrículas que deberán rellenarse con la información necesaria para poder hacer uso de ella. Por un lado, esta en el lado izquierdo arriba la entrada de la ruta donde se almacenan los audios de voz a analizar. Y por otro lado, están las cuadrículas correspondientes a la información personal del paciente. Seguido del botón para realizar la predicción.

El prototipo muestra una interfaz de usuario con un encabezado que incluye un botón de cierre (X) a la izquierda y un menú (≡) a la derecha. A la izquierda, hay un panel lateral con el título 'Ruta de la carpeta:' y un campo de entrada de texto. El área principal de la interfaz está titulada 'Clasificador de Audios' y contiene un formulario con los siguientes campos:

- Nombre: [Campo de entrada]
- 1º Apellido: [Campo de entrada]
- 2º Apellido: [Campo de entrada]
- Número de historia: [Campo de entrada]
- Edad: [Campo de entrada]
- Género: (Mujer/Hombre/Indefinido) [Campo de entrada]

Figura B.7: Formato del prototipo

Fuente: Elaboración propia

Tras ello, se devolverá el resultado con la siguiente estructura. Comienza por devolver los datos personales del paciente, seguido de los resultados obtenidos para cada tipo de audio analizado. Después, imprime como conclusión el resultado final conjunto de todas las predicciones. Además, devuelve también una representación visual mediante una gráfica de tarta que expresará en porcentajes en cuantos de los audios analizados han sido detectados los signos de la enfermedad de Parkinson. Todo ello, se puede ver en el ejemplo planteado.

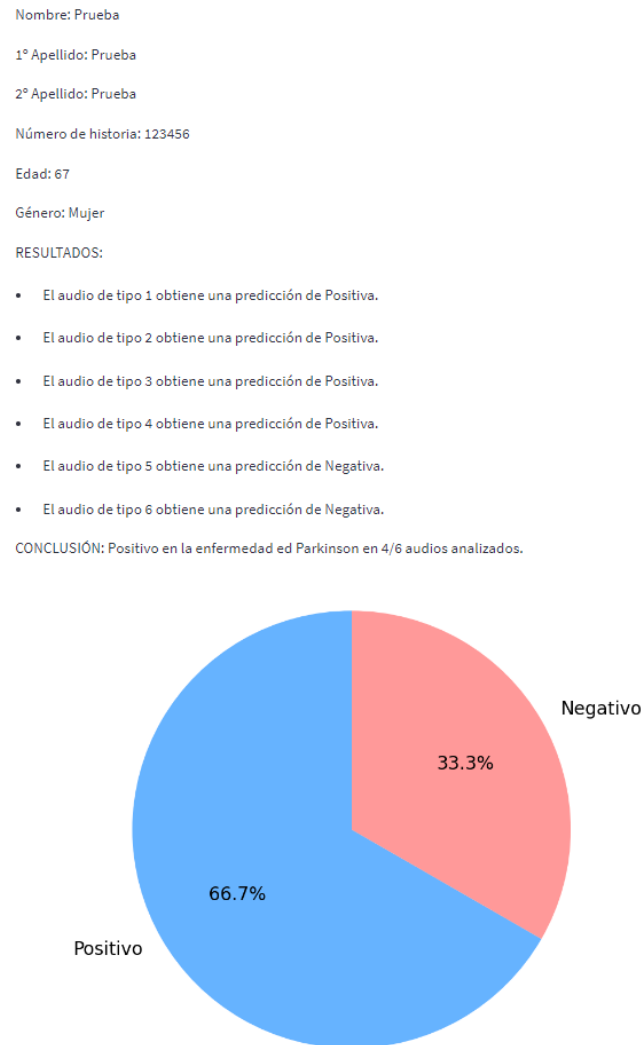


Figura B.8: Ejemplo respuesta del prototipo.

Fuente: Elaboración propia

Finalmente, para un mejor entendimiento de los pasos que se dan en una demostración como esta, se ha creado un diagrama de flujo que indicará de forma resumida el funcionamiento de los procesos que componen una ejecución del programa como esta [18].

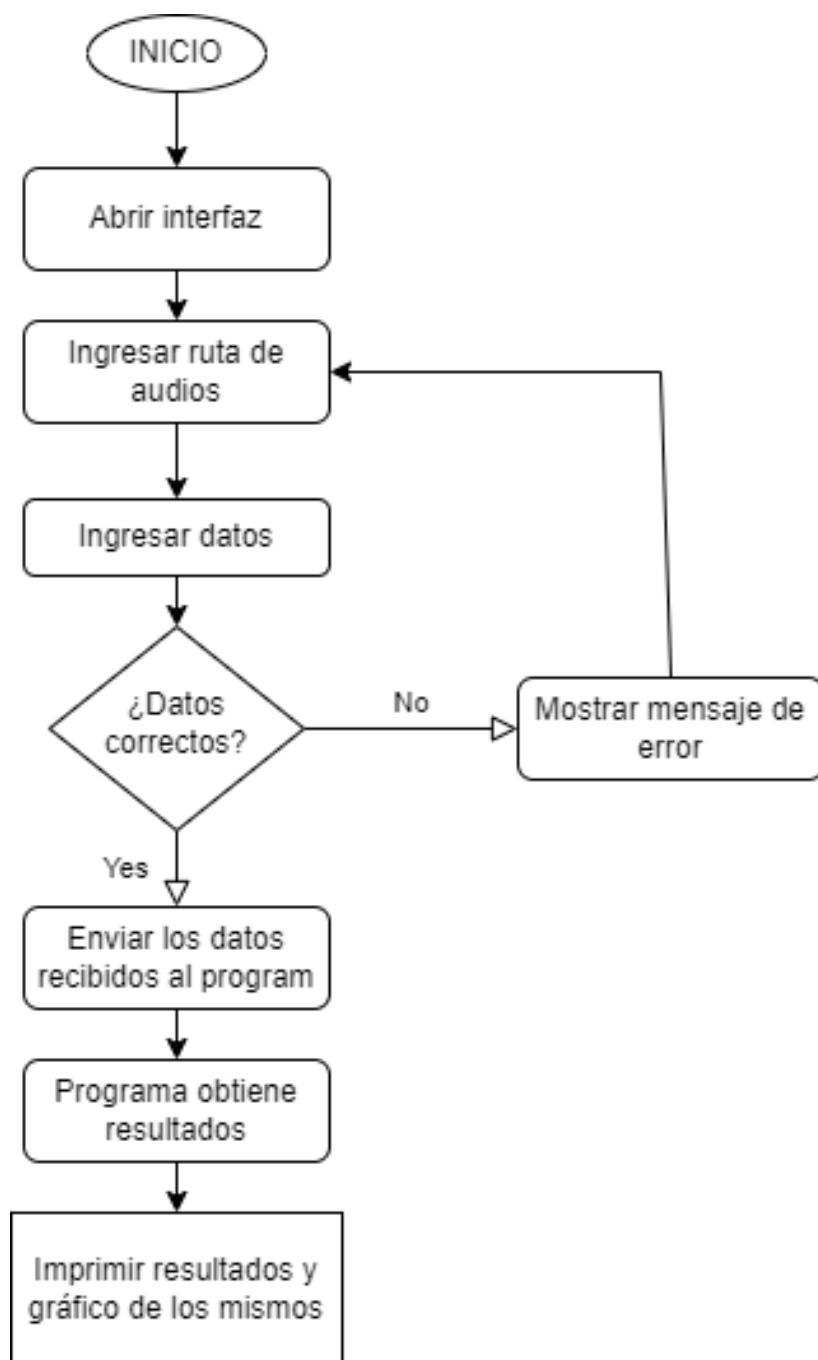


Figura B.9: Diagrama de flujo.

Fuente: Elaboración propia



## Apéndice C

---

# Manual del desarrollador / programador / investigador.

---

### C.1. Estructura de directorios

En este apartado se explica como esta organizada la estructura de los directorios que contendrán todo el proyecto en conjunto, de una manera ordenada y clara en el caso de querer acceder a cualquiera de los archivos o información que contienen.

En total esta ordenado todo dispuesto en tres directorios principales.

El primer directorio es el denominado ***Base\_de\_datos***, en este se encuentran otro dos directorios y tres archivos de datos:

- Directorio *Audios\_todos*: el cual contiene todos los audios con los que se ha trabajado a lo largo de todo el proyecto, identificados con una nomenclatura específica (explicada en el apartado descripción clínica de los datos en el anexo descripción de adquisición y tratamiento de datos), para poder identificar y diferenciar de manera clara cual es cual.
- Directorio *Documentos\_grabaciones*: contiene el documento de consentimiento para la recogida de audios, *Consentimiento Informado Grabacion Voz.pdf* y el documento aprobado por el comité ético para la realización del proyecto, *Aprobación Comité Ético Proyecto Grabación Audios .pdf*.
- Archivo *Datos\_control.xlsx*: con los datos del grupo control.

- Archivo *Datos\_sujetos.xlsx*: con los datos del grupo de sujetos.
- Archivo *Datos\_Parkinson.csv*: con los datos de ambos grupos de forma conjunta.

El segundo directorio principal está nombrado como **Memoria**. Este únicamente contiene una carpeta con todas las imágenes utilizadas a lo largo de la memoria. Y tanto la versión final de los anexos como de la memoria en formato *pdf* ambos.

El último directorio es el que contiene el programa completo por lo que ha sido denominado como **Programa**. Y en consecuencia el que más contenido tiene. Por ello, dentro del mismo se han diferenciado otras cuatro subcarpetas:

- Directorio **Clasificadores**:
  - Directorio *Clasificador\_arbol\_decision*:
    - Directorio *.ipynb\_checkpoints*:
      - ◇ *Clasificación Árboles de decisión -checkpoint.ipynb*
      - ◇ *Clasificación Árboles de decisión -Prueba-checkpoint.ipynb*
    - Directorio *modelos\_entrenados\_arbol\_decision*: contiene todos los modelos de clasificación de árbol de decisiones la entrenados para cada tipo de subconjunto y audios en formato “*joblib*”:
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo1\_Pythorch.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo1\_spicy.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo2\_Pythorch.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo2\_spicy.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo3\_Pythorch.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo3\_spicy.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo4\_Pythorch.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo4\_spicy.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo5\_Pythorch.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo5\_spicy.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo6\_Pythorch.joblib*
      - ◇ *modelo\_entrenado\_DecisionTreeClassifier\_tipo6\_spicy.joblib*

- Directorio *unio\_carac\_Pythorch*: contiene los archivos csv con las características de los audios extraídas con la librería Pythorch.
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_1.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_2.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_3.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_4.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_5.csv*
  - ◊ *unificacionv\_caracteristicas\_audios\_Vgg-pythorch\_6.csv*
- Directorio *union\_carac\_Spicy*: contiene los archivos csv con las características de los audios extraídas con la librería Spicy.
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_1.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_2.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_3.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_4.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_5.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_6.csv*
- Archivo *Clasificación Árboles de decisión .ipynb*: función para entrenar los modelos de entrenamiento con el algoritmo árbol de decisiones, evaluar el rendimiento de cada uno y almacenarlos en archivos con formato ".joblib".
- Archivo *Clasificador\_arbol\_decision\_metricas.csv*: las métricas obtenidas de cada entrenamiento de cada modelo.
- Directorio *Clasificador\_KNN*:
  - Directorio *.ipynb\_checkpoints*:
    - ◊ *Clasificación KNN-checkpoint.ipynb*
  - Directorio *modelos\_entrenados\_KNN*: contiene todos los modelos de clasificación de KNN entrenados para cada tipo de subconjunto y audios en formato ".joblib":
    - ◊ *modelo\_entrenado\_KNeighborsClassifier\_tipo1\_Pythorch.joblib*
    - ◊ *modelo\_entrenado\_KNeighborsClassifier\_tipo1\_spicy.joblib*
    - ◊ *modelo\_entrenado\_KNeighborsClassifier\_tipo2\_Pythorch.joblib*
    - ◊ *modelo\_entrenado\_KNeighborsClassifier\_tipo2\_spicy.joblib*
    - ◊ *modelo\_entrenado\_KNeighborsClassifier\_tipo3\_Pythorch.joblib*
    - ◊ *modelo\_entrenado\_KNeighborsClassifier\_tipo3\_spicy.joblib*

