Predicting Song Popularity Using Machine Learning Techniques

Aldridge Fonseca
Dept. of Computer Science
San Jose State University
San Jose, CA USA
aldridge.fonseca@sjsu.edu

Sushant Lokhande
Dept. of Computer Science
San Jose State University
San Jose, CA USA
sushant.lokhande@sjsu.edu

***Abstract*—With the rise of streaming platforms like Spotify, the music industry has grown exponentially, yet success remains concentrated among a small fraction of artists. This project investigates the factors influencing song popularity by analyzing audio features alongside textual features like lyrical complexity and sentiment. Using the Spotify 1 Million Tracks and Genius Song Lyrics datasets, six machine learning models were evaluated. XGBoost emerged as the most effective model, achieving 74.28% accuracy in multi-class classification. Binary classification further improved accuracy to 84.42%. These findings showcase the potential of combining diverse features for understanding song success.**

***Index Terms*—Audio and Textual Features, Machine Learning Models, Song Popularity Prediction, Spotify and Genius Datasets**

I. Introduction

Music is an important part of human culture, having the ability to cross borders and unite people through shared experiences. The introduction of streaming services like Spotify, resulted in the music industry growing exponentially and generating billions of dollars in revenue each year. But this wealth is not distributed equally, the top 1% of artists account for over 80% of the industry's revenue.

The phenomenon is often termed as the "superstar effect" stresses on the importance of predicting song popularity. The ability to predict successful songs can lead to great profitability for large and smaller artists alike. It also holds importance for producers and marketers across the industry. Machine learning models and data driven methodologies have made it possible to learn patterns that lead to a song's success.

In this project, we aim to analyse and predict song popularity using both audio and textual features. Audio features like danceability, energy, and acousticness can provide understanding of a songs structure and style. Textual features on the other hand can capture lyrical complexity and sentiment to form a representation of each song.

Two datasets were used in this project, the Spotify 1 Million Tracks Dataset and Genius Song Lyrics Dataset. Both datasets were combined together to find the relationship between various characteristics and popularity. Key challenges that this project addressed include predicting song popularity with the inherent subjectivity of different tracks. The need to preprocess their varied sources of data and the design of machine learning models that can capture the

II. Related Works

The growth of music content on platforms like Spotify creates challenges in predicting song popularity. Several studies have attempted to address this issue by analyzing key features contributing to a song's success.

Mousmi et al. [1] utilized Apache Spark for music genre classification, demonstrating the efficiency of distributed computing frameworks for processing massive datasets. Their work highlighted the importance of genre as a factor in song popularity. However, they did not consider other influential features, such as lyrical content or artist metadata.Incorporating text insights like sentiment analysis and artist collaborations alongside genre classification could lead to better predictive models [1].

Napier and Shamir [2] explored the role of lyrical sentiment in song popularity, finding that positive emotional tones often resonate with listeners and increase audience engagement. Their study revealed that songs with uplifting messages tend to gain visibility through user engagement metrics, such as playlist inclusion and sharing. Despite these findings, their approach focused solely on lyrical sentiment, overlooking the potential to enhance predictive accuracy by integrating audio features and metadata, such as release year or collaborations [2].

Li [3] analyzed audio features from Spotify, including energy, danceability, and speechiness, along with release year. These characteristics were found to influence song popularity. High-energy tracks with rhythm and favorable tempos, such as those in rock or EDM genres, tend to appeal to listeners. However, the study lacked a focus on lyrical content and sentiment, leaving room for improvement by integrating text features with audio analysis. Combining these aspects could provide a more comprehensive understanding of what drives song success [3].

Other studies expanded on these foundations. For instance, a Kaggle dataset study integrated sentiment analysis, profanity detection, and artist popularity, achieved 85% accuracy using Random Forest classifiers [4]. This work showed that combining multiple features beyond audio could enhance prediction

models. Similarly, emotion classification using LSTM-based models achieved 91.08% accuracy by capturing complex dependencies in lyrical content, emphasizing the potential of deep learning approaches [5].

