Proyecto 4 : Whale Calls

Integrantes:

Jorge Sebastian Tenorio Romero (100%)
Gonzalo Daniel Suárez Torres (100%)
Esteban Andre Vásquez Grados (100%)
Jorge Alonso Castillo Aliaga (100%)

Docente: Cristian López del Álamo
Curso: Machine Learning
Grupo de Proyecto: 8

***Abstract*—El proyecto número 4 de Machine Learning se centra en la clasificación de llamadas de ballenas para prevenir colisiones, basado en señales de audio. Utilizando un subconjunto de un extenso conjunto de datos de segmentos de audio anotados, donde se aplicarán técnicas de extracción de características de audio para obtener los vectores característicos.**

I. Introducción

A. Descripción General

El problema se basa en clasificar distintas señales de audios en si se tratan de sonidos de ballenas francas o no. En ese sentido, las muestras positivas contienen ”up-calls” de ballena franca, una vocalización común con una firma acústica de aproximadamente 60Hz-250Hz. Adicionalmente, se debe considerar que la clasificación puede ser desafiante debido a la presencia de ruido antropogénico en la banda de baja frecuencia, siendo estos como ejemplo el ruido de barcos, perforaciones, amontonamiento u operaciones navales.

B. Objetivo General

El objetivo de este proyecto es desarrollar un sistema de Machine Learning capaz de clasificar segmentos de audio para determinar si contienen o no llamadas de ballenas francas. Utilizando métodos de clasificación, se pretende mejorar la precisión en la identificación de estas llamadas.

C. Objetivos Específicos

1) Implementar distintos modelos de clasificación, verificando la eficacia y precisión de los resultados a partir de diferentes métricas.
2) Comprender la naturaleza de los datos para un posterior análisis de los testeos llevados a cabo.
3) Comparar los diferentes modelos propuestos respecto al problema establecido en torno a los resultados que estos arrojen.

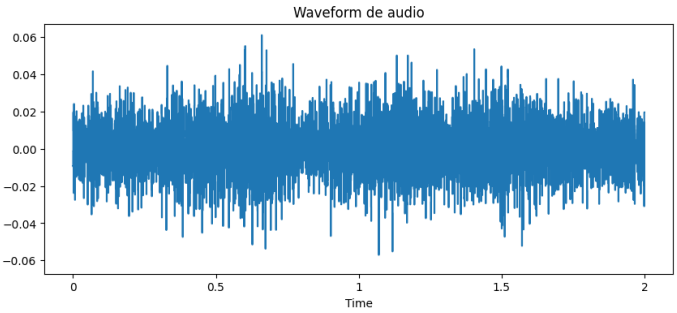
II. Dataset

A. Descripción

El dataset está compuesto por segmentos de audio de dos segundos de duración, muestreados a una frecuencia de 2kHz. Cada segmento de audio en el apartado de Train está etiquetado como ”Right Whale” o ”No Whale”. A continuación se detallan algunos de los archivos y carpetas que componen al dataset.

- `test/`: Subcarpeta que contiene los audios del apartado de testing.
- `train/`: Subcarpeta que contiene los audios del apartado de training.
- `train.csv`: Contiene los ids y las clases de la data de Train. (Siendo clasificaciones entre ”Right Whale” o ”No Whale” como se explicó)
- `sample_submission.csv`: Contiene el ejemplo de subida para la competición de Kaggle.

A continuación se muestra la imagen de un waveform de un audio del dataset:



B. Extracción de Características

Para la extracción de características se uso la libreria de librosa, la cual es utilizada para procesar los audios y extraer sus vectores característicos MFCC, la forma de la matriz MFCC es (20, 230). Esto significa que cada ventana de análisis del audio se representa mediante un vector de 20 dimensiones y el audio completo se ha dividido en 230 ventanas de análisis. Además, se hizo uso de la librería Pickle para almacenar los resultados del encoding tanto de Train y Test en una primera pasada. Esto con el fin de ahorrarse tiempo en cada ejecución.

C. Estructura del dataset

El dataset se compone de dos carpetas con los audios, Test y Train. Cada audio tiene como nombre un índice asociado partiendo desde 0. En el caso de Train, se pueden saber las etiquetas de los audios debido al archivo csv proporcionado. A continuación están los detalles de las carpetas:

Train data:

1) 10934 audios totales (samples)
2) 5467 etiquetados RightWhale
3) 5467 etiquetados NoWhale
4) 2 segundos de duración en promedio

Test data:

1) 1962 audios totales (samples)
2) 2 segundos de duración en promedio

Se debe notar que los audios proporcionados en Test no tienen etiquetas como lo es en el caso de Train. Debido a esto y con la finalidad de poder tener métricas del desempeño de los modelos, se decidió por destinar el 80% de los audios de la carpeta de Train a netamente el training de los modelos y el 20% restante al testing. Por otro lado, los audios de la carpeta de Test se tomaron como base para un segundo testing, destinado a la publicación de resultados en la competición de Kaggle asociada al presente proyecto.