- ◇ *modelo\_entrenado\_KNeighborsClassifier\_tipo4\_Pythorch.joblib*
  - ◇ *modelo\_entrenado\_KNeighborsClassifier\_tipo4\_spicy.joblib*
  - ◇ *modelo\_entrenado\_KNeighborsClassifier\_tipo5\_Pythorch.joblib*
  - ◇ *modelo\_entrenado\_KNeighborsClassifier\_tipo5\_spicy.joblib*
  - ◇ *modelo\_entrenado\_KNeighborsClassifier\_tipo6\_Pythorch.joblib*
  - ◇ *modelo\_entrenado\_KNeighborsClassifier\_tipo6\_spicy.joblib*
- Directorio *unio\_carac\_Pythorch*: contiene los archivos csv con las características de los audios extraídas con la librería Pythorch.
  - ◇ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_1.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_2.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_3.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_4.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_5.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Vgg-pythorchv\_6.csv*
- Directorio *union\_carac\_Spicy*: contiene los archivos csv con las características de los audios extraídas con la librería Spicy.
  - ◇ *unificacion\_caracteristicas\_audios\_Scipy\_1.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Scipy\_2.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Scipyv\_3.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Scipy\_4.csv*
  - ◇ *unificacion\_caracteristicas\_audios\_Scipy\_5.csv*
  - ◇ *unificacion\_caracteristicasv\_audios\_Scipyv\_6.csv*
- Archivo *Clasificación KNN.ipynb*: función para entrenar los modelos de entrenamiento con el algoritmo KNN, evaluar el rendimiento de cada uno y almacenarlos en archivos con formato ".joblib".
- Archivo *Clasificador\_KNN\_metricas.csv*: las métricas obtenidas de cada entrenamiento de cada modelo.
- Directorio *Clasificador\_random\_forest*:
  - Directorio *.ipynb\_checkpoints*:
    - ◇ *Clasificación Random Forest-checkpoint.ipynb*
  - Directorio *modelos\_entrenados\_RandomForest*: contiene todos los modelos de clasificación de Random Forest entrenados para cada tipo de subconjunto y audios en formato ".joblib":

- ◊ *modelo\_entrenado\_RandomForest\_tipo1\_Pythorch.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo1\_spicy.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo2\_Pythorch.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo2\_spicy.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo3\_Pythorch.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo3\_spicy.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo4\_Pythorch.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo4\_spicy.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo4\_Pythorch.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo5\_spicy.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo6\_Pythorch.joblib*
- ◊ *modelo\_entrenado\_RandomForest\_tipo6\_spicy.joblib*
- Directorio *unio\_carac\_Pythorch*: contiene los archivos csv con las características de los audios extraídas con la librería Pythorch.
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_1.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_2.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_3.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_4.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_5.csv*
  - ◊ *unificacionv\_caracteristicas\_audios\_Vgg-pythorch\_6.csv*
- Directorio *union\_carac\_Spicy*: contiene los archivos csv con las características de los audios extraídas con la librería Spicy.
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_1.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_2.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_3.csv*
  - ◊ *unificacion\_caracteristicas\_audiosv\_Scipy\_4.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_5.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_6.csv*
- Archivo *Clasificación Ramdom Forest.ipynb*: función para entrenar los modelos de entrenamiento con el algoritmo Random Forest, evaluar el rendimiento de cada uno y almacenarlos en archivos con formato ".joblib".
- Archivo *Clasificador\_Random\_Forest\_metricas.csv*: las métricas obtenidas de cada entrenamiento de cada modelo.

- Directorio *preparar\_base\_datos\_entrenamiento*: donde se encuentran los archivos necesarios para preparar los datos antes de comenzar con los entrenamientos.
  - Directorio *unio\_carac\_Pythorch*: contiene los archivos csv con las características de los audios extraídas con la librería Pythorch.
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_1.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_2.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_3.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_4.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_5.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_6.csv*
  - Directorio *union\_carac\_Spicy*: contiene los archivos csv con las características de los audios extraídas con la librería Spicy.
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_1.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_2.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_3.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_4.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_5.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_6.csv*
  - *Preparar\_conjunto\_datos.ipynb*
- Directorio ***Extraccion\_caracteristicas***: donde se extraerán las características de los audios con dos librerías distintas creando así los dos subconjuntos de datos con los que se trabaja.
  - Directorio *Caracteristicas\_Pytorch*:
    - ◊ Directorio *.ipynb\_checkpoints*
    - ◊ *vgg-pytorch\_extaer\_carac-checkpoint.ipynb*
    - ◊ Directorio *csv\_tipos\_Pytorch*: almacenamiento de los conjuntos de datos ya extraídos.
      - ◊ *caracteristicas\_audios\_Vgg-pythorch\_1.csv*
      - ◊ *caracteristicas\_audios\_Vgg-pythorch\_2.csv*
      - ◊ *caracteristicas\_audios\_Vgg-pythorch\_3.csv*
      - ◊ *caracteristicas\_audios\_Vgg-pythorch\_4.csv*
      - ◊ *caracteristicas\_audios\_Vgg-pythorch\_5.csv*
      - ◊ *caracteristicas\_audios\_Vgg-pythorch\_6.csv*

- ◊ Archivo *Extraer\_caract\_Pytorch.py*
  - ◊ Archivo *textitvgg-pytorch\_extaer\_carac.ipynb*
- Directorio *Caracteristicas\_Spicy*: beginitemize
- Directorio *.ipynb\_checkpoints*
  - ◊ *Scipy\_extraer\_caracteristicas-checkpoint.ipynb*
- Directorio *csv\_tipos\_Spicy*: almacenamiento de los conjuntos de datos ya extraídos.
  - ◊ *caracteristicas\_audios\_Scipy\_1.csv*
  - ◊ *caracteristicas\_audios\_Scipy\_2.csv*
  - ◊ *caracteristicas\_audios\_Scipy\_3.csv*
  - ◊ *caracteristicas\_audios\_Scipy\_4.csv*
  - ◊ *caracteristicas\_audios\_Scipy\_5.csv*
  - ◊ *caracteristicas\_audios\_Scipy\_6.csv*
- Archivo *Extraer\_caract\_Spicy.py*
- Archivo *Scipy\_extraer\_caracteristicas.ipynb*
- Directorio *Union\_dataset\_completos*:
  - Directorio *.ipynb\_checkpoints*
    - ◊ *Union\_csv-checkpoint.ipynb*
  - Directorio *unio\_carac\_Pythorch*: almacenamiento de los conjuntos de datos tras unirlos.
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_1.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_2.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_3.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_4.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_5.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Vgg-pythorch\_6.csv*
  - Directorio *union\_carac\_Spicy*: almacenamiento de los conjuntos de datos tras unirlos
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_1.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_2.csv*
    - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_3.csv*

- ◊ *unificacion\_caracteristicas\_audios\_Scipy\_4.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_5.csv*
  - ◊ *unificacion\_caracteristicas\_audios\_Scipy\_6.csv*
  - Archivo *Union\_dataset.py*
  - Archivo *textitUnion\_csv.ipynb*
- Directorio ***Preprocesado***: este directorio contienen tanto las funciones para preprocesar los audios como ya los audios preprocesados y ordenados.
  - Directorio *.ipynb\_checkpoints*:
    - *Preprocesado de los audios-checkpoint.ipynb*
  - Directorio *Audios\_preprocesados*: contiene los audios tras preprocesarlos.
  - Directorio *Reordenar\_tipos*:
    - Directorio *.ipynb\_checkpoints*:
      - ◊ *Reordenar audios segun el tipo-checkpoint.ipynb*
    - Directorio *Tipos\_audios*: se compone de seis subcarpetas que contienen los audios de cada tipo correspondientes.
      - ◊ Directorio *palabra\_campana\_4*
      - ◊ Directorio *palabra\_gato\_5*
      - ◊ Directorio *palabra\_petaca\_6*
      - ◊ Directorio *vocal\_A\_1*
      - ◊ Directorio *vocal\_I\_2*
      - ◊ Directorio *vocal\_U\_3*
    - *Reordenar audios segun el tipo.ipynb*
  - *Preprocesado de los audios.ipynb*
  - *Preprocesado\_Eliminar\_silencios.py*
  - *Preprocesado\_Normalizar\_volumen.py*



- Directorio ***Proptotipo***: contiene todos los archivos y funciones necesarias para la correcta ejecución tanto del prototipo en el notebook, como para la ejecución de la interfaz.
  - Directorio *.ipynb\_checkpoints*:
    - *Prototipo-checkpoint.ipynb*
    - *streamlit\_app-checkpoint.ipynb*
  - Directorio *\_\_pycache\_\_*:
    - *Extraer\_caract\_Pytorch.cpython-39.pyc*
    - *Extraer\_caract\_Spicy.cpython-39.pyc*
    - *Preprocesado\_Eliminar\_silencios.cpython-39.pyc*
    - *Preprocesado\_Normalizar\_volumen.cpython-39.pyc*
    - *Union\_dataset.cpython-39.pyc*
  - Directorio *características\_Spicy*: carpeta vacía que necesita el prototipo para su correcta ejecución.
  - Directorio *Modelos*: contiene los modelos que necesita el prototipo.
    - *modelo\_entrenado\_KNeighborsClassifier\_tipo1\_spicy.joblib*
    - *modelo\_entrenado\_KNeighborsClassifier\_tipo2\_spicy.joblib*
    - *modelo\_entrenado\_KNeighborsClassifier\_tipo3\_spicy.joblib*
    - *modelo\_entrenado\_KNeighborsClassifier\_tipo4\_spicy.joblib*
    - *modelo\_entrenado\_KNeighborsClassifier\_tipo5\_spicy.joblib*
    - *modelo\_entrenado\_KNeighborsClassifier\_tipo6\_spicy.joblib*
  - Directorio *paciente\_prueba1*: tiene en su interior los seis audios del paciente de prueba 1.
  - Directorio *paciente\_prueba2*: tiene en su interior los seis audios del paciente de prueba 2.

- Directorio *paciente\_prueba3*: tiene en su interior los seis audios del paciente de prueba 3.
- Directorio *paciente\_prueba4*: tiene en su interior los seis audios del paciente de prueba 4.
- Directorio *paciente\_prueba5*: tiene en su interior los seis audios del paciente de prueba 5.
- Directorio *paciente\_prueba6*: tiene en su interior los seis audios del paciente de prueba 6.
- *Extraer\_caract\_Pytorch.py*
- *Extraer\_caract\_Spicy.py*
- *Preprocesado\_Eliminar\_silencios.py*
- *Preprocesado\_Normalizar\_volumen.py*
- *Prototipo.ipynb*
- *streamlit\_app.ipynb*
- *streamlit\_app.py*
- *Union\_dataset.py*

## C.2. Compilación, instalación y ejecución del proyecto

La compilación, instalación de las herramientas necesarias y la ejecución de este proyecto han sido explicadas en el apéndice B Documentación del usuario.

## *Apéndice D*

---

# **Descripción de adquisición y tratamiento de datos**

---

En este apartado, se hablará por un lado de los datos en crudo, esto es de los audios y la información obtenida en el proceso de recogida de audios en el apartado D.1 Descripción clínica de los datos.