Finally, research on Billboard Top 100 songs examined lyrical complexity, focusing on word count, repetition, and structure. This study found that artist popularity and initial chart position were stronger predictors of success than lyrical complexity alone [6].

The integration of additional metadata, such as collaborations and release periods, was suggested as a way to strengthen predictive models.Collectively, these studies emphasize the need for holistic approaches that combine lyrical, audio, and metadata features. By capturing the interactions among these factors, predictive models can accurately forecast song popularity.

III. Dataset

A. Spotify 1 Million Tracks

The Spotify 1 Million Tracks dataset was sourced from Kaggle. It is one of the largest collections of metadata and audio features of songs available on Spotify. The data was extracted using the Spotipy API and contains information on 1 million songs released between 2000 and 2023. Metadata includes features such as artist names, track titles, and release years. It also contains detailed audio attributes like danceability, energy, tempo, and valence. The dataset provides numerical as well as categorical insights that makes the basis to understand the properties of songs that contribute to their popularity.

With over 61,000 unique artists and 82 genres, the dataset is large and diverse. The audio descriptors such as loudness , speechiness, acousticness and instrumentalness allow the capture of patterns in the structure of a song. Additionally, the dataset includes a metric called popularity score which ranges from 0 to 100. The popularity score that is provided by spotify is calculated using multiple parameters. While the actual algorithm is proprietary, it has been assumed to include streaming frequency, recent listener activity, and track engagement metrics such as saves, shares, and playlist inclusions .The dataset spans over two decades and provides opportunity to analyze the trends and changes in musical styles over time. By including features such as energy and valence change across songs and then map these changes across different popularities we can identify correlations between temporal trends and song popularity.

For our project, this dataset established the relationship between a song's features and its popularity. It helps us both define what is success for a song and also what makes it successful.

B. Genius Song Lyrics

The Genius Song Lyrics Dataset is a dataset that was scraped from the Genius website. The Genius website is a community driven platform where users upload and annotate different works such as songs and poems. This dataset builds upon the 5 Million Song Lyrics Dataset by detecting the native language of each row. The data spans work as recent as 2022, offering a recent collection of songs for analysis.

The dataset contains titles, artist names, release years, views, and lyrics making it a great resource for understanding textual and structural patterns in different songs. The included language annotations in the dataset allow us to focus on specifically English language songs for our project.The tag column in the dataset serves as a flexible filter that categorizes songs into genres such as pop, rock and rap. Non-musical pieces such as 'misc' can be easily identified using the tag column as well.

The unique feature of this dataset is its textual content in the form of song lyrics. Unlike numerical datasets, song lyrics require greater preprocessing due to its varied and unstructured format. By analyzing the lyrics we can identify patterns in language, word choice and structure and how they might influence popularity. This is an incredibly critical factor as lyrics play a pivotal role in the success of a song especially in genres like rap and pop.

The dataset also contains a views lyrics page views column which represents the number of times a song's lyric page has been viewed on the Genius website. While this metric is much smaller as compared to the number of streams on Spotify, it can serve as a strong indication of the audience's interest in a song. A higher count might suggest that the listeners are actively engaging with a song or deeply connecting with it. This behaviour can precede a song's rise to popularity, making it valuable to gain insights into listener engagement that may not be captured only by metrics of streaming.

IV. Data Preprocessing

Data preprocessing is a key component in any machine learning or data-driven project as it ensures that the resulting data is clean and structured. For this project, we processed two datasets: Spotify 1 Million Tracks and Genius Song Lyrics.

A. Preprocessing Spotify Dataset

The Spotify dataset was extracted using the Spotify APIs, resulting in it being relatively clean and already structured. The following steps were performed to ensure consistency and data quality:

- Handling missing values: Rows that contained null or missing values were identified and dropped.
- Normalization: Certain columns were normalized to ensure consistency given the large amount of features that were present.

