How To Make A Hit Song On Spotify Using Machine Learning

1st Junseong Lee
University of California, Davis
Davis, United States
prlee@ucdavis.edu

2nd Rishi Kumar
University of California, Davis
Davis, United States
rshkumar@ucdavis.edu

3rd Patrick Wang
University of California, Davis
Davis, United States
pchwang@ucdavis.edu

4th Vincent Xayasak
University of California, Davis
Davis, United States
vxxayasak@ucdavis.edu

5th Zishuo Li
University of California, Davis
Davis, United States
zshli@ucdavis.edu

***Abstract*—With over 11 million artists and 675 million users, Spotify has become a leading platform for music distribution, making it increasingly challenging to identify what makes certain songs more popular than others. This project aims to predict a song's popularity using machine learning techniques applied to the Spotify Tracks Dataset, which contains over 100,000 tracks with features such as danceability, energy, duration, and more. We conduct exploratory data analysis (EDA) to uncover patterns and correlations, and evaluate the effectiveness of various regression models, including Linear Regression, Decision Trees, Random Forests, and Artificial Neural Networks. Given the potential for feature collinearity, we hypothesize that nonlinear models will outperform linear ones in predicting popularity. Our findings aim to provide insights into the key elements that contribute to musical success.**

***Index Terms*—machine learning, music features, popularity.**

I. Introduction and Background

Today, music listening has been changed by streaming platforms, with Spotify emerging as one of the most dominant forces in the industry. As of 2025, Spotify hosts over 11 million artists and serves more than 675 million users worldwide. The platform's ease of access and global reach have led to an unprecedented volume of music being released daily, making it increasingly difficult to predict which songs will resonate with audiences and rise to prominence. While talent, timing, and marketing undoubtedly play a role, there remains a critical question: What features of a song contribute to its popularity?

Understanding the elements that drive a song's success can be incredibly valuable, not only to artists aiming to optimize their creative output, but also to producers, music marketers, and data-driven platforms curating user experiences. By leveraging data analytics and machine learning, we can identify and quantify patterns in musical features that are commonly associated with higher popularity metrics.

In this study, we aim to predict a song's popularity based on its audio and metadata features, using machine learning as a tool for uncovering the underlying factors that influence listener engagement. Specifically, we utilize the Spotify Tracks Dataset, a comprehensive collection comprising over 100,000 tracks. Each entry in the dataset includes features such as artist name, track duration (in milliseconds), whether the track is explicit, and a variety of audio-based metrics such as danceability, energy, valence, tempo, and more.

Our research methodology begins with exploratory data analysis (EDA) to investigate the relationships and correlations between different song features and their popularity. By visualizing distributions, examining feature importance, and identifying outliers, we aim to gain an initial understanding of which characteristics might be most influential. We then proceed to train several machine learning models, including regression and ensemble methods, to predict popularity scores and evaluate their performance.

The ultimate goal of this research is to shed light on the measurable attributes that contribute to a song's popularity. Our findings can provide actionable insights for emerging artists, producers, and industry stakeholders seeking to navigate the complex and competitive landscape of the music industry. Moreover, this study contributes to the growing body of work at the intersection of music and data science, illustrating how computational methods can be used to decode creative success.

This study explores the use of various regression approaches, including Linear Regression, Decision Trees, Random Forests, and potentially Artificial Neural Networks, to model and predict the popularity of songs based on Spotify track features. By comparing the performance of these models, we aim to identify the most effective techniques for understanding and predicting musical popularity in a data-driven framework.

II. Literature review

In this section, we review representative studies that have explored feature-based modeling for hit-song prediction, identify common methodological frameworks.

A. Feature Correlation and Early Models

Pachet and Roy [1] performed a large-scale empirical analyses of "hit-song science," using Echo Nest descriptors such as loudness, tempo, timbre, and danceability alongside basic metadata (e.g., artist familiarity). By training decision-tree classifiers, they demonstrated that tracks with higher danceability and energy were substantially more likely to correlate with high popularity scores. Importantly, their practical takeaway was that while individual features can signal potential, audio descriptors must be combined with marketing and social signals for deployment in a commercial setting.

B. Temporal Dynamics and Advanced Feature Engineering

Recognizing that static summaries lose important time-varying patterns, Herremans *et al.* [2] extracted statistical moments (mean, variance, skewness) over sliding windows fo timbre-PCA coefficients and rhythm intervals. Their logistic regression model achieved over 80% accuracy on a dance-music subset, suggesting that capturing fluctuations-such as build-ups and breakdowns-improves classification of listener engagement moments. In practice, this insight informs playlist algorithms that aim to maintain listener attention by sequencing track with complementary dynamic profiles.

