Análisis Acústico Sobre Señales de Audio de Aves:

Extracción de Características, Modelado y Clasificación con Redes Neuronales

David Mauricio López

La Salle Barcelona - Universidad Ramon Llull

Minería de Datos

Ester Vidaña Vila

Marzo, 2023

1. Introducción al Problema

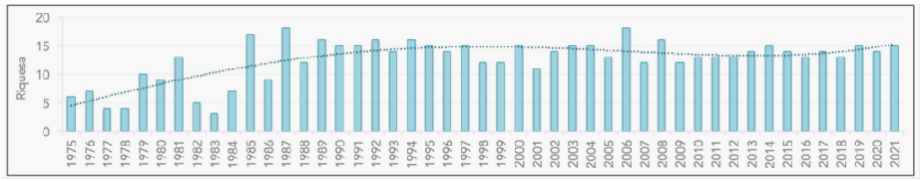
La creciente crisis de biodiversidad ha generado una urgente necesidad de adoptar estrategias innovadoras para comprender y conservar los ecosistemas naturales. En este contexto, el Parque Natural Aiguamolls del Empordà, ubicado en Girona, ha sido un área de especial interés debido a su rica diversidad de aves desde 1994. Sin embargo, el estudio y monitoreo de estas aves han enfrentado desafíos significativos. Tradicionalmente, este proceso se lleva a cabo de manera manual y costosa, mediante la labor de expertos humanos. Estos métodos, además de ser laboriosos, tienen un alcance limitado y exponen a los investigadores a diversos riesgos.

1.1 Indicador de Riqueza y Diversidad Avícola:

El seguimiento ornitológico del Parque Natural Aiguamolls del Empordà nos ofrece una herramienta clave: el indicador de riqueza.

Figura 0.

Evolución de los valores de riqueza de anátidos invernantes en los censos de enero en los Aiguamolls de l'Empordà, Periodo 1975-2021



Nota: este grafico muestra la evolución del número de especies que han estado en el parque desde 1975 hasta 2021. Tomado de Informe seguimiento ornitológico en parque natural de los aguamolles del emporda año 2021(p.52) por parque natural de los aguamolles del emporda, 2022.

Este indicador nos permite observar las especies que habitan una determinada localidad y proporciona información valiosa sobre la diversidad y abundancia de aves en el área. En el caso de los

Aiguamolls del Empordà, se han registrado un total de 35 especies de anátidas a lo largo de los censos de pájaros invernantes. Sin embargo, más de la mitad de estas especies son consideradas raras y de presencia ocasional, mientras que habitualmente se observan entre 12 y 15 especies de anátidas distintas. El gráfico siguiente muestra que, desde la declaración del Parque Natural, la riqueza de especies de anátidas se ha mantenido estable en los Humedales del Empordà, con valores medios de 14 especies en los últimos años.

Este conjunto de indicadores nos ofrece una visión completa y detallada de la diversidad avícola en el Parque Natural Aiguamolls del Empordà, permitiendo una mejor comprensión de su ecología y el diseño de estrategias efectivas para su conservación y manejo sostenible.

1.2 Incorporación de Tecnología: Machine Learning

En respuesta a esta problemática, surge la necesidad de explorar nuevas tecnologías, como el machine learning, para mejorar la eficiencia y precisión del estudio de aves en el Parque Natural Aiguamolls del Empordà. Esta tecnología promete automatizar el proceso de recopilación y análisis de datos, ofreciendo una alternativa innovadora y efectiva para comprender la dinámica de las poblaciones avícolas en el área.

Para lograr un impacto altamente positivo en la extracción y procesamiento de los audios, se propone la implementación de un modelo de machine learning end-to-end. Este modelo, diseñado específicamente para el estudio de las aves en el Parque Natural Aiguamolls del Empordà, se entrenará utilizando datos acústicos recopilados de 20 especies avícolas. La ventaja de este enfoque de machine learning radica en su capacidad para aprender y adaptarse a partir de los datos, permitiendo una comprensión más profunda y detallada del comportamiento de las aves. Al analizar patrones y correlaciones en los datos acústicos, el modelo podrá identificar distintas vocalizaciones y comportamientos, lo que a su vez facilitará la obtención de información relevante para la conservación de los hábitats de las aves.

Este modelo no solo mejorará la eficiencia en la extracción y procesamiento de los audios, sino que también abrirá nuevas posibilidades para realizar análisis más sofisticados y precisos. Con una comprensión más completa del comportamiento avícola, estaremos en una mejor posición para diseñar estrategias de conservación más efectivas y realizar tomas de muestras acústicas más exhaustivas y significativas.

2. Desarrollo Metodológico: Arquitectura y Flujo de Procesamiento de Señales Acústicas

En el marco de este proyecto, se ha desarrollado un pipeline que integra el procesamiento y análisis profundo de señales de audio provenientes de distintas especies de aves. Entendido el contexto del proyecto, es fundamental hacer especial énfasis en el enfoque metodológico aplicado en las diferentes etapas críticas del ejercicio, cómo pueden ser: Análisis exploratorio de los datos y audios, extracción inicial de características acústicas y aplicación de técnica de modelado y evaluación. A continuación, se describen los componentes claves del proyecto, destacando su importancia y contribución final en el desarrollo del Modelo de Clasificación.

