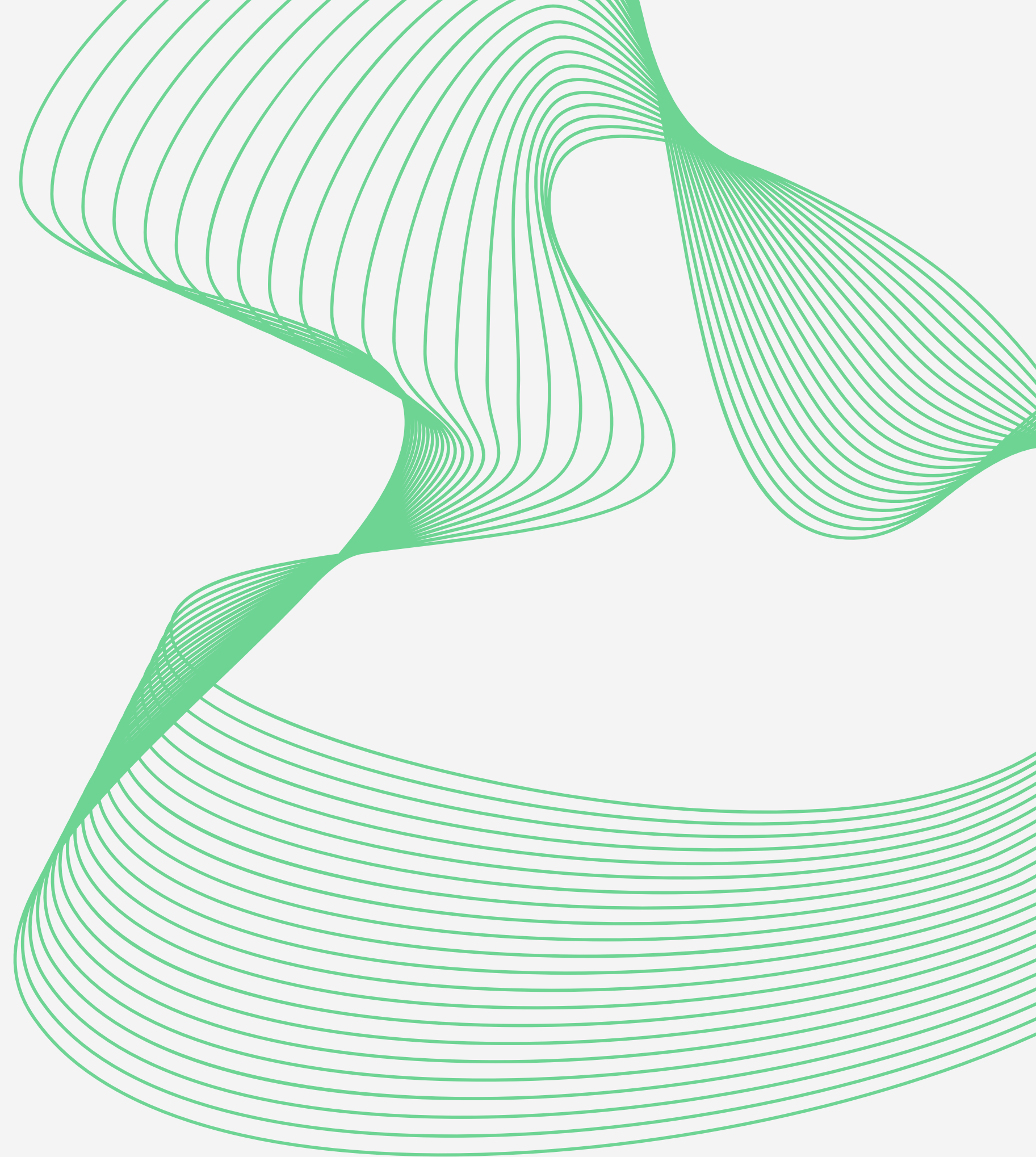


# Propuesta MACHINE LEARNING



# Introducción

- Predecir la “danceability” de las canciones para abordar necesidad de negocio de Spotify.
- Análisis de datos
- Modelo ML



# Análisis de datos

Dataset de Música → Kaggle


No valores vacíos + Estandarizados


50.000 filas → Canciones

17 columnas → Variables: popularidad, volumen, energía..

#	Column	Non-Null	Count	Dtype
0	instance_id	50000	non-null	float64
1	artist_name	50000	non-null	object
2	track_name	50000	non-null	object
3	popularity	50000	non-null	float64
4	acousticness	50000	non-null	float64
5	danceability	50000	non-null	float64
6	duration_ms	50000	non-null	float64
7	energy	50000	non-null	float64
8	instrumentalness	50000	non-null	float64
9	key	50000	non-null	object
10	liveness	50000	non-null	float64
11	loudness	50000	non-null	float64
12	mode	50000	non-null	object
13	speechiness	50000	non-null	float64
14	tempo	50000	non-null	object
15	obtained_date	50000	non-null	object
16	valence	50000	non-null	float64
17	music_genre	50000	non-null	object

Volumen (dB) 

