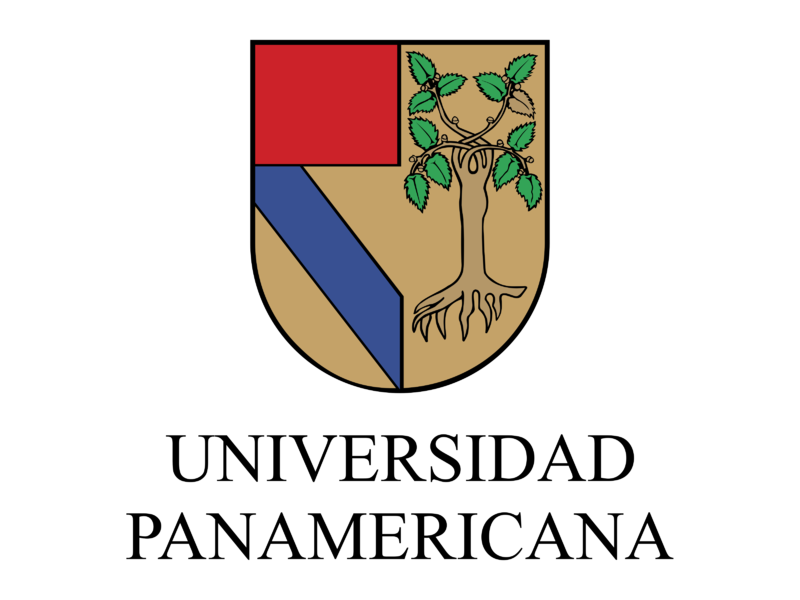
**Reporte proyecto redes neuronales**



Materia: Robótica industrial

Carlos Eduardo Pérez López 0196484

Aileen Sandoval González 0196737

Raúl Jiménez García 0196892

**Objetivo:**

El objetivo de este proyecto es entrenar una red neural para que pueda predecir cual es la emoción del habla de una persona mediante un audio. Esto se ejecutará mediante un código en lenguaje de programación Python.

**¿Qué es?**

El reconocimiento de la emoción del habla, abreviado como SER por sus siglas en inglés, es el acto de intentar reconocer la emoción humana y los estados afectivos del habla. Esto está capitalizando el hecho de que la voz a menudo refleja la emoción subyacente a través del tono y el pitch. Este es también el fenómeno que emplean animales como perros y caballos para poder comprender las emociones humanas. SER es difícil porque las emociones son subjetivas y anotar audio es un desafío.

**Requerimientos:**

Librerías:

* Librosa
* Soundfile
* Numpy
* Sklearn
* Pyaudio

Audios de aprendizaje:

* Esta es la base de datos audiovisuales de Ryerson del conjunto de datos de discurso y canción emocional, y se puede descargar de forma gratuita. Este conjunto de datos tiene 7356 archivos calificados por 247 personas 10 veces en cuanto a validez emocional, intensidad y autenticidad. El conjunto de datos completo es de 24.8 GB de 24 actores, pero hemos reducido la frecuencia de muestreo en todos los archivos, y puede descargarlo aquí <https://drive.google.com/file/d/1wWsrN2Ep7x6lWqOXfr4rpKGYrJhWc8z7/view>.

Una interfaz de audio para poder hacer tu código Python\_

* Nosotros utilizamos Google Colab por que tiene una muy buena capacidad de procesamiento para redes neuronales (operaciones matemáticas).
* Otro que también se puede usar es Jupyter notebook.

**Procedimiento:**

1.- Para empezar, vamos a instalar las librerías con el siguiente comando:

1. !pip install librosa soundfile numpy sklearn

2.- Nosotros tuvimos problemas (ERROR: Command errored out with exit status 1) al instalar la librería pyaudio, lo que se hizo fue instalar las otras librerías menos esta:

1. !pip install librosa soundfile numpy sklearn

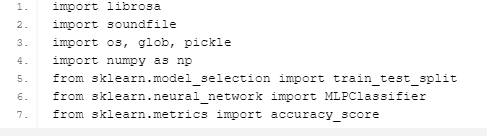
Después usamos esta la siguiente línea desinstalando, lo que nos impide instalar pyaudio:

1. !apt install libasound2-dev portaudio19-dev libportaudio2 libportaudiocpp0 ffmpeg

Para después instalar la librería:

1. !pip install pyaudio

3.- Hacemos las importaciones necesarias:



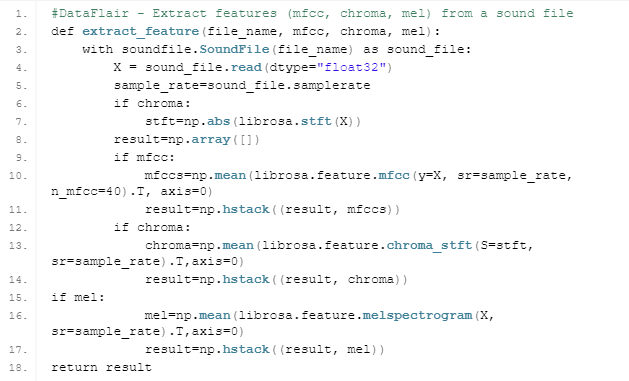
4.- Definimos una función extract\_feature para extraer las funciones mfcc, chroma y mel de un archivo de sonido. Esta función toma 4 parámetros: el nombre del archivo y tres parámetros booleanos para las tres características:

mfcc: coeficiente cepstral de frecuencia de mel, representa el espectro de potencia a corto plazo de un sonido