2.1 Arquitectura Integrada del Pipeline del Proyecto

Antes de profundizar sobre los aspectos técnicos aplicados en el proyecto, es fundamental realizar un análisis exploratorio de los datos. Esta fase preliminar implica una revisión meticulosa de los metadatos proporcionados, entender su estructura y funcionalidad. Adicionalmente, evaluamos aspectos cómo la calidad, duración y variedad de muestras de los audios, lo cual nos ayudara a visualizar los requerimientos prácticos relacionados con la manipulación de audios.

Figura 1.

Elaboración propia. (2024). Análisis Exploratorio de Señales de Audios Aves: Verificación tipos de datos.

```
[ ]  Motacilla_flava[0].dtype

     dtype('float32')
```

Figura 2.

Elaboración propia. (2024). Análisis Exploratorio de Señales de Audios Aves: Verificación Frecuencias de Audio para la especie 'Acrocephalus_Arundinacus'.

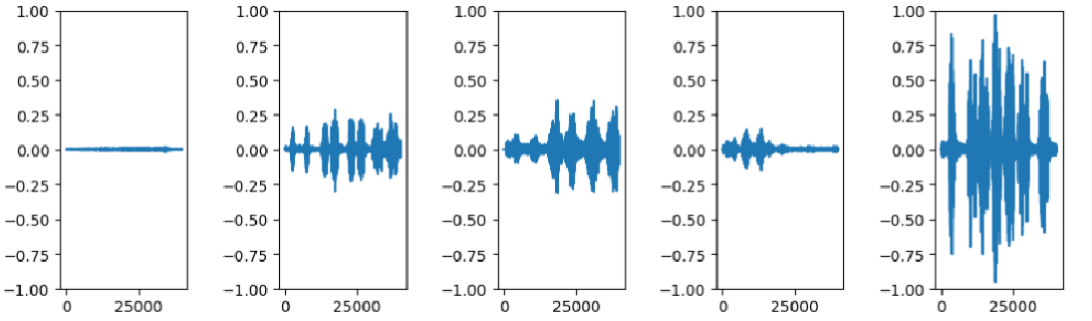


Figura 3.

Elaboración propia. (2024). Análisis Exploratorio de Señales de Audios Aves: Ejemplo audio de la especie 'Acrocephalus_Arundinacus'.

```
[ ]  # Escuchar un ejemplo de los Acrocephalus_Arundinacus
     ipd.Audio("/content/drive/MyDrive/Proyecto Mineria de Datos/audio_files/Acrocephalus arundinaceus/XC417157.mp3")
```

2.2 Recolección y Preparación de Datos

El inicio de este Pipeline consiste en la recopilación sistemática de audios por cada especie, lo que no representaría un problema porque se nos presentan los audios clasificados por cada especie. Esta organización facilita la extracción automatizada de las características, así nos aseguramos de que cada muestra de audio se asocie correctamente con su especie respectiva. Esta distribución se puede visualizar en la siguiente figura:

Figura 4.

Elaboración propia. (2024). Análisis Exploratorio de Señales de Audios Aves: Jerarquía y orden de carpetas por cada especie y sus respectivos audios.

2.3 Metodología de la Extracción de Características

La extracción de características constituye la etapa más crítica en el análisis de los audios, ya que las características seleccionadas ejercen una influencia determinante en la eficiencia del modelo de clasificación. Este proyecto busca emplear algoritmos y códigos que aíslen las características acústicas por cada audio de su especie respectiva, algunas de estas características pueden incluir: Tasa de Cruce por Cero (ZCR), Coeficientes Cepstrales de Frecuencia Mel (MFCCs) y Centroide Espectral.

Luego, se realiza un análisis detallado de las características mencionadas, para comprender el aporte específico que brindan sobre el análisis de los audios. Este entendimiento es fundamental para elegir de forma adecuada, las características que se integrarán dentro del modelo de clasificación, así garantizando un desempeño óptimo en la identificación precisa de cada especie de ave.

- **Tasa de Cruce por Cero (ZCR)**: Es una forma muy sencilla para calcular el número de veces que la señal de audio cruza el eje horizontal. Una señal de voz oscila lentamente (por ejemplo, una señal de 100 Hz cruzará el cero 100 por segundo), mientras que una fricativa

sorda puede tener 3000 cruces por cero por segundo. *Nagesh Singh, Chauhan. (2020). Audio Data Analysis Using Deep Learning with Python. KDnuggets.*

- **Coeficientes Cepstrales de Frecuencia Mel (MFCCs)**: Los coeficientes cepstrales de frecuencia Mel (MFCC) de una señal son un pequeño conjunto de características (generalmente entre 10 y 20) que describen de manera concisa la forma general de una envolvente espectral. Por ejemplo, modela las características de la voz humana. *Nagesh Singh, Chauhan. (2020). Audio Data Analysis Using Deep Learning with Python. KDnuggets.*

Estas características no solo brindan una comprensión profunda sobre las propiedades acústicas inherentes de los audios, sino que robustecen las condiciones adversas en las que se encuentran dichas grabaciones, cómo pueden ser: El ruido de fondo y otros factores externos. A través de la extracción de estas características identificamos diversas ventanas temporales a lo largo del audio, permitiéndonos examinar las variaciones con respecto al tiempo. Este análisis temporal es crucial para el reconocimiento de aspectos complejos dentro de las señales de audio, cómo es el caso del canto de las aves.