B. Preprocessing Genius Dataset

The Genius dataset, in comparison to the Spotify dataset, required a large amount of preprocessing due to its unstructured format, irregularities in the lyrics, and additional metadata. The Genius dataset initially contained over 5.1 million records. After preprocessing, the dataset was reduced to approximately 3.2 million records.

1) Filtering and Cleaning Rows: The dataset initially contained a lot of non-music entries and incomplete records that needed to be removed for any meaningful analysis. Specific filters were applied to remove:

- Tracks with non-English lyrics.
- Rows with missing tags or those tagged as miscellaneous.
- Instrumental songs with no lyrics, as these could not be used in our analysis.

2) Lyric Cleaning and Standardization: The raw lyrics data in Genius was incredibly messy and required significant cleaning to make it suitable for analysis. The following steps were undertaken:

- Lyrics often included special characters such as square brackets, parentheses, or newline indicators that disrupted analysis. Regular expressions were applied to replace these characters with whitespace.
- Punctuations were removed to prevent them from being treated as unique tokens during analysis.
- Slang words like 'lovin'' were replaced with their formal forms, such as 'loving', to ensure consistency.
- All lyrics were converted to lowercase to standardize the data and avoid case sensitivity issues.

3) Derived Features Extraction: To extract more meaningful insights, derived features were generated from the cleaned lyrics:

- Word Count: The total number of words in the song's lyrics was calculated, excluding section labels such as '[Verse 1]' or '[Chorus]'.
- Number of Parts: The occurrences of sections such as '[Verse 1]' or '[Chorus]' were counted.
- Unique Word Count: The number of unique words in each song's lyrics was computed after removing section labels and special characters.
- Presence of Specific Sections: Binary indicators were added to determine if the song had specific sections such as 'has_chorus', 'has_intro', 'has_outro', or 'has_bridge'.

4) Sentiment Analysis: Sentiment analysis was applied to the cleaned lyrics to understand the emotional tone of each song. The TextBlob library was used to compute the following sentiment scores:

- Sentiment Polarity: A continuous score ranging from -1 to 1 representing the overall sentiment of the lyrics.
- Sentiment Categories: Lyrics were categorized into three sentiment types based on the polarity score:
  - Negative Sentiment: Polarity score < 0.
  - Neutral Sentiment: Polarity score = 0.
  - Positive Sentiment: Polarity score > 0.

5) Additional Textual Features: Further analysis of the cleaned lyrics was performed to give two additional features

- Average Word Length: Calculated as the mean length of words in the lyrics, excluding special characters and section labels.
- Stopword Count: The number of common stop words such as 'the', 'is', 'and' was counted using the NLTK library.

C. Merging the Spotify and Genius Datasets

The Spotify and Genius datasets were merged to gain insights from both sources. The direct integration posed a challenge as the unique identifier used by Spotify and Genius were different. The Spotify Dataset uses a track ID , while the Genius dataset did not include this identifier. Therefore a join operation was performed on the artist name and track name after standardizing their format.

Columns representing the artist and tack names in both the datasets were renamed to maintain consistency. Both the columns were converted to lowercase and stripped of leading and trailing whitespaces to prevent any misses due to formatting inconsistencies. An inner join was performed on the cleaned artist name and track name columns.

After the merging process the combined dataset consisted of approximately 140,000 rows of data across genres and spanning many years. This data containing audio attributes as well as textual features was able to provide a robust foundation for the models trained.

