



**PREDICTION MODEL**

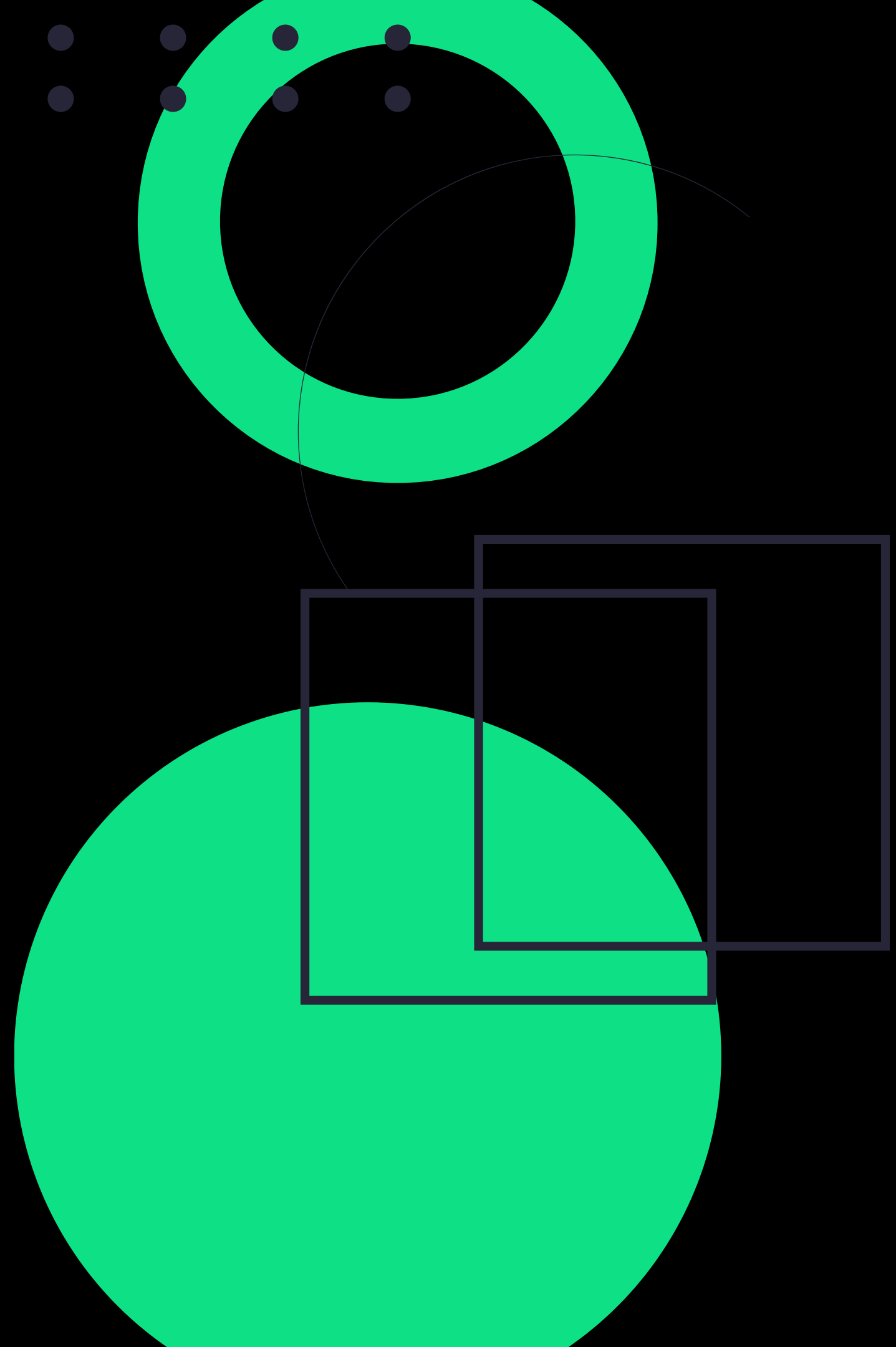
# **SPOTIFY RECOMMENDATIONS**

Check the success of your next hit!

Carlota Gordillo | Celia Manzano | Laura Sánchez

# Project Index

- 01 Preview Analysis
- 02 Data Selection and Preparation
- 03 Feature Engineering and Selection
- 04 Model Building and Evaluation
- 05 Hyperparameter Tuning and Model Optimization
- 06 Insights and Impact
- 07 Future Work and Improvements



# Our data set

## Danceability

Measure of how danceable the song is, based on rhythm, stability, beat strength and regularity (0 to 1).

## Energy

Perceived level of intensity and activity of the song. High values represent fast and loud songs (0 to 1).

## Valence

Emotional positivity of the song. High values indicate positivity (joy, euphoria), low values indicate sadness (0 to 1).

## Tempo

Estimated tempo of the song in beats per minute (BPM).

## Instrumentalness

Estimated tempo of the song in beats per minute (BPM). Predicts the amount of vocal elements in a song. Higher values indicate more instrumentals (0 to 1).

## Loudness

Overall track volume in decibels (dB). Generally ranges from -60 to 0 dB.

## Popularity

Measure of how popular the song is (0 to 100)



# 232.725 Tracks

## Key

Pitch of the song represented as an integer (0 = Do, 1 = Do#, ... 11 = Si).

## Mode

Song mode: Major (1) or Minor (0).

## Speechiness

Number of spoken words in the song. High values indicate spoken content (such as podcasts).

## Acousticness

Probability that the track is acoustic. Higher values indicate more acoustic content (0 to 1).

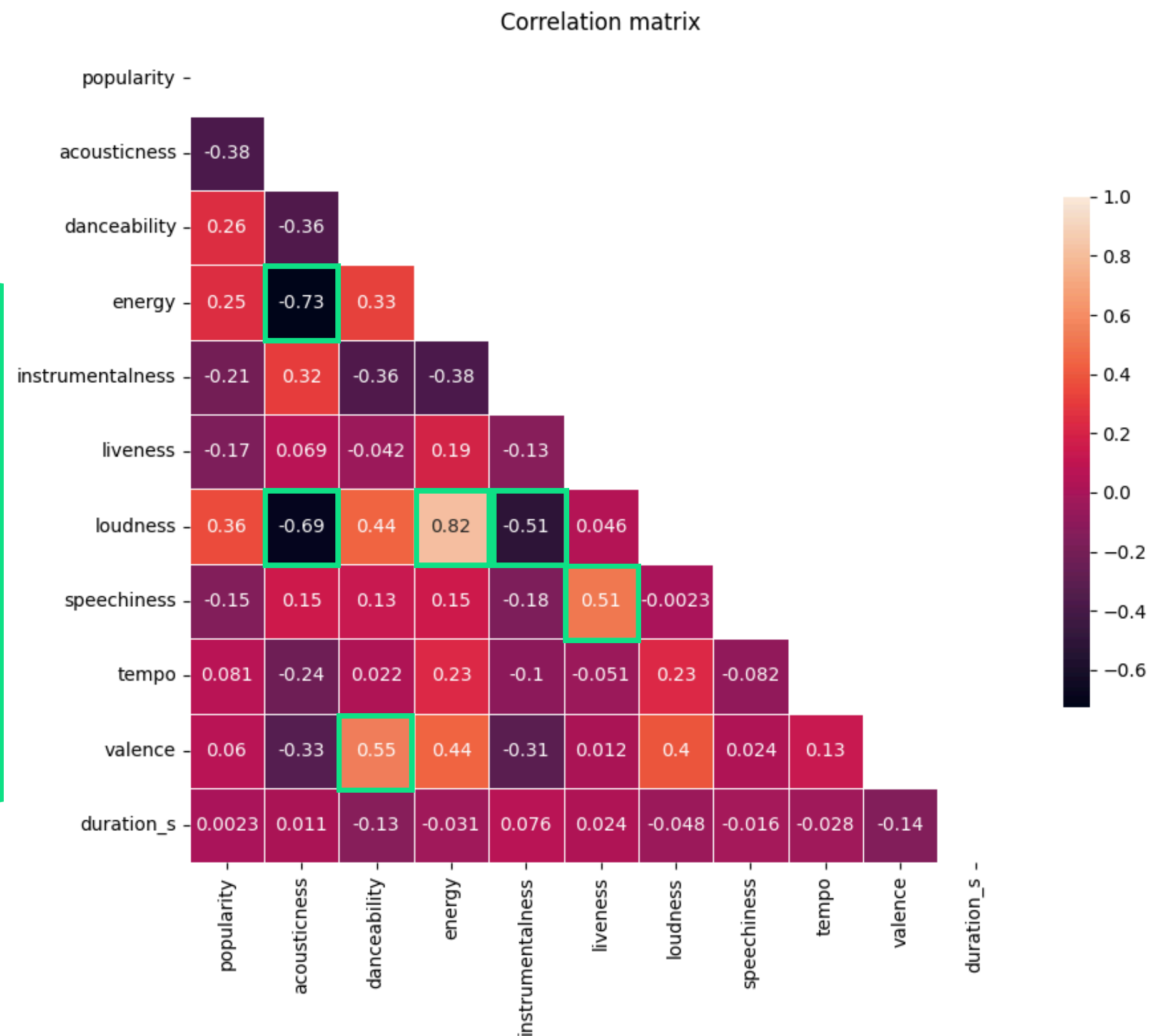
## Liveness

The likelihood that the song was recorded live. Higher values indicate a more 'live' environment (0 to 1).