Valencia 

Energía 

Hablabilidad 

Acústica 

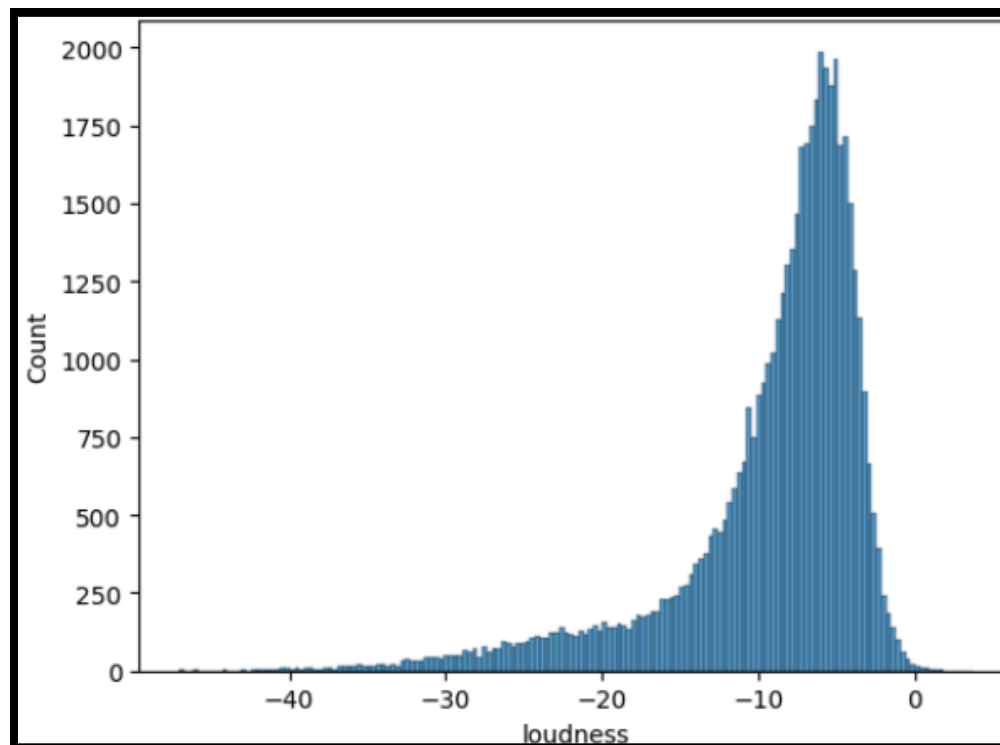
Instrumentalidad 

Popularidad 

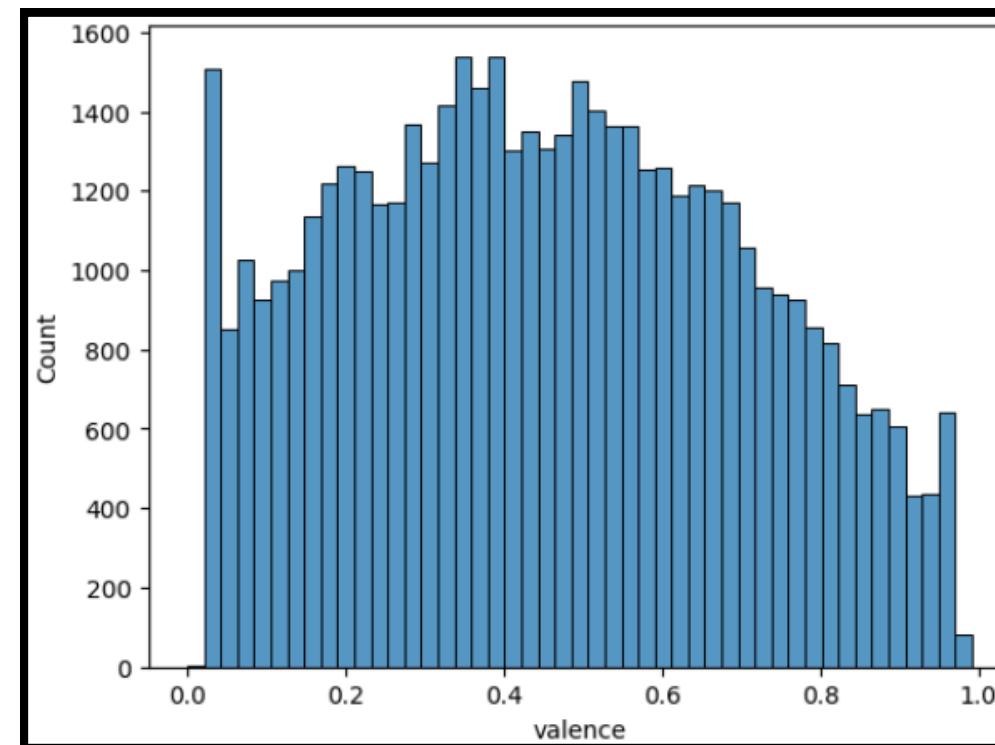


# Análisis de datos

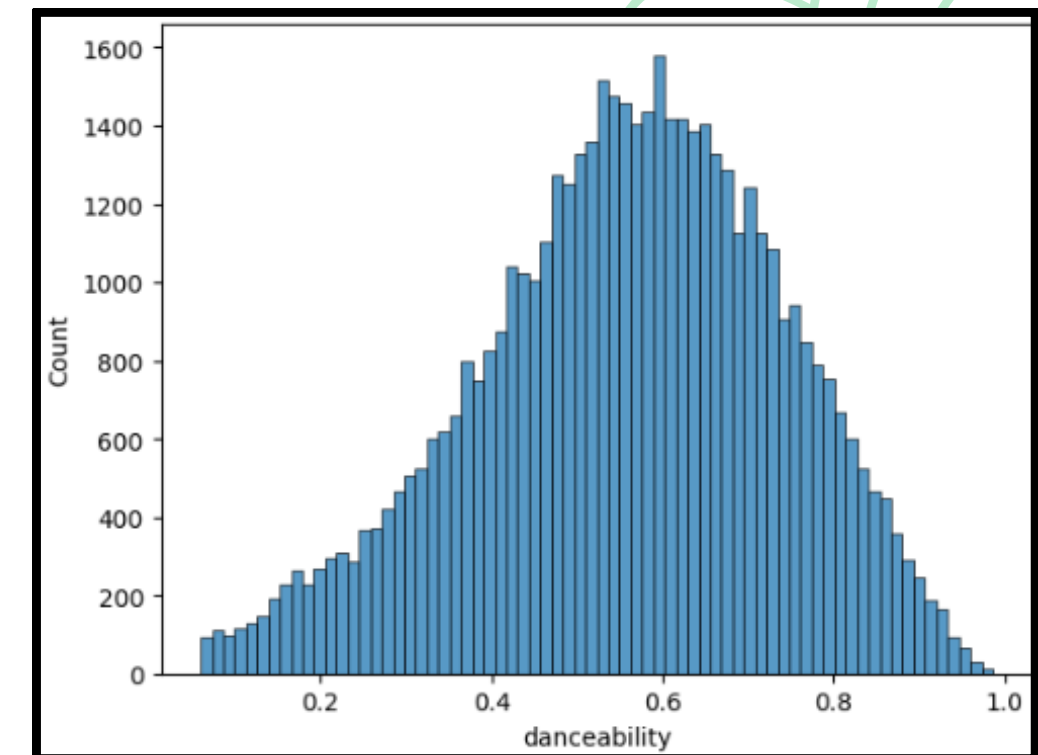
## Volumen (dB)



