Proyecto Final HyAIA

CLASIFICACIÓN DE GÉNEROS DE MÚSICA

María Gabriela Ramírez Castillo

10 diciembre 2025

OBJETIVO

- Clasificar géneros musicales a partir de características numéricas.
- Entrenar una red neuronal MLP usando TensorFlow/Keras

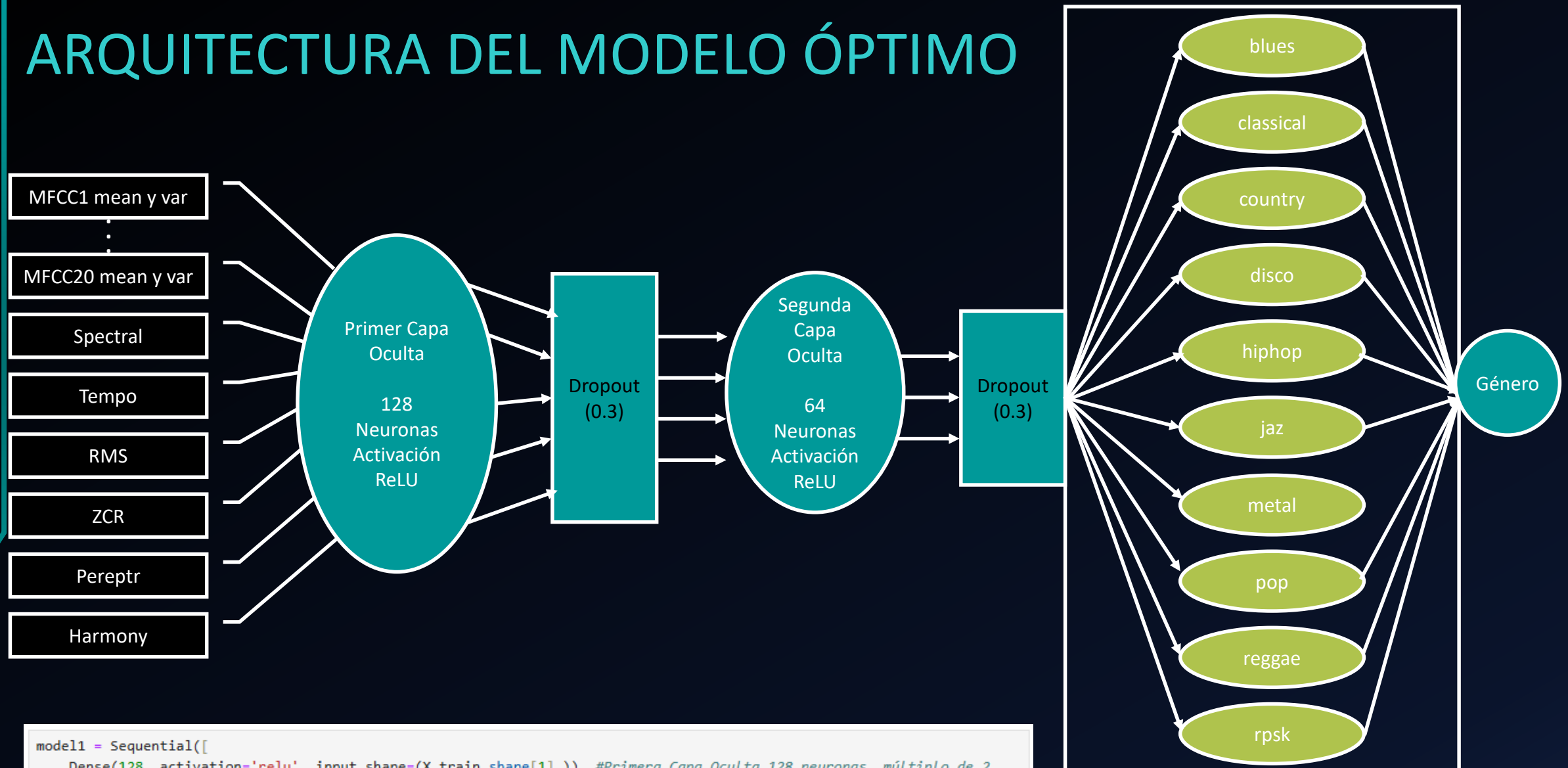
EXPLORACIÓN DE DATOS

- Archivo: features_30_sec.csv
- 1,000 registros
- 60 variables (58 numéricas, 2 texto)
- 10 géneros musicales