## Duration\_ms

Song duration in milliseconds.

# Objective & Data Selection



## Objective

Anticipate a song's popularity on Spotify based on its musical features

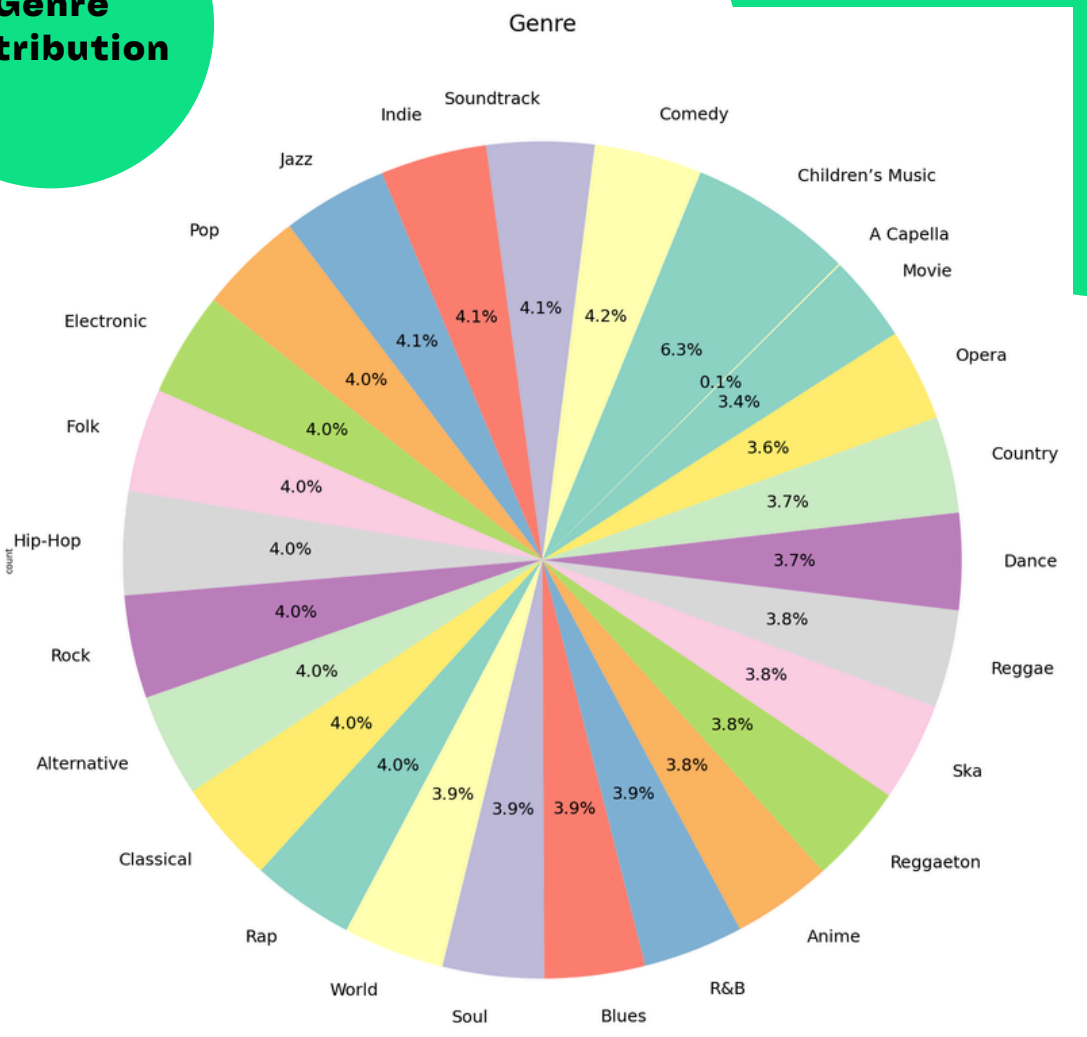
## Target and Features

- **Target:** 'Popularity'
- **Feature:** the other variables

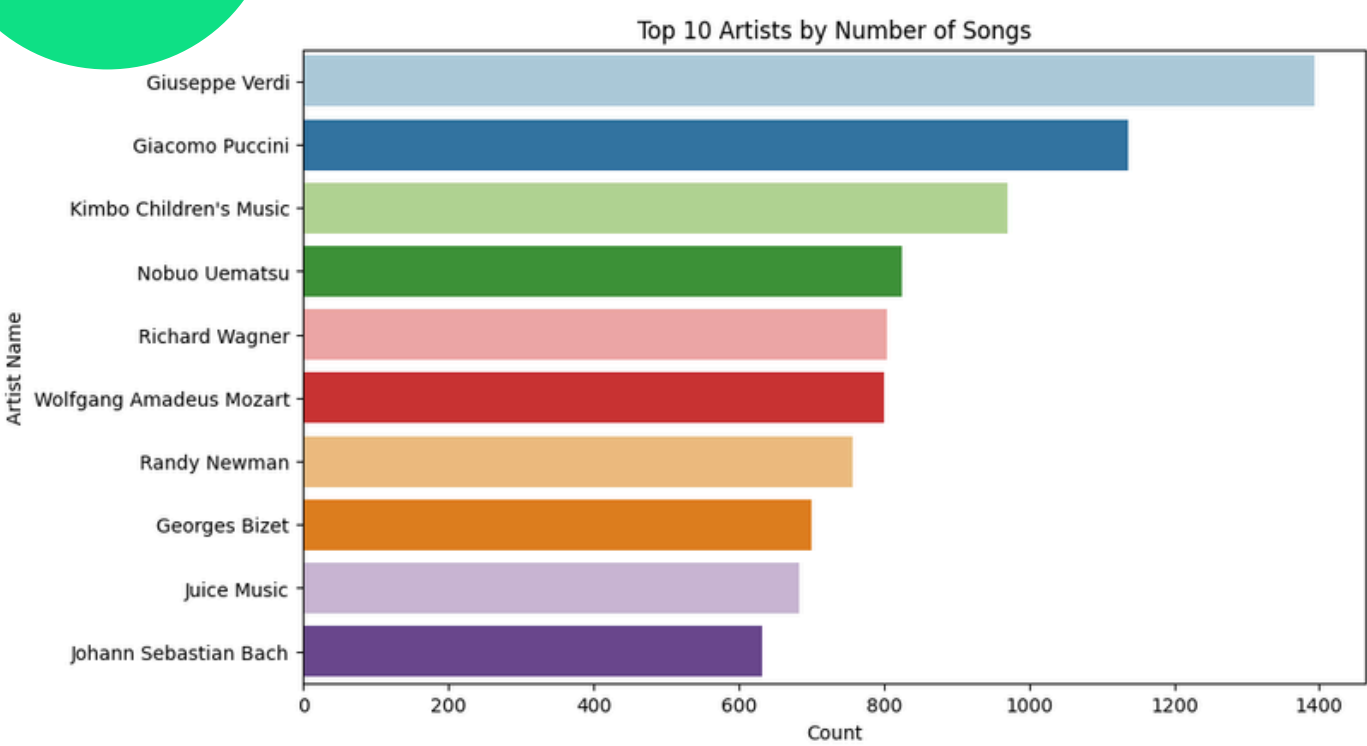
# Data Analysis

"⚠ In our data analysis, we detected some duplicate songs that did not appear as duplicates due to different genres assigned to them. Although the songs were the same, the genre distinction caused them to be treated as separate entries in the dataset."

## Genre distribution



## Top artists by songs number



## Top 30 popular genre & popularity

genre	mean_popularity	top_artist
Pop	66.590667	Drake
Rap	60.533795	Drake
Rock	59.619392	The Beatles
Hip-Hop	58.423131	Eminem
Dance	57.275256	Chris Brown
Indie	54.701561	G-Eazy
R&B	52.308719	Chris Brown
Alternative	50.213430	Five Finger Death Punch
Folk	49.940209	Bob Dylan
Soul	47.027836	John Legend
Country	46.100416	George Strait
Jazz	40.824383	Miles Davis
Electronic	38.056095	Moby
Reggaeton	37.742915	Daddy Yankee
Children's Music	36.202426	Kimbo Children's Music
Reggae	35.589328	Bob Marley & The Wailers
World	35.523145	Hillsong Worship
Blues	34.742879	Phish
Soundtrack	33.954800	Hans Zimmer
Classical	29.282195	Wolfgang Amadeus Mozart
Ska	28.612351	NOFX
Anime	24.258729	Nobuo Uematsu
Comedy	21.342630	George Carlin
Opera	13.335628	Giuseppe Verdi
Movie	12.174097	Randy Newman
A Capella	9.302521	The Singers Unlimited