III. Metodología

En las primeras semanas de trabajo del proyecto se tuvieron en cuenta múltiples modelos para el escenario descrito. En específico, se trabajó con MLP, SVM, Regresión Logística, Decision Tree y KNN como alternativas para abordar el problema establecido. Sin embargo, al hacer distintas pruebas en las implementaciones hechas, optamos con quedarnos con 3 opciones que fueron las que mejores resultados dieron y que notamos que podrían ajustarse mejor al problema.

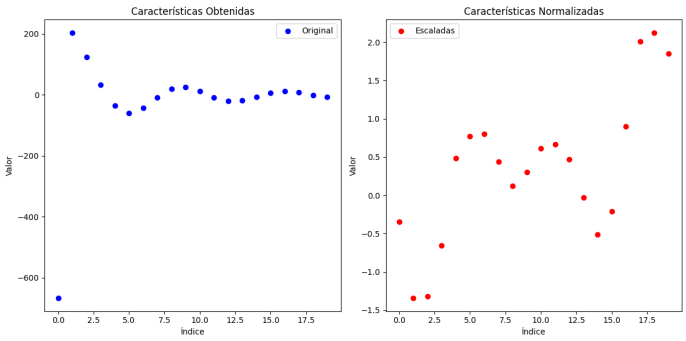
A. Normalización

1) StandardScaler: Para normalizar las características obtenidas del dataset se utilizó StandardScaler de la librería SKLEARN, ya que uniformiza las características extraídas logrando dar un peso de importancia similar a todas ellas. Es importante que los valores numéricos no sean demasiado grandes para algunos modelos, ya que sus derivadas serán muy grandes también, lo que puede conllevar a que el modelo nunca converja. Este método de estandarización consiste en operar de la siguiente forma cada característica:

$$w_{\text{normalizado},i} = \frac{w_i - \mu_i}{\sigma_i}$$

Donde W_i es la característica original, μ_i es la media de la característica y σ_i es la derivación estándar de la característica.

Al evaluar el cambio que tienen las características de un audio al azar al someterse al escalamiento, se muestra la siguiente gráfica:



donde se puede ver cómo los datos se encuentran en un rango de valores más corto (de [-600, 200] a [-1.5, 2]) y de manera más agrupada.

B. Modelos de clasificación y redes neuronales

Después de normalizar la data, se procede la implementación de los modelos de aprendizaje supervisado y redes neuronales de los cuales se realizará la comparación mediante las métricas seleccionadas. Los modelos propuestos para el presente proyecto son:

1) MLP: Usando MLP podemos aprender relaciones entre las características espectrales de las señales de audio MFCC vectoriales obtenido de los audios, así como encontrar patrones complejos con datos de alta dimensionalidad.
Este modelo permite poder realizar diferentes optimizaciones que permitan mejorar la eficacia de la predicción. Entre esas están técnicas de regularización como Drop Out o Batch Normalization, que son algunas de las que se usaron para el presente proyecto. Igualmente, se optaron por explorar dos tipos de funciones de activación para las capas hidden: ReLU y Leaky ReLU.

2) Decision Tree: Es un método que consiste en una partición recursiva de los elementos, sea basándose en ganancia de la información o la impureza de Gini.

El uso de Desicion Tree en el proyecto se debe a su capacidad para capturar relaciones no lineales complejas, seleccionar automáticamente las características más relevantes, en este caso la relación entre los audios con sus etiquetas.

3) Regresión Logística: Se trata de una técnica que trata de modelar la probabilidad de que un elemento pueda pertenecer a una clase o a otra, apoyándose de la función sigmoide (logística también llamada), un umbral de probabilidad y una función de pérdida. En general, la regresión logística es rápida y fácil de implementar, además de requerir menos recursos computacionales a comparación de otros métodos.

C. Métricas

Para evaluar nuestros modelos seleccionamos 4 métricas que retornaran la correctitud de los modelos. Estas son:

1) Matriz de confusión: La matriz de confusión nos permite visualizar fácilmente y de manera rápida las etiquetas obtenidas a partir del modelo, esto mediante la vista de cuatro cuadrantes que nos indican los falsos negativos, verdaderos negativos, falsos positivos y verdaderos positivos.

2) Precisión: A partir de la matriz de confusión obtenida de alguno de los modelos propuestos se procede a calcular la precisión obtenida. Esta métrica nos índica el porcentaje de data correctamente etiquetada por el modelo.