	filename	length	chroma_stft_mean	chroma_stft_var	rms_mean	rms_var
0	blues.00000.wav	661794	0.350088	0.088757	0.130228	0.002827
1	blues.00001.wav	661794	0.340914	0.094980	0.095948	0.002373
2	blues.00002.wav	661794	0.363637	0.085275	0.175570	0.002746
3	blues.00003.wav	661794	0.404785	0.093999	0.141093	0.006346
4	blues.00004.wav	661794	0.308526	0.087841	0.091529	0.002303

label	filename
blues	100
classical	100
country	100
disco	100
hiphop	100
jazz	100
metal	100
pop	100
reggae	100
rock	100

ARQUITECTURA DEL MODELO ÓPTIMO




```
model1 = Sequential([
    Dense(128, activation='relu', input_shape=(X_train.shape[1],)), #Primera Capa Oculta 128 neuronas, múltiplo de 2
    Dropout(0.3),
    Dense(64, activation='relu'),   #Segunda Capa Oculta 64 neuronas
    Dropout(0.3),
    Dense(10, activation='softmax') #Capa final, 10 neuronas
])
model1.summary() #Línea proporcionada por compañera pra
```

FASE DE ENTRENAMIENTO – DOS MODELOS

PARÁMETROS	Modelo 1	Modelo 2
ÉPOCAS	50	30
BATCH _ SIZE	32	16
SPLIT	20%	20%
OPTIMIZACIÓN	ADAM	
LOSS	CATEGORICAL_CROSSENTROPY	
VALIDACIÓN	ACCURACY	

FASE DE ENTRENAMIENTO - CÓDIGO

- MODELO 1 - ÓPTIMO

```
history1 = model1.fit(X_train, y_train, epochs=50, batch_size=32, validation_split=0.2, verbose=1)
```

```
Epoch 1/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.1906 - loss: 2.2903 - val_accuracy: 0.3688 - val_loss: 1.9006
```

```
Epoch 50/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8922 - loss: 0.3166 - val_accuracy: 0.7625 - val_loss: 0.7890
```

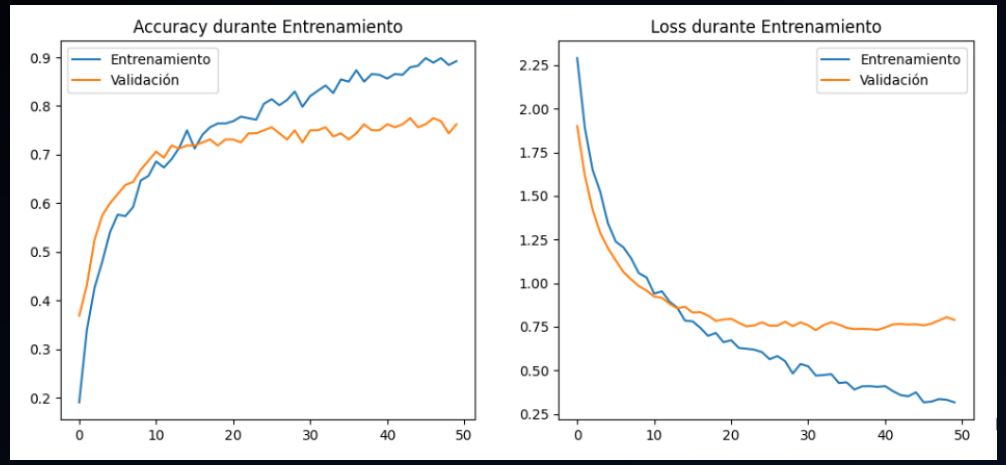
- MODELO 2

```
history2 = model2.fit(X_train, y_train, epochs=30, batch_size=16, validation_split=0.2, verbose=1)
```

```
Epoch 1/30
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - accuracy: 0.1500 - loss: 2.3668 - val_accuracy: 0.3438 - val_loss: 1.9685
```