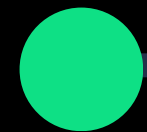
## Top 30 popular artists

	artist_name	mean_popularity	genre
	Pedro Capó	87.000000	Pop
1	Mario Bautista	85.000000	Pop
2	Mau y Ricky	83.000000	Pop
3	Paloma Mami	82.000000	Pop
4	Ninho	82.000000	Hip-Hop
5	Kris Kross Amsterdam	82.000000	Pop
6	Martin Garrix	81.857143	Pop
7	Sofia Reyes	81.500000	Dance
8	NSG	81.000000	Hip-Hop
9	Anitta	81.000000	Pop
10	Kenny Man	81.000000	Pop
11	Heuss L'enfoiré	81.000000	Hip-Hop
12	Billie Eilish	80.500000	Pop
13	Piso 21	80.500000	Pop
14	Grupo Arranke	80.000000	Pop
15	Coolio	80.000000	Hip-Hop
16	juan karlos	80.000000	Indie
17	Lele Pons	80.000000	Pop
18	JENNIE	80.000000	Pop
19	4 Non Blondes	80.000000	Pop
20	Rombai	80.000000	Pop
21	Ramz	80.000000	Hip-Hop
22	XO Cupid	79.500000	Dance
23	Drax Project	79.000000	Pop
24	Katrina & The Waves	79.000000	Pop
25	Joel Adams	79.000000	Pop
26	Silk City	78.500000	Pop
27	Tommy Boysen	78.000000	Reggaeton
28	Dennis Lloyd	78.000000	Pop
29	Daniel Powter	78.000000	Pop

# Methodology for creating a ML Model

## CHOOSING VARIABLES TO CREATE THE MODEL

STEP 1

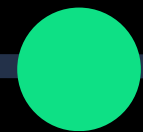


### Removing columns

Dropping NON-ESSENCIAL columns  
(['genre', 'artist\_name',  
'track\_name', 'track\_id'])  
and do get.dummies()  
\*\*For regression problem  
we also remove the less  
correlated variables

## CREATING A CLASSIFICATION FOR POPULARITY

STEP 2



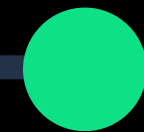
### Discretization of popularity column

We transformed this column  
dividing the numerical values  
into 5 categories:

- 'Very Low' (0-25)
- 'Low' (25-50)
- 'Medium' (50-75)
- 'High' (75-90)
- 'Top' (90-100)

## PERFORMING TRAIN TEST SPLIT

STEP 3

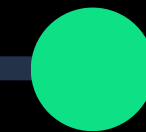


### Splitting data for training the model

We split the dataset for  
80% of the data for  
training the model and 20%  
of the data for testing the  
model using as target  
column the new popularity  
column

## NORMALIZATION OF DATA

STEP 4

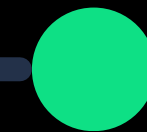


### Using MinMaxScaler()

A technique used to  
scale the features of  
the data so that all  
values are within a  
specific range, typically  
between 0 and 1.

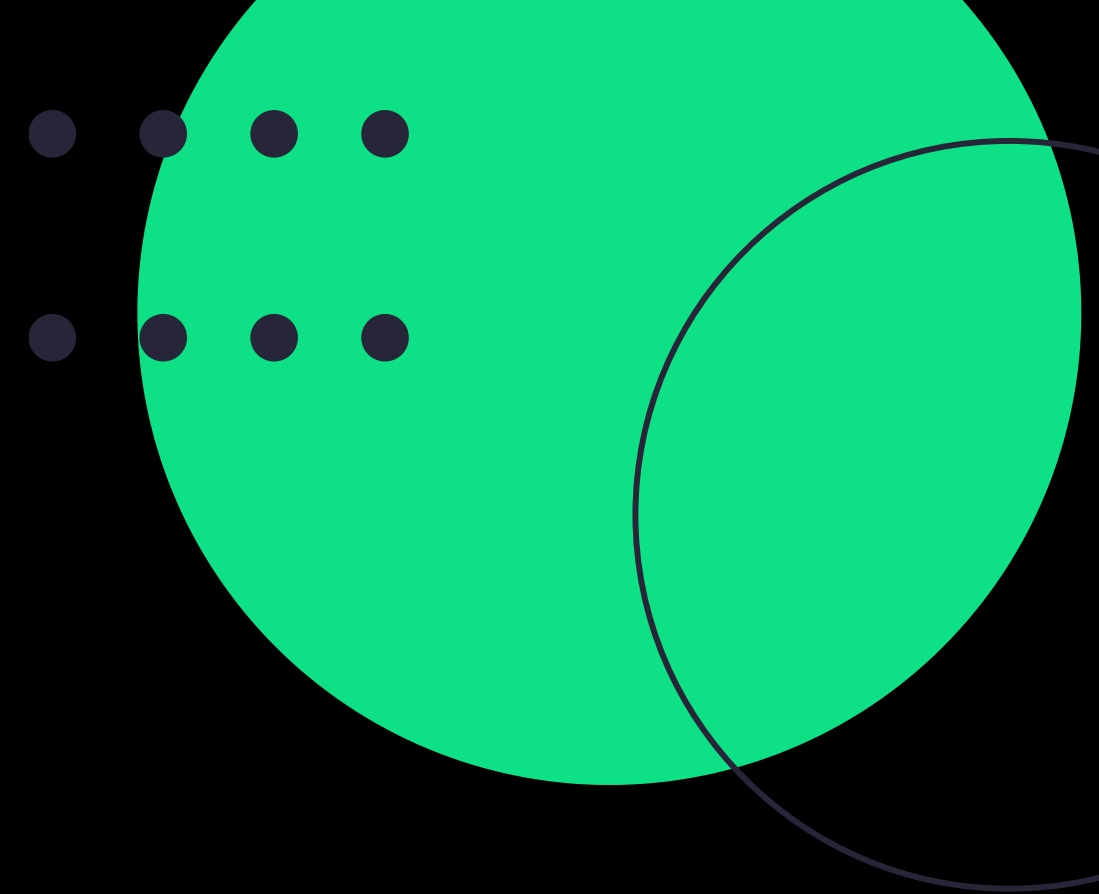
## PCA ANALYSIS

STEP 5



### Choosing best components

Dimensionality reduction  
technique transforms a  
set of possibly correlated  
variables into a smaller  
set of uncorrelated  
variables called principal  
components





# Changes made to improve our ML performance

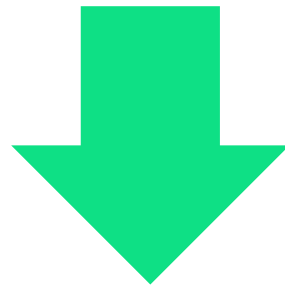
## Do a label encoding for categorical variables

- The unique values from the `time_signature` column are extracted, and a number is assigned to each category using a mapping dictionary.
- In the `mode` column, 'Major' is replaced with 1 and 'Minor' with 0.
- The unique values from the `key` column are extracted and mapped to unique numbers using a mapping dictionary.