## Valencia

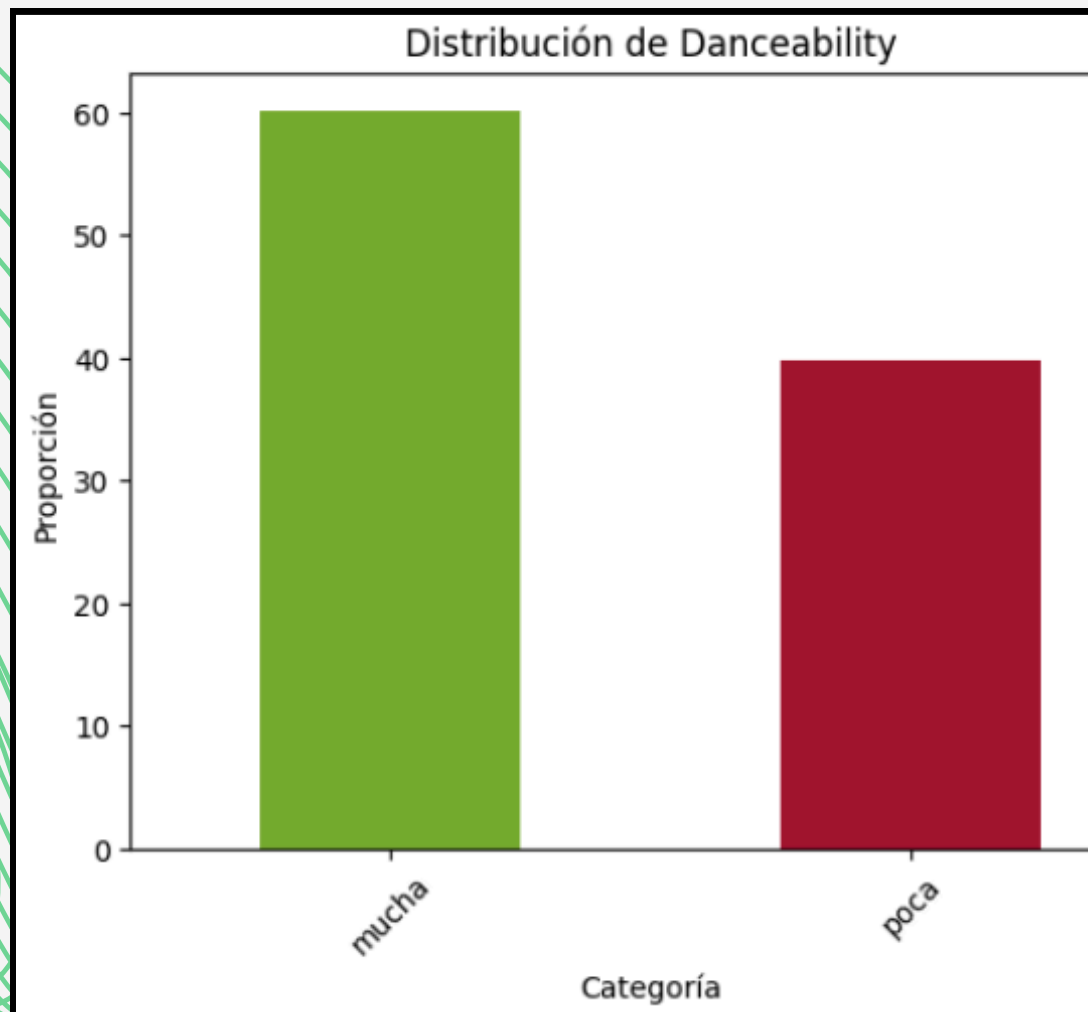


## Danzabilidad



# Preprocesamiento de datos

## Distribución del target



Mucha 'danceability' 60%  
Poca 'danceability' 40%

## Codificación del target



1

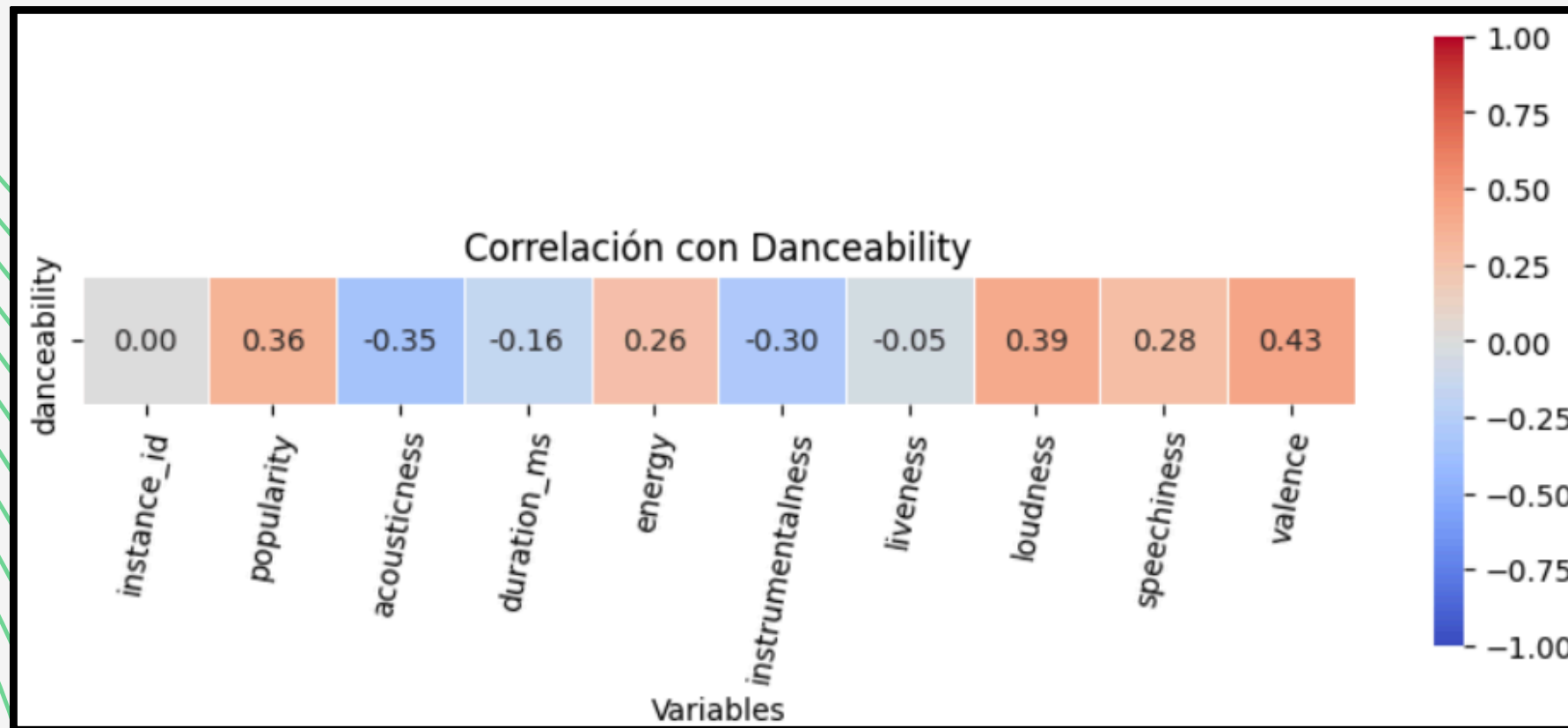
0



Problema de clasificación

# Preprocesamiento de datos

## Correlaciones con el target



### Positivamente correladas

- **Popularidad** (0.36)
- **Energía** (0.26)
- **Volumen** (0.39)
- **Hablabilidad** (0.28)
- **Valencia** (0.43)