Y por otro lado en cambio, de la base de datos creada tras extraer diversas características de esos mismos audios. La cual se utilizará para entrenar los modelos de clasificación y finalmente desarrollar el prototipo final denominado Clasificador de audios. Explicado todo detenidamente en el anexo D.2 Descripción formal de los datos.

## **D.1. Descripción clínica de los datos**

Los datos base para este proyecto han sido unas pistas de audio, grabadas con diversos móviles en formato “.wav”. En total se disponen de 390 pistas de audios, 198 audios provenientes de pacientes control, esto es, de pacientes que no padecen de la enfermedad de Parkinson, y 192 de pacientes diagnosticados con enfermedad de Parkinson en diferentes estadios.

Los audios pueden ser agrupados por su contenido. Y en este caso se pueden diferenciar seis tipos de audios, todos ellos nombrados en base al tipo de contenido de cada uno.

- Los nombrados que terminan en 1, contienen la vocal “A” mantenida de forma constante.

- Los que terminan en 2, contienen la vocal “I” mantenida de forma constante.
- Los que terminan en 3, contienen la vocal “U” mantenida de forma constante.
- Los que terminan en 4, contienen la palabra “campana”.
- Los que terminan en 5, contienen la palabra “gato”.
- Los que terminan en 6, contienen la palabra “petaca”.

A la hora de la recogida de los audios se han nombrado siguiendo un patrón específico, para poder diferenciar cada audio a que paciente pertenecían y el contenido de los audios.

El formato ha sido el siguiente: IDXX\_D-M-A\_Y.wav

- XX: número identificador del paciente.
- D-M-A: fecha de la grabación.
- Y: Identificador del contenido del audio (Número variable entre el 1 y el 6).

Además de los audios también se recopiló información relevante para el desarrollo del proyecto, así como la edad del paciente, el sexo del paciente y el estadio del desarrollo de la enfermedad de Parkinson en la escala *MDS – UPDRS*. En el caso de los pacientes control, este último fue rellenado con el valor de 0, indicativo de no haber signos de la enfermedad en la escala *MDS – UPDRS*.

Por otro lado, tras haber extraídos las características fonéticas de los audios, partiendo de dichos 390 audios, se ha obtenido diferentes conjuntos de datos. Cada conjunto está compuesto por características extraídas a través de un algoritmo que usa una librería distinta y que contiene el nombre de cada archivo de audio seguido de 10 columnas con los valores de diferentes características extraídas.

Finalmente se ha hecho un Join <sup>1</sup> con la información de los pacientes y con las características obtenidas en cada caso, logrando de esta manera un conjunto de datos con mucha información para poder utilizarlo al entrenar los modelos de clasificación automática.

---

<sup>1</sup>Función para combinar variables de dos o más conjuntos basándose en un campo común entre ellas.

## D.2. Descripción formal de los datos.

Se han creado en total dos conjuntos de datos distintos partiendo del análisis de todos los audios, para ello se han extraído diferentes características para componer cada conjunto. Además, para realizar el proceso de extracción de características se ha hecho uso de dos librerías distintas para cada caso.

El primer conjunto de características ha sido obtenido mediante la librería ‘*Spicy*’ mayoritariamente y se han extraído las siguientes características, 1º derivada de la frecuencia fundamental, 2º derivada de la frecuencia fundamental, Jitter, Shimmer, APQ, PPQ, energía logarítmica, amplitud media del audio y la duración del audio.

Todas ellas, se han almacenado en archivos de formato “.csv”, con una estructura por columnas por características, identificando cada fila con el nombre de cada archivo de audio, número de id del paciente y el tipo de audio al que se corresponde. Después, se ha hecho un *join* (con el punto de unión el id del paciente en ambos archivos), con otro archivo “.csv”, unificando de esta manera tanto los datos de las extracciones de las características como los datos clínicos relevantes de cada paciente. Obtenido de esta manera finalmente un conjunto de datos por cada tipo de audio con el que se trabajará.

Ejemplo del primer conjunto de datos creado, al que denominaremos de aquí en adelante como conjunto de características Spicy:

Genero	Edad	ID	Estadio	archivo	tipo	ff1	ff2
Mujer		78 ID151		0 ID151_16-2-2 1.0		3,83E-02	-1,41E-03
Mujer		56 ID152		0 ID152_17-2-2 1.0		2,90E-02	1,87E-02
Hombre		56 ID153		0 ID153_17-2-2 1.0		-2,25E+16	-1,50E+16
Mujer		66 ID154		0 ID154_17-2-2 1.0		-3,49E-01	-2,47E-01
Mujer		58 ID155		0 ID155_17-2-2 1.0		1,54E-01	1,08E-01
Mujer		58 ID156		0 ID156_17-2-2 1.0		3,77E-02	2,51E-02
Hombre		78 ID157		0 ID157_17-2-2 1.0		1,48E-02	9,85E-03
Hombre		67 ID158		0 ID158_17-2-2 1.0		-5,20E-03	-3,93E-03
Mujer		92 ID159		0 ID159_17-2-2 1.0		-4,70E-02	-3,13E-02
Mujer		50 ID160		0 ID160_19-02- 1.0		-1,92E-02	-2,09E-03
jitter	shimmer	log_energy	apq	ppq	Amplitud media	duracion	
-1,40E+15	1,82E-01	1,94E+16	2,27E+12	2,89E+15	2,57E+16	3,17E+00	
-8,44E+15	1,54E-01	2,03E+16	3,45E+16	4,65E+13	2,63E+16	1,57E+04	
-4,44E+16	1,55E-01	1,98E+15	1,05E+16	1,36E+16	2,22E+15	7,88E+03	
-2,84E+15	1,22E-01	1,90E+16	1,59E+13	2,15E+13	3,78E+16	4,26E+03	
-1,13E+16	1,14E-01	1,83E+16	3,37E+15	4,37E+15	1,86E+16	5,05E+03	
-3,03E+16	1,24E-01	2,06E+15	1,43E+13	1,92E+13	2,71E+16	1,05E+04	
-6,52E+15	8,91E-02	2,08E+15	3,16E+13	4,02E+16	3,45E+16	1,49E+04	
-2,77E+15	1,04E-01	2,04E+16	5,53E+12	7,12E+12	1,92E+16	7,04E+03	
-6,75E+15	1,23E-01	2,05E+16	1,39E+16	1,79E+16	2,22E+15	8,71E+03	
-3,13E+14	1,72E-01	1,54E+16	1,36E+16	4,03E+15	6,63E+15	1,41E+04	

Figura D.1: Ejemplo conjunto de datos extraídos con Spicy

Para el segundo conjunto de características, se ha utilizado la librería ‘*Torch*’ extrayendo las siguientes, MFCC, la amplitud en decibelios, media del espectrograma, densidad espectral de potencia (PSD), loudness y el centroide espectral. Haciendo con estas extracciones el mismo procedimiento que en el caso del primer conjunto. Obteniendo finalmente otros seis conjuntos diferentes de cada tipo de audio con estructura similar a los conjuntos anteriores.

Estructura del segundo conjunto de datos, al que a partir de ahora llamaremos conjunto de características torch:

Genero	Edad	ID	Estadio	archivo	tipo	amplitude_to	loudness	mfcc	psd	spectral_cent	spectrogram
Mujer	60	ID150		0	ID150_16-2-2 2.0	-3.997.069	-22.243.849	-16.495.598	34.336.107	26.542.778	0.64969814
Mujer	78	ID151		0	ID151_16-2-2 2.0	-351.483	-14.331.858	-14.778.366	15.522.028	21.258.718	3.061.758
Mujer	56	ID152		0	ID152_17-2-2 2.0	-36.855.324	-11.617.362	-14.761.337	24.702.545	19.516.483	48.062.606
Hombre	56	ID153		0	ID153_17-2-2 2.0	-46.072.514	-13.019.314	-15.618.957	1.924.682	13.798.083	37.850.149
Mujer	66	ID154		0	ID154_17-2-2 2.0	-3.876.945	-12.851.757	-142.731.285	22.731.693	1.229.181	43.928.537
Mujer	58	ID155		0	ID155_17-2-2 2.0	-38.814.064	-15.865.065	-15.989.177	10.339.837	16.816.471	20.375.626

Figura D.2: Ejemplo conjunto de datos extraídos con Pytorch

Con los conjuntos de datos creados, se ha utilizado tres modelos de calificación diferentes para entrenarlos. Y para entrenar estos modelos, se ha optado por seguir el proceso de validación cruzada con cada subconjunto de datos (recordar que los subconjuntos estarán compuestos con la información extraída de 65 audios, todos del mismo tipo).

## *Apéndice E*

---

# Manual de especificación de diseño

---

Para explicar de forma visual el diseño procedimental, esto es, el enfoque del diseño de software de la estructura y funcionamiento para realizar una tarea en objetivo, en este caso obtener resultados de si son detectables los signos de la enfermedad de Parkinson en los audios de voz de los pacientes, se ha hecho uso de un diagrama de secuencias.

Ya que es muy útil para representar las interacciones entre los diferentes objetos que componen el programa a lo largo del tiempo y con el orden indicado [14].

Tal y como podemos visualizar en el diagrama de secuencias el total se pueden diferenciar tres objetos, donde la interfaz será el intermediario entre el usuario y el programa.

Por ello, tal y como se representa en el diagrama, se comenzará con el ingreso de los datos necesarios en la interfaz por parte del usuario, siendo estos mandados por el interfaz al propio programa. Donde se harán los procedimientos necesarios, para finalmente obtener un resultado y graficarlo. Este mismo resultado y gráfico tiene que devolverse al usuario, lo cual lo hace la interfaz, siendo de nuevo el intermediario mensajero. Remarcando la importancia de una interfaz intuitiva y de fácil uso para que la comunicación entre el usuario y el programa sea posible.

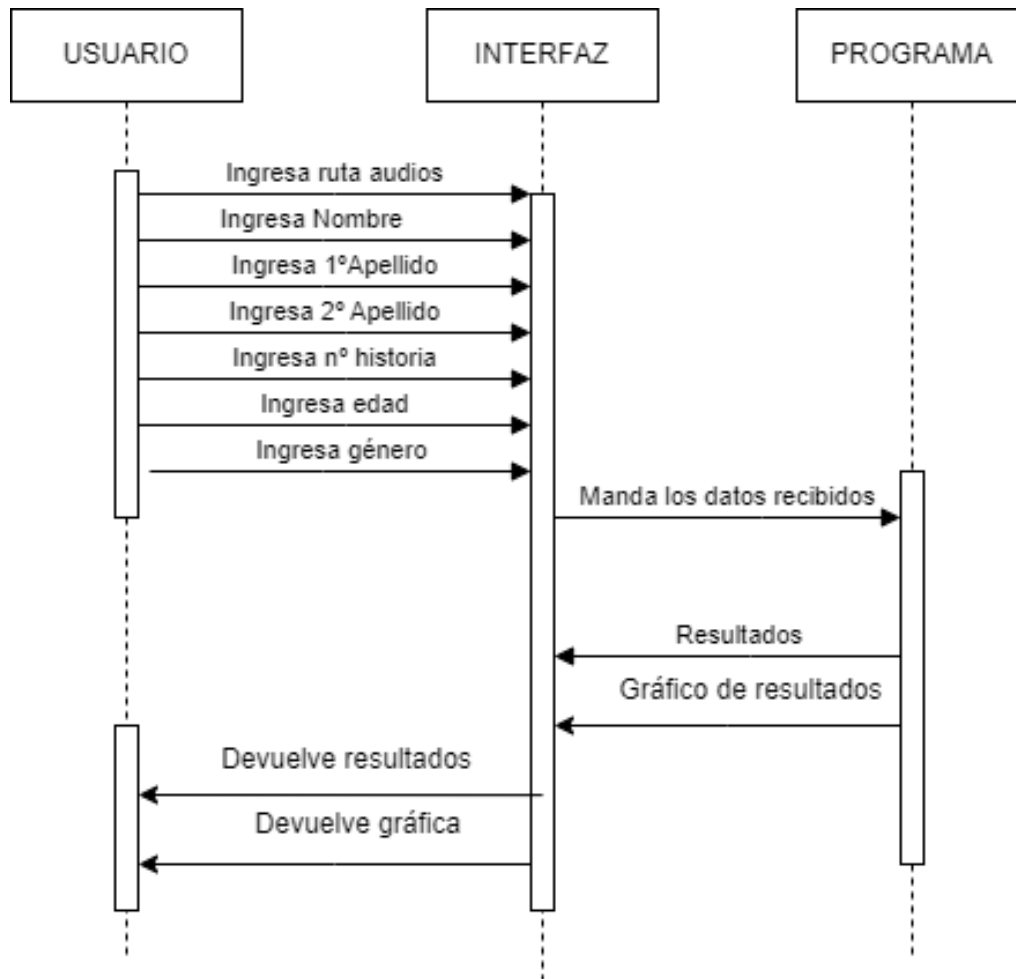


Figura E.1: Diagrama de secuencias.