# Balance strategy

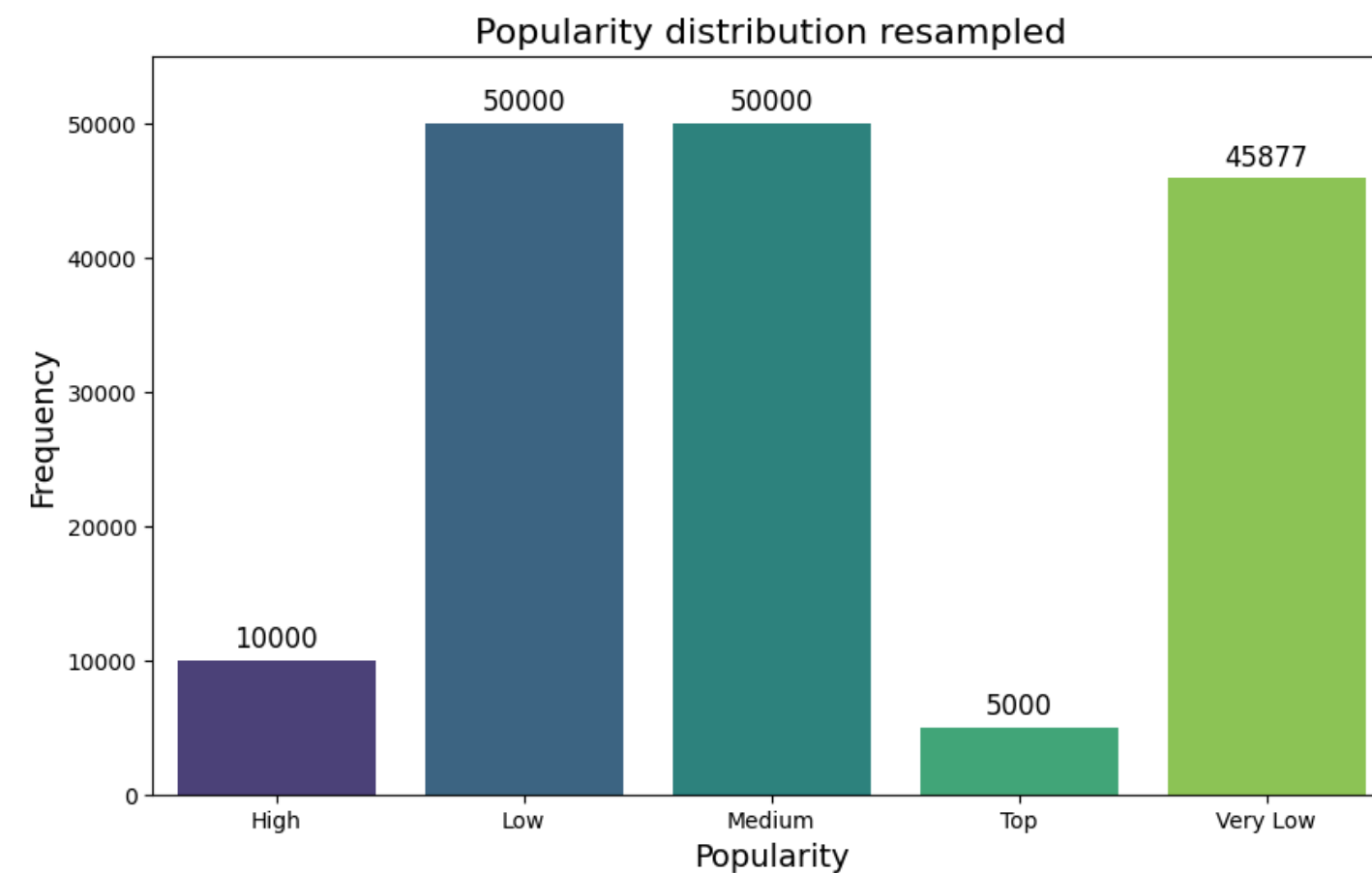
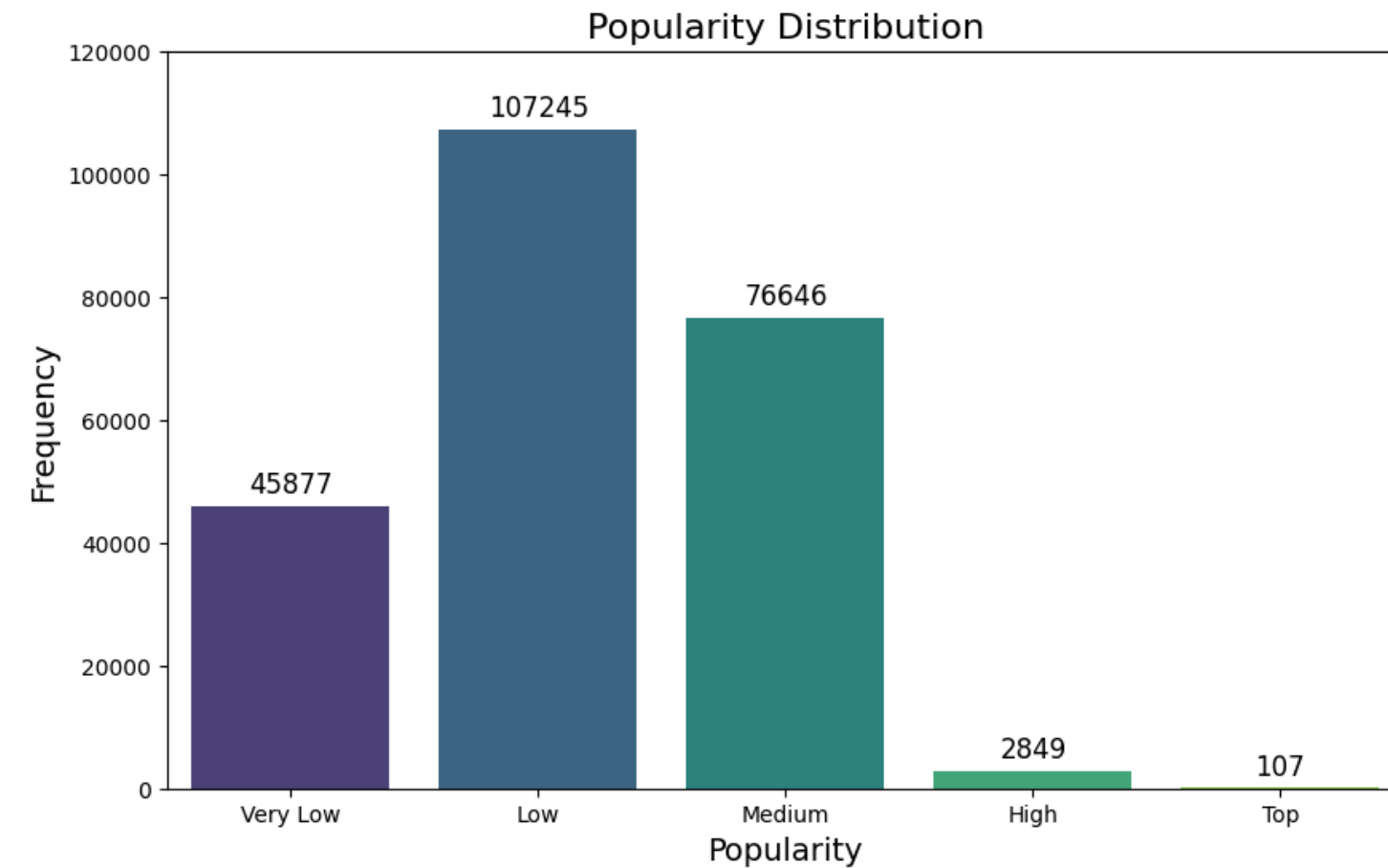
## Popularity

Integer variable in range 0 to 100



Classification variable:

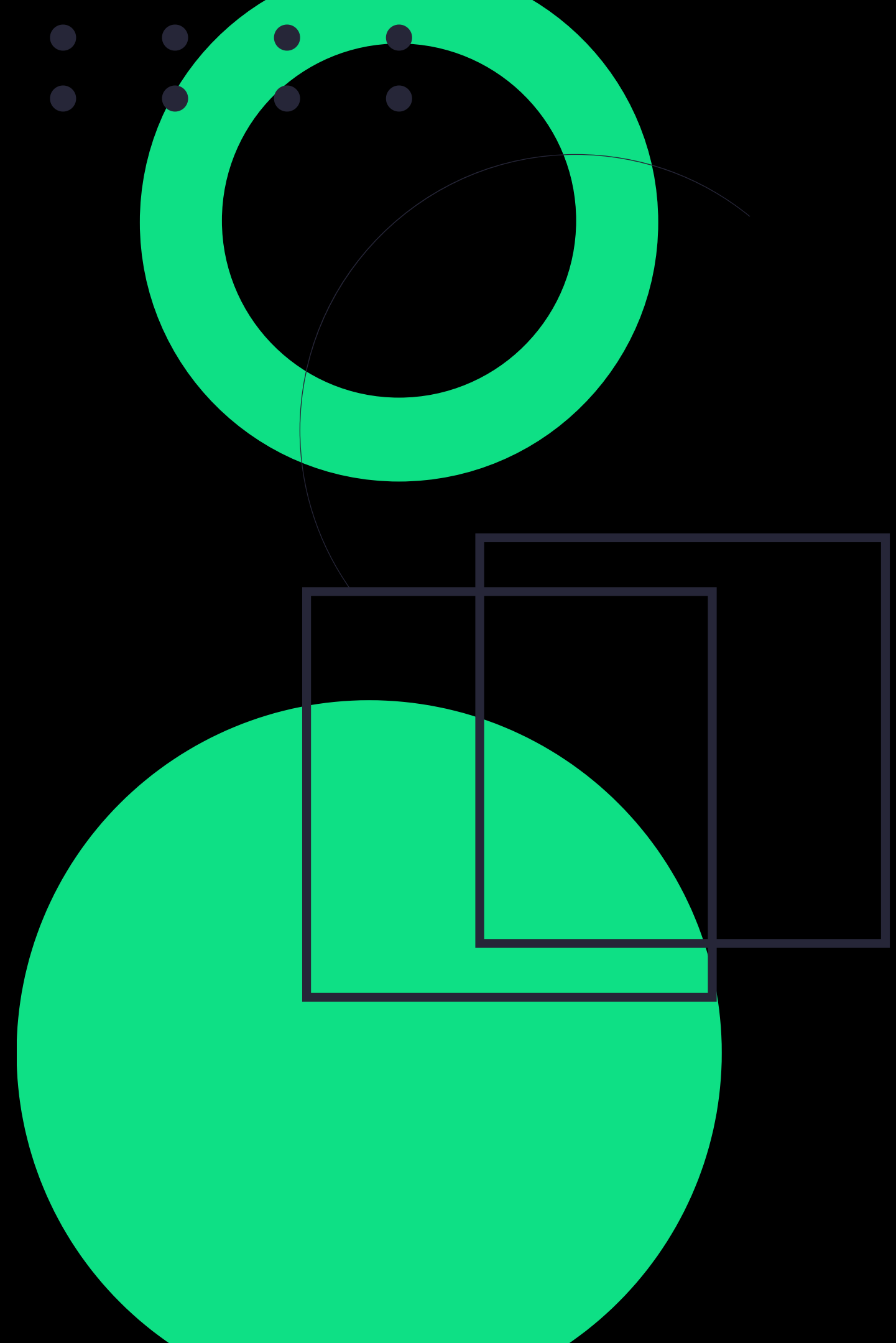
- **Very Low:** 0 to 25
- **Low:** 25 to 50
- **Medium:** 50 to 75
- **High:** 75 to 90
- **Top:** 90 to 100





# Approach to the ML project

- 01** A ML project of a classification problem (5 classes)
- 02** A ML project of a regression problem
- 03** New idea: ML project of a classification problem with ONLY 2 classes



# Feature Engineering and Selection

Model for a Classification problem

## K-NN

Classifies data points based on the majority class of their k nearest neighbors (3) in the feature space.