2.3.1 Extracción de Características con Librosa

Después de preparar adecuadamente los datos para el análisis, se emplea la librería Librosa para realizar una extracción exhaustiva y precisa de dichas características. Este proceso es la consolidación de todas las etapas anteriores y a continuación, detallamos cómo se aplicaron las funcionalidades de Librosa dentro del marco del proyecto y sus facetas:

- **Tasa de Cruce por Cero (ZCR)**: Utilizamos '`librosa.feature.zero_crossing_rate()`' para calcular la ZCR por cada segmento de audio, en este caso, solo se incorporará este cálculo en el segmento dónde se encuentra el canto del ave, este procedimiento se realiza para robustecer la efectividad del modelo de clasificación más adelante. La ZCR es una medida eficaz para

identificar las partes del audio con mayor actividad, lo que resulta muy útil en nuestro ejercicio, ya que distingue segmentos del audio vocales y no vocales.

Figura 5.

Elaboración propia. (2024). Uso de biblioteca Librosa para la extracción de ZCR.

```
# Calcular el Zero Crossing Rate (ZCR)
zcr = librosa.feature.zero_crossing_rate(signal)
```

- **Coeficientes Cepstrales de Frecuencia Mel (MFCCs)**: Para extraer el MFCCs utilizamos la librería '**`librosa.feature.mfcc()`**', seleccionando adecuadamente un número de coeficientes que se adapte mejor a nuestro modelo, con la finalidad de capturar la esencia de los audios sin entrar en redundancia de la información.

Figura 6.

Elaboración propia. (2024). Uso de biblioteca Librosa para la extracción de MFCCs.

```
# Calcular los Mel-Frequency Cepstral Coefficients (MFCCs)
mfccs = librosa.feature.mfcc(y=signal, sr=sr, n_mfcc=13)
```

Cómo conclusión, este enfoque detallado en la etapa de extracción asegura que el modelo de clasificación reciba datos de alta calidad y relevancia, resultando en la maximización de su desempeño y precisión. En consecuencia, esta fase del proyecto reafirma la importancia de una selección cuidadosa de características de análisis, dado que más adelante observaremos su impacto en los resultados del modelo de clasificación.

2.4 Estructuración de los Datos

Una vez extraídas las características deseadas, se organizan en una estructura tabular usando la librería de pandas para facilitar la visualización, análisis y manipulación de los datos. Cada fila del

Dataframe se dedicó al segmento de audio correspondiente al canto del ave anteriormente aislado para su debida clasificación, lo cual permitió generar una correlación directa entre los datos y su origen.

A continuación, se detalla un segmento del código empleado para ilustrar el proceso de exportación del Datafame 'df', que contiene las características extraídas, a un archivo CSV. Este archivo se almacena posteriormente en Drive, facilitando así la visualización y el análisis de las clases contenidas.

Figura 7.

Elaboración propia. (2024). Exportación de Características Acústicas a CSV.

```
#Crear un DataFrame de Pandas con las características extraídas
df = pd.DataFrame(all_features, columns=["Especie", "Nombre_Audio", "ZCR"] + [f"MFCC_{i}" for i in range(1, 14)])
```

```
csv_filename = "/content/drive/MyDrive/Proyecto Mineria de Datos/CARPETA FINAL (NO TOCAR)/Dataset_Pajaros_Features_Audio_Final.csv"
df.to_csv(csv_filename, index=False)
```

Figura 8.

Elaboración propia. (2024). Visualización del Conjunto de Datos Final: Características Extraídas.

	Especie	Nombre_Audio	ZCR	MFCC_1	MFCC_2	MFCC_3	MFCC_4	MFCC_5	MFCC_6
0	Acrocephalus arundinaceus	XC417157	0.124023	-529.779907	53.188835	-21.022627	15.877716	-10.117560	4.505464
1	Acrocephalus arundinaceus	XC417157	0.180176	-494.047180	57.014755	-15.270870	0.154132	-10.710044	6.403300
2	Acrocephalus arundinaceus	XC417157	0.234375	-439.205719	55.601654	-49.169914	26.791927	19.355724	4.738024
3	Acrocephalus arundinaceus	XC417157	0.226562	-412.112122	29.842804	-98.413208	46.521858	38.899101	-28.511703
4	Acrocephalus arundinaceus	XC417157	0.218750	-359.968689	40.367714	-104.717545	45.786652	36.640564	-12.324357

2.5 Preparación para el Modelado

Tras haber obtenido el Dataframe completo, se proceden a preparar los datos para su uso en el modelo de clasificación. Esto incluye la división del conjunto de datos en subconjuntos de entrenamiento y prueba, estandarización de características para mejorar la convergencia del modelo y la aplicación de técnicas de codificación sobre las etiquetas de especies para convertirlas a un formato

numérico para que el modelo pueda procesar con mayor efectividad los datos. Estos procesos se presentarán a continuación:

- **Preprocesamiento de los Datos**: Este proceso comienza con la separación del conjunto de datos en variables independientes (X) que constituyen las características y la variable dependiente (y) que actúa como la etiqueta. Luego se procede a eliminar las columnas irrelevantes para el entrenamiento del modelo. En este contexto, eliminamos las columnas de 'Especie' y 'Nombre de Audio' de (X). A continuación, la columna 'Especie' se designa cómo nuestra variable objetivo (y), mientras que 'Nombre de Audio' se descarta porque no aporta información relevante para el modelo más adelante.