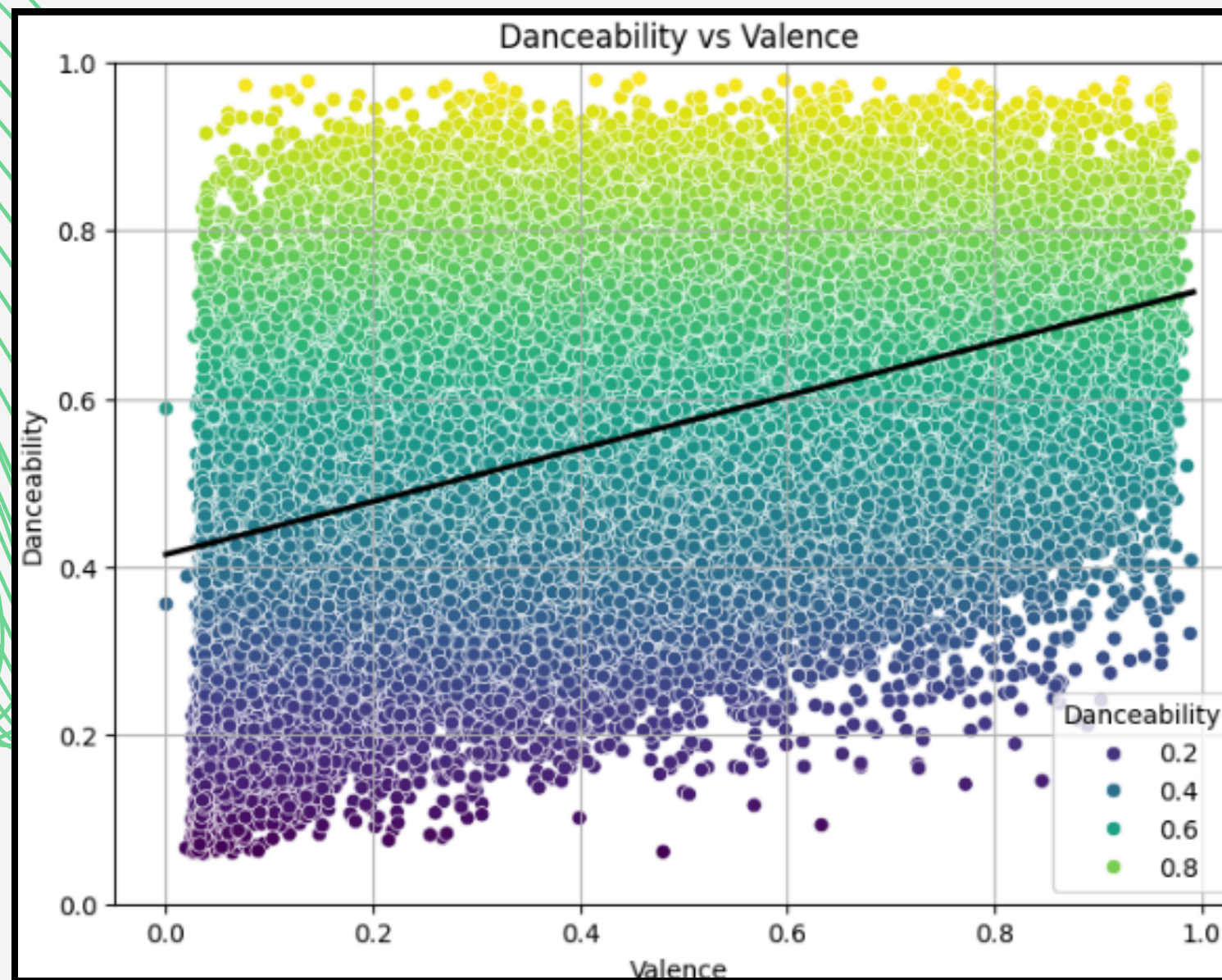
### Negativamente correladas

- **Acústica** (-0.35)
- **Instrumentalidad** (-0.3)