C. Metadata and Engagement Signals

Moving beyond audio, Li [3] analyzed over 50,000 Spotify tracks with ordinary least squares regression on features like danceability, acousticness, and track duration, and augmented them with release-year and artist follower counts. They showed that while core audio features explain a sizable fraction of variance, incorporating release timing and audience size boosts explained variance by up to 10 percentage points. Nijkamp [4] similarly found that combining audio descriptors with contextual metadata (e.g., genre tags, label promotion budgets) yields more robust popularity estimates, highlighting the importance of non-audio information in applied forecasting systems.

D. Model Interpretability and Deployment

In these models, interpretability is as critical as raw accuracy. While tree-based models (Pachet & Roy [1]) surface simple rules-"if danceability $>$ 0.7 and loudness $>$ $-$6dB, recommend for summer playlists"-linear models allow straightforward attribution of feature contributions (Li [3]). Our worked adopts both paradigms, training random forests for performance and examining permutation importance to derive actionable insights.

III. Exploratory Data Analysis

Dataset Description:

This dataset contains 114,000 rows and 21 columns, making it a large and comprehensive collection centered on music tracks, their attributes, and associated data. It brings together both quantitative audio features and qualitative identifiers, offering a broad base for examining musical characteristics, artist patterns, and genre trends across a substantial number of songs.

The dataset covers a wide range of musical characteristics, combining audio features and descriptive data to enable in-depth exploration of tracks, artists, and genres. It includes audio based attributes such as danceability, energy, loudness, tempo, and valence, which reflect the emotional qualities of the music. This combination of features allows for a well-rounded exploration of both the objective and subjective aspects of the music landscape, making the dataset suitable for tasks ranging from descriptive analysis to predictive modeling.

The key attributes within the dataset can be grouped into several categories. Identifiers such as track_id, track_name, artists, and album_name provide the essential labels and references for each entry. Numerical features include popularity, duration_ms, key, loudness, mode, tempo, and time_signature, which capture measurable aspects of the song. Audio features variables like danceability, energy, speechiness, acousticness, instrumentalness, liveness, and valence describe the emotional profile of the track. The explicit attribute marks whether a song contains explicit content. Finally, the track_genre offers categorical labels that organize tracks by genre, allowing for segmentation and classification.

Exploratory Data Analysis:

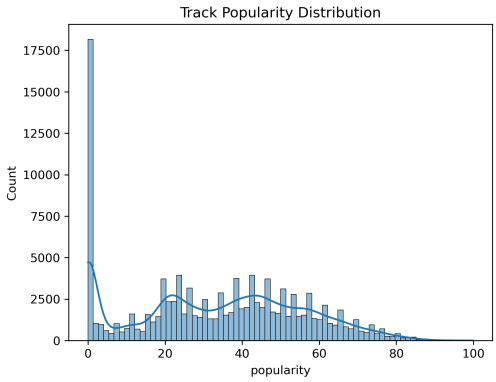

Fig. 1. Popularity distribution of original data

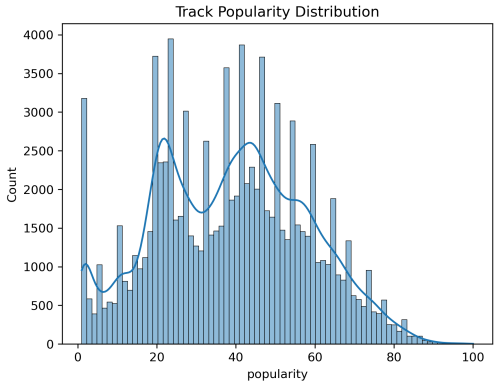

Fig. 2. Popularity distribution excluding zeros and missing values

From (Fig. 1), we can see that there is a disproportionately high number of values with a popularity score of 0. These

may add significant noise to the model, reducing the accuracy of our predictions since it may not be an accurate reflection of a song's effectiveness. Even after removing all observations with a popularity score of 0, there are still 97,980 observations, so we aren't losing a significant amount of data from doing so. We don't know why there are so many observations with a popularity score of 0, but our model will still be able to train on observations where the popularity score is close to 0 to assign lower popularity scores when predicting on the test data when necessary (Fig. 2). The resulting distribution (Fig. 2) is approximately bimodal and skewed to the right due to higher popularity scores existing near 100.

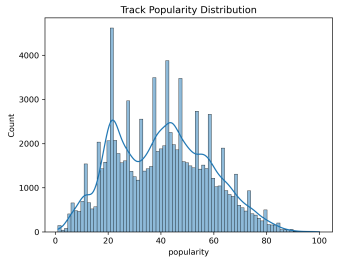

Fig. 3. Popularity distribution excluding October 2022 data

We also chose to remove all songs with a release date in October 2022. This decision was based on the observation that songs released very recently to the creation of the dataset, may not have had enough time to accumulate streams or popularity, resulting in artificially low or inconsistent popularity scores. Including these songs could introduce bias or noise into the model, especially if their popularity is still evolving. By removing this subset, we aim to ensure that the model is trained on songs with more stable and representative popularity data, thereby improving the overall reliability of our predictions (Fig. 3).

