Summary Report

Model to predict song popularity

A classification model based on spotify data to predict the popularity of a song

Proposed by:

Alejandro Gutierrez Werner

Apilash Balasingham

Fhadah Alromy

Lucia Pellicer Cascales

Luis Soto Guareschi

Ignacio Sahonero Vadillo

Professor:

Alvaro Jose Mendez Lopez

MBDS: Master in Business Analytics and Data Science

Machine Learning II

Madrid, Spain

www.ie.edu

November, 2025

Table of Contents

Executive Summary

This project set out to predict a track's popularity before a track is even released. We worked with a dataset of 2,200 songs that included details ranging from how danceable or energetic a song is, the song's overall mood, to how well-known is the artist. The results of our exploratory data analysis confirmed the dataset was sound in structure, appropriate for further analysis, and ready to move forward with model development.

We tested several machine-learning models including Naive Bayes, Decision Trees, KNN, and different boosting methods and found that XGBoost gave us the most reliable results. After fine-tuning the model, it reached 78.5% accuracy with a 0.75 ROC-AUC, which shows it can confidently distinguish between songs that are likely to perform well and those that aren't.

From a business perspective, these insights can help record labels, producers, and streaming platforms spot promising songs early. This allows them to focus their marketing efforts, plan releases more strategically, and align content with what listeners are already leaning toward.

Data Audit, Cleaning and Manipulation.

The dataset contains 2,200 records and 19 columns with detailed information about tracks, artists, and audio features such as danceability, energy, loudness, and tempo.

The data audit showed that the dataset is well-structured, with no duplicate entries and only a few missing values across numerical columns, which were dropped to ensure data consistency.

During preprocessing, the artist_genres column was transformed into multiple binary columns, each representing a specific genre, through one-hot encoding. However we only used the most popular genres to avoid having hundreds of columns. Afterward, the original column was removed to avoid redundancy. Additionally, a target variable named popular was created to classify songs based on their popularity: tracks with a popularity score equal to or above the 75% quantile were labeled as True (popular), and the rest as False. The distribution of genres was also analyzed, revealing that 167 out of 2,199 songs had no assigned genres, and a bar chart was produced to visualize the most frequent ones.

Overall, the dataset is clean, consistent, and fully prepared for subsequent exploratory analysis and predictive modeling.

Testing models

To determine which algorithm offered the best foundation for further optimization, we tested multiple models, including Decision Tree, Random Forest, Boosting, Naïve Bayes, and K-Nearest Neighbors (KNN). Model performance was primarily evaluated using Accuracy as the main metric, with F1-score providing additional insight into the balance between precision and recall. Among these, Boosting consistently delivered the highest performance and that is why we chose it to face the project.

Boosting Model

To predict how popular a song might become, we decided to use the **XGBoost** Classifier. We chose it because it works really well with structured data, handles complex non-linear patterns, and has built-in ways to avoid overfitting all of which make it a strong fit for this kind of problem.

Once the data was cleaned and we created the binary "popular" label, we split the dataset into training and testing sets using a 75/25 split. We used stratified sampling so both sets kept the same balance of popular and non-popular songs.

Model Performance

General Model Fitness

The classification model selected for this project was the **XGBoost Classifier**, chosen for its efficiency, ability to handle non-linear relationships, and built-in regularization features that reduce overfitting. After parameter tuning using **GridSearchCV**, the optimized model achieved a **test accuracy of 78.6%** and a **ROC-AUC score of 0.74**, demonstrating strong predictive performance. The optimal parameters included 150 estimators, a learning rate of 0.05, and a maximum tree depth of 3, balancing complexity and generalization.

Model Evaluation

The final model's effectiveness was assessed through a confusion matrix and ROC curve. The ROC curve indicated good discrimination between popular and non-popular songs, with an optimal decision threshold of 0.31. However, the confusion matrix revealed that the model prioritizes sensitivity over precision—it correctly identifies most songs that become popular (true positives) but also predicts some non-popular tracks as popular. This behavior suggests the model is tuned to minimize false negatives rather than false positives, making it particularly useful for exploratory or discovery-focused business applications where missing a potential hit carries higher cost than overestimating one.