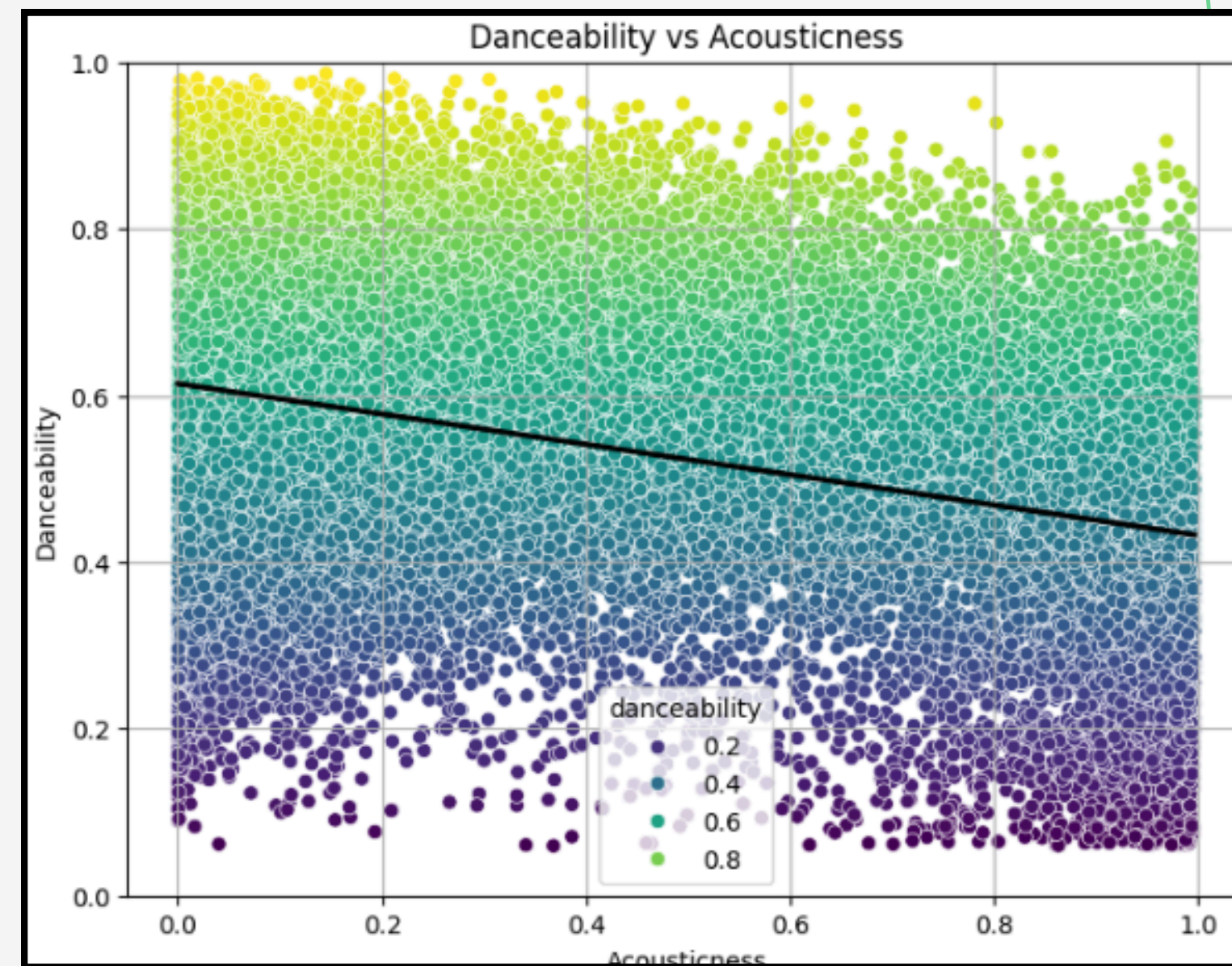


# Correlaciones

## Valencia vs Danzabilidad



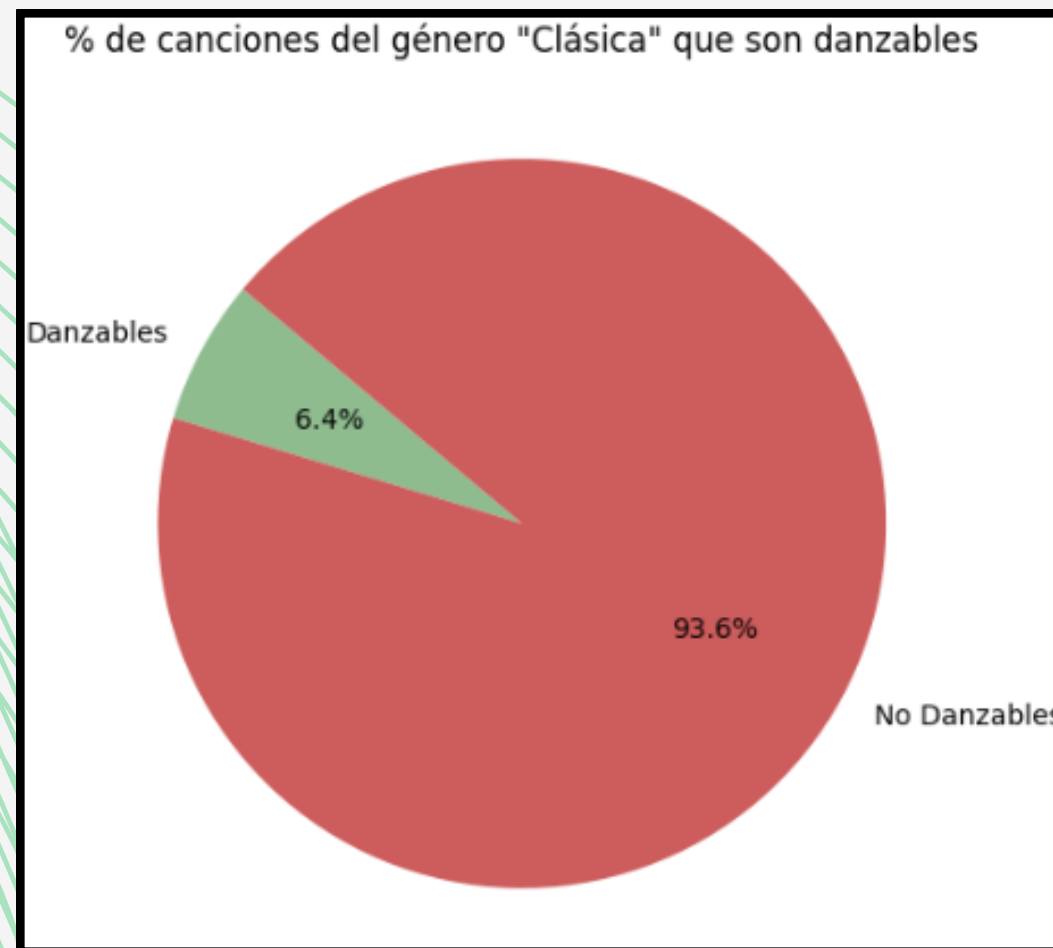
## Acústica vs Danzabilidad



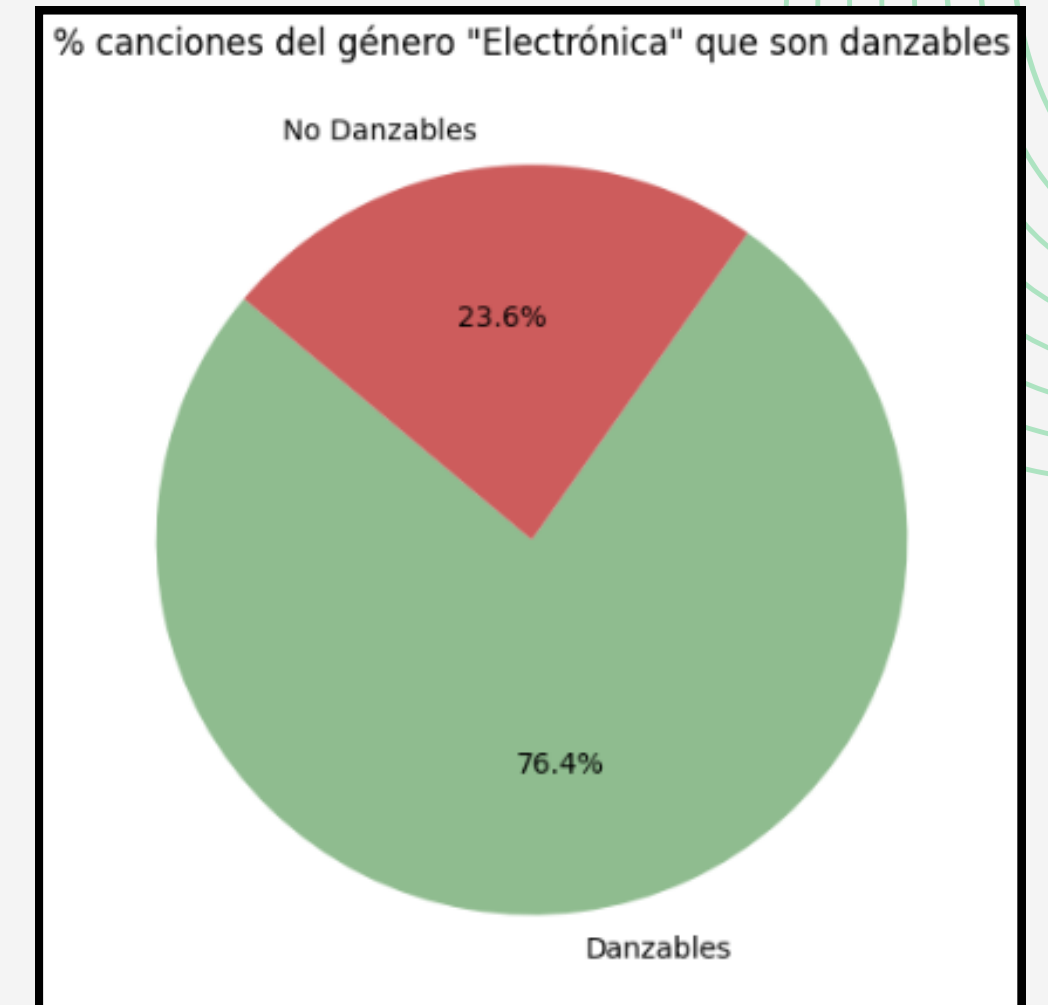


# Entendiendo correlaciones

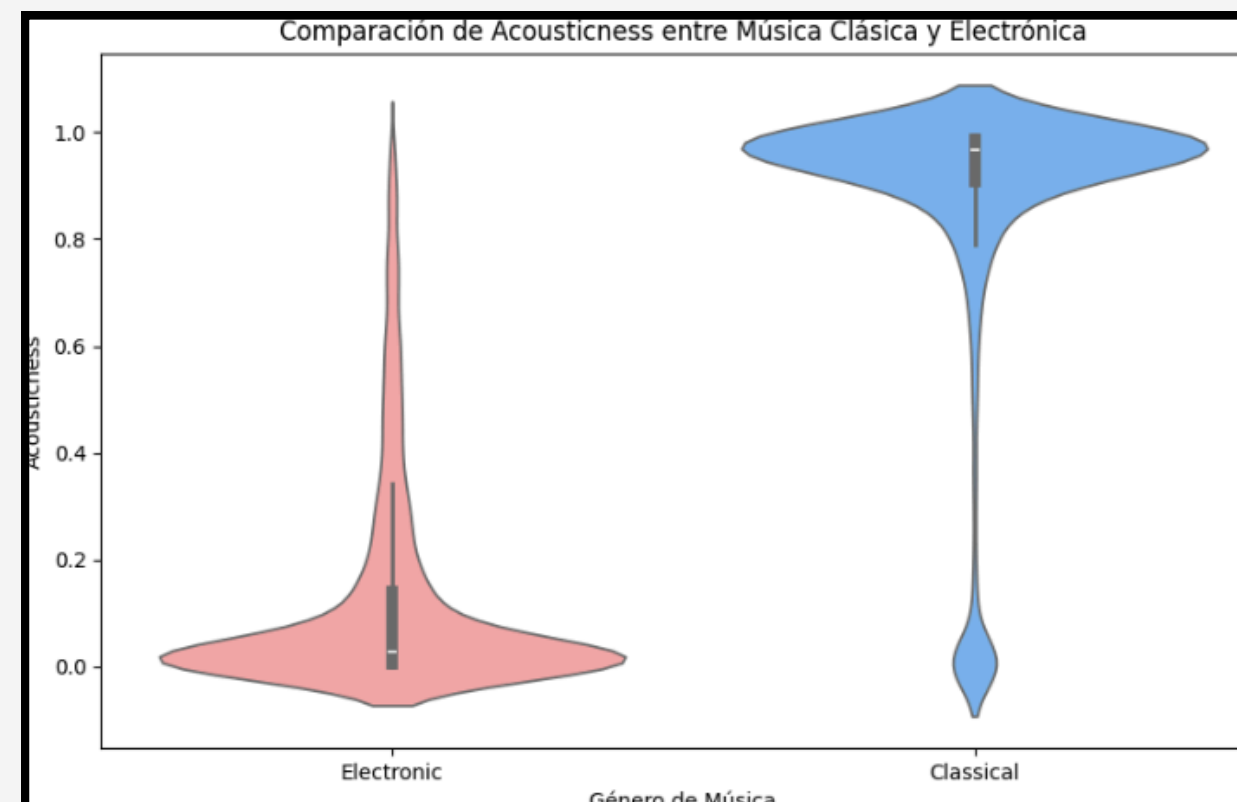
## Música clásica vs. danzabilidad



## Música electrónica vs. danzabilidad



## Acústica por género musical





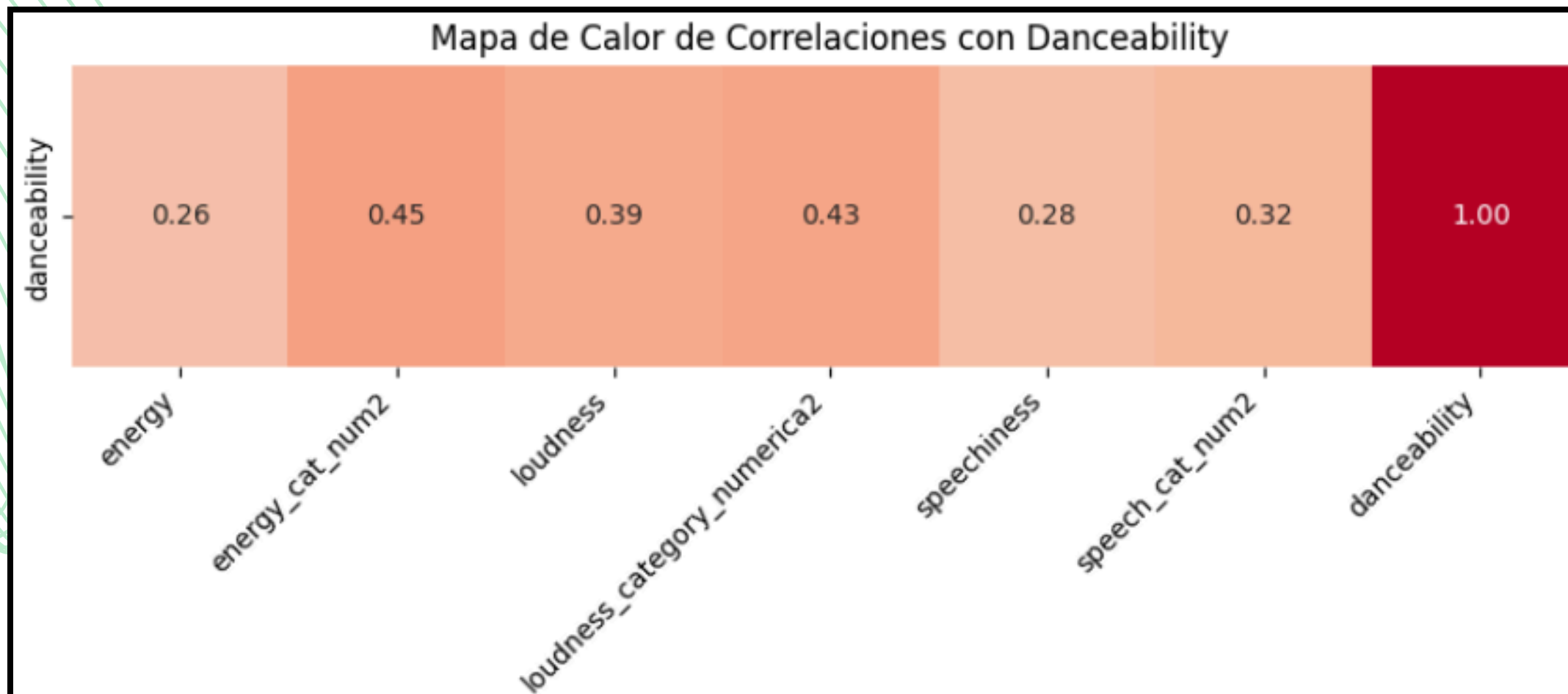
# Aumentando correlaciones

1. Dividir la variable en rangos y asignar números
2. Agrupar la variable por la media de "danceability"
3. Reajustar los valores si fuera necesario

```
music.loc[music["energy"] < 0.2, "energy_cat_num"] = 0
music.loc[(music["energy"] >= 0.2) & (music["energy"] < 0.4), "energy_cat_num"] = 1
music.loc[(music["energy"] >= 0.4) & (music["energy"] < 0.6), "energy_cat_num"] = 2
music.loc[(music["energy"] >= 0.6) & (music["energy"] < 0.8), "energy_cat_num"] = 3
music.loc[(music["energy"] >= 0.8) & (music["energy"] < 0.9999), "energy_cat_num"] = 4
```

```
music.groupby('energy_cat_num')['danceability'].mean().sort_values(ascending=True)
```

```
music.loc[music['energy_cat_num'] == 0, "energy_cat_num2"] = 0
music.loc[music['energy_cat_num'] == 1, "energy_cat_num2"] = 2
music.loc[music['energy_cat_num'] == 4, "energy_cat_num2"] = 4
music.loc[music['energy_cat_num'] == 3, "energy_cat_num2"] = 6
music.loc[music['energy_cat_num'] == 2, "energy_cat_num2"] = 7
```