F1 SCORE: 0.5879

## Logistic Regression

Estimates the probability that a given input belongs to a particular class using a logistic (sigmoid) function.

F1 SCORE: 0.530

## Random Forest

Creates a collection (a "forest") of decision trees during training and combines their outputs to make predictions.

F1 SCORE: 0.7023

## XGBoost

Technique for building an ensemble of decision trees sequentially to improve model accuracy.

F1 SCORE: 0.53

# Feature Engineering and Selection

Model for Regression problem

## K-NN

Classifies data points based on the majority class of their k nearest neighbors (80) in the feature space.

SCORE: 0.29

## Bagging model

Combine predictions from multiple models (usually decision trees) trained on different subsets of the training data to improve accuracy and reduce overfitting.

SCORE: 0.298

## Random Forest

Creates a collection (a "forest") of decision trees during training and combines their outputs to make predictions.

SCORE: 0.446

## Gradient Boosting

It belongs to the family of boosting algorithms, which build an ensemble of weak learners (typically decision trees) sequentially

SCORE: 0.515

## Ada Boost

Iteratively train a series of models, each focusing on the errors made by previous models, and to give more weight to the harder-to-predict examples.

SCORE: 0.5180

## XGBoost

Technique for building an ensemble of decision trees sequentially to improve model accuracy.

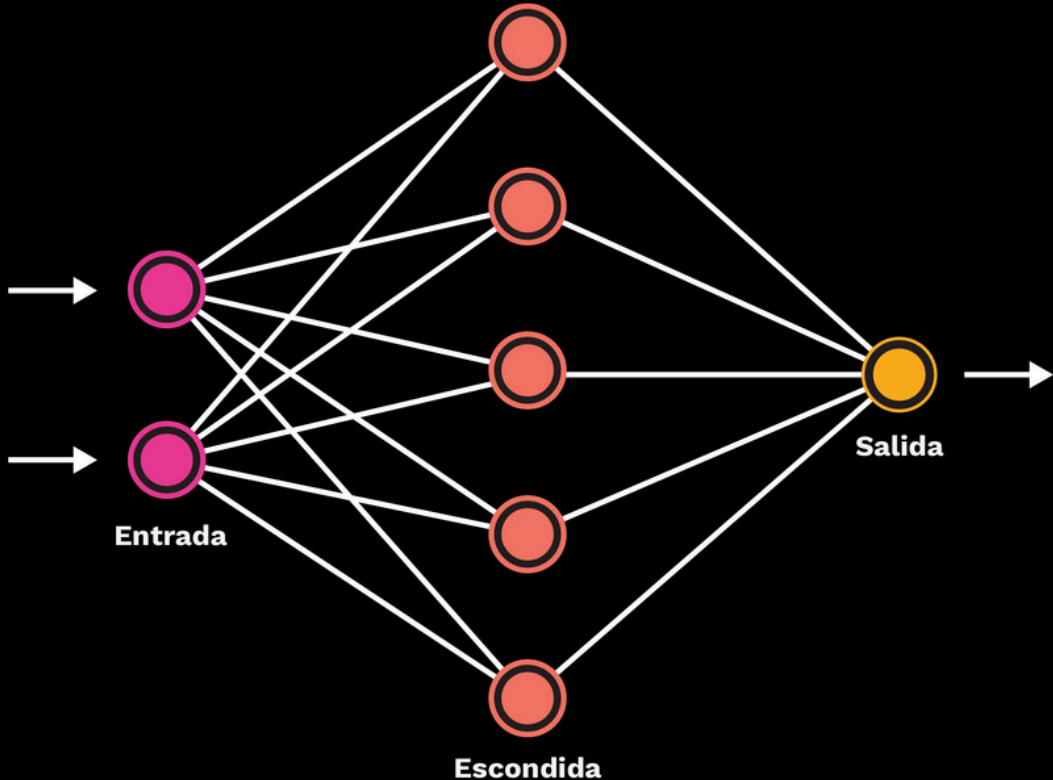
SCORE: 0.399

# Feature Engineering and Selection

Model for Regression and Classification problem

## MPL Regressor

Type of artificial neural network (ANN) used for regression tasks. It's part of the scikit-learn library and implements a feedforward neural network with one or more hidden layers. The network is trained to learn the relationship between the input features (independent variables) and the continuous target variable (dependent variable) in a regression problem.



## NN Structure

```
# model
model = Sequential([
    Dense(256, activation = 'relu',input_shape=(X_train_norm.shape[1],)),
    Dropout(0.3),
    Dense(128, activation = 'relu'),
    Dropout(0.3),
    Dense(64, activation = 'relu'),
    Dense(len(np.unique(y_train)), activation='softmax')
])

# Callbacks para EarlyStopping y reducción dinámica de la tasa de aprendizaje
early_stopping = EarlyStopping(monitor='val_loss', patience=5, verbose=1, restore_best_weights=True)
lr_reduction = ReduceLROnPlateau(monitor='val_loss', patience=3, verbose=1, factor=0.5, min_lr=0.0001)

# compile model
model.compile(optimizer=Adam(learning_rate=0.0005),
              loss=focal_loss(),
              metrics=['accuracy'])

# train the model
history = model.fit(
    X_train_norm, y_train_encoded,
    validation_data=(X_test_norm, y_test_encoded),
    epochs=100,
    batch_size=32,
    callbacks=[early_stopping, lr_reduction],
    verbose=1
)
```

Regression

Loss : 0.70

Classification

Accuracy: 0.557

Loss : 0.02

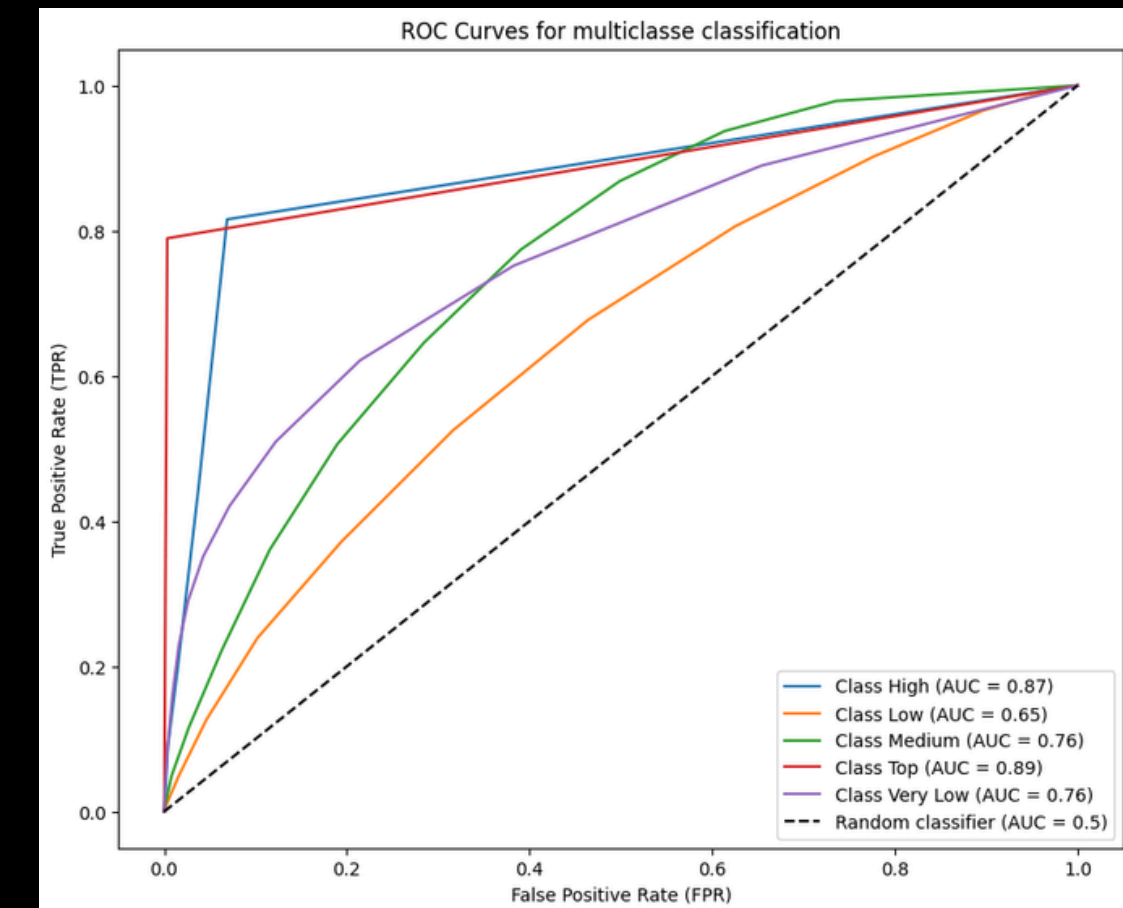
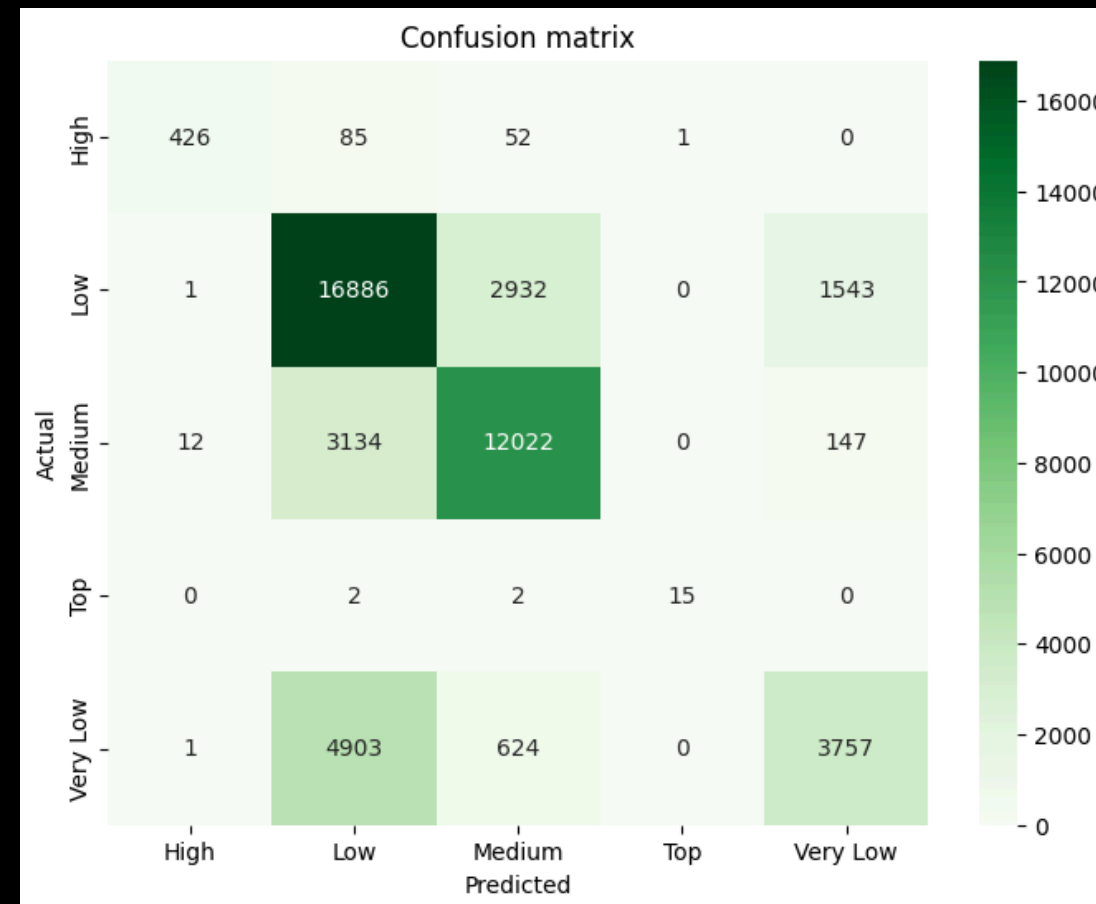
# Model selection & evaluation