Fuente: Elaboración propia

## E.1. Diseño arquitectónico

El diseño arquitectónico de un programa es la descripción de la estructura y organización que tiene el diseño según los componentes que tenga. Representando el papel de cada uno y las funciones que realiza. Por ello, se ha optado por el uso de un diagrama de clases para la representación visual del diseño del mismo, para expresar los componentes y las relaciones entre ellos.

De tal manera, que cada cuadrícula representa una clase, con sus atributos y métodos asociados, así como las interacciones entre ellos[2].



Este ha sido el diagrama de clases realizados para expresar el diseño arquitectónico del programa realizado.

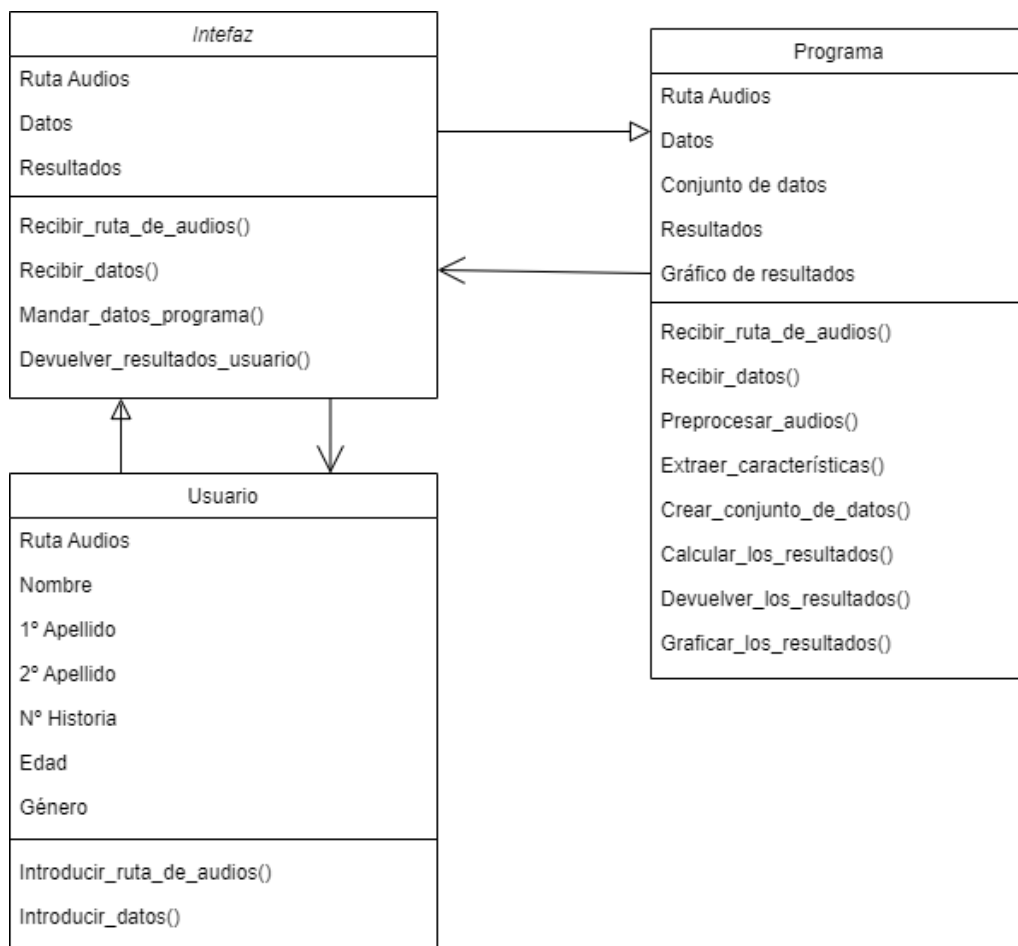


Figura E.2: Diagrama de clases.

Fuente: Elaboración propia

En este diagrama, podemos diferenciar tres clases, la clase del usuario, que tendrá como atributos los datos de los que dispone y los métodos que realiza, mandándole su información a la clase de la interfaz, que obtendrá del anterior los atributos necesarios para realizar sus funciones y interactuar con la última clase que representa al programa. El cual, obtiene como atributos además de los recibidos, los creados por el mismo mediante distintos métodos y tras realizar todos los procesos necesarios devuelve los resultados a la interfaz y esta al usuario.



## *Apéndice F*

---

# **Especificación de Requisitos**

---

### **F.1. Diagrama de casos de uso**

Un diagrama de casos de uso es una representación para poder identificar los roles de un sistema y la interacción de este con el programa en este caso. Su importancia radica en que reúne los requisitos mínimos para el uso del programa, siendo en este caso unas entradas de datos y una ruta de carpeta que contienen los audios a estudiar[15].

Este es el diagrama creado para la representación del uso del este programa.

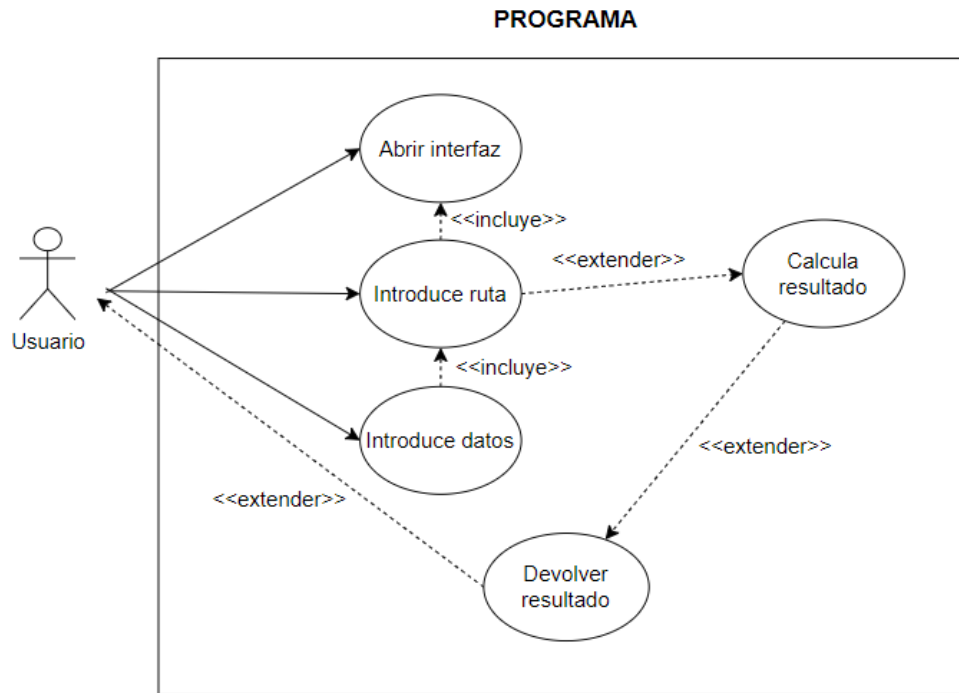


Figura F.1: Diagrama de casos de uso.

Fuente: Elaboración propia

## F.2. Explicación casos de uso.

Este diagrama de casos de uso es muy simple, ya que únicamente tiene un rol, el del usuario. Este será el encargado de introducir la información necesaria para el correcto funcionamiento del programa. De esta manera, es el que introduce tanto la ruta del directorio que contiene los audios a analizar como los datos personales del paciente. Cabe mencionar que, para introducir los datos, será necesario meter la ruta anteriormente y en consecuencia haber abierto el programa para ello.

Después, dentro del programa, se realizarán los procesos necesarios para finalmente poder calcular el resultado final. Estos mismos procesos, serán realizados de forma encadena en consecuencia de haber introducidos todos los datos necesarios de forma correcta. Además, este resultado será devuelto al usuario, de forma textual como con una representación gráfica.

## *Apéndice G*

---

# **Estudio experimental**

---

### **G.1. Cuaderno de trabajo.**

Se ha utilizado el siguiente apartado para hacer una demostración práctica del prototipo se han realizado pruebas con 6 pacientes distintos. Para comprobar que el funcionamiento es el correcto, se hará el análisis del caso de tres pacientes con el prototipo programado y ejecutado directamente en el notebook. Y el análisis de los otros tres, haciendo uso de la interfaz desarrollada.

Dichos pacientes han sido seleccionados con distintas características, para comprobar el correcto funcionamiento del programa frente a casos distintos. Empezamos con los ejemplos de los pacientes 1, 3 y 5, con el programa en el notebook.

El paciente 1, se sabe que padece de Parkinson, por lo que vamos a comprobar que efectivamente el resultado obtenido con el prototipo es el mismo. Y en consecuencia su conclusión es adecuada.

Este paciente tiene una edad de 72 años y es Mujer. En este caso le denominaremos paciente ID107, para mantener la confidencialidad del mismo.

Primeramente, hay que introducir los datos necesarios mediante inputs:

```

Ruta directorio con audios:.\paciente_prueba1\
Nombre:ID107
1º Apellido:ID107
2º Apellido:ID107
Número de historia:1111111
Edad:72
Género: (Mujer/Hombre/Indefinido)Mujer

```

Figura G.1: Introducción de los datos paciente ejemplo.

Fuente: Elaboración propia

Tras ejecutarlo el resultado se imprimirá de la siguiente manera.