Train vs Test Evaluation

The base model initially achieved a train accuracy of 99.9% and a test accuracy of 75.6%, suggesting overfitting. After hyperparameter optimization, the train accuracy slightly decreased to 85.5%, while test accuracy improved to 78.6%, reflecting a 3.9% performance gain and better generalization to unseen data. This improvement indicates that the model maintains strong predictive power without relying too heavily on training data patterns, making it robust for real-world applications.

Conclusions & Business Recommendations

Our optimized XGBoost model achieved a test accuracy of 78.6% and a ROC-AUC of 0.74, indicating that machine learning techniques can meaningfully anticipate the popularity of songs before release. This predictive capacity confirms that certain musical and artist-level features—such as energy, danceability, and artist popularity—carry strong explanatory power for market performance.

Beyond technical performance, the findings suggest that data-driven modeling can enhance strategic decision-making within the music industry. Record labels could integrate such models into early A&R screening—the process by which labels evaluate and select artists or songs for investment—to identify high-potential tracks, while streaming platforms could leverage them to improve recommendation systems and playlist optimization.

The model's current thresholding approach emphasizes recall over precision, meaning it is particularly sensitive to detecting songs that could become popular, even if some non-popular tracks are mistakenly classified as potential hits. This makes the model valuable for exploratory and discovery-oriented business applications, where identifying potential successes is more critical than avoiding false positives. Future research could improve precision and generalization by incorporating temporal trends, lyrical sentiment, and social-media signals to capture emerging popularity dynamics more accurately.

Overall, this work demonstrates how predictive analytics can bridge artistic creativity and business intelligence, transforming subjective industry intuition into quantifiable evidence for more efficient and data-informed decision-making.

Business Applications and Market Opportunities

The developed model and accompanying deployment code offer a practical decision-support tool for the music industry. By inputting a song's core attributes—such as energy, danceability, loudness, and artist popularity—the system generates a direct prediction of whether a track is likely to be "popular" or "not popular."

Record labels can integrate this predictive tool into their A&R (Artists and Repertoire) evaluation process to screen new tracks prior to release, supporting data-driven prioritization of marketing budgets and talent investments. Producers can leverage the model's insights to align their creative decisions with characteristics associated with past commercial success, while streaming platforms can embed the tool into recommendation systems to identify high-potential tracks and refine playlist curation.

The ready-to-deploy code transforms machine learning research into an actionable industry asset, enabling more efficient promotion strategies, better allocation of creative and financial resources, and stronger alignment between artistic output and audience preferences. However, because the model optimizes for higher sensitivity, while aiming for a low false negative rate—capturing most songs that may become popular—it can also yield some false positives. For this reason, predictions should be complemented with expert industry judgment to validate artistic and market potential, ensuring that algorithmic insights enhance rather than replace human expertise.

Considerations and Limitations

One of the principal drawbacks of using a boosting model is its interpretability, even though the model is very useful and accurate, it doesn't mean that we can understand how each feature affects the results individually. In addition the model's performance and applicability are constrained by several factors. First, it depends on Spotify-specific audio metrics such as danceability and energy, which are not universally available or consistently defined across platforms. Additionally, since the dataset captures a static moment in time, shifting musical trends and cultural preferences could reduce predictive accuracy, requiring ongoing retraining. The model demonstrates strong recall but moderate precision—it successfully identifies most songs that become popular but also classifies some non-popular tracks as popular. While this favors discovery and minimizes missed opportunities, it may lead to overestimating potential hits. Moreover, genre imbalance and reliance on Spotify's internal feature extraction may introduce bias toward mainstream patterns. Ethically, predictive tools should complement rather than replace human artistic judgment, ensuring that innovation and creative diversity are not constrained by algorithmic recommendations.

Technical Annex

Technical Annex

This document provides detailed technical documentation of the data preprocessing and model development process, including key results, tables, and visualizations extracted from the analysis notebooks.