## Random Forest

Accuracy: 0.7113  
Precision: 0.7126  
Recall: 0.7113  
F1 Score: 0.7023

### Classification Report:

	precision	recall	f1-score	support
High	0.97	0.76	0.85	564
Low	0.68	0.79	0.73	21362
Medium	0.77	0.78	0.78	15315
Top	0.94	0.79	0.86	19
Very Low	0.69	0.40	0.51	9285
accuracy			0.71	46545
macro avg	0.81	0.70	0.74	46545
weighted avg	0.71	0.71	0.70	46545



### High & Top Categories:

- High precision (0.97 for "High", 0.94 for "Top") → The model correctly predicts these classes when it identifies them.
- Lower recall (0.76 for "High", 0.79 for "Top") → The model fails to capture all cases in these categories.

### Low & Medium Categories:

- Balanced recall (~0.78-0.79) → Model performs consistently here.

### Very Low Category:

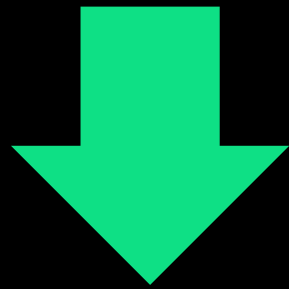
- Recall is only 0.40, meaning the model struggles to identify songs in this category.
- Could be due to class imbalance or overlapping feature distributions.

# Hyperparameter Tuning and Model Optimization

## New idea: Divide differently “Popularity”

Popularity

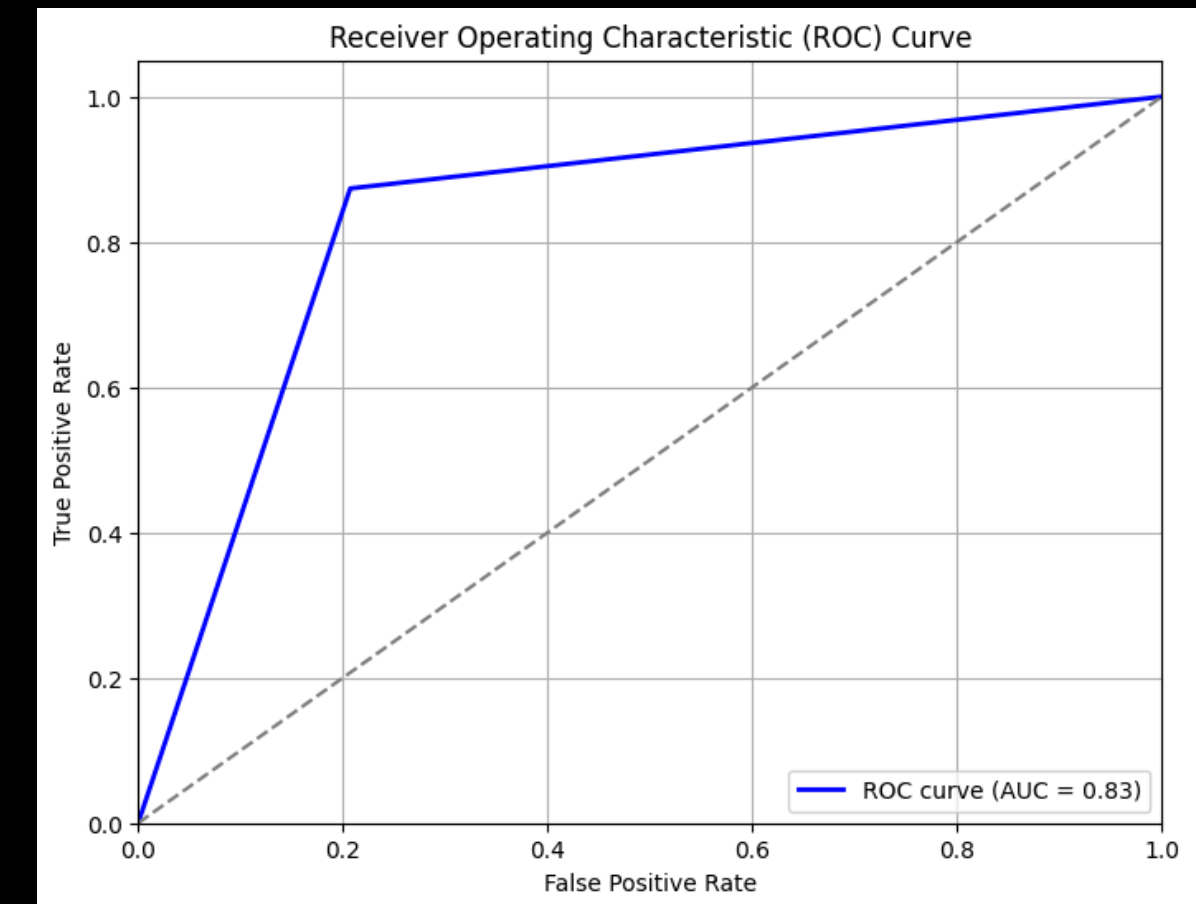
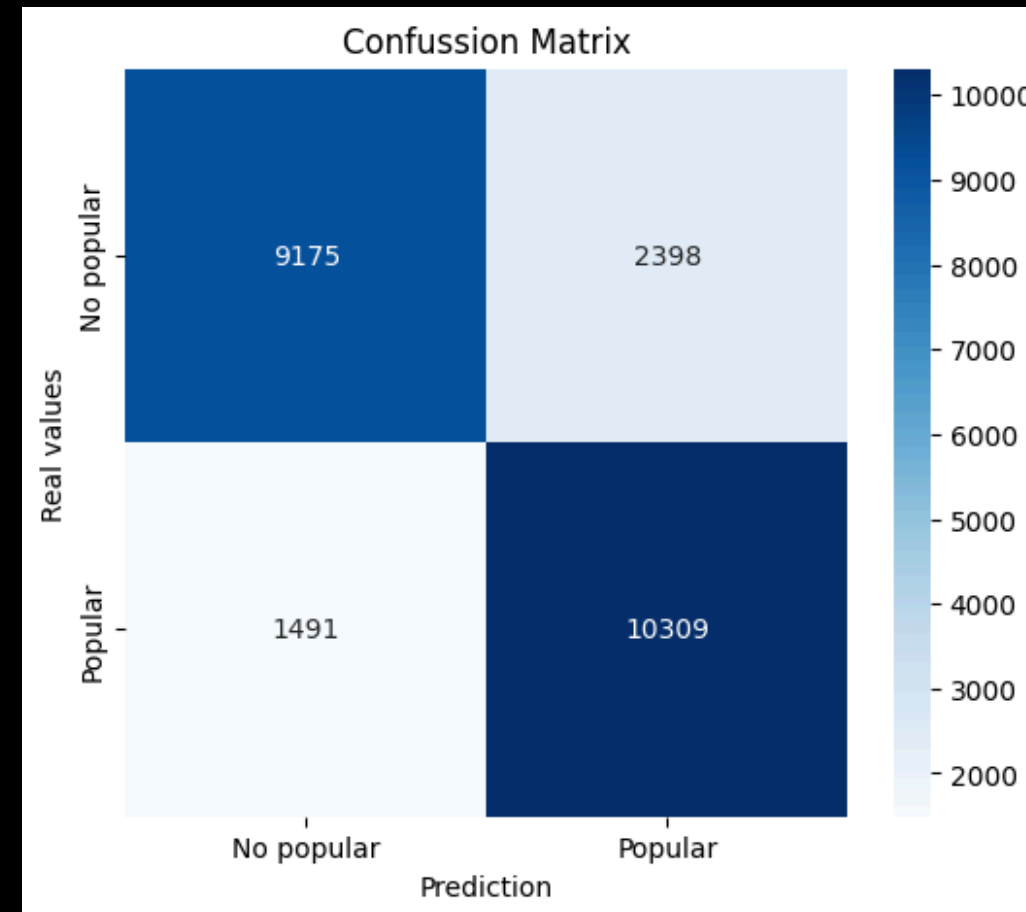
Integer variable in range 0 to 100