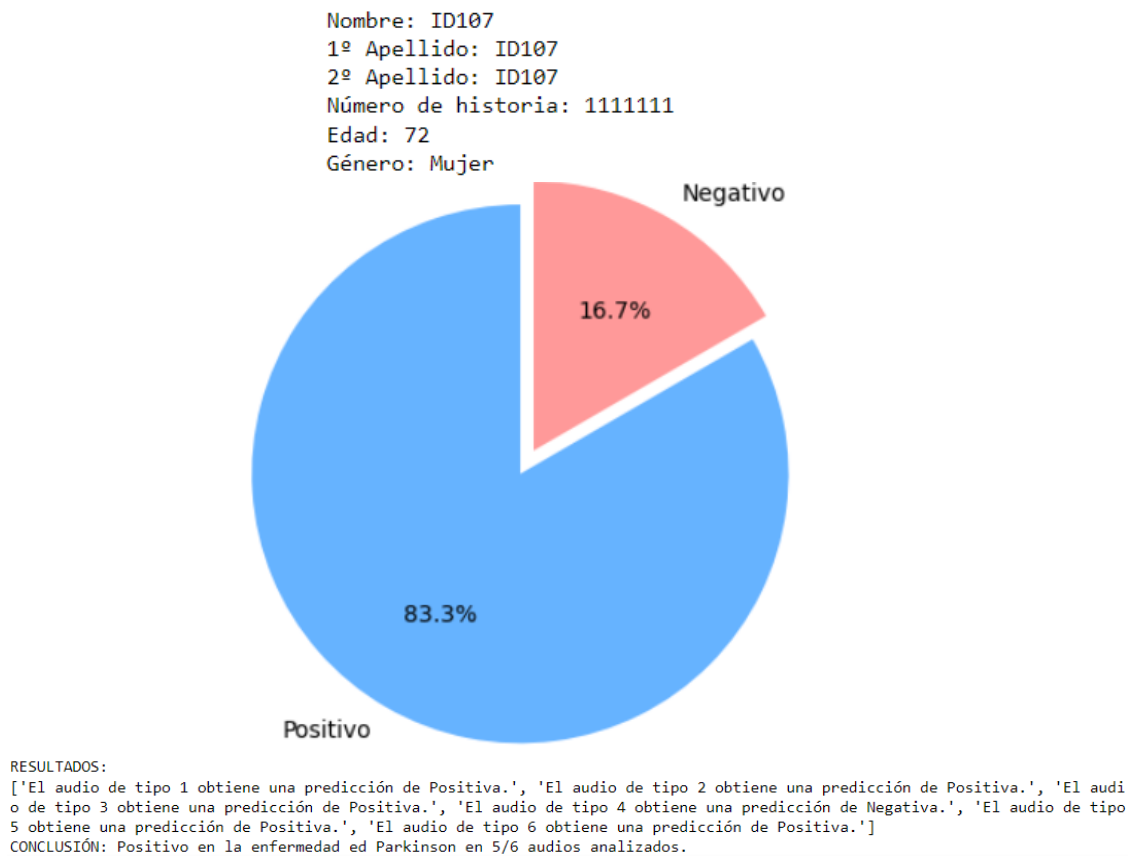


Figura G.2: Resultado paciente ejemplo.

Fuente: Elaboración propia

Donde se puede visualizar que las conclusiones coinciden con el diagnóstico de este mismo paciente y el prototipo ha resultado que son detectables los signos de la enfermedad de Parkinson en 5 de los 6 audios analizados.

Para el caso del paciente 3, también estamos ante un caso positivo de la enfermedad de Parkinson. En este caso se trata de una mujer de 74 años, a la cual se le ha adjudicado un identificador, ID123.

Se introducen los datos de la siguiente manera.

```
Ruta directorio con audios:.\paciente_prueba3\  
Nombre:ID123  
1º Apellido:ID123  
2º Apellido:ID123  
Número de historia:3333333  
Edad:74  
Género: (Mujer/Hombre/Indefinido)Mujer
```

Figura G.3: Introducción de los datos paciente ejemplo.

Fuente: Elaboración propia

Y devuelve el siguiente resultado, donde vuelve a confirmarse que efectivamente este paciente padece de la enfermedad de Parkinson. Detectada en los 6 audios disponibles.

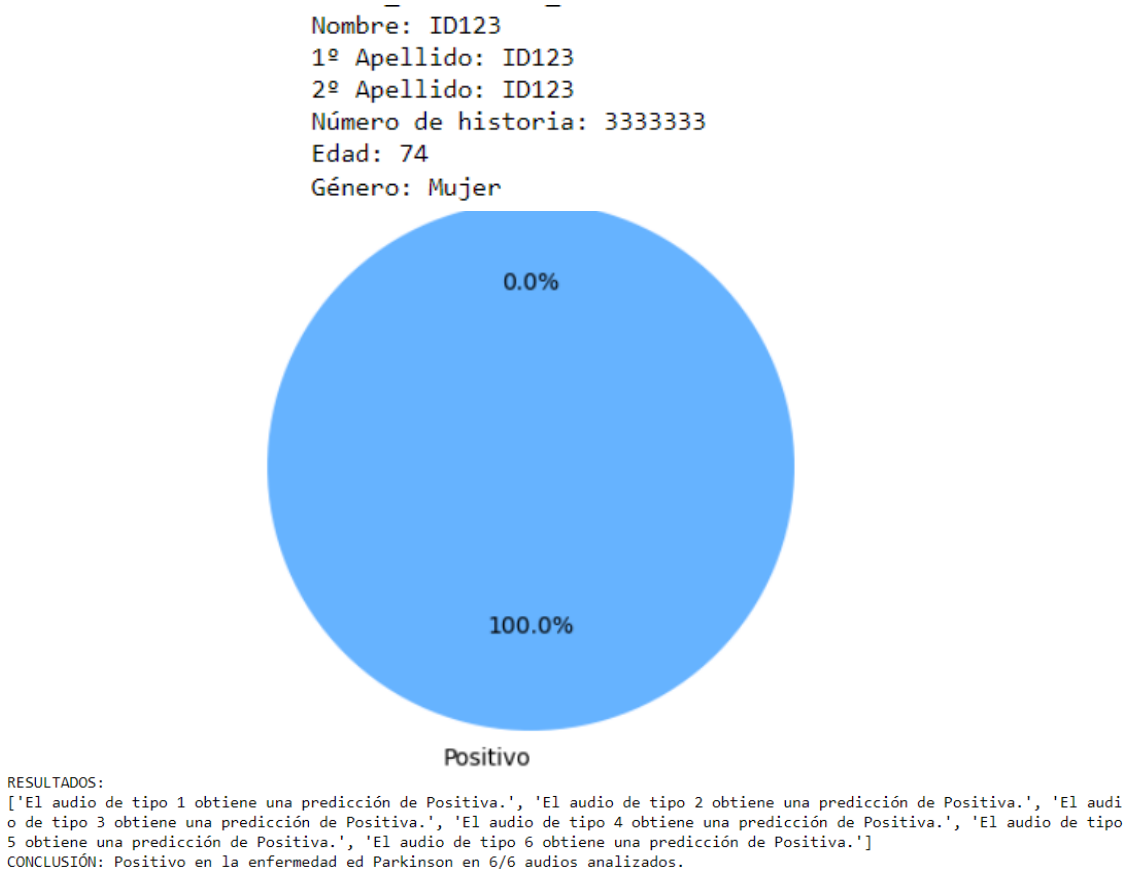


Figura G.4: Resultado paciente ejemplo.  
Fuente: Elaboración propia



Como tercer ejemplo en el uso del prototipo tenemos al paciente 5, el cual no está diagnosticado con la enfermedad, por lo que los resultados deberían ser negativos. Esta mujer de 50 años, será identificada con el ID160.

Se introducen sus datos de igual manera a los casos anteriores.

```
Ruta directorio con audios:.\paciente_prueba5\  
Nombre:ID160  
1º Apellido:ID160  
2º Apellido:ID160  
Número de historia:5555555  
Edad:50  
Género: (Mujer/Hombre/Indefinido)Mujer
```

Figura G.5: Introducción de los datos paciente ejemplo.

Fuente: Elaboración propia

Donde se obtienen los siguientes resultados. Confirmando que este paciente no presenta signos de la enfermedad de Parkinson en ninguno de los audios analizados.

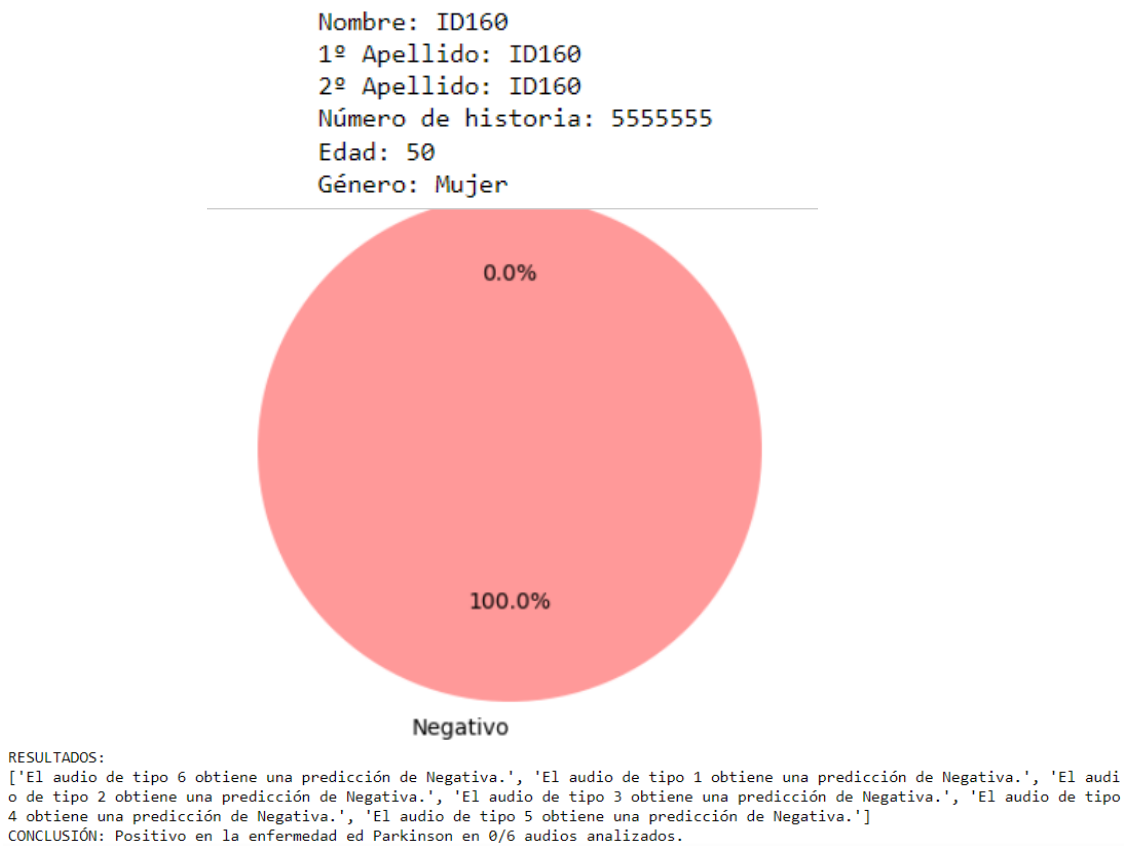


Figura G.6: Resultado paciente ejemplo 3.

Fuente: Elaboración propia

Por otro lado, se hará la prueba los pacientes 2, 4 y 6, pero en este caso haciendo uso de la interfaz para ello.

El paciente 2, denominado con un ID111, es una mujer de 78 años, la cual esta ya previamente diagnosticada con la enfermedad de Parkinson. Se comienza introduciendo los datos de la siguiente manera.

Ruta de la carpeta:

C:\Users\Usuario\Desktop\ingeniería de la t

Clasificador de Audios

Nombre:

ID111

1º Apellido:

ID111

2º Apellido:

ID111

Número de historia:

2222222

Edad:

78

Género: (Mujer/Hombre/Indefinido)

Mujer

Realizar predicción

Figura G.7: Introducción de los datos paciente ejemplo.

Fuente: Elaboración propia

Obteniendo así el siguiente resultado, donde efectivamente indica que se han detectado signos de la enfermedad en 4 de los 6 audios analizados.

Nombre: ID111

1° Apellido: ID111

2° Apellido: ID111

Número de historia: 2222222

Edad: 78

Género: Mujer

RESULTADOS:

- El audio de tipo 1 obtiene una predicción de Positiva.
- El audio de tipo 2 obtiene una predicción de Positiva.
- El audio de tipo 3 obtiene una predicción de Positiva.
- El audio de tipo 4 obtiene una predicción de Positiva.
- El audio de tipo 5 obtiene una predicción de Negativa.
- El audio de tipo 6 obtiene una predicción de Negativa.