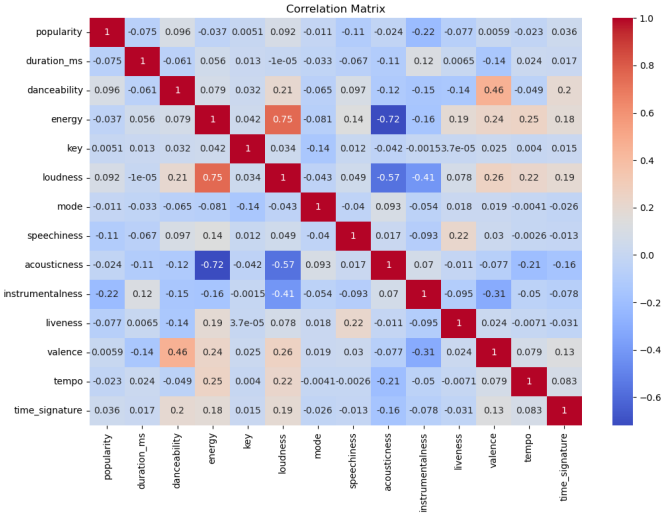

Fig. 4. Heatmap

From the correlation heatmap, we can observe that most features do not exhibit strong linear relationship with one another, with the exception of a few, such as acousticness, loudness, and energy (Fig. 4). These relationships make sense intuitively, as louder tracks tend to be more energetic, while high acousticness is often associated with softer or less energetic compositions. However, popularity does not appear to be strongly correlated with any single feature in the dataset, indicating that no individual audio characteristic is a clear predictor of a song's success on its own. It also suggest that popularity is likely influenced by a more complex combination of features, rather than by any one attribute in isolation.

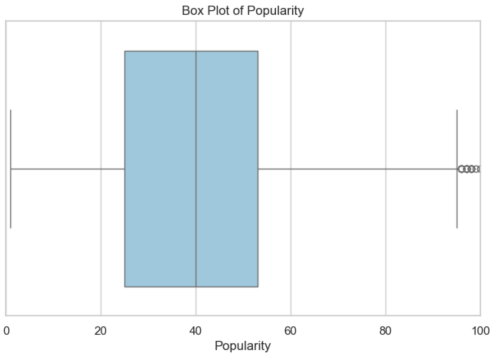

Fig. 5. Box plot of popularity

The variable Popularity exhibits a moderately right-skewed distribution (Fig. 5). Most values fall within the interquartile range (IQR), which is approximately 25 to 55. The median popularity score is 40.0, indicating that half the entries fall below this value. The mean is 40.41, slightly higher than the median, confirming a mild skew toward higher values. The standard deviation is 17.89, suggesting a moderate spread around the average, with a wide range of scores. The minimum popularity score is 1, while the maximum reaches 100. Notably, there are 25 outliers in the dataset values that lie significantly above the typical range. These outliers include repeated high scores such as 96, 97, 98, 99, and 100, indicating that a small subset of items are exceptionally popular compared to the rest.

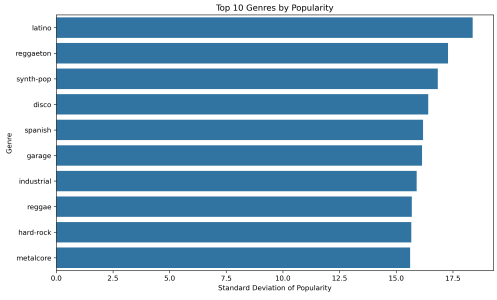

Fig. 6. Standard deviation of genre popularity

The top 10 music genres with the highest standard deviation in popularity offer a fascinating look at the volatility within these styles. What's interesting is that genres like latino,

reggaeton, and synth-pop have the highest standard deviation (Fig. 6), suggesting that while these genres are generally popular, their tracks can be very polarizing. This could also imply that the audiences for these genres might be more dynamic and niche in their preferences, with sub-genres or artists that have either followings or mainstream crossovers. Such variability could also reflect the ongoing evolution of these genres and their shifting relationship with trends and cultural influences.

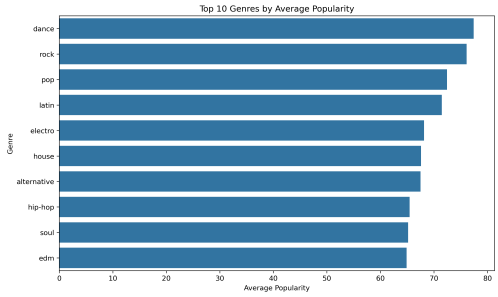

Fig. 7. Average popularity of song genre

The top 10 genres by average popularity shows which musical styles resonate most consistently with listener. Dance, rock, and pop leads this list (Fig. 7), reflecting their steady presence in mainstream music consumption. This consistent popularity might also signal that these genres have become more ingrained in the general listening habits of audiences, offering a reliable soundtrack to people's daily lives regardless of the ever-shifting music scene.