Classification variable:

- No popular  $\leq 55$
- Popular  $> 55$

Apply undersampling to ensure the classification and distribution of both categories (“Popular and not popular”)



Accuracy: 0.8336  
Precision: 0.8355  
Recall: 0.8336  
F1 Score: 0.8333





## Danceability

Lower danceability is generally associated with reduced popularity. This suggests that songs that are more rhythmically engaging and suitable for dancing tend to be more popular among listeners.

## Acousticness

A decrease in acousticness tends to correlate with an increase in popularity.

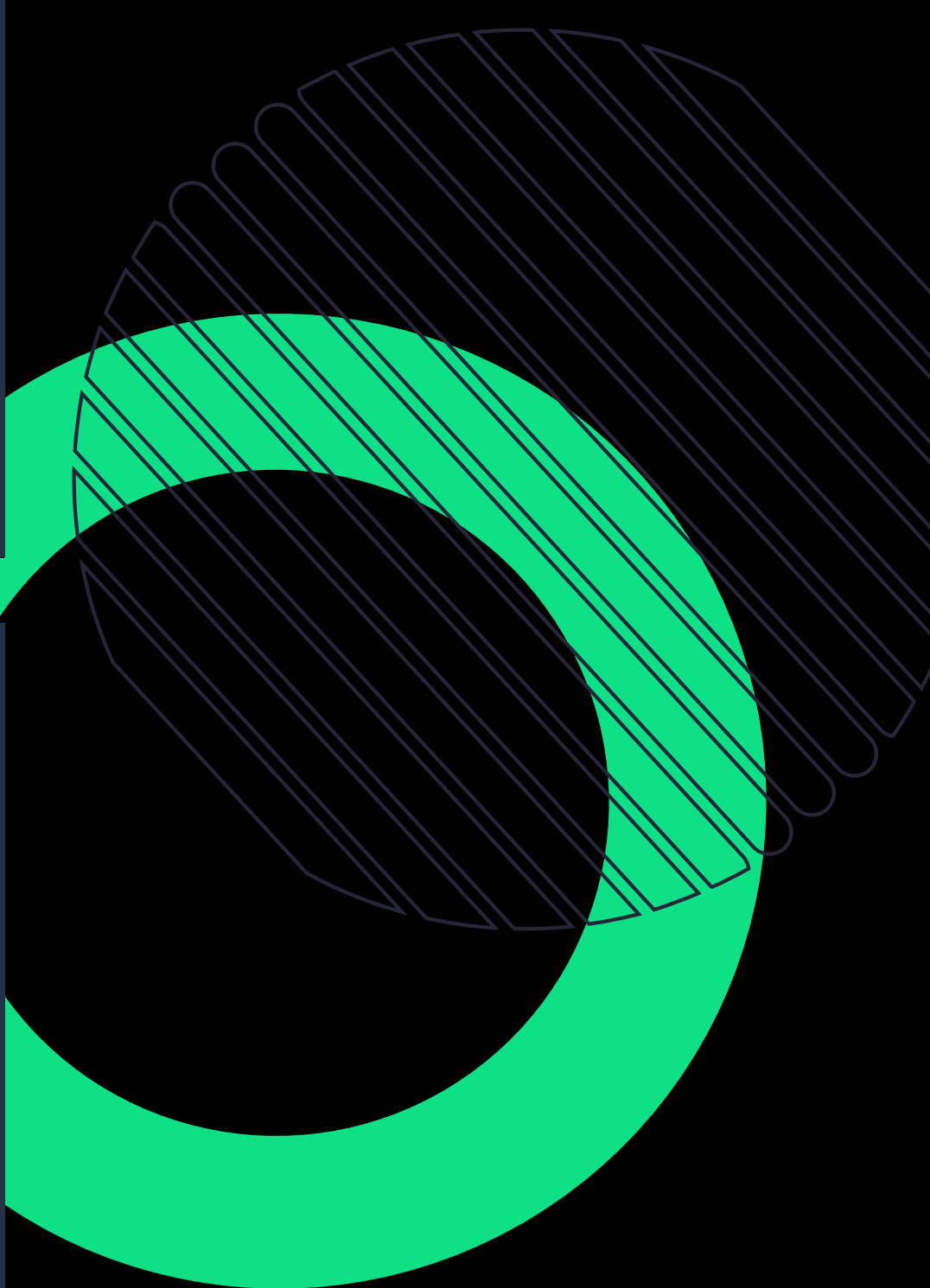
# INSIGHTS & IMPACT

## Energy

This suggests that songs with higher energy levels are somewhat more likely to be popular. However, the relationship is not as strong, other factors also play significant roles in determining its popularity.

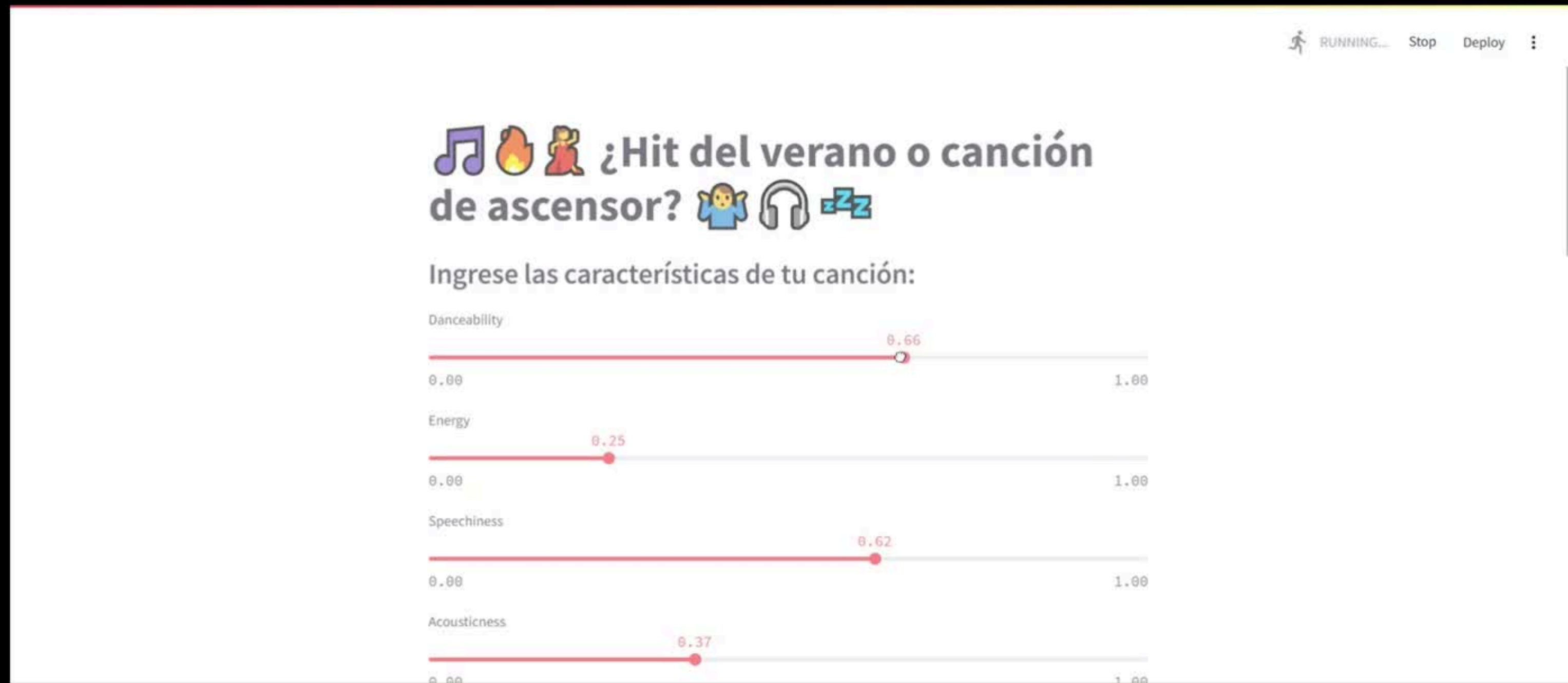
## Speechiness

This indicates that songs with less words tend to be more popular. It suggests that tracks with a large number of words capture less attention and engagement from listeners.