V. Exploratory Data Analysis

TABLE I
KEY FEATURES USED IN THIS PROJECT

Feature Name	Description
track_name	Song title
artist_name	Name of the artist
lyrics	Song lyrics
popularity	Song popularity (0–100)
genre	Music genre
sentiment	Overall sentiment score
positive_sentiment	Binary value: positive sentiment
negative_sentiment	Binary value: negative sentiment
neutral_sentiment	Binary value: neutral sentiment
lyric_page_counter	Number of views on lyric page
word_count	Total word count in lyrics
num_parts	Number of distinct song sections
year_y	Release year of the track

A. Data Overview

The newly fabricated and combined dataset consisted of 139,433 records and 39 Columns. Each row represented a song which was defined by a variety of attributes that captured lyrical and non-lyrical aspects. The features included song metadata such as the track name, artist name, year of release, and genre. The dataset also consisted of numerical audio features acousticness, loudness, duration of the song, and danceability. All these features give an idea on the musical variety of the track.

B. Popularity Analysis

The average popularity score across all tracks is about 31, with a standard deviation of about 15. The popularity distribution, shown in Fig. 1, shows a skewed distribution,with most tracks between 20 and 40.

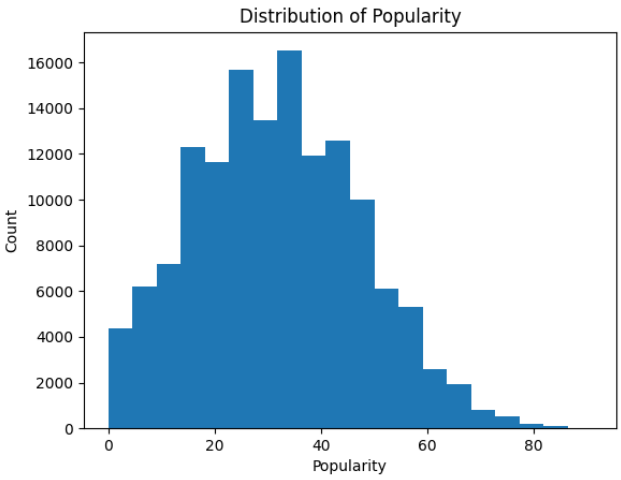

Fig. 1. Distribution of Popularity

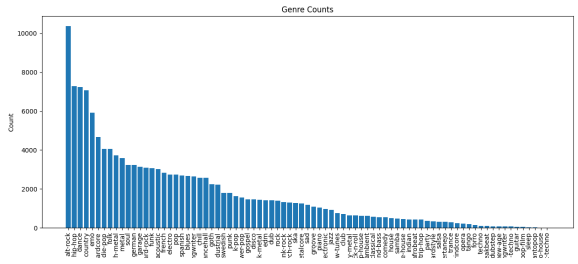

Fig. 2. Genre Counts

C. Genre Distribution

The dataset includes 81 unique genres, with the highest representation being from alt-rock , hip-hop , and dance . A bar chart shows genre distribution is presented in Fig. 2

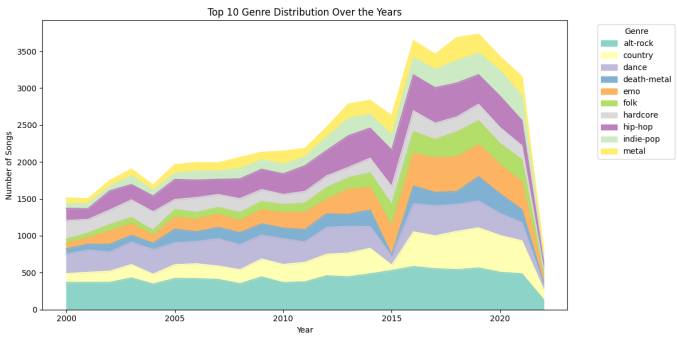

Fig. 3. Genre Distribution Over the Years

D. Temporal Trends

To explore the year on year trends, the top 10 genres by track count were plotted over the years 2000–2023. Fig. 3, revealed trends such as the rise of hip-hop and the constant popularity of alt-rock. Additionally, average track popularity has increased steadily, peaking in 2023 with a score of 54 as shown in Fig. 4.

