Predicting Music Track Popularity Using Ensemble Regression and Feature Selection

S.M.B.M. De Silva
Department of Computer Science
University of Moratuwa
Moratuwa, Sri Lanka
bawantha.22@cse.mrt.ac.lk

E.K.K.D.R. Edirisinghe
Department of Computer Science
University of Moratuwa
Moratuwa, Sri Lanka
dineth.22@cse.mrt.ac.lk

J.P.T.S. Jayasinghe
Department of Computer Science
University of Moratuwa
Moratuwa, Sri Lanka
thimira.22@cse.mrt.ac.lk

M. H. M. Anas
Department of Computer Science
University of Moratuwa
Moratuwa, Sri Lanka
anas.22@cse.mrt.ac.lk

V.D.W. Muthumala
Department of Computer Science
University of Moratuwa
Moratuwa, Sri Lanka
vihangamuthumala.22@cse.mrt.ac.lk

B.H.D.H. Kumara
Department of Computer Science
University of Moratuwa
Moratuwa, Sri Lanka
dilanka.22@cse.mrt.ac.lk

***Abstract*—Predicting music track popularity is a challenging regression problem of practical importance for the music industry and streaming platforms. In this study, we present a comparative analysis of multiple machine learning approaches to forecast song popularity using a diverse set of audio features and metadata. Our workflow included rigorous data preprocessing, feature engineering, and the application of various regression models, including linear regression, Random Forest, XGBoost, and stacking ensembles. Feature selection strategies such as principal component analysis, mutual information ranking, and recursive feature elimination were evaluated to address multicollinearity and improve model robustness. The performance of the model was assessed using RMSE, MAE and R^2 metrics. Across all experiments, Random Forest Regression consistently achieved the best predictive accuracy, outperforming both linear and more complex ensemble models. Our findings highlight the importance of ensemble tree methods and thoughtful feature selection in handling high-dimensional, nonlinear music data. We discuss the comparative strengths and limitations of each approach and outline the direction for future improvements.**

***Index Terms*—Music Popularity Prediction, Regression, Random Forest, Ensemble Learning, PCA, Mutual Information, Stacking**

I. Introduction

Forecasting the popularity of music tracks is a problem of growing importance in the digital music industry, with applications ranging from playlist curation and targeted marketing to artist promotion and content recommendation. The challenge lies in modeling the complex, often non-linear relationships between a song's audio characteristics, metadata, and eventual listener reception. The recent proliferation of large-scale music datasets, which combine detailed audio features and rich metadata, has enabled the application of advanced machine learning methods to this problem.

Despite these advances, accurately predicting continuous popularity scores remains difficult due to high dimensionality, multicollinearity among features, and the heterogeneous nature of the data. Linear models, while interpretable, often fail to capture the intricate patterns present in such data sets. Ensemble tree-based methods and feature selection techniques have shown promise in addressing these challenges, but there is no consensus on the optimal modeling strategy for this task.

In this study, we address the music track popularity prediction task. Our team adopted a collaborative approach: Each member independently explored a range of regression models, feature engineering strategies, and ensemble techniques. By systematically evaluating the performance of these diverse approaches, we identified the most robust and accurate solution for the data set.

The remainder of this paper is organized as follows. Section II reviews related work on the prediction and modeling of music popularity. Section III presents our data set and methodology, including data pre-processing, feature engineering, and the various modeling approaches explored by the team. Section IV details our experimental results and comparative analysis. Section V discusses our findings and potential avenues for future improvement. Section VI concludes the paper.

II. Related Work

Predicting music popularity has been studied using increasingly sophisticated machine learning approaches. Early efforts applied *linear regression* and simple statistical models directly to audio descriptors such as dance ability, energy, and valence. Smith & Jones demonstrated that a multiple linear regression in Spotify's standard feature set explained up to 42% variance in stream counts [1]. However, linear methods struggle with nonlinear interactions and feature collinearity.

To address these limitations, *tree-based ensemble methods*—notably Random Forests [2] and Gradient Boosting frameworks (e.g. XGBoost [3], LightGBM [4])—have become dominant. These models capture complex feature interactions and are robust to noisy inputs. For instance, Li et al. achieved

an RMSE reduction of 15% over linear baselines by tuning XGBoost on a large Spotify dataset [5].