One key factor influencing why certain songs become more popular than others is their exposure through social media and streaming platforms. With the rapid rise of platforms like TikTok, Instagram Reels, and curated Spotify playlists, a song can quickly reach viral status when snippets are used in short videos or featured on popular playlists. In addition, collaborations with well-known artists can significantly boost a song's reach, tapping into the established fan bases of multiple performers. Playlists and algorithmic recommendations also push these songs to the forefront, helping them stand out among countless new releases.

Two major X-factors that drive a song's popularity are social media exposure and artist fame. Platforms like TikTok and Instagram Reels allow short, catchy snippets of music to go viral, often turning obscure tracks into overnight hits through user-generated content and trends. At the same time, song released by or featuring famous artists naturally gain more attention due to their established fan bases and media presence. These factors can spike a song's popularity in unexpected ways and are nearly impossible to predict and quantify ahead of time.

IV. Proposed Methodology

1. Data Collection

To begin the modeling process, we obtained our dataset from a curated collection available on Kaggle. This dataset includes a broad range of song-level information sourced from Spotify's API, offering a structured and comprehensive view of both musical and contextual metadata. The dataset features key audio characteristics such as energy, tempo, danceability, valence, acousticness, intrumentalness, speechiness and loudness, all of which are quantitative indicators generated by Spotify's audio analysis algorithms. These features provide insight into the acoustic and rhythmic qualities of each track. Each track ID of the dataset corresponds to a song and the specific album in which it was released. For additional context, we wanted to also identify when each track was released due to popularity scores potentially being particularly low if a track was released immediately before the dataset was made. This might be because it may take time for newly released songs to gain traction. Since the dataset info was collected in 2022 between October 20 and October 22, we web-scraped release dates for all tracks of our dataset using each track ID.

2. Feature Selection

The primary objective is to retain features that contribute meaningfully to predicting the popularity of a song. At the same time, our aim is to remove features that are redundant or noninformative and therefore not useful for generalization.

We began by removing features that serve as unique identifiers or provide little or no predictive value:

- track_id, track_name, and album_name are identifiers that are unique to each track and do not carry generalizable patterns that a model could learn from.
- artist, although potentially influential due to artist popularity, was excluded because it introduces significant sparsity and overfitting risk. With 28,851 unique artists, one-hot encoding this feature would be computationally infeasible and likely lead to poor generalization. Additionally, our goal is to predict popularity based on audio content rather than artist fame.

We retained the core set of audio feature generated by Spotify's API, including energy, danceability, acousticness, valence, speechiness, intrumentalness, loudness and tempo. All quantitative standardized metrics describing the acoustic and rhythmic characteristics of each track. They were included because they are directly relevant to the nature of music and are likely to influence how listeners perceive and engage with a track, and thus its popularity.

The track_genre feature, while categorical, was retained because it provides high-level contextual information about the type of music. With 114 unique genres, one-hot encoding was computationally reasonable and allowed us to capture genre-based trends without a significant loss of generalization.

We also engineered an additional feature: the release date of each track, which we web-scraped using track IDs. While this feature was not used in our model, it helped us clean the dataset by filtering out tracks released in October 2022. Our final cleaned dataset was saved as Cleaned_Dataset.csv.

3. Model Selection and Training

- **Linear Regression:** This serves as a baseline model for regression analysis. It assumes a linear relationship between independent features and the popularity score. Although relatively simple, it offers useful interpretability and can work well when relationships between variables are mostly linear.
- **Decision Tree (with Pruning):** A single decision tree will be trained using recursively partitioning. To avoid overfitting, pruning techniques such as cost-complexity pruning will be applied. Pruning decision trees are interpretable and can capture non-linear relationships, though they often lack the predictive strength of ensemble models.
- **Decision Tree (Bagged):** To improve the variance of issues of a single decision tree, bagging will be used to create a group of trees trained in bootstrapped subsets of the data. This technique reduces overfitting and increases the stability of predictions while maintaining moderate interpretability.
- **Decision Tree (Random Forest):** An extension of bagged trees, Random Forest introduces additional randomness by selecting a random subset of features at each split. This helps de-correlate the trees and further reduces overfitting. Random Forests are typically robust, perform well across various datasets, and provide feature importance metrics, which are valuable for interpretation.
- **Decision Tree (XGBoost):** Extreme Gradient Boost is an open source machine learning library that boosts decision trees by combining the predictions of multiple decision trees. It works by continuously building models, with each new model focusing on correcting the errors from old models.
- **Decision Tree (LightGBM):** A LightGBM model will be implemented to evaluate the effectiveness of gradient boosting methods for song popularity prediction. LightGBM is a highly efficient framework based on decision tree algorithms, designed for fast training and high accuracy. It handles large-scale data and high-dimensional features well by leveraging histogram-based learning and leaf-wise tree growth strategies. LightGBM also supports advanced features such as regularization, categorical feature handling, and parallel learning, making it well-suited for complex prediction tasks with structured data.
- **Neural Networks:** A feedforward neural network will be implemented to assess the potential of deep learning methods. Neural networks are capable of learning complex nonlinear patterns, especially in high-dimensional spaces. The architecture will include one or more hidden layers with appropriate activation functions and regularization techniques such as dropout or batch normalization to prevent overfitting.
- **K-Fold Cross Validation:** To ensure the robustness and generalization of model performance, the K-Fold Cross Validation will be applied during model training and evaluation. This involves dividing the training data into k subsets, training the model k - 1 times, and validating it on the remaining fold, repeating this process k times. The average performance across all folds will serve as a more reliable estimate of how the model will perform on unseen data, helping to avoid bias due to data splits.
- **Hyperparameter Tuning:** Choosing the right hyperparameter values for our nonlinear models is crucial for better predictions. This motivated us to use a more robust algorithm for selecting hyperparameters, such as Grid Search and Bayesian Optimization, rather than manually adjusting values. Between these two algorithms, our experiments suggested Bayesian Optimization was more successful at pinpointing the best hyperparameter values. It's capable of searching through a continuous range of values in an optimized manner, which allowed it to discover more optimal values than manual inputs of values to test through Grid Search. As a result, we will be using Bayesian Optimization to find the best hyperparameter values for all of our following nonlinear models.