Data Preparation

Data Loading and Initial Exploration

The dataset was loaded from spotify-songs.xlsx containing Spotify song data with the following initial structure:

- General features:
  - year, track_name, track_popularity, album, artist_name, artist_genres, artist_popularity
- Audio features:
  - danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration_ms

Missing Values Analysis

Missing Values Count: 1 row with some nulls in some columns
Action Taken: Since only one null value was present across all audio features, the row was dropped using df.dropna(), resulting in a clean dataset with no missing values.

Feature Engineering

Album Songs Count

A new feature album_songs was created to capture the number of songs per album (per artist). This feature was added to capture potential album-level effects on song popularity.

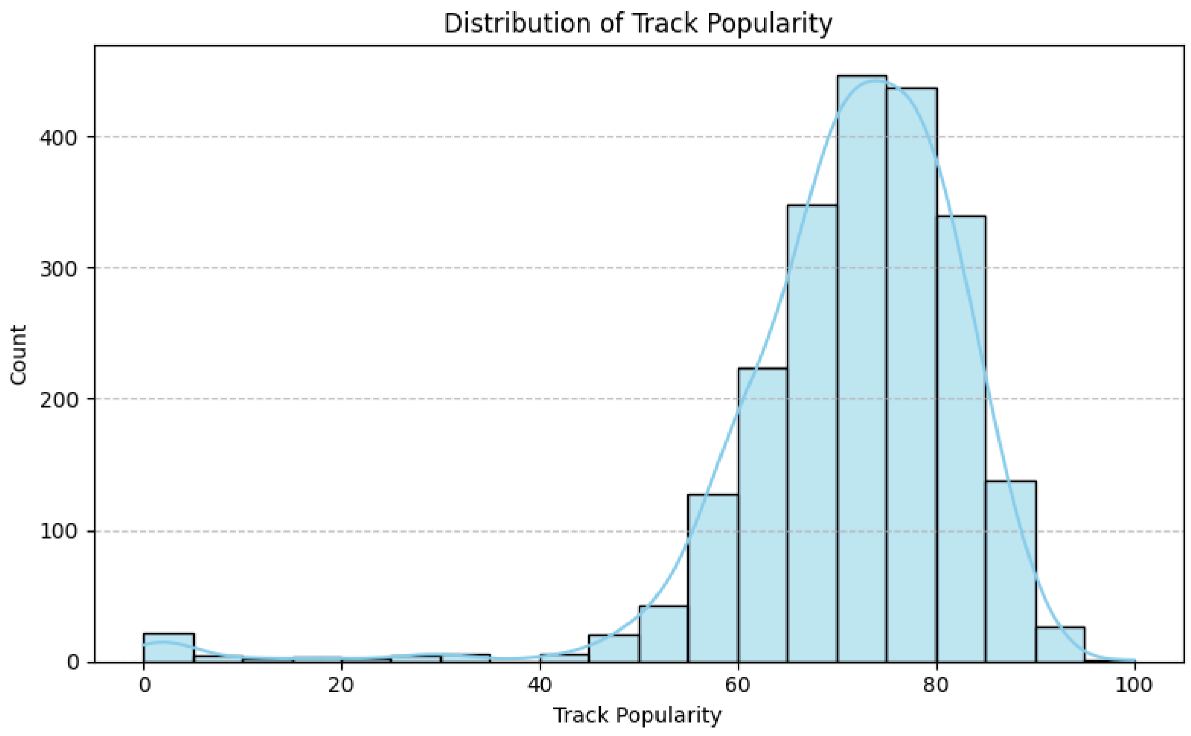
Column Removal

The following columns were dropped as they were not suitable for machine learning:

- year: Temporal information not used in this analysis
- track_name: Text identifier, not a predictive feature
- album: Text identifier, not a predictive feature
- artist_name: Text identifier, not a predictive feature

Target Variable Creation

Track Popularity Distribution
The track_popularity distribution was analyzed using a histogram, showing the distribution of popularity scores across the dataset.