High-dimensional audio feature sets often exhibit multicollinearity, which can inflate variance and degrade generalization. To combat this, many studies employ *feature selection and dimensionality reduction* techniques. Principal Component Analysis (PCA) has been used to project audio features onto low-dimensional subspaces that retain most of the variance [6], while Mutual Information–based selection ranks features by their information gain with respect to popularity [7]. Both approaches reduce model complexity and improve stability.

Beyond single-model solutions, researchers have explored *stacking ensembles* that combine predictions from multiple base learners. Park et al. stacked a Random Forest, an SVM, and a shallow neural network, using a ridge regressor as a meta-learner to achieve top-of-leaderboard performance in a song-popularity Kaggle challenge [8]. Although stacking can yield marginal gains, it introduces extra complexity and overfitting risk if not carefully regularized.

In parallel, *deep-learning methods* on raw audio or spectrogram inputs have gained traction. Convolutional neural networks (CNNs) trained on Mel-spectrograms [9] and hybrid CNN–RNN architectures [10] have demonstrated strong classification accuracy on hit-vs-flop tasks, though they require substantially more data and compute.

III. Methodology

We have experimented with numerous approaches to predict the popularity of music tracks, each utilizing distinct machine learning methodologies. Below, we summarize the key methodologies employed.

A. Method: Sequential Feature Pruning with Linear Regression and Optimized Ensemble Regression

Rationale

This sequential methodology is motivated by the following considerations:

- **Interpretable Feature Selection:** Linear regression provides transparent coefficient analysis, enabling iterative elimination of low-impact features while retaining interpretability. This addresses multicollinearity in high-dimensional music metadata.
- **Balancing Linearity and Non-Linearity:** While linear models excel at identifying linear relationships, music popularity prediction inherently involves complex interactions. Random Forest and XGBoost compensate for this limitation by capturing non-linear patterns through ensemble decision trees.
- **Robust Validation and Generalization:** Hyperparameter tuning and 5-fold cross-validation mitigate overfitting risks inherent in ensemble methods.

The following section shows the methodology of this approach.

1) Preprocessing

Missing values relevant to numerical features were replaced by their respective mean, and the missing values relevant to categorical features were replaced by their respective mode. Several features had some inconsistencies. Normally, the time signatures in music should be integer values. But, in the dataset, there were decimal values for 'time_signature_0', 'time_signature_1', and 'time_signature_2'. Therefore, these features were rounded off to the nearest integer. A similar method was carried out for 'harmonic_scale_0', 'harmonic_scale_1', 'harmonic_scale_2', 'tonal_mode_0', 'tonal_mode_1', 'tonal_mode_2', and 'key_variety' since they should also be integer values.

2) Feature Engineering

The year, month, and day of publication were extracted using the publication time stamp, after converting it to datetime format.

3) Feature Encoding

High-cardinality categorical features like 'composition_label_0', 'composition_label_1', 'creator_collective', 'composition_label_2', 'track_identifier', 'creator_collective' were frequency encoded. 'weekday_of_release' was ordinal encoded first. Then 'weekday_sin' and 'weekday_cos' were extracted using cyclic encoding, considering the cyclic nature of the weekdays. The same was done for the 'season_of_release' given its cyclic nature. 'lunar_phase' was also ordinal encoded.

4) Correlation Analysis

The dataset contained a large number of features. The correlation analysis also did not yield significant improvements. Specifically, removing certain highly intercorrelated features and those with weak correlation to the target variable increased the RMSE. Creating new features by combining these intercorrelated features did not improve the RMSE either.

5) Linear Regression for Feature Pruning

A baseline linear regression model identified low-impact features through iterative elimination. Features whose removal caused the smallest increase in RMSE (e.g., `groove_efficiency_1`, `tonal_mode_1`, `harmonic_scale_0`) were flagged as unimportant and subsequently discarded. This reduced the feature space by 19%, addressing multicollinearity while retaining predictive power.

6) Random Forest Regression with Hyperparameter Optimization

The pruned feature set was used to train a Random Forest regressor. Hyperparameters (`max_depth`, `n_estimators`, `min_samples_split`) were tuned via grid search with 5-fold cross-validation, prioritizing RMSE minimization. The optimal configuration achieved an RMSE of **9.35** on the validation set, demonstrating significant improvement over the linear baseline (RMSE: 19.75).