3) Recall: A partir de la matriz de confusión obtenida de alguno de los modelos propuesto se procede a calcular el recall obtenido. Esta métrica no solo sirve para determinar las etiquetas correctas, sino también, para evaluar las etiquetas mal etiquetadas.

4) F1 score: Se calcula a partir de los resultados de mi metricas de presición y recall, esta métrica es buena usarlo en caso tengamos un dataset desbalanceado.

D. Training y testing

Una vez logrado la implementación de los tres modelos seleccionados y las métricas, se procede a separar la data de entrenamiento en un 80% para training y 20% para testing. A continuación, se realiza el entrenamiento de dichos modelos mediante la variación de hiperparámetros de cada modelo y la comparación mediante las métricas propuestas. Finalmente, se selecciona el mejor modelo obtenido para predecir la data propuesta y subir los resultados a la competencia Kaggle.

IV. Implementación

El proyecto fue trabajado en Google Colaboratory haciendo uso implementaciones pasadas de los modelos explicados, así como librerías externas para la extracción de características, el manejo de operaciones de matrices, visualización de datos, entre otros. https://colab.research.google.com/drive/1rxO5v2gYQHatq-1W4PAHTvDcCYZtBvrk?usp=sharing

V. Experimentación

1) MLP: Se tomaron en cuenta las pérdidas de entrenamiento con ReLU y Leaky ReLU como funciones de activación en las capas hidden, abarcando como hiperparámetro variable el tamaño de las capas ocultas.

TABLE I
PÉRDIDA CON RELU

Capa Oculta	Épocas				
	0	500	1000	1500	2000
16	0.5408	0.4641	0.4597	0.4608	0.4586
32	0.5165	0.4469	0.4451	0.4422	0.4417
64	0.4952	0.4288	0.4238	0.4211	0.4187

TABLE II
MÉTRICAS CON RELU

Capa Oculta	Métricas			
	Accuracy	Precision	Recall	F1 Score
16	0.7805	0.7629	0.8333	0.7966
32	0.7888	0.7703	0.8413	0.8042
64	0.7846	0.7585	0.8546	0.8037

TABLE III
PÉRDIDA CON LEAKY RELU

Capa Oculta	Épocas				
	0	500	1000	1500	2000
16	0.5349	0.4635	0.4587	0.4642	0.4656
32	0.5151	0.4473	0.4421	0.4488	0.4400
64	0.4974	0.4234	0.4239	0.4214	0.4167

TABLE IV
MÉTRICAS CON LEAKY RELU

Capa Oculta	Métricas			
	Accuracy	Precision	Recall	F1 Score
16	0.7814	0.7555	0.8520	0.8008
32	0.7915	0.7710	0.8475	0.8074
64	0.7892	0.7833	0.8174	0.8000

2) Decision Tree: Debido a la naturaleza de entrenamiento del modelo, se muestran como resultados los siguientes aspectos.

TABLE V
MÉTRICAS CON INFORMATION GAIN

Métrica	Valor
Accuracy	0.7170
Precision	0.7341
Recall	0.7074
F1-score	0.7205

TABLE VI
MÉTRICAS CON GINI IMPURITY

Métrica	Valor
Accuracy	0.7023
Precision	0.7182
Recall	0.6959
F1-score	0.7069

3) Regresión Logística: Se tomó en cuenta como parámetro variable el valor de alpha, el cual afecta directamente a las derivadas.

TABLE VII
PÉRDIDA

Alpha (α)	Épocas				
	0	500	1000	1500	2000
$2 * 10^{-1}$	1.8274	0.2112	0.2111	0.2110	0.2109
$2 * 10^{-2}$	1.8274	0.2139	0.2118	0.2116	0.2115
$2 * 10^{-3}$	1.8274	0.4529	0.2459	0.2297	0.2142
$2 * 10^{-4}$	1.8274	1.6515	1.4779	1.3094	1.1478

TABLE VIII
MÉTRICAS

Alpha (α)	Métricas			
	Accuracy	Precision	Recall	F1 Score
$2 * 10^{-1}$	0.7531	0.7442	0.7943	0.7684
$2 * 10^{-2}$	0.7513	0.7441	0.7890	0.7659
$2 * 10^{-3}$	0.7554	0.7269	0.8422	0.7803
$2 * 10^{-4}$	0.3077	0.3047	0.2668	0.2845

VI. DISCUSIÓN

A. Extracción de Características

El proceso para obtener las Frecuencias de Mel (MFCC) son una representación del espectro de potencia a corto plazo de un sonido, los MFCC capturan características importantes como el timbre o la entonación del sonido. La libería utilizada nos brinda 20 características por entrada, esto encaminó a no utilizar alguna metodología de redución de caracterisitcas, ya que, al ser un número reducido de estas es muy probable que se pierda gran parte de la información extraída. Por otro lado, si se optaba por realizar la extracción por, por ejemplo espectrogramas, se obtendría una matriz de dimension 128x87 por audio, la cual conllevaría realizar un pre procesamiento de estas caracteristicas obtenidas y también de reducción de dimensionalidad de las mismas.