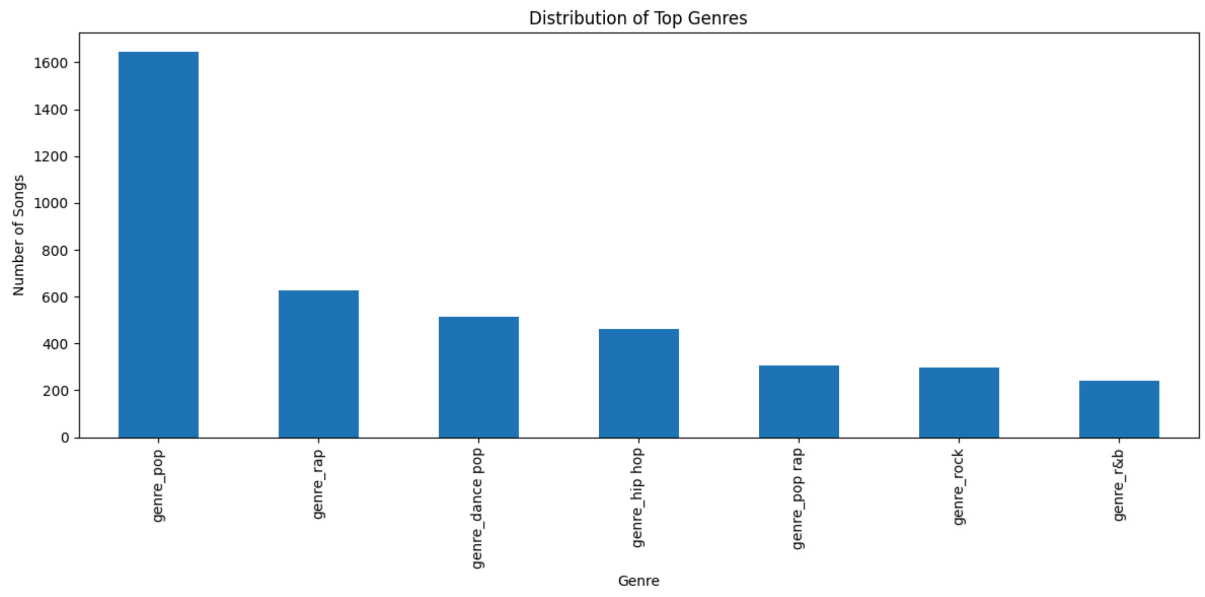
Target Variable Definition:
A binary target variable popular was created using the 75th percentile threshold:

This approach classifies songs in the top 25% of popularity as "popular" (True) and the remaining 75% as "not popular" (False).

Genre Encoding

Genre Analysis
The artist_genres column contained string representations of genre lists. The following process was applied:

Genre Extraction: All unique genres were extracted from the dataset
Genre Frequency Analysis: Genres were filtered to include only those appearing in at least 10% of songs (219.9 occurrences minimum)
Top Genres Selected (with occurrence counts):



Genre Coverage: 167 songs (7.6% of the dataset) had no genres assigned to any of the selected categories.

Final Dataset Structure

After preprocessing, the dataset contained 2199 rows and 22 columns (21 features, 1 target column)

Final Features:

- Artist_popularity
- Audio features: danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration_ms
- Album_songs
- Genre features: genre_pop, genre_rap, genre_dance pop, genre_hip hop, genre_pop rap, genre_rock, genre_r&b

Data Export

The preprocessed dataset was exported to spotify-songs-preprocessed.csv for use in model training.

Testing Model

Train-Test Split

The dataset was split into training and testing sets with the following configuration:

- Test size: 25%
- Random state: 100

- Stratify = y (to maintain class distribution across splits)

Base Model Performance

A baseline XGBoost classifier was trained with default parameters

Base Model Results

- Train Accuracy: 0.998787 (99.88%)
- Test Accuracy: 0.756364 (75.64%)

The large gap between train and test accuracy (99.88% vs 75.64%) indicates significant overfitting in the base model, suggesting the need for hyperparameter tuning and regularization.