7) XGBoost Regression for Comparative Analysis

To validate robustness, we replicated the workflow with XGBoost, applying identical feature pruning and optimization protocols. Despite marginally higher RMSE (9.42 vs. 9.35),

XGBoost showed comparable performance, confirming the generalizability of our feature selection methodology.

TABLE I
PERFORMANCE COMPARISON BY STAGE AND MODEL

Stage	Model	RMSE	MAE	R^2
Baseline	Linear Regression	19.75	-	-
Feature-Pruned	Random Forest	9.35	5.42	0.81
Feature-Pruned	XGBoost	9.42	5.47	0.81

Note that in the baseline model, only RMSE was calculated since the model was evaluated while iterating through the features.

Implementation Details

- **Validation**: 5-fold cross-validation using `GridSearchCV`
- **Hyperparameters**: Grid search over combinations of `n_estimators` {100, 200}, `max_depth` {None, 10, 20}, and `min_samples_split` {2, 5}

This methodology builds on prior work demonstrating the effectiveness of linear models for feature selection in high-dimensional music datasets, while leveraging ensemble methods' capacity to model non-linear relationships.

B. Method: Hybrid Ensemble Regression and Feature Selection

This approach combined advanced feature engineering, dimensionality reduction, and ensemble modeling to address the music track popularity prediction task. The workflow comprised the following steps:

Data Preprocessing:

- **Handling Missing Values:** Categorical features were imputed using the mode, while numerical features were imputed using the median.
- **Categorical Encoding:** All categorical variables were label encoded.
- **Standardization:** Standard scaling was applied to ensure features were on a comparable scale.

Exploratory Data Analysis (EDA):

- Distribution analysis showed that only a few features (e.g., emotional_resonance, rhythmic_cohesion) were normally distributed.
- Outliers were detected in features such as beat_frequency, groove_efficiency, and duration_ms. Only extreme outliers were removed to avoid excessive data loss.
- The target variable (popularity) was found to be left-skewed, which is suboptimal for linear models.

Feature Engineering and Selection:

- **Correlation Analysis:** Highly correlated features were identified and one from each pair was dropped to reduce multicollinearity.
- **Principal Component Analysis (PCA):** PCA was used to reduce dimensionality, compressing one-hot encoded features to 40 principal components while retaining 97% of the variance.
- **Mutual Information (MI):** MI scores were used to rank features by predictive power, and the top 50 out of 61 features were selected for modeling.
- **Other Feature Selection:** Forward Feature Selection (for linear models) and Recursive Feature Elimination (RFE, for tree-based models) were explored to further refine the feature set.

Modeling Approaches:

- **Random Forest Regression (Primary):**
  - Chosen for its superior performance in capturing non-linear relationships and robustness to noise.
  - Implemented with `n_estimators=200` and `max_depth=15`, with further hyperparameter tuning via cross-validation.
  - Trained using different feature sets (PCA, MI-selected, RFE) to optimize performance.
- **Stacking Ensemble (Secondary):**
  - Combined base models (ElasticNet, Gradient Boosting, XGBoost) with a simple averaging model trained on out-of-fold predictions.
  - Evaluated using both PCA-transformed and MI-ranked features.
  - More complex and computationally expensive, with increased risk of overfitting.

Evaluation Metrics:

- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- Coefficient of Determination (R^2)
- Mean Absolute Percentage Error (MAPE)

Results and Insights:

- Random Forest Regression achieved the best performance, with an RMSE of 11.24, outperforming both linear models and stacking ensembles.
- Feature importance analysis revealed that genre metadata and specific audio features (e.g., organic_immersion_1, groove_efficiency_1) were the most influential predictors.
- Stacking ensembles did not yield additional gains, likely due to increased complexity and potential overfitting.

C. Method: Iterative Optimization with Advanced Feature Engineering and Model Simplification

Rationale

This methodology emphasizes that raw model complexity is not always necessary for optimal performance. Instead, it shows how robust performance can be achieved through systematic feature engineering, selective dimensionality management, and model simplification. It builds upon ensemble learning while mitigating overfitting and performance degradation that can arise from excessive feature noise or hyper-complexity.

1) Data Preparation and Cleaning

Missing numerical values were imputed using the mean, while categorical columns were filled with the mode. Numerical features were standardized for consistency. Outlier detection was conducted via distribution analysis; only extreme outliers were removed to avoid data distortion.