Finalmente, se consolidan estos datos en sus variables respectivas y empleamos la función '**`train_test_split()`**' de Scikit-learn (Biblioteca de aprendizaje automático) para dividir el conjunto de datos. Esta división reserva el 20% de los datos para pruebas y el 80% restante para en entrenamiento, cómo se puede observar en la siguiente figura:

Figura 10.

Elaboración propia. (2024). División del Conjunto de Datos en Entrenamiento y Prueba.

```
#Dividimos datos de Train y Test
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.8, test_size=0.2, random_state=45)
```

- **Estandarización de Características**:

Dado que las características acústicas pueden variar en magnitud, escala y rango, estandarizar estas características es fundamental para el óptimo rendimiento del modelo de clasificación. La estandarización se encarga de ajustar nuestros datos para que tengan una media de cero y una desviación de uno, lo que facilitan a gran escala la convergencia de nuestro modelo. *Conectando ideas. (2020). Machine Learning: Análisis de Componentes Principales (PCA) para la Mejora de Resultados.*

En el contexto del proyecto, se aplicó dicha estandarización por medio de la biblioteca Scikit-learn empleando la función '**StandardScaler()**', la cual se aplicó respectivamente al conjunto de entrenamiento y prueba asegurando así la uniformidad en la escala, equilibrando la influencia de cada característica en el aprendizaje del modelo, cómo se puede observar en la siguiente figura:

Figura 11.

Elaboración propia. (2024). Estandarización de las Características del Conjunto de Datos.

```
#Escalamos los datos
scaler = StandardScaler()
X_train_sc = scaler.fit_transform(X_resampled)
X_test_scaled = scaler.transform(X_test)
```

- **Codificación de Etiquetas**:

En el contexto de nuestro modelo de clasificación, es crucial trabajar con variables numéricas. Como las etiquetas son categóricas, específicamente los nombres de las especies de las aves, hay que transformarlos a formato numérico. Para lograr este objetivo debemos implementar la función '**LabelEncoder()**' de la biblioteca Scikit-learn, la cual nos permite realizar este procedimiento de transformación de etiquetas categóricas a etiquetas numéricas únicas por cada especie de ave.

Figura 12.

Elaboración propia. (2024). Codificación de las Etiquetas Categóricas.

```
# Codificar las etiquetas
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)
```

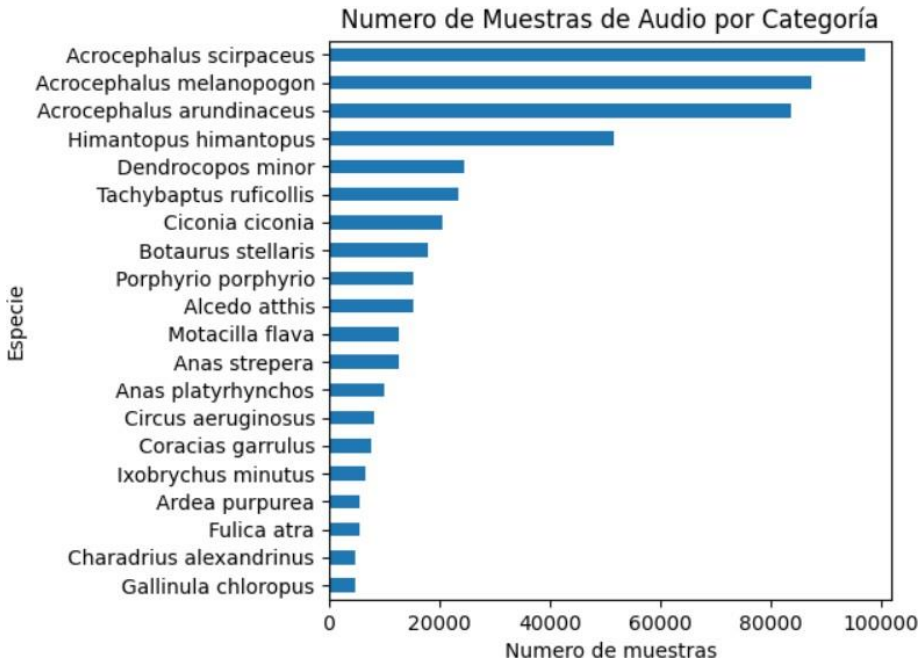
- **Balanceo de Clases:**

Profundizando en los conceptos anteriormente mencionados, entendemos que para la construcción de un modelo de clasificación es necesario realizar un análisis exhaustivo de los datos. Ante este análisis le sigue un proceso de transformación y estandarización, fundamental para la

adecuada preparación de los datos. Al examinar el conjunto de datos, se identifica un desequilibrio significativo entre clases, lo que podría afectar negativamente el rendimiento del modelo. Por consiguiente, es fundamental emplear técnicas de sobremuestreo, como SMOTE, para equilibrar la cantidad de muestras entre clases. Esta técnica crea muestras sintéticas de las clases minoritarias, mejorando así la capacidad del modelo para aprender por igual de todas las clases. A continuación, se presentará una gráfica de las clases antes de aplicar técnicas de sobre muestreó.

Figura 13.

Elaboración propia. (2024). Visualización del Conjunto de Datos Final: Balance de Clases previa a la aplicación de técnica de muestreo SMOTE.



