In [1]:
```
import os

import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

import sklearn

import plotly.express as px
import plotly.graph_objects as go

from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import StandardScaler
from plotly.subplots import make_subplots
```

In [2]:
```
# Obtener el directorio actual de trabajo
directorio_actual = os.getcwd()

# Especificar la ruta relativa desde el directorio actual
ruta_csv_relativa = os.path.join('..', 'data', '01_raw','spotify.csv')

# Cargar el archivo CSV
spotify = pd.read_csv(ruta_csv_relativa)
```

In [3]:
```
# Limpiar los datos (eliminar filas con valores nulos)
spotify.dropna(inplace=True)

# Definir una variable dependiente (ejemplo: popularidad > 50)
spotify['popular'] = (spotify['popularity'] > 50).astype(int)

# Seleccionar las variables independientes (solo datos numéricos)
X = spotify[['duration_ms', 'danceability', 'energy', 'key', 'loudness',
             'mode', 'speechiness', 'acousticness', 'instrumentalness',
             'liveness', 'valence', 'tempo', 'time_signature']]

# Definir la variable dependiente
y = spotify['popular']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Escalar los datos
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Crear y entrenar el modelo
model = LogisticRegression(max_iter=2000)  # Aumentar a 2000 iteraciones
model.fit(X_train_scaled, y_train)

# Hacer predicciones
y_pred = model.predict(X_test_scaled)

# Evaluar el modelo
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[17308     5]
 [ 5480     7]]
              precision    recall  f1-score   support

           0       0.76      1.00      0.86     17313
           1       0.58      0.00      0.00      5487

    accuracy                           0.76     22800
   macro avg       0.67      0.50      0.43     22800
weighted avg       0.72      0.76      0.66     22800
```