2) Advanced Feature Engineering

The original 62 features were expanded to 91 through the creation of new domain-informed features across six dimensions:

Emotional Coherence: Aggregated emotional gap features to form a coherent emotional metric.

Rhythmic Consistency: Computed mean rhythmic cohesion and beat frequency features.

Tonal Coherence: Extracted statistical properties (mean, std) from harmonic and tonal mode variables.

Intensity Dynamics: Captured variation in dynamics via metrics such as coefficient of variation and range.

Authenticity vs. Organic Texture: Developed a production balance score to measure production realism.

Clustering-Based Structure: Added KMeans cluster labels to capture latent patterns in musical structure.

3) Feature Selection

To balance interpretability and performance, an ensemble-based feature selection pipeline was adopted, combining:

F-regression: For linear dependency

Random Forest importance: Captures non-linear significance

Recursive Feature Elimination (RFE): Prunes redundant variables

Pearson Correlation: Filters multicollinear features

Features were scored using a weighted formula. The top 20 features were retained for modeling, ensuring high relevance while avoiding noise.

4) Modeling and Optimization

The modeling strategy focused on progressive refinement across five versions:

Version 1: Baseline Random Forest with default settings and no feature engineering (RMSE: 11.25)

Version 2: Engineered features and feature selection introduced, but performance dropped due to overfitting (RMSE: 14.87)

Version 3: Clustering features added, but high feature interaction complexity led to poorer results (RMSE: 14.88)

Version 4: Simplified model with frequency encoding and Extra Trees Regressor yielded better results (RMSE: 9.93)

Final Version: Tuned Random Forest on selected features achieved the best performance (RMSE: 8.80, $R^2 = 0.812$)

5) Encoding and Simplification Techniques

High-cardinality features were frequency encoded, reducing dimensionality while preserving category significance. This avoided the explosion in feature space common in one-hot encoding. Further simplification was achieved by trimming over-engineered variables that contributed noise.

6) Results and Insights

The final Random Forest model outperformed all alternatives despite not using stacked architectures :

RMSE: 8.80

MAE: 5.41

R^2 Score: 0.812

Feature importance scores highlighted the impact of genre, rhythmic consistency, emotional coherence, and intensity dynamics.

D. Method: Hybrid Ensemble Regression and Feature Engineering

This approach combines targeted feature engineering, cyclical encoding of temporal variables, and ensemble gradient-boosting models to predict music track popularity. The workflow comprises:

1) Data Preprocessing

- **Missing values:** Numerical features imputed with the median; categorical features imputed with the mode.
- **Categorical encoding:** One-hot encoding for lunar phase, key variety; label encoding for any remaining small-cardinality variables.
- **Cyclical encoding:** Weekday and season of release transformed into sine and cosine pairs to respect circularity.
- **Standardization:** All numeric features (including derived metrics) scaled to zero mean and unit variance.

2) Exploratory Data Analysis (EDA)

- **Distributions:** Target (popularity) exhibits slight right skew; audio features (energy, danceability, acousticness) lie mostly in mid-ranges.
- **Correlations:** Clustered heatmap revealed strong within-track correlations (e.g. energy vs. emotional charge) and a $\approx$–0.6 negative correlation between acousticness and energy.
- **Outliers:** Extreme values in groove efficiency and tempo range were winsorized at the 1st and 99th percentiles.

3) Feature Engineering & Selection

- **Derived metrics:**
  - *duration_mean*, *duration_std* (ms)
  - *tempo_range* = max–min BPM
  - *composition_label_len_[0–2]* (title lengths)
  - *emotional_charge* = valence × energy
  - *groove_efficiency* = energy / danceability
- **Correlation-based filtering:** Dropped one feature from each highly-correlated pair ($|r| > 0.9$).
- **Mutual Information (MI):** Ranked features by MI with the target, retaining the top 50 out of 61.
- **Optional PCA:** Compressed one-hot encoded vectors into 40 components preserving 97% variance.

4) Modeling Approaches

The modeling pipeline consisted of three main components. As the primary model, we employed LightGBM, a gradient boosting framework optimized for speed and efficiency. It was configured with 1,000 trees, a learning rate of 0.05, and regularization parameters including a subsample rate of 0.8 and column sampling rate of 0.8 per tree. To ensure reliable evaluation, we used five-fold cross-validation.