CONCLUSIÓN: Positivo en la enfermedad ed Parkinson en 4/6 audios analizados.

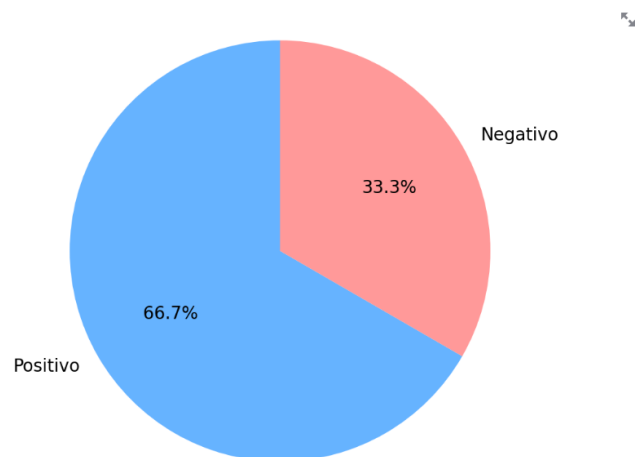


Figura G.8: Resultado paciente ejemplo.

Fuente: Elaboración propia

Para el ejemplo del paciente 4, tenemos un hombre de 67 años, al cual se le ha asignado el ID182. Y se trata de un paciente sin enfermedad de Parkinson diagnosticada.

Donde tras introducir los datos como se puede visualizar en la imagen de abajo.

Ruta de la carpeta:

C:\Users\Usuario\Desktop\ingeniería de la s

Clasificador de Audios

Nombre:

ID182

1º Apellido:

ID182

2º Apellido:

ID182

Número de historia:

4444444

Edad:

67

Género: (Mujer/Hombre/Indefinido)

Hombre

Realizar predicción

Figura G.9: Introducción de los datos paciente ejemplo.

Fuente: Elaboración propia

Devolverá el resultado de la siguiente manera, donde efectivamente indica que no ha encontrado signos de la enfermedad en ninguno de los audios.

Nombre: ID182

1º Apellido: ID182

2º Apellido: ID182

Número de historia: 4444444

Edad: 67

Género: Hombre

RESULTADOS:

- El audio de tipo 1 obtiene una predicción de Negativa.
- El audio de tipo 2 obtiene una predicción de Negativa.
- El audio de tipo 3 obtiene una predicción de Negativa.
- El audio de tipo 4 obtiene una predicción de Negativa.
- El audio de tipo 5 obtiene una predicción de Negativa.
- El audio de tipo 6 obtiene una predicción de Negativa.

CONCLUSIÓN: Positivo en la enfermedad ed Parkinson en 0/6 audios analizados.

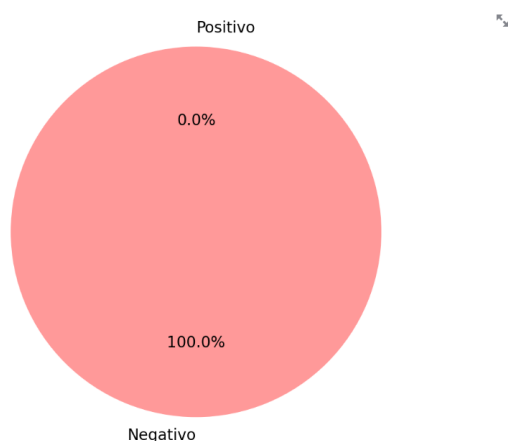


Figura G.10: Resultado paciente ejemplo.

Fuente: Elaboración propia

Por último, tenemos el ejemplo del paciente 6, que es una mujer de 74 años, sin diagnóstico de Parkinson y denominada con el ID174.

Se introducen los datos de igual manera.

Ruta de la carpeta:

C:\Users\Usuario\Desktop\Ingeniería de la s

Clasificador de Audios

Nombre:

ID174

1º Apellido:

ID174

2º Apellido:

ID174

Número de historia:

666666

Edad:

74

Género: (Mujer/Hombre/Indefinido)

Mujer

Realizar predicción

Figura G.11: Introducción de los datos paciente ejemplo 3.

Fuente: Elaboración propia

Con el siguiente resultado. Donde se confirma que no se ha encontrado ningún signo aparente de padecer la enfermedad para el caso de los 6 audios distintos.

Nombre: ID174

1° Apellido: ID174

2° Apellido: ID174

Número de historia: 666666

Edad: 74

Género: Mujer

RESULTADOS:

- El audio de tipo 1 obtiene una predicción de Negativa.
- El audio de tipo 2 obtiene una predicción de Negativa.
- El audio de tipo 3 obtiene una predicción de Negativa.
- El audio de tipo 4 obtiene una predicción de Negativa.
- El audio de tipo 5 obtiene una predicción de Negativa.
- El audio de tipo 6 obtiene una predicción de Negativa.

CONCLUSIÓN: Positivo en la enfermedad ed Parkinson en 0/6 audios analizados.

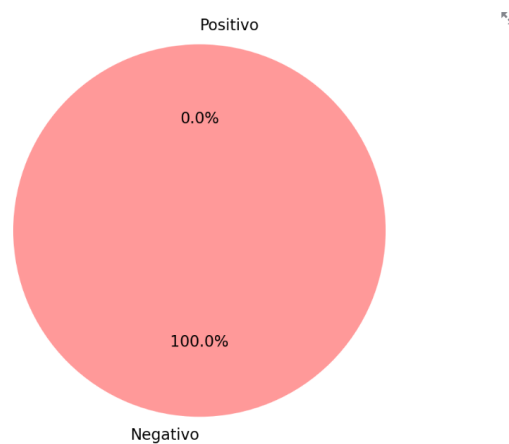


Figura G.12: Resultado paciente ejemplo 3.

Fuente: Elaboración propia

Tras la ejecución de los seis paciente ejemplos, se puede determinar que tanto el prototipo como la interfaz tiene una alta efectividad a la hora de detectar correctamente la enfermedad de Parkinson en todos los ejemplos realizados. Corroborando de esta manera los altos valores de rendimiento que ha obtenido en las métricas del modelo de entramiento para la composición



del prototipo, explicados con detenimiento en el apartado de resultados y discusión de la memoria.

## G.2. Detalle de resultados.

En esta sección se han incluido todos los resultados, representados con un gráfico para una mejor visualización de cada caso. La síntesis y análisis de los mismos están redactados en la memoria.

Son gráficos de barras donde en el eje Y se representan las cuatro métricas más relevantes para evaluar el rendimiento de un modelo y en el eje X, se representa la escala de los valores que pueden tener cada uno de ellos.

Empezamos con la comparación de los dos conjuntos para los audios de tipo 1 (vocal A) con el modelo de árbol de decisiones.

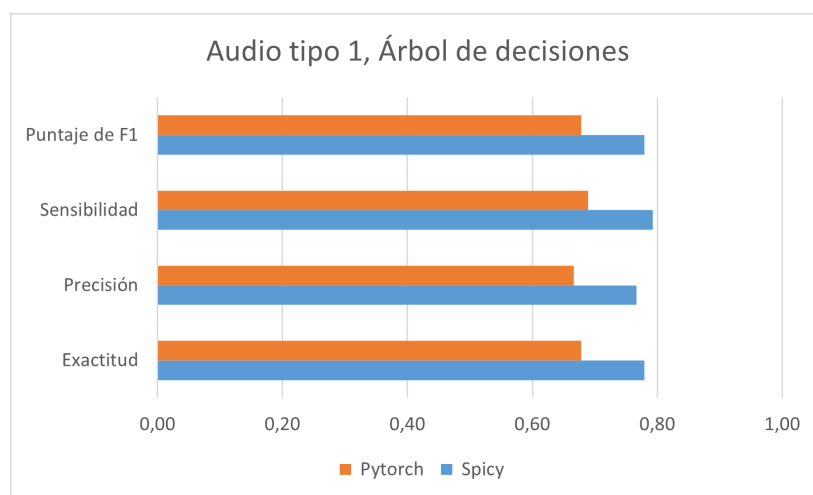


Figura G.13: Audio tipo 1 (vocal A), árbol de decisiones.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 2 (vocal I) (vocal I) con el modelo de árbol de decisiones.

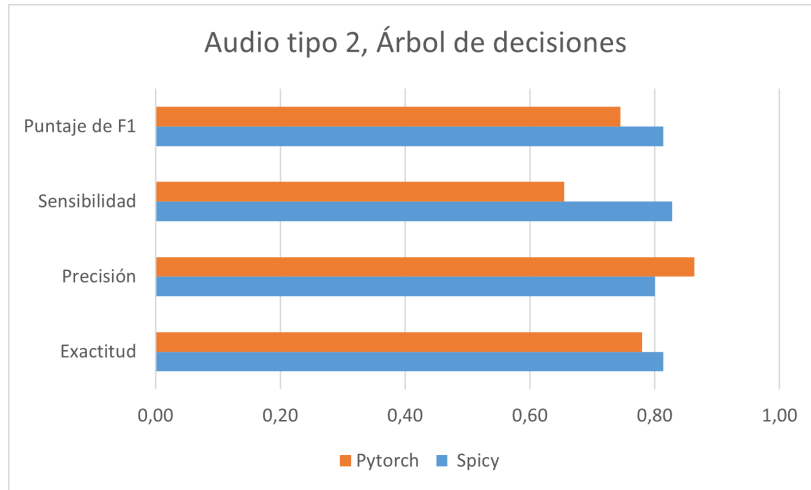


Figura G.14: Audio tipo 2 (vocal I), árbol de decisiones.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 3 (vocal U) con el modelo de árbol de decisiones.

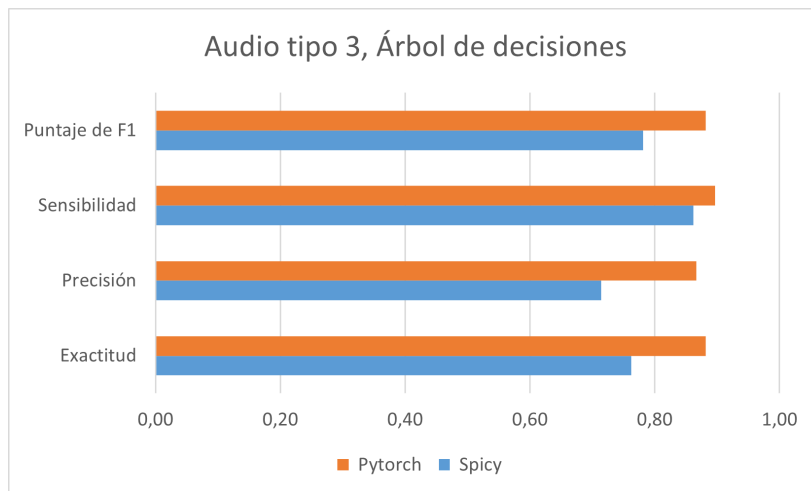


Figura G.15: Audio tipo 3 (vocal U), árbol de decisiones.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 4 (palabra campana) con el modelo de árbol de decisiones.