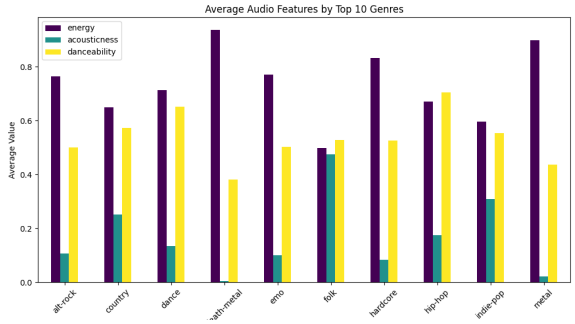

Fig. 4. Average Song Popularity Over Years

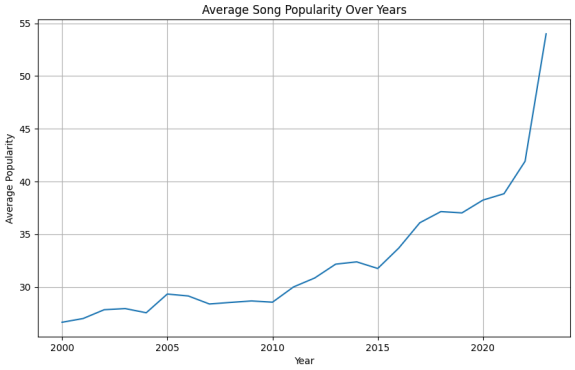

Fig. 5. Audio Features by Genre

E. Audio Feature Trends

The top 10 genres were compared based on audio features: energy, acousticness, and danceability. Death metal had high energy while folk music had a higher acoustic score. These trends are plotted in Fig. 5.

F. Lyrical Complexity

Analyzing lyrical complexity, which is measured by unique word count can highlight the variability among genres. Hip-hop exhibited the highest average complexity, while folk and alt-rock had lower averages. A box plot showing this variability is shown in Fig. 6.

VI. Feature Engineering

A. Target Variable Transformation

The primary objective of this project was to predict the popularity of the songs. Initially, the variable popularity was represented as a numerical value ranging from 0 to 100. Predicting a numerical target variable directly through regression based models was explored but it had biased results. The reason behind was the skewed distribution of popularity scores.

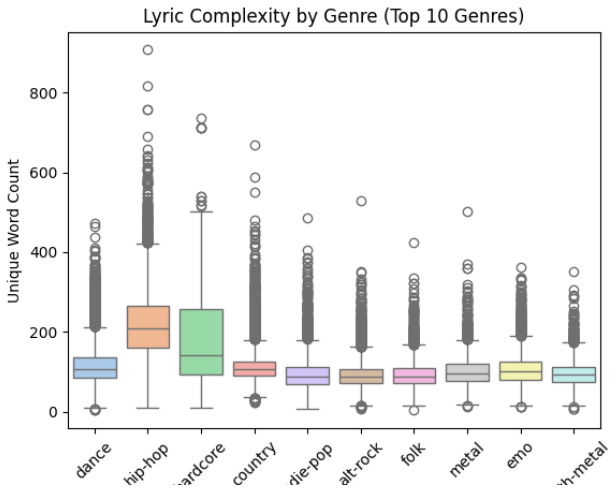

Fig. 6. Lyric Complexity by Genre

Most of the songs were classified into neutral range. Very few songs reached high popularity.

To overcome this, the target variable was transformed into a classification problem. The popularity scores were grouped into distinct classes. This allowed us to utilize classification models as classification models are better suited at handling categorical targets. Following approaches were evaluated to determine the most effective strategy for assigning a class to the popularity variable.