4. Model Evaluation Metric

To assess the performance of each model in predicting a song's popularity score, we utilize the following evaluation metrics:

- **R^2 Score (Coefficient of Determination):** This metric indicates how well the model captures the variance in the popularity score. An R^2 value of 1 implies perfect prediction, while a value of 0 indicates that the model performs no better than the mean of the target variable. R^2 is especially useful for comparing the explanatory power of different models.
- **Mean Squared Error (MSE):** MSE is the average squared difference between the predicted and actual values. It penalizes larger errors more than smaller ones due to the squaring term, making it sensitive to outliers. This metric is useful for evaluating the overall accuracy of predictions in the regression context.
- **Cross Validation Average:** To make sure that our evaluation is not biased by a specific train-test split, we employ the k-fold cross-validation. This involves dividing the training data into k sections (folds), training the model on k - 1 folds, and validating it on the remaining fold. This process repeats k times, and each fold serves as a validation set once. Then we can take the average of R^2 and MSE across folds to provide a different estimate of the model performance on unseen data.

We choose to use these different metrics because it helps us assess different aspects of model performance: R^2 shows explanatory power, MSE captures the magnitude of prediction errors, and cross-validation can show the generalization of different results.

5. Comparison and Selection Criteria of the Model

After training and evaluating all candidate models using R^2, MSE, and cross-validation averages, we will compare their performances to identify the most suitable model to predict song popularity. The selection of models will be based on a balanced consideration of predictive accuracy, generalization performance, and interpretability.

We prioritize models that:

- Achieve higher R^2 scores, indicating stronger explanatory power and better ability to capture variance in popularity scores.
- Maintain lower MSE, reflecting more accurate predictions and fewer large errors.
- Demonstrate stable cross-validation performance, suggesting that the model is not overfitting and is likely to generalize well to unseen data.

V. Experimental Results and Evaluation

The proposed methodology was implemented on Anaconda v24.9.2 using Python 3.12.7. We coded proposed models stated in our methodology and tested it on our Cleaned_Dataset.csv. For all subsequent models, we use the following predictors, with popularity score as our response variable.

Linear Regression

For linear regression, we obtained a train R^2 and MSE of 0.6040 and 126.78 respectively, while the test R^2 is 127.18 and 0.6076 respectively. The small difference between these error metrics suggests that this model is certainly not overfitting, but we don't have proof whether the model underfits from being too basic yet. To analyze whether our linear model is missing nonlinear behavior in the data, we analyze the following residual plot and Q-Q plot (Fig. 8 & 9).

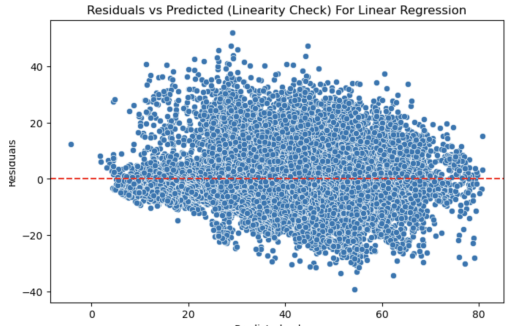

Fig. 8. Residuals vs Predicted From Linear Regression

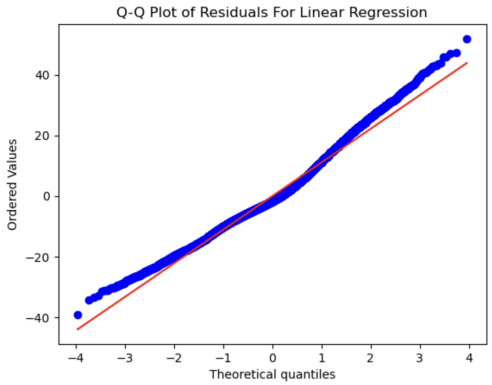

Fig. 9. Q-Q Plot of Residuals From Linear Regression

The residual plot shows points are scattered randomly, and there are no obvious nonlinear curves. However, the Q-Q plot slightly deviates from the red line at the tails of the plot, so some residuals are either much smaller or larger than what we'd expect if the residuals were normally distributed. This indicates that there's a subset of songs whhich our model can't accurately predict, either because they are outliers with unexpected popularity scores with those features or our linear model is too basic. To pinpoint the source of this error, we were motivated to fit more complex nonlinear models.