3. Análisis Comparativo de Modelos de Machine Learning

En el desarrollo del proyecto, se han explorado una serie de modelos Machine Learning para enfrentar el reto que supone la clasificación de audios de aves. Esta variedad de enfoques nos ha permitido tener una visión holística del proceso, dónde se llegó a adoptar una estrategia integral, que no solo mejoró la precisión, sino la robustez de las predicciones resultantes del modelo. A continuación, profundizaremos en la metodología de cada una y la implementación del modelo escogido:

3.1 Redes Neuronales Artificiales (ANN):

Se optó por las Redes Neuronales Artificiales (ANN) como base del modelo debido a su flexibilidad y capacidad de adaptarse a las demandas del proyecto. Aunque los modelos ANN pueden ser menos complejas y escalables a comparación de las redes neuronales convolucionales (CNN), su estructura permitió personalizar características esenciales para abordar nuestro desafío en específico. Este modelo se benefició del uso de funciones de activación, que facilitan el aprendizaje de patrones complejos y no lineales en los datos, una capacidad crucial para clasificar el canto de las aves en el estudio.

El modelo ANN incorpora capas densas, permitiendo capturar relaciones profundas que no se presentan dentro del canto del ave, lo que beneficia a gran escala la clasificación dentro del entrenamiento del modelo. Luego, a través de capas Dropout y técnicas de regularización, se reduce el sobreajuste, asegurando así que el modelo generalice bien los datos nuevos proporcionados en la prueba. También es importante resaltar que para la parte de optimización del rendimiento del modelo se integraron las siguientes funciones:

- **Descenso de Gradiente**: El descenso de gradiente es un algoritmo de optimización que se usa comúnmente para entrenar modelos de machine learning y redes neuronales. Los datos de entrenamiento ayudan a que estos modelos aprendan con el tiempo, y la función de costo

dentro del descenso de gradiente actúa específicamente como un barómetro, midiendo su precisión con cada iteración de actualizaciones de parámetros. Hasta que la función sea cercana o igual a cero, el modelo continuará ajustando sus parámetros para producir la menor cantidad de errores posible. *IBM. (2020). ¿Qué es el descenso de gradiente?*

- **Error cuadrático medio**: El error cuadrático medio es un valor único que proporciona información sobre la bondad del ajuste de la línea de regresión. Cuanto menor sea el valor de MSE, mejor será el ajuste, ya que los valores más pequeños implican menores magnitudes de error. *Rodrigo, Ricardo. (2020). Estudyando.Error cuadrático medio.*
- **Binary Cross-Entropy**: La entropía cruzada binaria y la pérdida de registros se refieren al mismo concepto y se usan indistintamente. Son funciones de pérdida diseñadas específicamente para problemas de clasificación binaria. En la clasificación binaria, hay dos clases, a menudo denominadas clase positiva (1) y clase negativa (0). La entropía cruzada binaria/pérdida de registros mide la diferencia entre las etiquetas reales y las probabilidades previstas de que los puntos de datos estén en la clase positiva. Penaliza las predicciones que son seguras pero erróneas. *Leiki, Igal. (2024). Understanding Binary Cross-Entropy and Log Loss for Effective Model Monitoring. Aporia.*

3.2 Decision Tree:

Se selecciono realizar un modelo Decision tree debido a que es un modelo de clasificación que nos permite predecir el valor de una variable, mediante la clasificación de información en función de

otras variables, es un modelo de fácil interpretación en donde la estructura costa de ramas y nodos de diferentes tipos:

- **Los Nodos Internos**: representan las características extraídas de los cantos de los pájaros, la tasa de cruce por (ZCR) y los Coeficientes Cepstrales de Frecuencia Mel (MFCCs).
- **Las Ramas**: representan la decisión en función de la probabilidad de ocurrencia que sea de X especie.
- **Los Nodos Finales**: representan el resultado de la decisión.

Esto permitió tomar decisiones de manera rápida y eficiente, de igual forma los Decisión tree tienden a que si hay clases dominantes es fácil que los árboles generen datos sesgados, aunque no requiere ninguna preparación de datos este modelo, para este caso de clasificación de aves en específico era de vital importancia obtener una adecuada preparación de datos, con el fin de obtener un resultado veraz

3.3 Random Forest:

Con la finalidad de obtener un mejor resultado del modelo se decidio implementar random forest, que en este caso actuara como ensamble, este en un modelo que nos permite implementar varios decision tree.

Para este problema de clasificación hemos decidido entrenar 200 árboles, cada árbol se entrena en un subconjunto de la serie de datos y arroja un resultado, posteriormente se combinan los resultados de cada árbol de decision para así generar una respuesta final, este método se conoce como "*bagging",* de esta manera logramos construir un modelo robusto a partir de modelos que no deben ser tan robustos.

3.4 Redes Neuronales Convolucionales (CNN):

En el proyecto, además de utilizar las Redes Neuronales Artificiales (ANN), se incorporó otro modelo de Red Neuronal pero más robusto, para analizar datos más complejos y que el resultado fuese

más preciso, por eso se eligió el Modelo de Red Neuronal Convolucional (CNN). Este enfoque aprovecha la capacidad de las CNN para interpretar patrones visuales complejos, por lo que es adecuado para trabajar con espectrogramas, que representan visualmente las frecuencias de los sonidos en el tiempo.