B. Evaluación de la clasificación de los modelos

1) MLP: Las redes neuronales multicapa (MLP) mostraron un rendimiento notable en la clasificación de los sonidos de ballenas debido a su capacidad para modelar relaciones no lineales complejas entre las características.

MLP utiliza múltiples capas ocultas con neuronas interconectadas que permiten al modelo aprender representaciones jerárquicas de los datos de audio. Esta estructura es capaz de capturar patrones y dependencias temporales en las características de los sonidos, mejorando la precisión de la clasificación.

La flexibilidad de las MLP para ajustar pesos mediante la retropropagación permite que el modelo se adapte a las variaciones en los datos de entrenamiento, identificando características importantes para distinguir entre ”NoWhale” y ”RightWhale”.

MLP requiere una cuidadosa afinación de hiperparámetros y un conjunto de datos suficientemente grande para evitar el sobreajuste y lograr un rendimiento óptimo.

2) Decision Tree: Los árboles de decisión nos proporcionaron resultados óptimos gracias a su capacidad para capturar relaciones no lineales entre características.

Durante la construcción del árbol, se seleccionan las características más relevantes que proporcionan una mayor ganancia de información o la mayor reducción de impureza. Esto significa que podemos identificar y enfocarnos en frecuencias específicas y patrones temporales que son más informativos para la clasificación de los sonidos de ballenas.

Esta capacidad de selección de características es una ventaja significativa, ya que permite al modelo ignorar ruidos o características menos relevantes, concentrándose en los aspectos del audio que son más distintivos para diferenciar entre las clases ”NoWhale” y ”RightWhale”. Como resultado, los árboles de decisión pueden adaptarse eficazmente a las particularidades de los datos de audio procesados.

3) Regresión Logistica: Mediante este método se pudieron obtener resultados decentes en la mayoría de variaciones de alpha. Sin embargo, se pudo ver que para valores muy pequeños de esta y tendiendo a cero los resultados podían llegar a ser considerablemente peores para las épocas tomadas en cuenta. Esto se puede deber en buena parte por la incidencia que tienen estas en la función de pérdida, ya que valores pequeños hacen que el proceso de aprendizaje se relentice y el modelo no aprenda mucho, como se puede evidenciar en la evolución de la pérdida según el número de épocas.

VII. Conclusiones

A. Resumen de resultados

Luego de haber obtenido las métricas, se procedió a comparar los modelos impementados, y en síntesis, se esperó obtener mejores resultados en orden descente mediante los modelos MLP, Regresión Logística y Decision Tree. Sin embargo, al ingresar a la competencia Kaggle las predicciones obtenidas en los experimentos, se logró nuestro más alto puntaje (0.6571) con el modelo Desicion Tree, con el cual se esperó obtener el peor puntaje. Debido a esto, no solo se espera mejorar el score de Kaggle al utilizar la data test al 100%, sino también, que entre los 3 modelos propuestos en este proyecto se obtenga el mejor resultado con el modelo MLP con función de activación RELU o Leaky RELU.

B. Limitaciones

- Se utilizó *Mel-Frequency Cepstral Coefficients (MFCC)* como técnica de extracción de características de audio, no se probó con otras técnicas como spectrograms o mel-spectrograms que se utilizan para capturar las frecuencias a lo largo del tiempo de manera más directa y visual, aunque no capturan las variaciones en la frecuencia tan detalladamente como los MFCC.
- Una gran limitante fue disponibilidad de la GPU de Google Colab para el entrenamiento de modelos basados en redes neuronales como MLP ya que se cuenta con una cuota de procesamiento limitado.

C. Recomendaciones

- Los vectores característicos deben contener valores normalizados para asegurar que todas las características contribuyan de manera equitativa al modelo y para mejorar la convergencia en los algoritmos de optimización.
- Usa Grid Search o Random Search para encontrar la mejor combinación de hiperparámetros para los modelos.
- Dado a que los resultados de las métricas propuestas fueron ligeramente superiores en los modelos MLP, se recomienda aumentar la experimentación sobre estos modelos. Esto mediante el aumento de un rango de 0 - 1000 épocas más y variación en la regularización dropout del modelo, así como el aumento de neuronas en las capas ocultas.