Hyperparameter Tuning with GridSearchCV

A comprehensive grid search was performed to optimize the XGBoost model

Grid Search Configuration

Scoring Metric: ROC AUC
Cross-Validation: 5-fold CV
Model Persistence: Best model saved to grid_roc.pkl
Best Parameters: Retrieved from grid_roc.best_params_ (specific values depend on the grid search execution)
Best Cross-Validation Score: Retrieved from grid_roc.best_score_ (ROC AUC score from training dataset)

Final Model Results

Train Accuracy: 0.85569 (85.56%)
Test Accuracy: 0.78600 (78.6%)

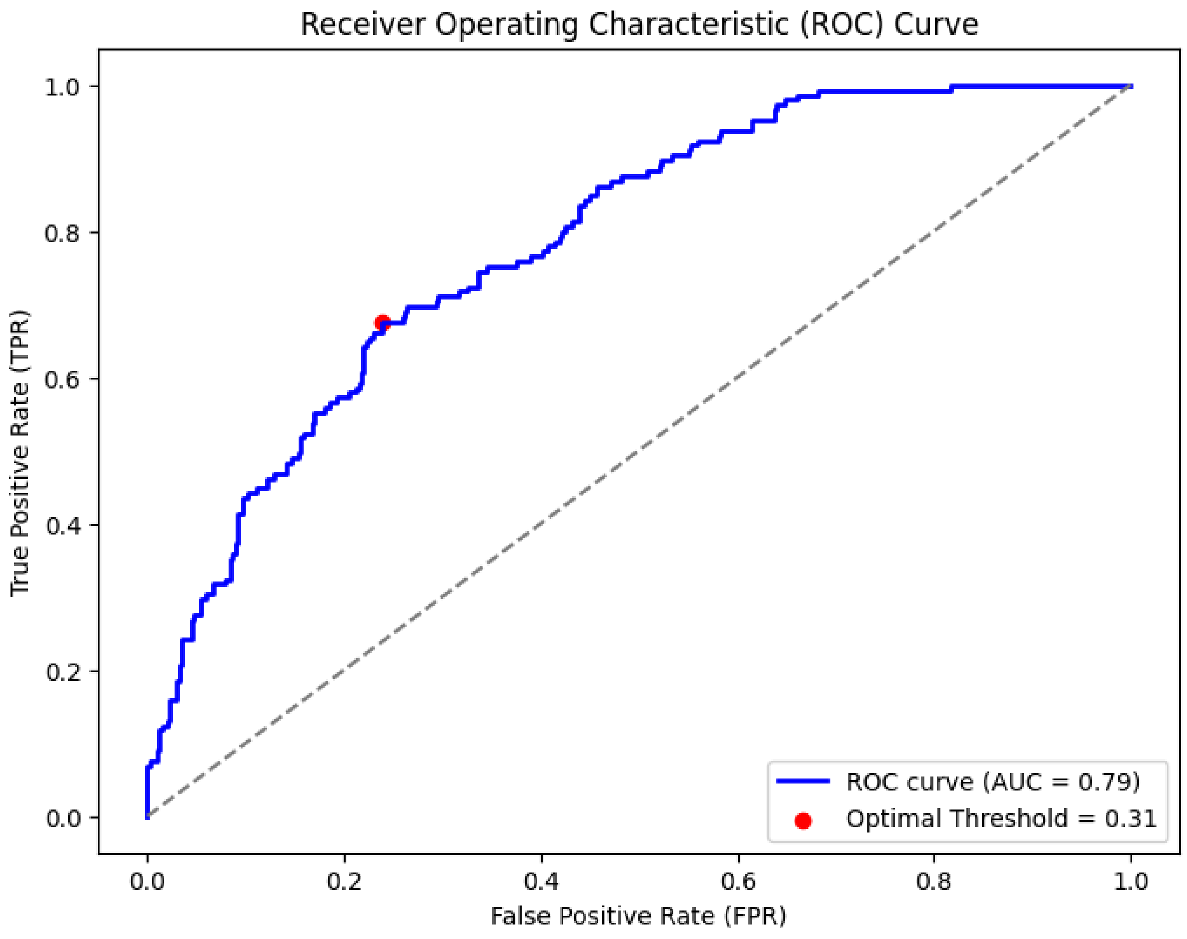
Model Comparison

Initial Accuracy: 0.7564 (75.64%)
Improved Accuracy: 0.78600 (78.6%)
Improvement: 3.9186% relative improvement