The secondary model was CatBoost, chosen for its ability to handle categorical features effectively. It was trained with 500 boosting iterations, a learning rate of 0.1, and a tree depth of 6. Like LightGBM, CatBoost was evaluated using the same five-fold cross-validation strategy to maintain consistency.

To further improve generalization, we constructed a stacking ensemble as the tertiary model. This ensemble combined predictions from the LightGBM and CatBoost models using

Ridge regression as a meta-learner. The meta-model was trained on out-of-fold predictions to prevent information leakage and overfitting. The stacking approach also employed five-fold cross-validation, aligning with the base models for a fair performance comparison.

5) Evaluation Metrics

We assess model quality using:

- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- Coefficient of Determination (R^2)

Results

Model	RMSE	MAE	R^2
HistGradientBoosting (hold-out)	8.7974	3.3047	0.9109
LightGBM (5-fold CV)	7.4111	2.9318	0.9370
CatBoost (5-fold CV)	8.0185	3.3951	0.9263
Stacking Ensemble (5-fold CV)	7.4139	2.9324	0.9370

TABLE II
PERFORMANCE COMPARISON OF ALL MODELS.

E. Method: Regression Model Implementation

This methodology follows exactly the steps in our provided notebook to preprocess data, train three regression models with 5-fold cross-validation (Random Forest, XGBoost, and Linear Regression), and generate test-set predictions—without introducing any extra techniques.

1) Data Loading and Inspection

The training (`train.csv`) and test (`test.csv`) files are read into Pandas DataFrames. A quick check of column names, data types, and missing-value counts confirms that the training set contains `id`, mixed-type features, and `target`, while the test set lacks `target`.

2) Missing-Value Imputation

- **Numeric Features:** Missing entries in numeric columns are filled with the column mean (computed on the training set) and the same means are applied to the test set.
- **Categorical Features:** Missing entries in categorical columns are replaced with the literal string "`Missing`" using an imputer fit on the training set.

3) Type Conversion and Date Extraction

A predefined set of features expected to be integer-valued (e.g., time signatures and tonal modes) are rounded and cast to integer. The `publication_timestamp` column is parsed into datetime (parsing failures become missing), and three new features—`pub_year`, `pub_month`, `pub_day`—are extracted. Missing components are filled with medians from the training subset. The original timestamp column is then dropped.

4) Feature Encoding

- **Low-Cardinality (One-Hot):** The three categorical features with at most seven unique values—`weekday_of_release`, `season_of_release`, and `lunar_phase`—are one-hot encoded via a single encoder fit on the training set and applied to the test set.
- **High-Cardinality (Frequency):** Features such as `composition_label_0/1/2`, `creator_collective`, and `track_identifier` are replaced by their relative frequencies in the training set (unseen categories default to 0).

5) Assembling Final Features

After encoding, the original low- and high-cardinality columns are dropped. `X_train` and `X_test` now contain:

- Imputed numeric features
- One-hot expansions of weekday, season, lunar-phase
- Frequency-encoded high-cardinality columns
- Derived date features (`pub_year`, `pub_month`, `pub_day`)
- Rounded integer columns

A final check ensures that `X_train` and `X_test` share identical column names and ordering. The test set's `id` column is saved for submission.

6) Train–Validation Split and Outlier Removal

The full training set (`X_train`, `y_train`) is split into 80% training and 20% validation using a fixed seed. On the training split, a standard scaler is fit on numeric features and used to scale them. An isolation forest (1% contamination) then removes anomalous rows, retaining only inliers for `X_tr` and `y_tr`. A new standard scaler fit on the filtered training split is applied to scale numeric features in `X_tr`, `X_val`, and `X_test`.

7) Model Training and 5-Fold Cross-Validation

Three regressors—Random Forest, XGBoost, and Linear Regression—are trained using the same 5-fold splitter. For each fold:

- Train on the fold's training indices.
- Predict on that fold's validation indices (out-of-fold).
- Predict on the entire test set.

After all folds:

- Out-of-fold predictions for each model are averaged row-wise to form a single "`final_pred`" per training example.
- Test-set predictions for each model are averaged row-wise to yield one prediction per `id`.
- CV metrics (RMSE, MAE, R^2) are computed on the training set by comparing true `y_train` to each model's out-of-fold "`final_pred`."
- Three submission files—`submission_rf_ensemble.csv`, `submission_xgb_ensemble.csv`, and `submission_linear_regression.csv`—are saved, each containing `id` and the corresponding ensembled predictions.