El modelo se construyó utilizando TensorFlow y Keras, comenzando con las capas convolucionales, las cuales aplican filtros para extraer estadísticas visuales relevantes para el entrenamiento, adicionalmente esta seguida de una capa MaxPooling que reduce la dimensionalidad de los datos, preservando características esenciales de los espectrogramas y se concluye con una capa Dropout.

Esta estructura, adaptada para analizar los espectrogramas, representa un componente clave para clasificar el canto de las aves, ofreciendo una perspectiva adicional basada en patrones visuales con mayor complejidad.

4. Análisis de rendimiento y eficacia: evaluación de modelos de aprendizaje automático en la clasificación de audio de aves

4.1 Redes Neuronales Artificiales (ANN):

El modelo de redes neuronales artificiales (ANN) se entrenó durante 10 épocas con un tamaño de lote de 32, utilizando el conjunto de datos de entrenamiento escalado y sobre muestreado, con una fracción del 20% de los datos de entrenamiento utilizados para validación, se logró obtener una accuracy del 88% lo cual es muy bueno, esto nos dice que del 100% de las predicciones que se hicieron 88% de ellas fueron correctas, además la métrica loss nos da un valor de 0,41 indica cuánto se desvían las predicciones del modelo de las etiquetas verdaderas en el conjunto de datos de prueba. En este caso, una pérdida de 0.41 es relativamente baja y sugiere que el modelo está haciendo predicciones bastante precisas en el conjunto de datos de prueba.

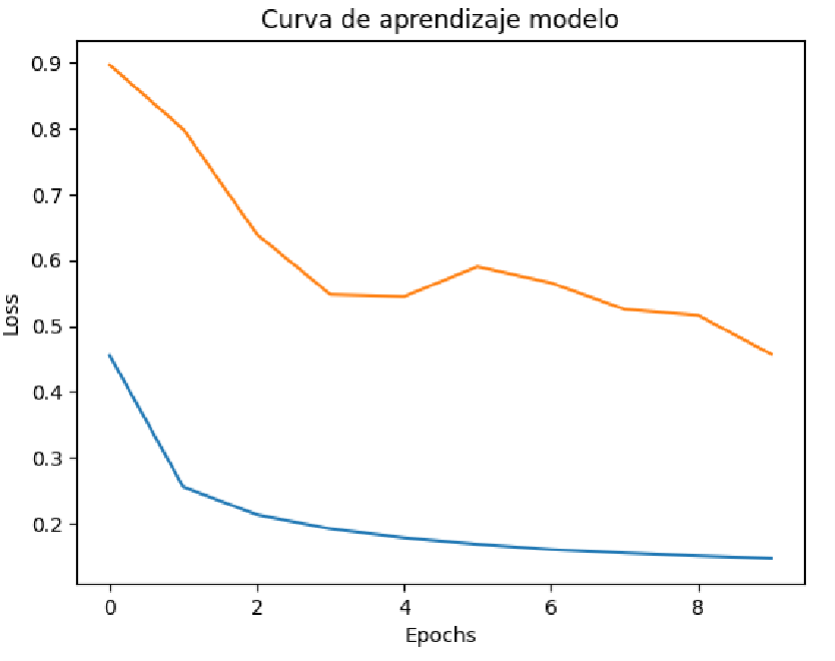
- **Macro avg**: Al utilizar la métrica macro avg se logro realizar el promedio de las métricas más relevantes como:
- **Precision**: se obtuvo un promedio de precisión del 0,84. Esto indica que, en promedio, el modelo ha identificado correctamente el 84% de todas las instancias clasificadas como positivas como verdaderas instancias positivas. En otras palabras, el modelo ha demostrado una capacidad del 84% para clasificar correctamente las muestras positivas, lo que sugiere un rendimiento bastante bueno en este aspecto
- **Recall**: se obtuvo un recall en promedio del 0.88. Esto indica que el modelo ha logrado identificar correctamente el 88% de todas las instancias positivas como verdaderas instancias positivas. En otras palabras, el modelo ha demostrado una capacidad del 88% para capturar y clasificar correctamente las muestras positivas, lo que sugiere un rendimiento bastante sólido en este aspecto.

- **F1-Score**: se logró obtener F1-Score del 0,86 que nos permite decir que el modelo logra un buen rendimiento.

Para realizar un análisis de manera explícita graficamos La pérdida, es una medida de cuánto se desvían las predicciones del modelo de las etiquetas verdaderas en el conjunto de datos de entrenamiento vs La pérdida de validación es similar a la pérdida, pero se calcula utilizando un conjunto de datos diferente al conjunto de entrenamiento, conocido como conjunto de validación.

Figura 15.

Elaboración propia. (2024). Plot de la Curva de Aprendizaje para Training Loss y Training Accuracy .