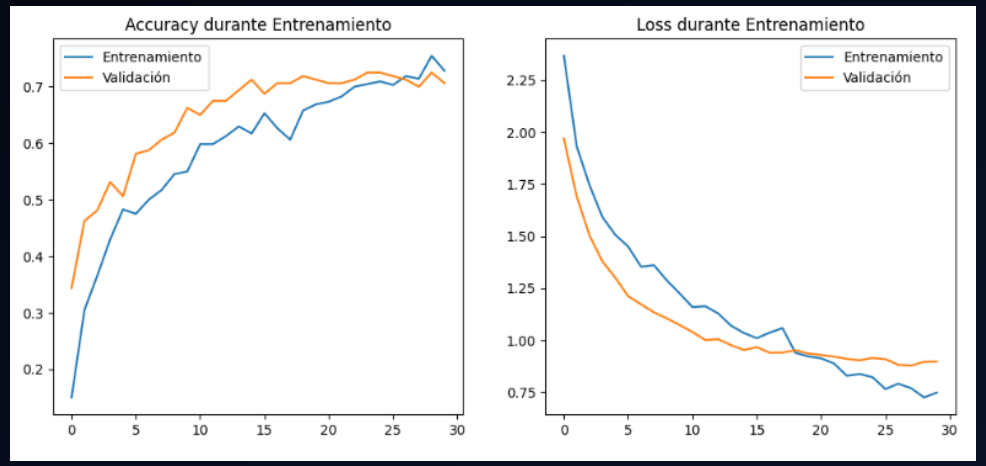
```
Epoch 30/30
40/40 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.7281 - loss: 0.7480 - val_accuracy: 0.7063 - val_loss: 0.8968
```

FASE DE ENTRENAMIENTO



Modelo 1:

- Mayor Precisión
  89%
- Menor Pérdida
  3.1%



Modelo 2:

- Menor Precisión
  72.8%
- Mayor Pérdida
  7.4%

PREDICCIÓN

- MODELO 1

Modelo 1

[34]:

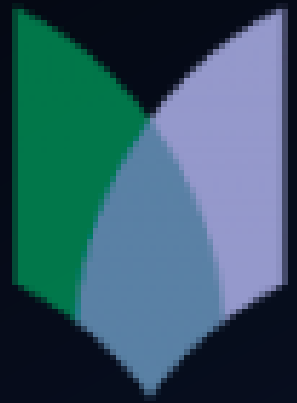
```
predictions = model1.predict(X_test[:5])
print("Predicciones Modelo 1:", predictions.argmax(axis=1))
pred_generos = encoder.inverse_transform(predictions.argmax(axis=1)) #convierte

print(pred_generos)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 81ms/step
Predicciones Modelo 1: [3 7 7 6 4]
['disco' 'pop' 'pop' 'metal' 'hiphop']
```

CONCLUSIÓN

- La aplicación de técnicas y hiperparametros; número de capas, de neuronas por capa, épocas, dropout, pueden mejorar, optimizar o debilitar, reducir en forma general, requiere de practicar.
- En proyecto avanzado, podría aplicar la implementación del modelo con espectrogramas (carpeta de imágenes PNG o audio WAV) mediante la implementación de CNN.
- Utilizar y evaluar el modelo con el dataset "features_3_sec.csv" que contiene 9,990 registros con menor tiempo.



ALINNCO