Model Evaluation Metrics

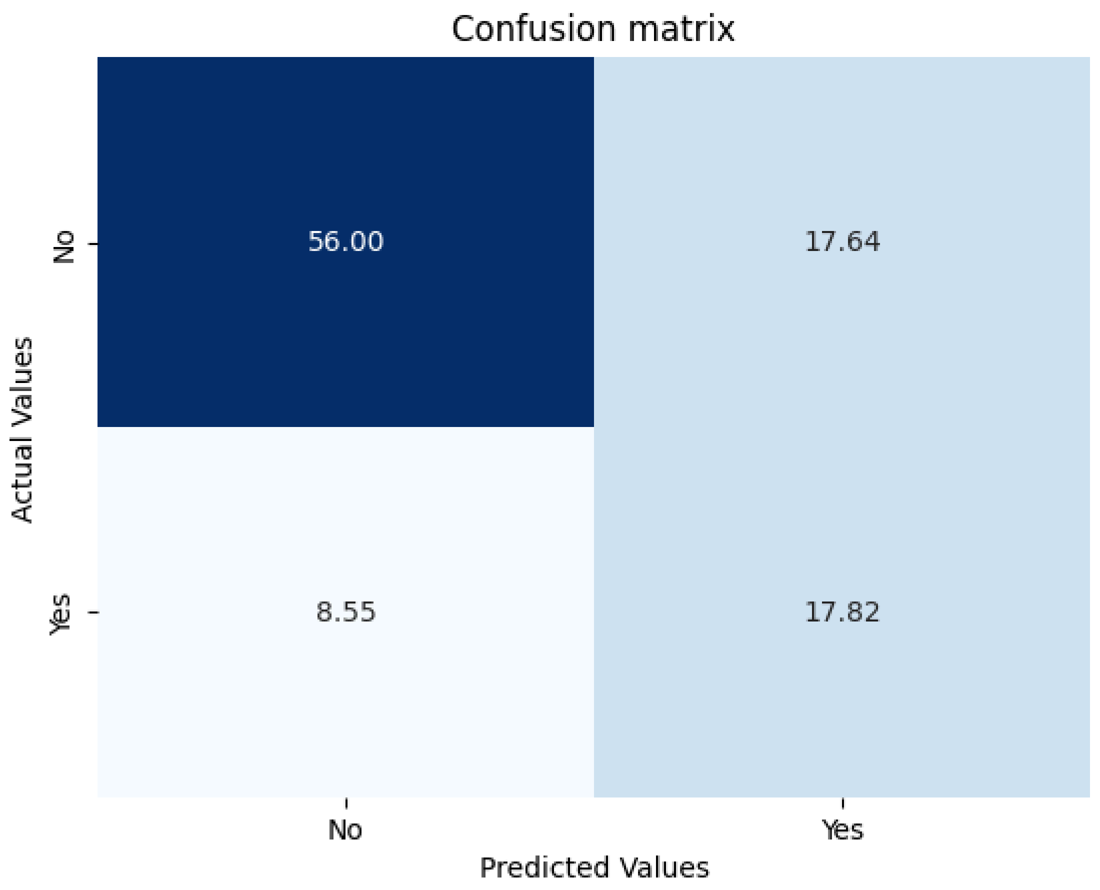
ROC Curve Analysis

Optimal Threshold: 0.31 (determined by maximizing TPR - FPR)



Confusion Matrix (%)

The confusion matrix for the test set was generated using an optimal classification threshold of 0.31, identified from the ROC curve as the point maximizing the trade-off between sensitivity and specificity:



Conclusion

From the confusion matrix we can deduce:

- True Negatives (TN): 56%
- False Negatives (FN): 8.55%
- False Positives (FP): 17.64%
- True Positives (TP): 17.82%

The model is not very conservative in predicting “Yes.”
It correctly identifies about 78% of the truly popular songs, showing good recall (sensitivity). However, its precision is lower, meaning that when it predicts a song will be popular, it is often wrong.

The model is good at detecting potential hits, but it tends to overpredict popularity, flagging many songs that don’t actually become popular.