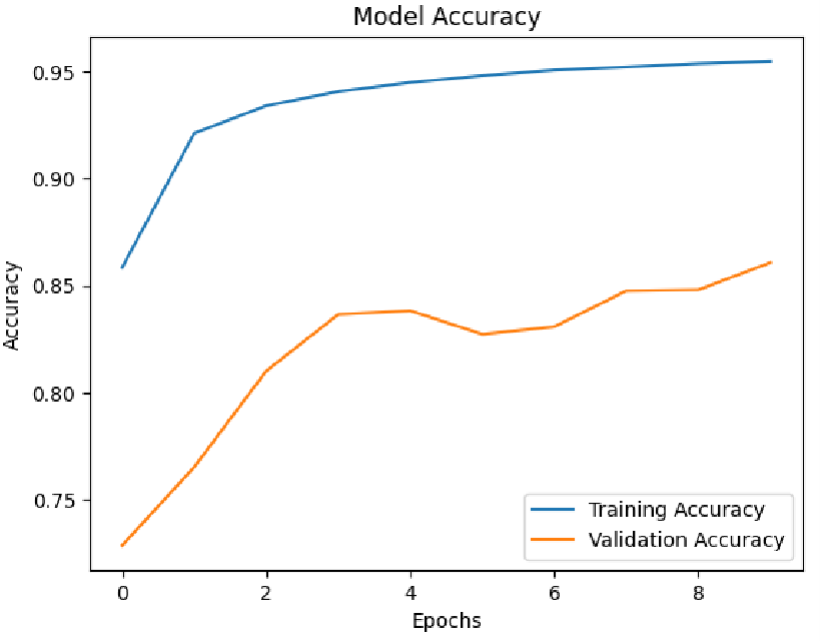


Como podemos observar en la gráfica las dos curvas disminuyen a medida que avanzan las épocas, lo que indica que el modelo está aprendiendo y mejorando su rendimiento.

Realizamos un plot para visualizar La precisión del conjunto de entrenamiento que indica la proporción de predicciones correctas realizadas por el modelo en los datos utilizados para entrenar. Mientras que la precisión del conjunto de validación indica la proporción de predicciones correctas en un conjunto de datos que no se utilizaron durante el entrenamiento, lo que ayuda a evaluar cómo el modelo generaliza a nuevos datos.

Figura 16.

Elaboración propia. (2024). Plot de la Curva de Training Accuracy vs Validation Accuracy.



Las curvas de precisión aumentaron a medida que avanza el entrenamiento, lo que indica que el modelo está mejorando su capacidad para hacer predicciones precisas.

4.2 Decision Tree:

Para medir el modelo se aplicaron las 4 métricas más relevantes:

- **Accuracy**: el modelo Decision Tree nos ha arrojado una precisión del 0,74 eso nos permite saber que de las predicciones que se realizaron, el modelo predijo correctamente el 74% de ellas.

  Figura 17.

  Elaboración propia. (2024). Accuracy Score Decision Tree.

```
Accuracy Score (Decision Tree): 0.74
```

- **Precision**: se obtuvo una precision del 0,74, esto representa que de todas las instancias positivas clasificadas solo el 74% de las mismas son verdades, este cálculo se realiza dividiendo los verdaderos positivos sobre la suma de verdaderos positivos y falsos positivos.

  Figura 18.

  Elaboración propia. (2024). Precision Score Decision Tree.

```
precision = precision_score(y_test, y_test_dt, average='weighted')
print(precision)
```

```
0.7493560387640216
```

- **Recall**: el recall al dividir los verdaderos positivos sobre la suma de los verdaderos positivos y falsos negativos, nos permitió calcular que el 74% de instancias positivas fueron identificadas correctamente por el modelo entre todas las instancias presentes en los datos de prueba.

Figura 19.

Elaboración propia. (2024). Recall Score Decision Tree.

```
recall = recall_score(y_test, y_test_dt, average='weighted')
print(recall)
```

```
0.742620203253214
```

- **F1-Score**: El F1-Score nos permitio obtener una combinación de la precision y el recall en un solo valor, donde arrojo que el rendimiento del modelo es del 0,74, sabiendo que el F1 score alcanza una precision y recall perfecto en 1 y su peor valor en 0.

Figura 20.

Elaboración propia. (2024). F1Score Decision Tree.

```
f1 = f1_score(y_test, y_test_dt, average='weighted')
print(f1)
```

```
0.7447018635686662
```

4.3 Random Forest:

- **Accuracy**: el modelo Random Forest permitió mejorar la precisión del 0,74 a un 0,77, esto se traduce en que el modelo ahora el 77% del total de las predicciones que se hicieron son correctas.

Figura 21.

Elaboración propia. (2024). Accuracy Score Random Forest.

```
print("Accuracy Score:", round(accuracy_score(y_test, y_test_rf),2))
```

```
Accuracy Score: 0.77
```

- **Precision**: en la precision se logró observar una mejoría significativa al mejorar 6 puntos porcentuales, ahora de las instancias positivas el 80% equivalen a verdaderas.
- **Recall**: tras la implementación de Random Forest, se observó un incremento notable en el recall del modelo, aumentando en 3 puntos porcentuales. Esto significa que ahora, el modelo logra identificar correctamente el 77% de las instancias positivas presentes en el conjunto de prueba.

  Figura 22.

  Elaboración propia. (2024). Recall Score Random Forest.

```
recall = recall_score(y_test, y_test_rf, average='weighted')
print("Recall Score:", round(recall, 2))

Recall Score: 0.77
```

- **F1-Score**: Con el método de ensamble utilizado logramos obtener un mejor rendimiento del modelo pasando de un 0,74 a un 0,78.

  Figura 23.

  Elaboración propia. (2024). F1 Score Random Forest.

```
f1 = f1_score(y_test, y_test_rf, average='weighted')
print("F1 Score:", round(f1, 2))

F1 Score: 0.78
```

4.4 CNN:

En el análisis de resultados para el Modelo CNN, revela una clara indicación de sobreajuste desde las fases iniciales del entrenamiento. Desde la primera época se evidencia una precisión del entrenamiento del 31.74 %, al contrario, la precisión de validación fue notablemente más baja que su contraparte con un valor del 1.44 %, lo que en conjunto elevada pérdida de validación del 8.75. Estos resultados, subrayan un desajuste entre el aprendizaje del modelo en los datos de entrenamiento y los datos de prueba.