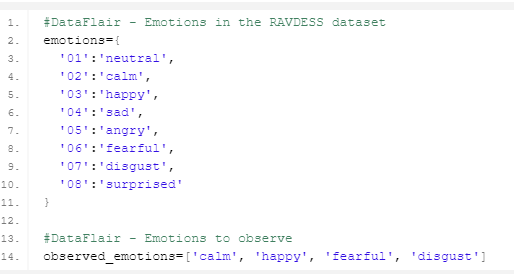
croma: pertenece a las 12 clases de tono diferentes

mel: Mel Frecuencia del espectrograma

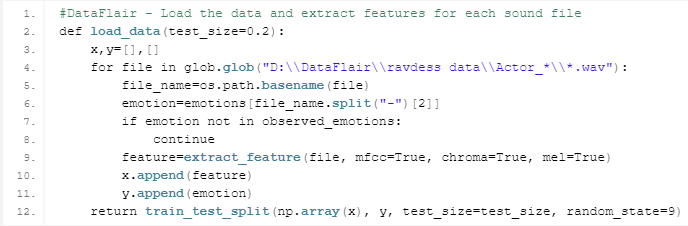
Abra el archivo de sonido con soundfile.SoundFile usando with-as para que se cierre automáticamente una vez que hayamos terminado. Lea de él y llámelo X. Además, obtenga la frecuencia de muestreo. Si chroma es True, obtenga la Transformada de Fourier a corto plazo de X.

Deje que el resultado sea una matriz numpy vacía. Ahora, para cada característica de las tres, si existe, llame a la función correspondiente desde librosa.feature (por ejemplo, librosa.feature.mfcc para mfcc), y obtenga el valor medio. Llame a la función hstack () desde numpy con el resultado y el valor de la característica, y almacene esto en el resultado. hstack () apila matrices en secuencia horizontalmente (en forma de columnas). Luego, devuelve el resultado.

5.- Ahora, definamos un diccionario para contener los números y las emociones disponibles en el conjunto de datos RAVDESS, y una lista para contener los que queremos: calma, felicidad, miedo y asco.



6.- Usando nuestro diccionario de emociones, este número se convierte en una emoción, y nuestra función verifica si esta emoción está en nuestra lista de emociones observadas; si no, continúa al siguiente archivo. Hace una llamada a extract\_feature y almacena lo que se devuelve en "característica". Luego, agrega la característica a xy la emoción a y. Entonces, la lista x contiene las características e y contiene las emociones. Llamamos a la función train\_test\_split con estos, el tamaño de la prueba y un valor de estado aleatorio, y lo devolvemos. Tenemos que estar muy atentos en el enlace de los archivos.



7.- Es hora de dividir el conjunto de datos en conjuntos de entrenamiento y prueba. Mantengamos el conjunto de prueba 25% de todo y usemos la función load\_data para esto.



8.- Observe la forma de los conjuntos de datos de entrenamiento y prueba:



9.- Y obtenga la cantidad de características extraídas:



10.- Ahora, inicialicemos un MLPClassifier. Este es un clasificador de perceptrón multicapa; optimiza la función de pérdida de registro utilizando LBFGS o el descenso de gradiente estocástico. A diferencia de SVM o Naive Bayes, el MLPClassifier tiene una red neuronal interna para fines de clasificación. Este es un modelo ANN de avance.



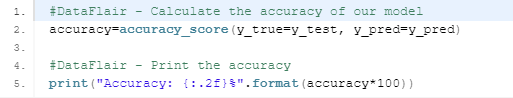
11.- Montar / entrenar a la modelo.



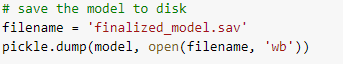
12.- Vamos a predecir los valores para el conjunto de prueba. Esto nos da y\_pred (las emociones predichas para las características del conjunto de prueba).



13.- Para calcular la precisión de nuestro modelo, llamaremos a la función precision\_score () que importamos de sklearn. Finalmente, redondearemos la precisión a 2 decimales e imprimiremos.



14.- Si el modelo es del agrado se guarda con el siguiente código, si no les gusta jugamos con los valores del punto 10. Para guardar el modelo se usará el siguiente código:



15.- Ahora, tenemos que hacer otro archivo para probar con tus propios audios. Se tienen que hacer los pasos del 1 al 5 en este nuevo documento. Cargaremos nuestros propios audios de la siguiente manera:



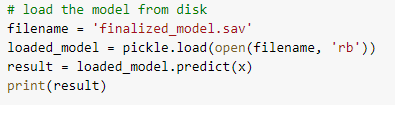
A que tener mucho cuidado con escribir bien la ubicación de los archivos de tus audios.

16.- Lo siguiente es elegir el audio que vamos a querer analizar, con las siguientes líneas



A que tener en cuenta que la ubicación de ese archivo es el de nosotros, asegúrate de poner la ubicación de tu audio.

17.- Para finalizar, tenemos que cargar el modelo que guardamos en el punto 14. Y ponemos a funcionar nuestro modelo. Nos entregará emoción de nuestro audio elegido.



Nota: los archivos que vas a poner a prueba tienen que estar en formato .wav con una resolución de 256 kbps.

**Conclusión:**

En este proyecto de Python, aprendimos a reconocer las emociones del habla. Usamos un MLPClassifier para esto e hicimos uso de la biblioteca de archivos de sonido para leer el archivo de sonido, y la biblioteca de librosa para extraer características de él. Aprendimos como guardar el modelo que mejor precisión nos entrego y como usar este, en otro archivo para analizar el audio que nosotros queramos. Este es un ejemplo del uso de las redes neuronales y de la inteligencia artificial.

**Referencias:**

<https://machinelearningmastery.com/save-load-machine-learning-models-python-scikit-learn/>

<https://data-flair.training/blogs/python-mini-project-speech-emotion-recognition/>