Decision Tree (Base)

After using Bayesian Optimization on the regular decision tree, the most notable takeaway from the optimal parameter values was that this decision trees was pruned quite aggressively. This is supported by the cost complexity pruning alpha value of about 0.05 and requiring a minimum of 60 samples to split at a node. This setup likely serves to prevent the model from overfitting. For our prediction results, we obtained a train R^2 and MSE of 0.57 and 137.32 respectively, while the test R^2 and MSE were 0.56 and 139.07. Surprisingly, this performed worse than our baseline linear model despite decision trees being able to capture nonlinearity in the data. Additionally, the small difference between the test and train MSE is reasonably small, so our model is still not overfitting. As previously, we can't conclude whether it's underfitting until we see future models performing better. We can also check the feature importances to see how strongly each feature influences splits of our decision tree.

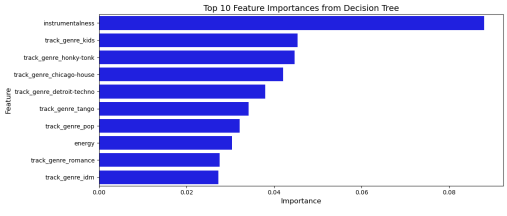

Fig. 10. Top 10 Feature Importances From Decision Tree

(Fig. 10) shows the instrumental feature is consistently providing the highest information gain for the feature to split

on, while all the other top features are just genres. We want to ensure our model is actually using song features and isn't overly tuned to our dataset's specific characteristics, so this is another potential limitation of this model.

Decision Tree (Bagged)

Bagged Forest trains multiple decision trees on different random subsets of the training data and then averaging their predictions to reduce variance and improve generalization. From Bayesian Optimization, we found the best hyper parameters which were 76 estimators with a maximum depth of 30 and a minimum of 4 samples per leaf. This model achieved a Train R^2 of 0.58 and a Test R^2 of 0.52. This indicates that while it did capture over half of the variance in the training data, it also generalized fairly well to unseen data. The Test MSE of 152.84 represents the average squared error between the model's predictions and actual popularity scores, which reflects reasonable accurate predictions given the wide range of song popularity values in the dataset. This shows that bagging helped stabilize decision tree performance and mitigate overfitting.

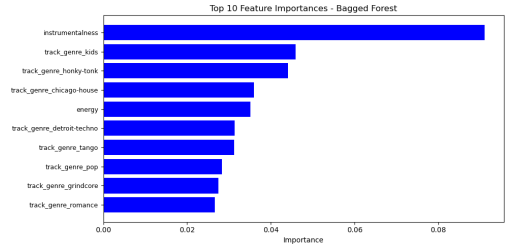

Fig. 11. Top 10 Feature Importance From Bagged Forest

(Fig. 11) shows that while instrumentalness is influencing popularity, the other top features are still genres, which is an issue. We want our model to use song features.

Decision Tree (Random Forest)

It's highly likely that having over 100 binary hot encoded genre features prevented the previous model from learning actual structure in the data. This is why we turned to random forests next, as it will randomly select a subset of predictors for each split. Also, an ensemble of models may yield better predictions than just the output of one model by learning different patterns in the data. For our random forest model, we achieve a train R^2 of 0.73 and a test R^2 of 0.58. This is higher than our previous decision trees' results, yet surprisingly slightly worse than our linear regression results. (Fig. 12) tells us whether our random forest model is overly influenced by the one hot encoded genres as the previous decision trees.

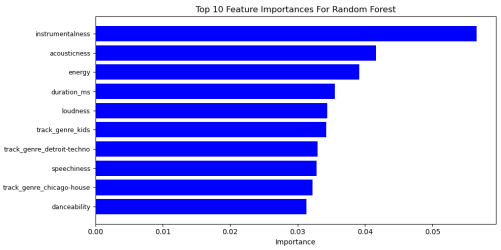

Fig. 12. Top 10 Feature Importance From Random Forest

The top 10 features are now song-specific characteristics (Fig. 12), indicating this model avoids the issue of being overly fixated on track genres, even if its prediction error is typically higher. Since overfitting was still not an issue with any model tested yet, we wanted to assess whether more complex models would yield better predictions.