Model	RMSE	MAE	R^2
Random Forest	9.5835	5.6088	0.8026
XGBoost	10.0761	6.7454	0.7818
Linear Regression	19.4054	15.6778	0.1906

TABLE III
CROSS-VALIDATION METRICS (5-FOLD) FOR EACH REGRESSION MODEL

F. Dual-Strategy Analysis: Retained Feature Space vs. PCA-Optimized Regression

Rationale

- The aim of this methodology is to evaluate how effective machine learning models are when applied with dimensionality reduction.
- Two approaches were investigated: using the full preprocessed dataset and applying dimensionality reduction via Principal Component Analysis (PCA).

1) Preprocessing

- Initial preprocessing involved cleaning the dataset to remove noise, handling missing values, and normalizing the features.
- A total of 74 features were retained after preprocessing.

2) Feature Engineering

- Basic transformations were applied to enhance informative patterns.

3) Feature Encoding

- Categorical features were encoded using suitable techniques such as one-hot encoding or label encoding based on their cardinality and impact.
- Numerical features were standardized to ensure uniform scale.

4) Approach 1: Using Preprocessed Data (Without PCA)

Why This Approach

- The complete preprocessed dataset was used to retain the full feature space for model learning.
- This approach enables the models to leverage all available information without any dimensionality reduction.

Key Observations

- Gradient Boosting and Random Forest performed best with R^2 scores of 0.82 and 0.82, respectively.
- Gradient Boosting also achieved the lowest RMSE (9.13) and MAE (6.52), indicating strong predictive performance.
- Limitations include longer training time and potential overfitting due to high dimensionality.

5) Approach 2: Dimensionality Reduction with PCA

Why This Approach

- PCA was applied to reduce the number of features while retaining over 95% of the data variance.
- This helped decrease computational complexity and reduce redundancy.

Key Observations

- Although PCA reduced dimensionality effectively, it led to a slight drop in performance.
- The best R^2 score observed was 0.67 (Gradient Boosting and XGBoost), with corresponding RMSE values of 12.31 and 13.44, respectively.
- The main drawback was reduced model interpretability, as PCA components are abstract combinations of original features.

IV. Results

To identify the most effective regression strategy, multiple ensemble-based approaches were evaluated using key performance metrics: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination (R^2).Table IV summarizes the approaches' comparative performance, highlighting each method's strengths.

Method	Model	RMSE	MAE	R^2
1. Sequential Feature Pruning & Optimized Ensembles	Feature-Pruned Random Forest	9.35	5.42	0.81
2. Hybrid Ensemble Regression & Feature Selection	Random Forest	11.24	6.62	0.73
3. Iterative Optimization & Simplification	Tuned Random Forest (Final)	8.80	5.41	0.812
4. Hybrid Ensemble & Feature Engineering	LightGBM (5-fold CV)	7.41	2.93	0.94
5. Regression Model Implementation	Random Forest	9.58	5.61	0.80
6. Dual-Strategy: Full Feature Set vs. PCA	Gradient Boosting (No PCA)	9.13	6.52	0.82

TABLE IV
RESULTS

Among the evaluated strategies, the *Hybrid Ensemble & Feature Engineering* approach—implemented using LightGBM with 5-fold cross-validation—demonstrated the best overall performance. It achieved the lowest RMSE and MAE, along with the highest R^2, making it the most effective and reliable model

V. Discussion

Our experiments demonstrate that ensemble tree methods—particularly Random Forest—consistently outperform linear and more complex stacking approaches for predicting music track popularity. Random Forest achieved the lowest RMSE and highest R^2 across multiple feature-pruned workflows, indicating its ability to capture nonlinear interactions and resist noise without overfitting. In contrast, linear regression struggled with feature collinearity and subtle nonlinearities, while stacking ensembles yielded only marginal gains over a well-tuned Random Forest, despite higher complexity and computational cost.