Se reconoce el desajuste general en el proyecto, por lo cual, se ha propuesto implementar estrategias de mejora, como la inclusión de normalización por lotes y aumento de las tasas de abandonó con la finalidad de no solo optimizar el rendimiento del modelo en los datos de entrenamiento sino también mejorar su capacidad de generalización de nuevos datos, abordando la problemática general del proyecto.

5. Conclusiones

5.1 Conclusiones Generales

Al proponer una solución óptima frente al problema de clasificación de especies de aves, podemos concluir varios aspectos:

- Para lograr un buen rendimiento de un modelo de machine learning debemos primar por obtener en primera estancia los datos relevantes o lo relevante de los datos, en este caso transformar los audios completos a solamente obtener el sonido de ellos, permitió a su vez un modelo con mejor calidad de datos y mucho más ligero
- El proceso de implementación de SMOTE para balancear las clases fue fundamental para evitar el sesgo del modelo hacia la clase mayoritaria. Al generar muestras sintéticas para nivelar las clases, pudimos mejorar significativamente el rendimiento del modelo en la clasificación de la clase minoritaria. Esto nos permitió lograr una mayor equidad en la capacidad predictiva del modelo, garantizando así que todas las clases fueran tratadas de manera justa y precisa en el proceso de clasificación. En última instancia, el uso de SMOTE contribuyó a mejorar significativamente la capacidad del modelo para generalizar y hacer predicciones precisas en conjuntos de datos desbalanceados, lo que resulta en un modelo más robusto y confiable para abordar la clasificación de las especies de aves.

- En conclusión, al comparar el rendimiento de los modelos de ensamble Random Forest con el modelo de Redes Neuronales Artificiales (ANN), observamos mejoras significativas en las métricas de evaluación. Sin embargo, a pesar de la implementación del método de ensamble

para mejorar las métricas del árbol de decisión, el modelo de ANN sigue destacándose como la solución preferida para el problema específico abordado.

El modelo de ANN muestra un rendimiento superior en métricas clave como la precisión, recall, F1-Score y accuracy en comparación con Random Forest. Esto sugiere que, para el conjunto de datos y la tarea en particular, la capacidad de adaptación y flexibilidad de las Redes Neuronales Artificiales permite capturar relaciones complejas de manera más efectiva, lo que resulta en una mejor capacidad de generalización y predicción.

Aunque Random Forest es un método de ensamble poderoso que puede mejorar el rendimiento de los árboles de decisión individuales, en este caso específico, el modelo de ANN se destaca como la opción preferida debido a su capacidad para proporcionar una solución más precisa y efectiva para el problema en cuestión.

En cuanto al modelo CNN, con su capacidad para captar características espaciales y temporales en los espectrogramas, ha demostrado ser adecuado para esta tarea de clasificación. Con la experimentación y el ajuste, el modelo aprendió representaciones distintivas de cada canto de ave para su especie respectiva, lo que se tradujo en mejoras en las métricas de rendimiento.

Aunque existe un margen de mejora, los resultados resaltan la capacidad inmanente de los modelos convolucionales en tareas cómo: La clasificación avanzada de factores profundos e innatos de los espectrogramas, cómo también la identificación de especies mediante claves auditivas (canto). Este avance enriquece los conocimientos colectivos de nosotros los investigadores sobre la aplicación de algoritmos de aprendizaje automático en campos dedicados a la apreciación del medio ambiente, sino que sienta las bases para futuras investigaciones en estas disciplinas de clasificación con Modelo de Aprendizaje Profundo.

6. Referencias

Singh Chauhan, Nagesh. (2020). Audio Data Analysis Using Deep Learning with Python. KDnuggets. https://www.kdnuggets.com/2020/02/audio-data-analysis-deep-learning-python-part-1.html

Leiki, Igal. (2024). Understanding Binary Cross-Entropy and Log Loss for Effective Model Monitoring. Aporia. https://www.aporia.com/learn/understanding-binary-cross-entropy-and-log-loss-for-effective-model-monitoring/

IBM. (2020). ¿Qué es el descenso de gradiente?. https://www.ibm.com/mx-es/topics/gradient-descent#:~:text=IBM-,%C2%BFQu%C3%A9%20es%20el%20descenso%20de%20gradiente%3F,machine%20learning%20y%20redes%20neuronales.

Rodrigo, Ricardo. (2020). Estudyando. Error cuadrático medio. https://estudyando.com/error-cuadratico-medio-definicion-y-ejemplos/

Benhamadi, Salim. (2023). ESC-50: CNN for Spectrogram Classification. Kaggle. https://www.kaggle.com/code/salimhammadi07/esc-50-cnn-for-spectrogram-classification#II.-Data-Exploration-and-Visualization

Mlearnere.(2021). Learning from Audio: Spectrograms. Medium. https://towardsdatascience.com/learning-from-audio-spectrograms-37df29dba98c

Tomaselli, Francesco. (2023). Urban sound classification. https://github.com/tomfran/urban-sound-classification?tab=readme-ov-file#urban-sound-classification

For additional information on APA Style formatting, please consult the APA Style Manual, 7th Edition.