# Streamlit Popularity Prediction

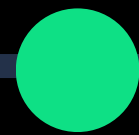


# Future work & improvements



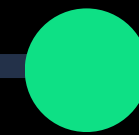
## Model Optimization

Fine-tune hyperparameters using advanced techniques  
Test additional models for comparison.  
Upgrade hardware (e.g., faster processors, more memory, GPUs) to run more complex models and accelerate training time.



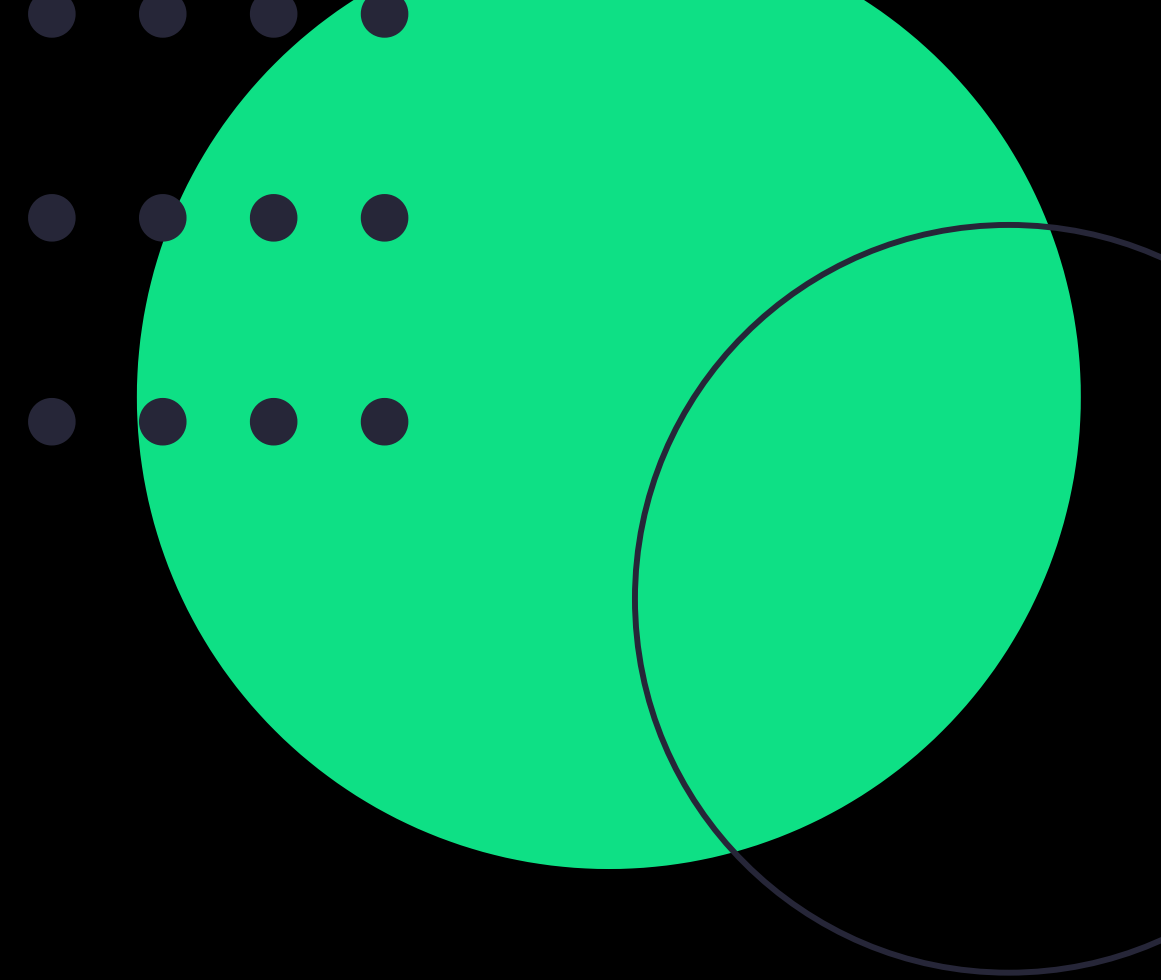
## Data Expansion

Incorporate more variables (e.g., playlist data) to enhance the model.  
  
Increase dataset size for better generalization.



## Distribution of the Popularity Variable

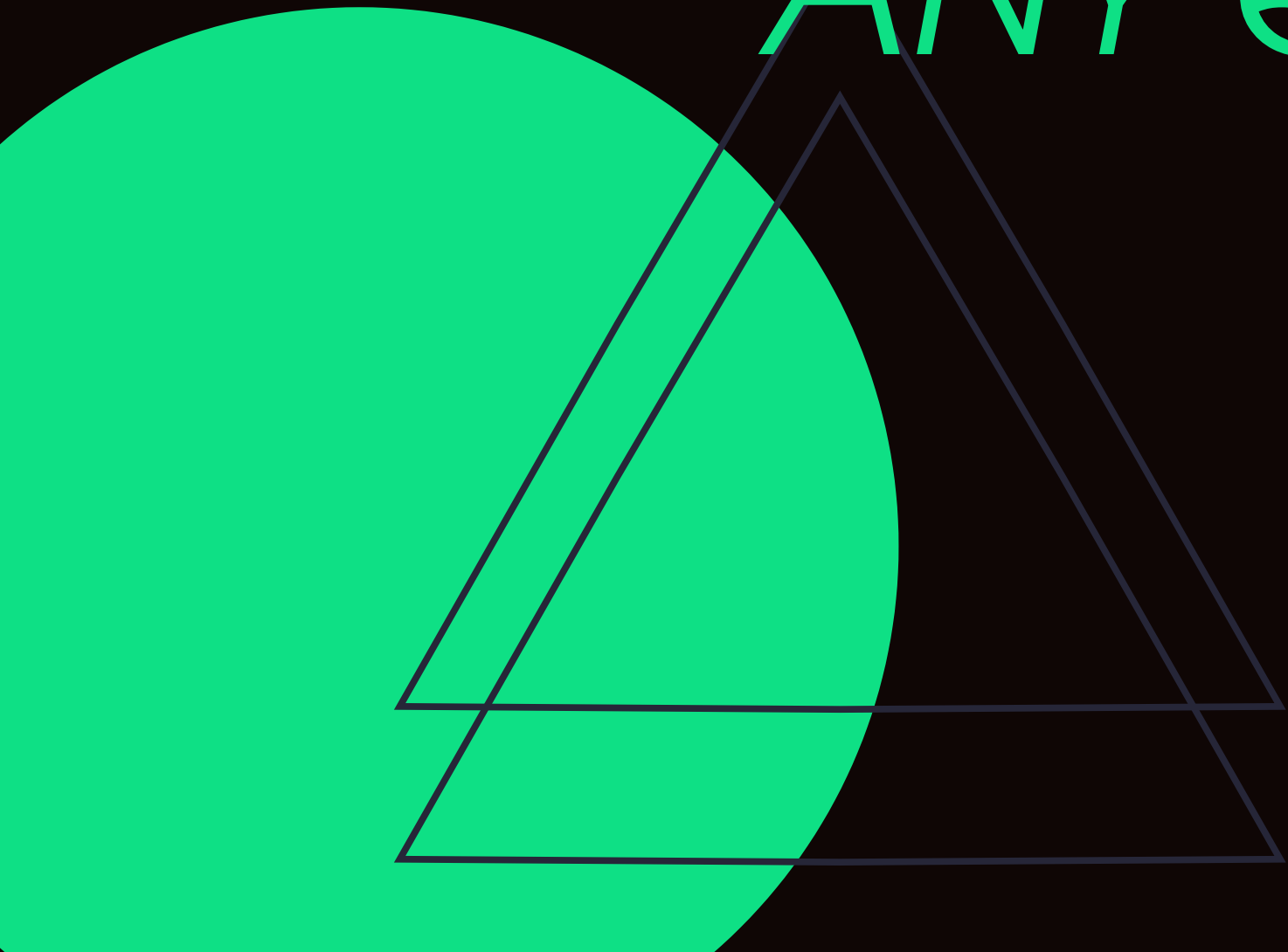
Test the trained models with a different split of the Popularity variable



# THANKS



## *ANY QUESTION?*



Celia Manzano | Laura Sánchez | Carlota Gordillo