XGBoost

After training our XGBoost model with one-hot encoded features, we achieved a high training R^2 of 0.92 and a test R^2 of 0.69, with a test MSE of 98.97. This indicates that the model fits the training data very well and generalizes moderately well on unseen data, although the performance gap between train and test scores suggests some degree of overfitting.

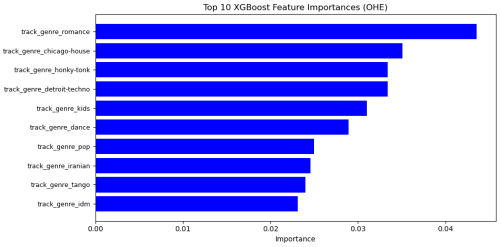

Fig. 13. Top 10 Feature Importances From XGBoost

We also visualized the top 10 most important features in (Fig. 13) based on their contribution to the model's splits. The plot shows that genre-related features play a dominant role in prediction, with track_genre_romance, track_genre_chicago-house, and track_genre_honky-tonk ranking as the top contributors. This is an issue as we don't want our model to predict a song's popularity based off of genres, so we tried LightGBM next.

Winning Model: LightGBM

Our LightGBM model achieved strong performance with a training R^2 of 0.98 and a test R^2 of 0.70, along with a significantly reduced test MSE of 95.07 compared to previous models. The small training MSE of 6.82 and the relatively low gap between training and test metrics suggest that the model generalizes well without substantial overfitting. These results mark LightGBM as our most successful model for predicting song popularity.

The feature importance plot for this model in (Fig. 14) highlights how it leverages actual song features rather than

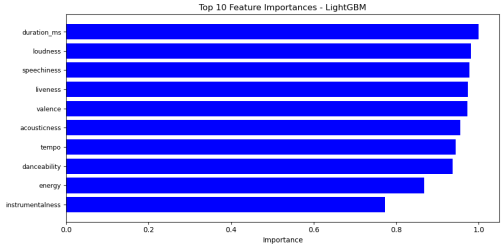

Fig. 14. Top 10 Feature Importances From LightGBM

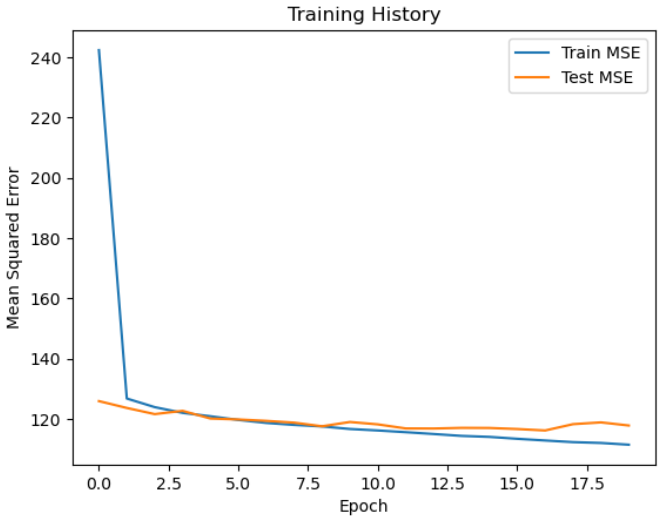

Fig. 15. Training History Of Neural Network

categorical genre information to make predictions. Features such as duration_ms, loudness, and speechiness are among the most influential, indicating the model now captures more nuanced aspects of the audio signals themselves. This shift from genre-dependence to content-based features suggests that the optimized LightGBM model better understands the important qualities that helps a track's popularity.

Neural Networks

Neural networks have extremely strong nonlinear fitting capabilities, but they require a large amount of data to avoid overfitting, and the greater the depth/width of the model, the more likely it is to lead to overfitting. Due to our limited sample size, other large-depth/wide models tend to show obvious overfitting trends, and easily turn to underfitting after regularization. And With low depth/small batches and introducing Bayesian optimization, we obtained a model with the best result.

Several typical problems can be seen from the training curve (Fig. 15). The MSE of the training set dropped sharply in the first few epochs and continued to slowly decrease while the test MSE stayed relatively flat with some small fluctuations. This suggests the model started to slightly overfit even with a shallow architecture. Although we did not train for many epochs, the test performance stopped improving past epoch 10. According to the curve, we limited training to 20 epochs and obtained our best result, the maximum value of R^2 = 0.63 and MSE = 117.49 on the test set.

VI. Conclusion and Discussion

This study aims to use different supervised learning models to analyze the contribution of audio attributes to the popularity of tracks and practice predicting the popularity of tracks. According to past studies, simple linear models have been able to capture a considerable amount of variance, but cannot model more complex nonlinear relationships in the data. So we use linear regression as a baseline (test set $R^2 \approx 0.61$, MSE ≈ 127), and gradually increase the model capacity and regularization.