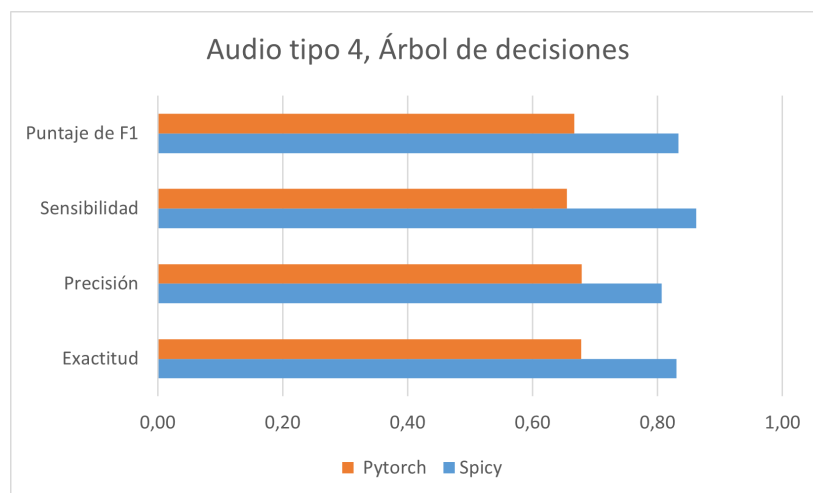


Figura G.16: Audio tipo 4 (palabra campana), árbol de decisiones.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 5 (palabra gato) con el modelo de árbol de decisiones.

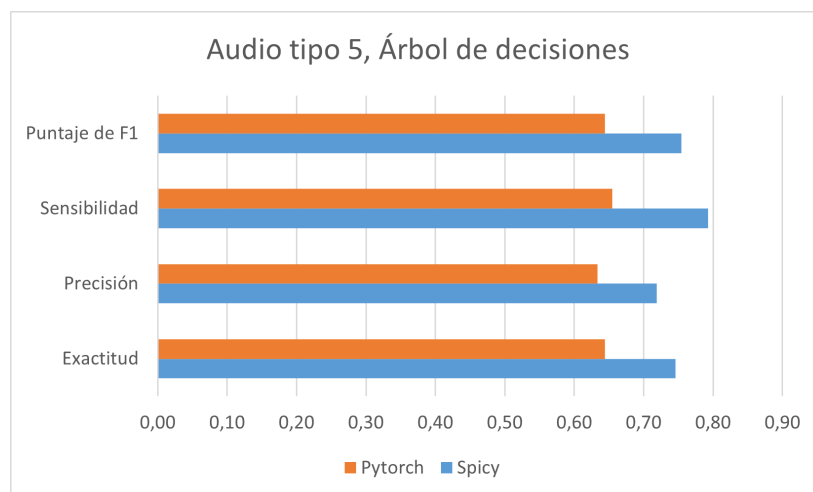


Figura G.17: Audio tipo 5 (palabra gato), árbol de decisiones.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 6 (palabra petaca) con el modelo de árbol de decisiones.

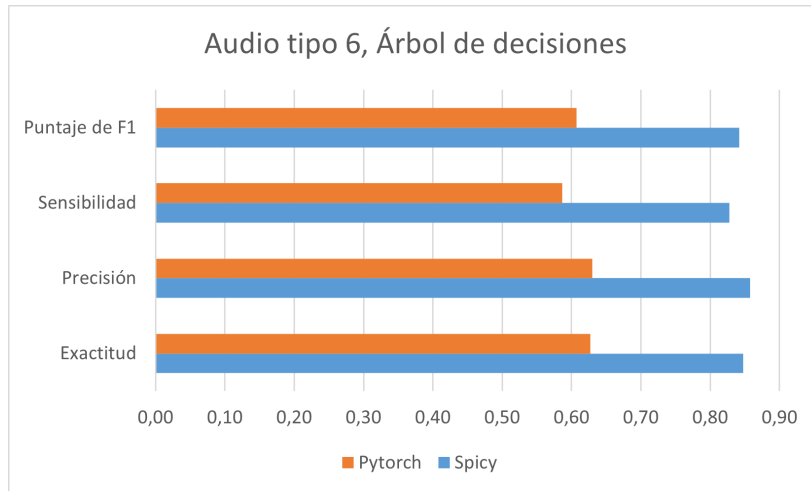


Figura G.18: Audio tipo 6 (palabra petaca), árbol de decisiones.

Fuente: Elaboración propia

Seguiremos haciendo la comparación, pero en este caso con las métricas obtenidas con el modelo de clasificación de KNN.

Comparación de los dos conjuntos para los audios de tipo 1 (vocal A) con el modelo de KNN.

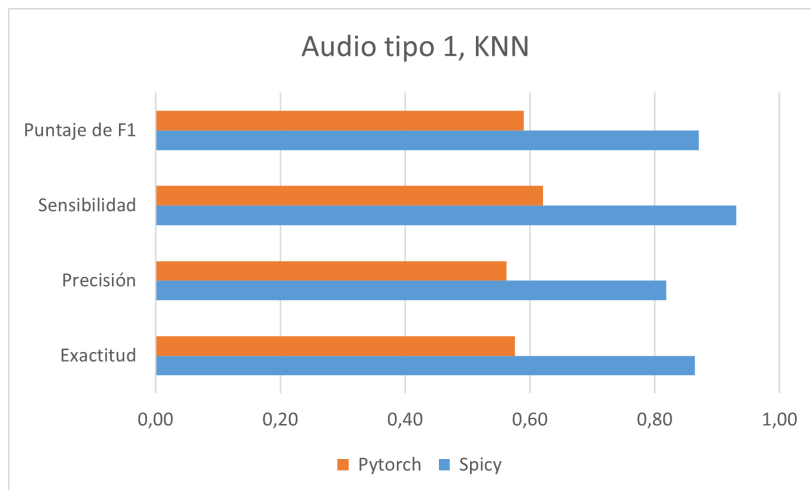


Figura G.19: Audio tipo 1 (vocal A), KNN.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 2 (vocal I) con el modelo de KNN.

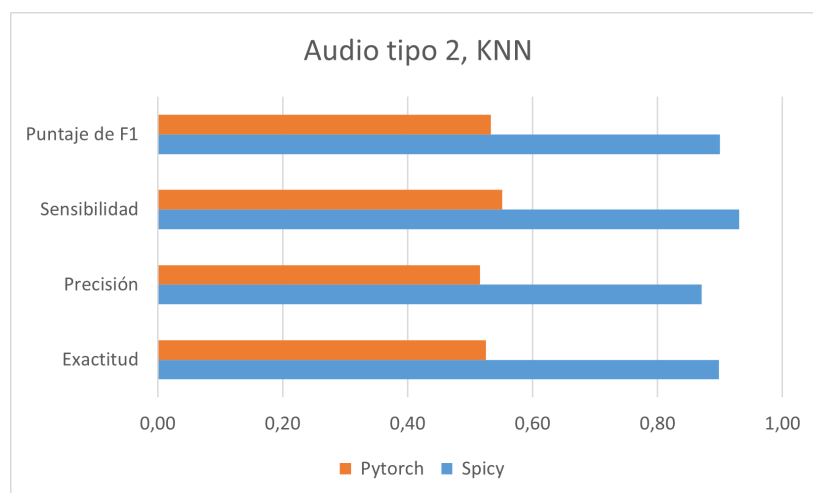


Figura G.20: Audio tipo 2 (vocal I), KNN.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 3 (vocal U) con el modelo KNN.

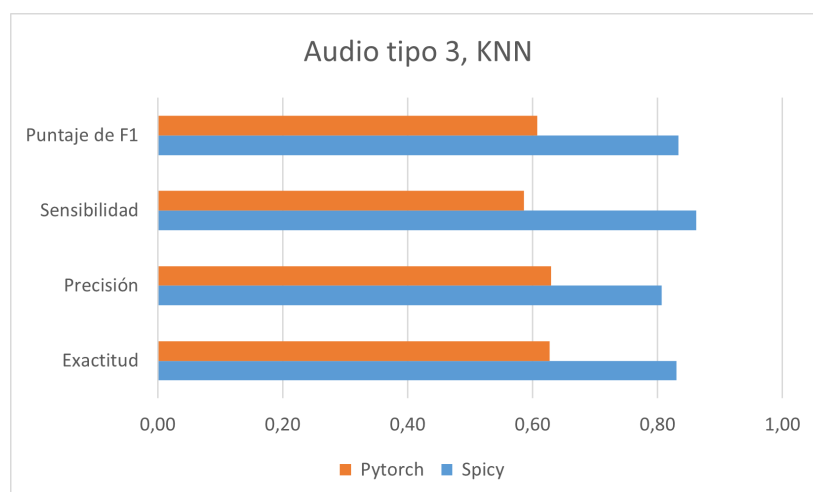


Figura G.21: Audio tipo 3 (vocal U), KNN.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 4 (palabra campana) con el modelo de KNN.

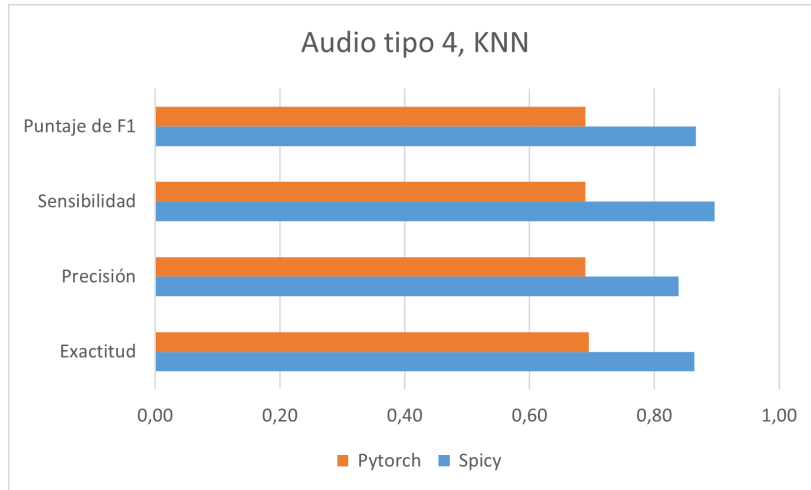


Figura G.22: Audio tipo 4 (palabra campana), KNN.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 5 (palabra gato) con el modelo de KNN.

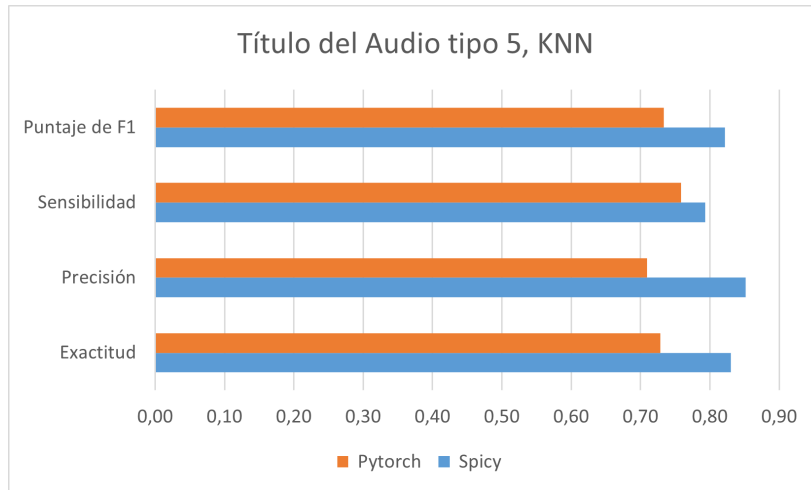


Figura G.23: Audio tipo 5 (palabra gato), KNN.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 6 (palabra petaca) con el modelo de KNN.

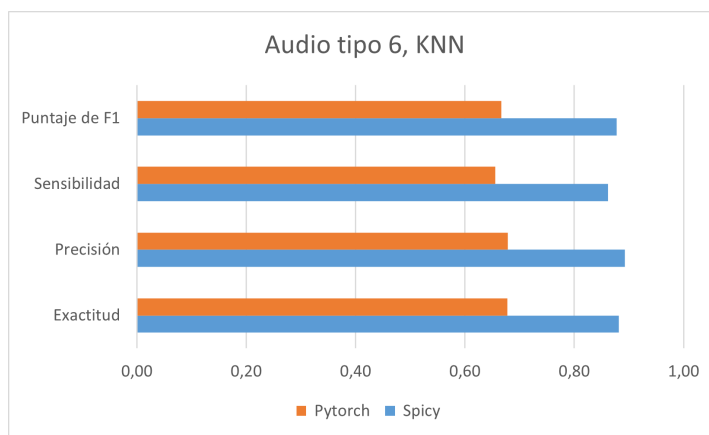


Figura G.24: Audio tipo 6 (palabra petaca), KNN.