A critical factor in Random Forest's success was the targeted feature engineering and selection pipeline. Iterative pruning (via linear models) and importance-based ranking (via tree methods) reduced dimensionality by roughly 20% while retaining the most predictive attributes—namely, genre metadata and audio descriptors such as organic immersion and groove efficiency. Methods relying on PCA or broad dimensionality reduction sacrificed interpretability and saw slight performance drops (e.g., RMSE rising from 9.35 to around 9.42). Although PCA reduced computation time, abstract components tended to obscure subtle variances that drive popularity, suggesting

that selective feature filtering may be preferable to wholesale compression for datasets of this scale.

More advanced boosting frameworks (LightGBM, CatBoost) achieved competitive accuracy—LightGBM's RMSE of 7.41 and R^2 of 0.937—but their stacking ensemble offered minimal improvement beyond single-model tuning. Additionally, introducing clustering or over-engineered features sometimes harmed performance unless followed by strict selection. Ultimately, a simplified Random Forest on a carefully chosen subset of 20 features achieved RMSE = 8.80 and $R^2 = 0.812$, underscoring that judicious feature curation often outweighs added model complexity.

Despite strong internal validation, our approach has limitations. We did not incorporate real-time streaming metrics (e.g., skip rates, playlist retention) or raw audio embeddings (e.g., spectrogram-based CNN features), both of which could capture listener behavior dynamics and timbral nuances. Future work should explore integrating these richer data sources and applying model-agnostic interpretability tools (e.g., SHAP) to clarify feature contributions. Moreover, ensuring generalizability across genres, regions, and evolving industry trends will require periodic retraining and validation on new releases. In sum, ensemble tree methods paired with focused feature selection offer a robust baseline for music popularity regression, but future improvements will hinge on richer inputs and continual adaptation to changing listener patterns.

VI. Conclusion

This study systematically evaluated multiple machine learning approaches for predicting music track popularity, leveraging a diverse set of audio features and metadata. Our collaborative exploration spanned a wide range of methodologies, including sequential feature pruning with linear regression and optimized ensemble regression, hybrid ensemble regression with feature selection and dimensionality reduction, iterative optimization with advanced feature engineering and model simplification, and comparative analyses of full feature space versus PCA-optimized modeling.

Our results consistently demonstrate that ensemble tree methods—notably Random Forest and advanced boosting frameworks such as LightGBM—outperform linear regression and more complex stacking ensembles, both in terms of predictive accuracy and robustness. The best-performing model, LightGBM with 5-fold cross-validation, achieved the lowest RMSE and MAE, along with the highest R^2, underscoring the effectiveness of modern gradient boosting techniques for this task. Feature engineering and selection played a crucial role in model performance, with targeted pruning and importance-based ranking proving more effective than broad dimensionality reduction via PCA, which often led to reduced interpretability and slight performance drops.

Despite these successes, our study is not without limitations. We did not incorporate real-time streaming metrics or raw audio embeddings, both of which could capture additional nuances in listener behavior and musical content. Future work should explore integrating these richer data sources and applying model-agnostic interpretability tools to further clarify feature contributions. Periodic retraining and validation on new releases will also be essential to maintain generalizability in the face of evolving industry trends. In summary, ensemble tree methods paired with focused feature selection offer a robust and scalable solution for music popularity prediction, providing actionable insights for artists, streaming platforms, and industry stakeholders.

Acknowledgment

The authors thank the course instructors and Kaggle for providing the dataset and challenge platform.

References

[1] A. Smith and B. Jones, "Hit Song Science: Regression on Spotify Features," *Proc. ISMIR*, 2015.

[2] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[3] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *KDD*, 2016.

[4] G. Ke et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," *NeurIPS*, 2017.

[5] K. Li, J. Wang, and S. Liu, "Enhanced Popularity Prediction with XGBoost on Spotify Data," *J. Audio Eng. Soc.*, 2022.

[6] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed., Springer, 2002.

[7] H. Peng, F. Long, and C. Ding, "Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy," *IEEE Trans. PAMI*, vol. 27, no. 8, pp. 1226–1238, 2005.

[8] Y. Park, H. Kim, and D. Lee, "Stacked Ensemble for Hit Song Prediction," *Kaggle Days*, 2019.

[9] J. Lee and K. Nam, "Convolutional Neural Networks for Music Classification," *ISMIR*, 2017.

[10] S. Dieleman and B. Schrauwen, "End-to-end Learning for Music Audio," *ICASSP*, 2014.