- Energía 0.26 – 0.45
- Volumen 0.39 – 0.43
- Hablabilidad 0.28 – 0.32

\*Multicolinealidad

# Modelado

## Variables predictoras útiles

- Popularidad
- Energía
- Volumen
- Letra
- Valencia
- Acústica
- Instrumentalidad

## Target

- Danzabilidad

## Resultados sobre TEST

	Modelo	Accuracy	Precision	Recall
0	RandomForest	0.7435	0.739586	0.887895
1	XGBClassifier	0.7554	0.769542	0.849313
2	GradientBoosting	0.7447	0.751733	0.861898
3	GradientBoostingKBest	0.7468	0.755426	0.858751
4	KNN	0.7164	0.750430	0.794668
5	SVC	0.6825	0.687779	0.868521
6	RandomForest_PCA	0.7337	0.741488	0.858255
7	RandomForestkfold	0.7488	0.757333	0.859414

# XGB Classifier

## ¿Cómo se ha entrenado?

- Pipeline
- Parámetros:
  - n\_estimators
  - max\_depth
  - learning\_rate
  - subsample
  - colsample\_bytree
  - Regularizaciones L1 y L2
- RandomizedSearch (cv=4)

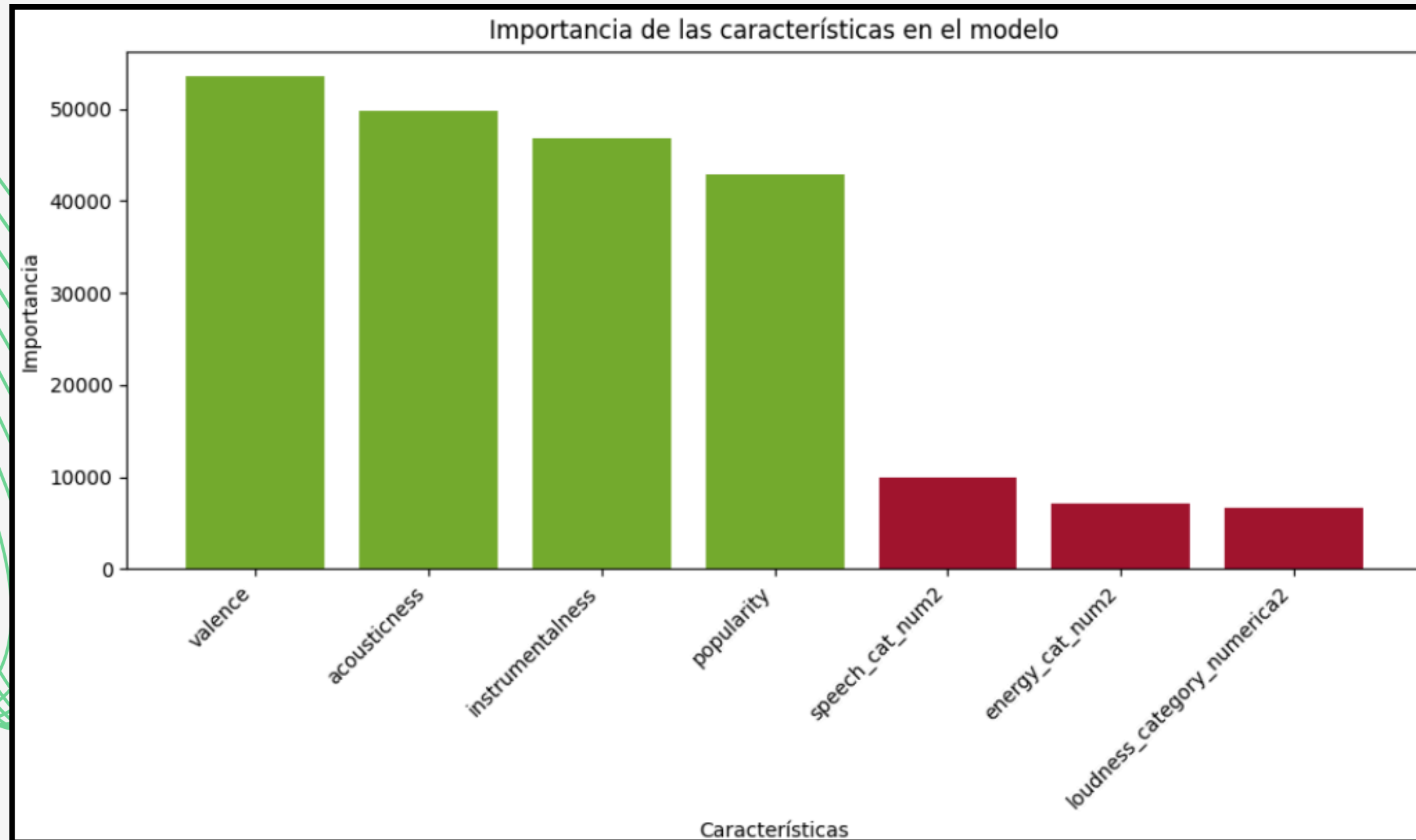
## Feature importance

	Feature	Importance
3	valence	53563.0
1	acousticness	49784.0
2	instrumentalness	46832.0
0	popularity	42932.0
5	speech_cat_num2	9911.0
6	energy_cat_num2	7162.0
4	loudness_category_numerica2	6696.0