In a production environment, it is crucial to understand why the model "favors" certain audio features or external signals. Therefore, we also analyze the contribution of different features to the output of different tree models. Neither a single pruned decision tree nor the variance-stabilizing bagging algorithm can bring better performance than the baseline model. Bace on our experiments, for both the base tree and the bagging tree, we observed that the track_genre feature absolutely dominated the output. This phenomenon may be because a large number of binary features are generated after one-hot encoding, which makes the tree model more inclined to choose these dimensions when splitting, resulting in bias. Which can explain the reason that the random forest with feature subsampling effectively balances the contribution of track_genre and other features to the output, providing better results.

Our experiments with gradient boosted trees further demonstrated the importance of eliminating the model's bias toward incorrect features. XGBoost captures most nonlinear features, but its large training-test set gap indicates residual overfitting and severe type dependence. In contrast, LightGBM combines leaf growth with histogram binning, achieving the best overall test performance while keeping the distribution of training and test sets close. Its feature importance curve shifts from one-hot–encoded genre features. to intrinsic audio descriptors (duration, loudness, voice, instrumentality), indicating that its decision surface is more meaningful in terms of musicality.

We also tested a compact neural network to explore the potential of deep models in popularity prediction. As shown in theory, neural networks have a strong ability to fit complex nonlinear patterns, but the premise is that there are sufficient training samples. In our case of about 50,000 records, even if the network width and depth are kept low, the model still shows a clear tendency to overfit. However, even so, after Bayesian optimization adjustment, our neural network model achieved $R^2 \approx 0.63$. Although the amount of data limits the full training of deep models, compact neural networks can still achieve better results than traditional linear regression models due to their ability to fit nonlinear relationships after reasonable regularization and hyperparameter optimization.

Prospect

This study only predicts based on audio signals and basic metadata, but does not involve more in-depth features such as the frequency span of human voices, whether there is human voice distortion caused by audio post-processing, and external signals such as advertising budget that can obviously affect the popularity of music. Introducing these features can better provide guidance for music producers.

Although LightGBM has performed best in this study, AutoML platforms (such as AutoGluon and Optuna) can still be explored in the future for broader model search and integration. Given the limitation of insufficient data, future work could also involve a hybrid architecture combining neural networks and gradient boosting (e.g., TabNet, DeepGBM), and balance prediction accuracy, model complexity, and inference speed using multi-objective Bayesian optimization.

Resources and Roadmap

GitHub Repo: https://github.com/VincentXayasak/Spotify-Song-Popularity-Predictor

Demo: https://drive.google.com/file/d/1R9Ig3bgpHp4_LSSE4HgP7MjVzPTzZALD/view?usp=sharing

Date	Task / Activity
4/24/2025	Assign project roles to each member.
4/25/2025	Collaborate online to write the 1-pager report.
4/29/2025	In-person meeting for dataset understanding and initial EDA. Explore features, target variables, missing values, and distributions. Produce initial plots and summary write-up.
5/6/2025	In-person meeting for deeper EDA and feature engineering. Analyze correlations, handle missing data, encode categorical variables, and refine feature selection. Submit mid-quarter report.
5/13/2025	In-person meeting: Scrape data and clean dataset by removing unnecessary or unwanted features.
5/20/2025	In-person meeting: Fit different model types and tune hyperparameters. Evaluate and compare baseline models.
5/27/2025	In-person meeting: Finalize best model choice and justify selection.
6/3/2025	In-person meeting: Write experimental results and evaluation section of the final report.
6/6/2025	Final in-person meeting: Complete final report and record demo video.
6/9/2025	Submit final report, demo, and source code.

Team Member Tasks

Vincent Xayasak (Group Leader): Held team meetings, managed tasks, contributed to EDA, contributed to model testing, wrote the experimental results and evaluation for XGBoost and LightGBM, wrote the Introduction and Background, created front-end website, and recorded the demo.

Patrick Wang: Contributed to EDA, contributed to model testing, wrote EDA, proposed methodology, and the experimental results and evaluation for neural networks.

Rishi Kumar: Led EDA strategy, identified a critical temporal bias issue causing faulty popularity scores, drove model design choices, wrote the experimental results and evaluation for Linear Regression, Base Decision Tree, and Random Forest.

Zishuo Li: Contributed to EDA report, contributed to model testing, helped write literature review, wrote experimental results and evaluation for ANN, and Conclusion.

Junseong Lee: Contributions to EDA, led research and writing of literature review, proofread and contribution to writing evaluation for ANN.

References

[1] F. Pachet and P. Roy, "Hit Song Science Is Not Yet a Science," in *Proc. Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, 2008.

[2] D. Herremans, D. Martens, and K. Sörensen, "Dance Hit Song Prediction," in *Proc. Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, 2014.

[3] K. Li, "Predicting Song Popularity in the Digital Age through Spotify's Data," *Electronic Workshop in Acoustic Data*, 2024.

[4] R. Nijkamp, "Prediction of Product Success: Explaining Song Popularity by Audio Features from Spotify Data," B.A. thesis, University of Twente, Enschede, The Netherlands, 2018.