Fuente: Elaboración propia

Por último, vamos a hacer uso de las métricas obtenidas con el modelo de clasificación Random Forest.

Comparación de los dos conjuntos para los audios de tipo 1 (vocal A) con el modelo de Random Forest.

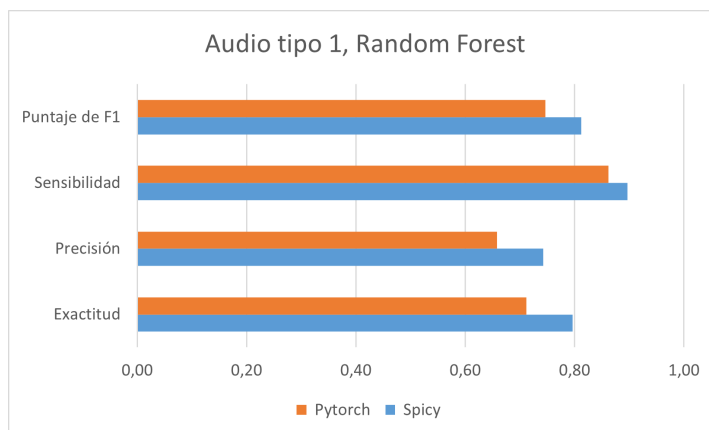


Figura G.25: Audio tipo 1 (vocal A), Random Forest.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 2 (vocal I) con el modelo de Random Forest.

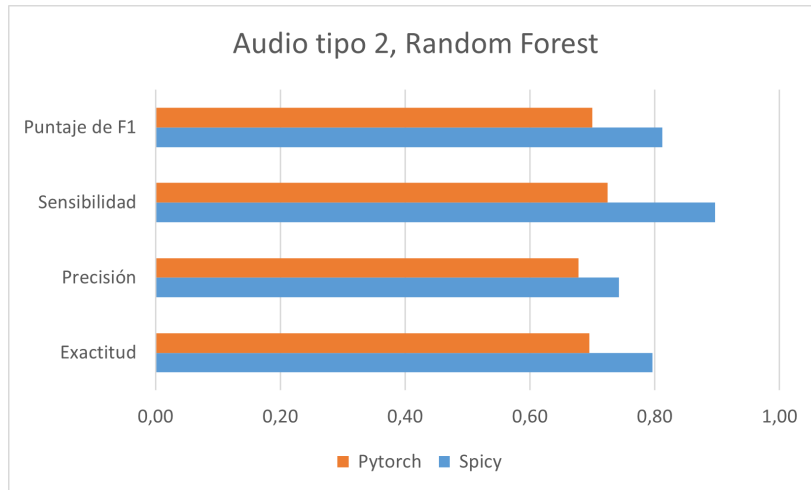


Figura G.26: Audio tipo 2 (vocal I), Random Forest.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 3 (vocal U) con el modelo de Random Forest.

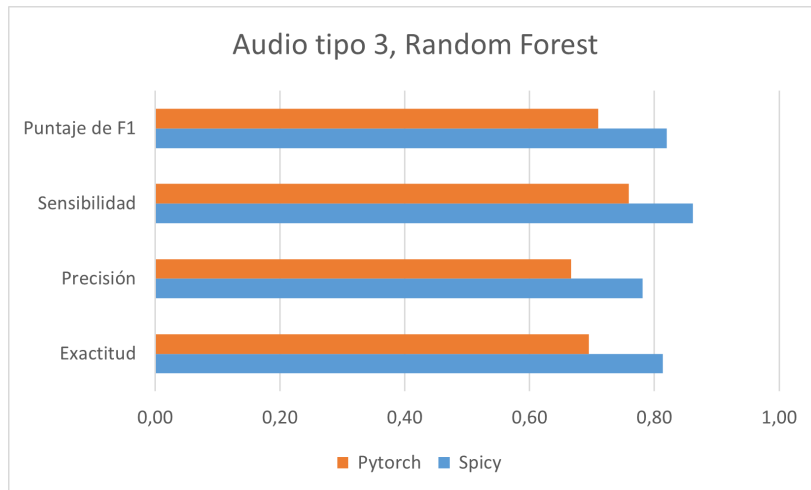


Figura G.27: Audio tipo 3 (vocal U), Random Forest.

Fuente: Elaboración propia



Comparación de los dos conjuntos para los audios de tipo 4 (palabra campana) con el modelo de Random Forest.

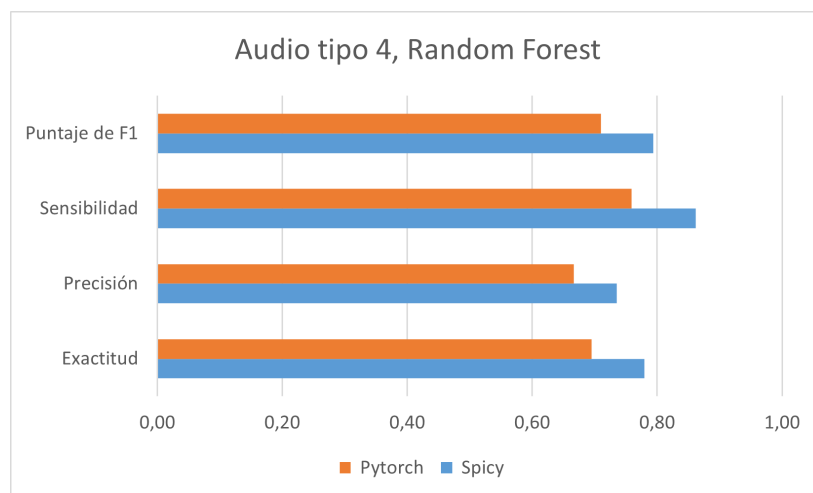


Figura G.28: Audio tipo 4 (palabra campana), Random Forest.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 5 (palabra gato) con el modelo de Random Forest.

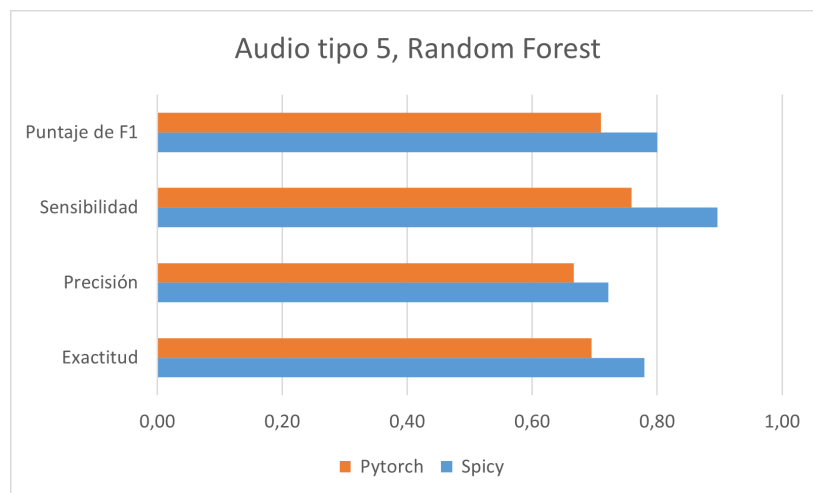


Figura G.29: Audio tipo 5 (palabra gato), Random Forest.

Fuente: Elaboración propia

Comparación de los dos conjuntos para los audios de tipo 6 (palabra petaca) con el modelo de Random Forest.

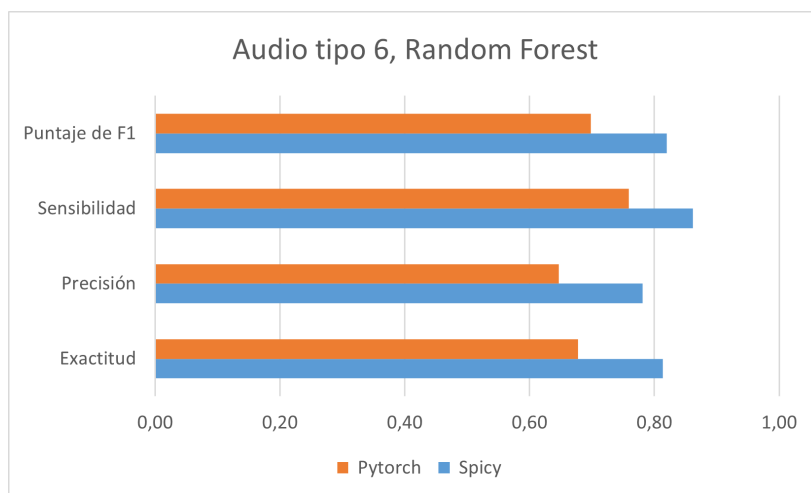


Figura G.30: Audio tipo 6 (palabra petaca), Random Forest.

Fuente: Elaboración propia

---

# Bibliografía

---

- [1] csv — csv file reading and writing — python 3.11.3 documentation. <https://docs.python.org/3/library/csv.html>.
- [2] Cómo crear un diagrama de clases [+ejemplos]. <https://es.venngage.com/blog/diagrama-de-clases/>. (Accessed on 07/03/2023).
- [3] librosa — librosa 0.10.1dev documentation. <https://librosa.org/doc/main/>.
- [4] Numpy documentation. <https://numpy.org/doc/>.
- [5] os — miscellaneous operating system interfaces — python 3.11.3 documentation. <https://docs.python.org/3/library/os.html>.
- [6] pandas documentation — pandas 2.0.1 documentation. <https://pandas.pydata.org/docs/>.
- [7] Project jupyter | home. <https://jupyter.org/>.
- [8] pydub/api.markdown at master · jiaaro/pydub. <https://github.com/jiaaro/pydub/blob/master/API.markdown#audiosegment>.
- [9] scikit-learn: machine learning in python — scikit-learn 1.2.2 documentation. <https://scikit-learn.org/stable/>.
- [10] Scipy api — scipy v1.10.1 manual. <https://docs.scipy.org/doc/scipy/reference/>.
- [11] shutil — high-level file operations — python 3.11.3 documentation. <https://docs.python.org/3/library/shutil.html>.

- [12] Streamlit documentation. <https://docs.streamlit.io/>. (Accessed on 06/24/2023).
- [13] Torchaudio documentation — torchaudio 2.0.1 documentation. <https://pytorch.org/audio/stable/index.html>.
- [14] Tutorial de diagrama de secuencia uml | lucidchart. <https://www.lucidchart.com/pages/es/diagrama-de-secuencia>. (Accessed on 07/03/2023).
- [15] Tutorial de diagramas de casos de uso ( guía con ejemplos ). <https://createely.com/blog/es/diagramas/tutorial-diagrama-caso-de-uso/>. (Accessed on 07/03/2023).
- [16] Users guide — matplotlib 3.7.1 documentation. <https://matplotlib.org/stable/users/index.html>.
- [17] ¿qué es python? - explicación del lenguaje python - aws. <https://aws.amazon.com/es/what-is/python/>.
- [18] ¿qué es un diagrama de flujo? | lucidchart. <https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo>. (Accessed on 07/03/2023).
- [19] ¿cómo instalar python? - el pythonista. <https://elpythonista.com/como-instalar-python>. (Accessed on 07/03/2023).