# XGB Classifier

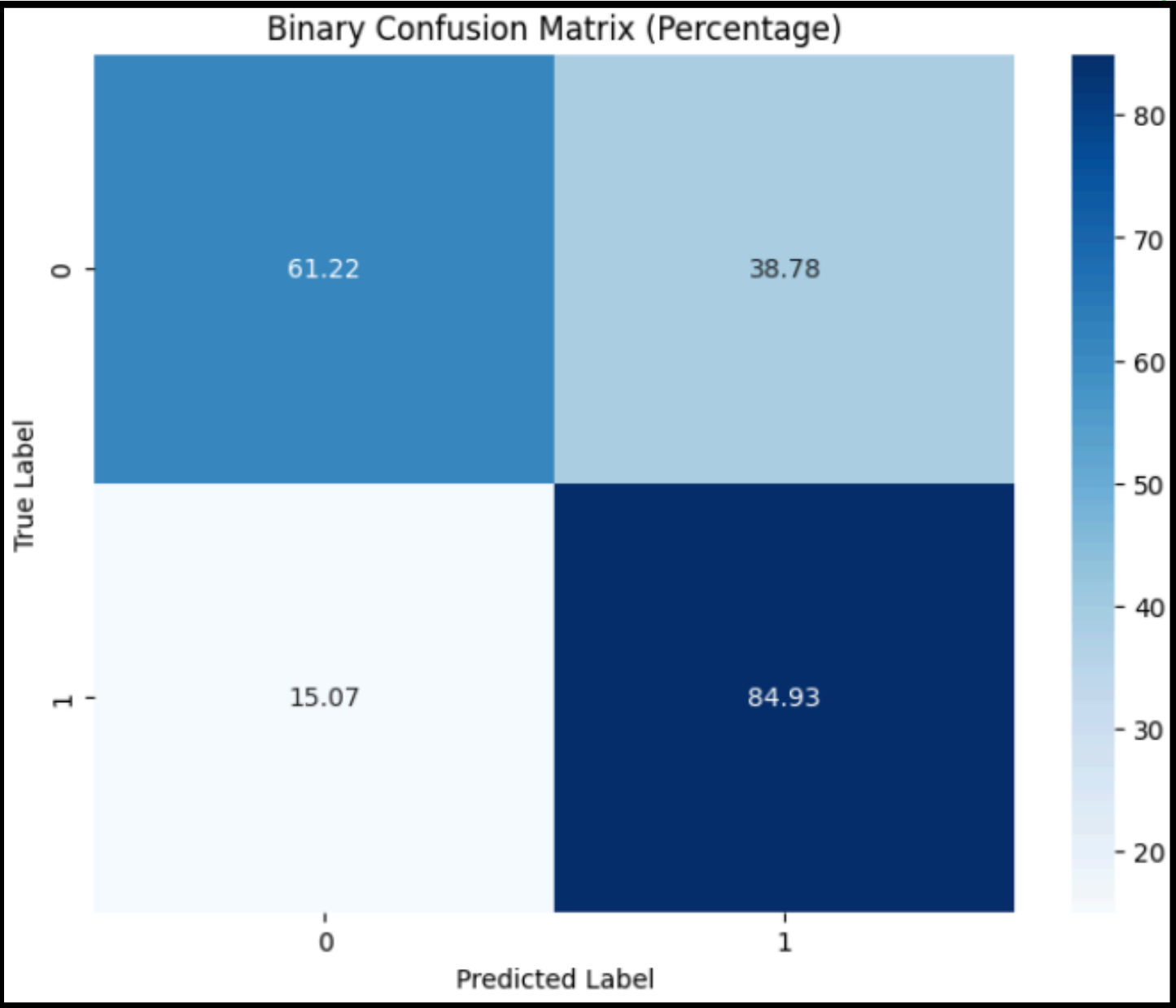
## Feature importante



- **Valencia: 25%**
- **Acústica: 23%**
- **Instrumentalidad: 21%**
- **Popularidad: 20%**
- Letra/habla: 5%
- Energía: 3%
- Volumen: 3%

# Evaluación

ACCURACY	75%
PRECISION	77%
RECALL	85%
F1 SCORE	87%

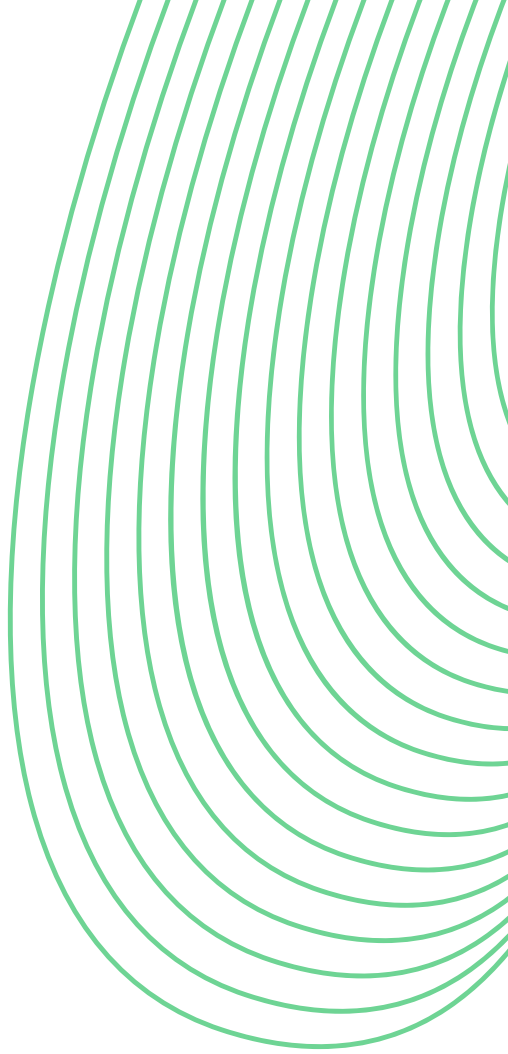


# Limitaciones y mejoras

**Problema de regresión**

**División del target en 3 rangos: desproporción + resultados mejorables**

**Mayor preprocesamiento de datos y jugar con nuevas variables**





# Streamlit

## Importar modelo y setear la página: Home + Predicción

```
pickle_in = open('../models_class/final_model2.pkl', 'rb')
modelo_rf = pickle.load(pickle_in)

st.set_page_config(page_title="Predicción de danzabilidad", page_icon=":music")

seleccion = st.sidebar.selectbox("Selecciona una opción", ["Home", "Predicción"])
```

## Input de variables + transformación

```
elif seleccion == "Predicción":
    st.title("¡Vamos a predecir la danceability de la canción!")
    nombre_cancion = st.text_input("Ingresa el nombre de la canción")
    popularity = st.slider("Popularity", min_value=-60.0, max_value=0.0, value=-10.0)
    acousticness = st.slider("Energy", min_value=0.0, max_value=1.0, value=0.7)
    instrumentalness = st.slider("Instrumentalness", min_value=0.0, max_value=1.0, value=0.1)
    valence = st.slider("Valence", min_value=0.0, max_value=1.0, value=0.1)
    loudness = st.slider("Loudness", min_value=0.0, max_value=1.0, value=0.1)
    speechiness = st.slider("Speechiness", min_value=0.0, max_value=1.0, value=0.1)
    energy = st.slider("Energy", min_value=0.0, max_value=1.0, value=0.1)
```

## Predicción en DF

```
prediccion = [-1]
if st.button("Predecir"):
    prediccion = modelo_rf.predict(datos)
    st.write("Resultado de la predicción:")
    if prediccion[0] == -1:
        st.write("Haz tu predicción")
    elif prediccion[0] < 0.5:
        st.write("La canción", nombre_cancion, "tiene poca danceability.")
    else:
        st.write("La canción", nombre_cancion, "tiene mucha danceability.")
    df_resultado = pd.DataFrame({
        'Nombre de la canción': [nombre_cancion],
        'Popularity': [popularity],
        'Acousticness': [acousticness],
        'Instrumentalness': [instrumentalness],
        'Valence': [valence],
        'Loudness Category': [loudness_category_numerica2],
        'Speechiness Category': [speech_cat_num2],
        'Energy Category': [energy_cat_num2],
        'Predicción': [prediccion[0]]
    })
    st.write("Resultados de la predicción:")
    st.write(df_resultado)
```

## Acceso a Streamlit

**¡Muchas  
gracias!**