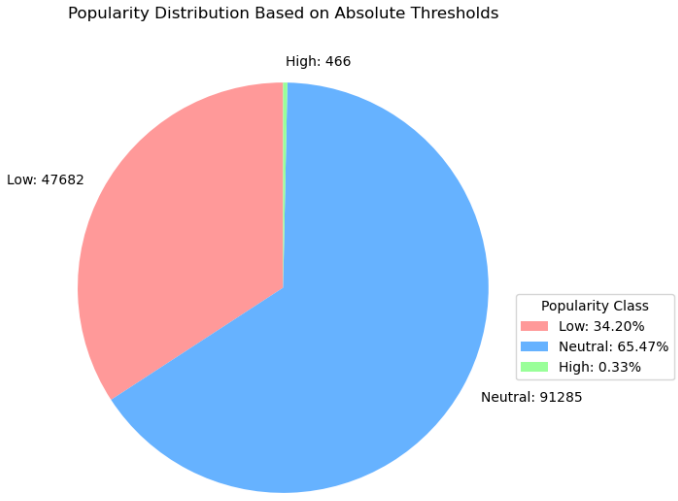

Fig. 7. Popularity Distribution Based on Absolute Thresholds

1) Absolute Threshold Classification: In the initial attempt, fixed thresholds were defined:

- Low Popularity: Scores below 25
- Neutral Popularity: Scores between 25 and 75
- High Popularity: Scores above 75

While this method was straightforward , it resulted in an extremely imbalanced dataset. The class distribution was highly dominated by the "Neutral" Category. Only 0.33% of the songs were classified as "High" Popularity, whereas the majority fell into the neutral category. This imbalance can impact the ability of models to generalize effectively. Hence this approach was not implemented.

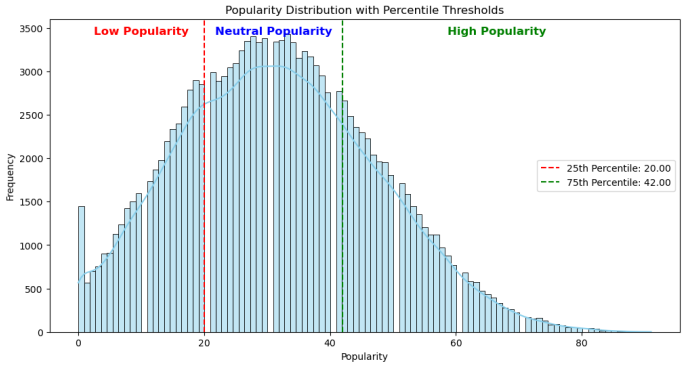

Fig. 8. Popularity Distribution Based on Percentile Thresholds

```
popularity_class
1    68914
0    35808
2    34711
Name: count, dtype: int64
```

Fig. 9. Distribution of Popularity Classes in Percentile Based Approach

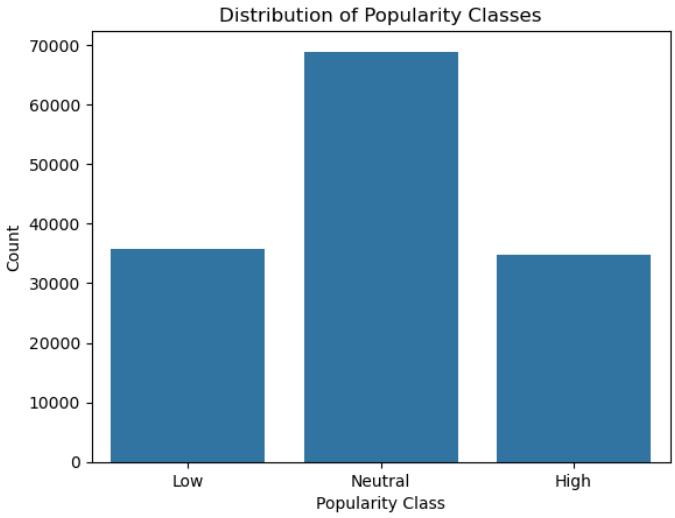

Fig. 10. Distribution of Popularity Classes

2) Percentile Based Classification: To tackle the imbalance issue, a percentile-based approach was adopted. Here, the

popularity score was divided based on quartiles. For this the following logic was used:

- Low Popularity (Class 0): Bottom 25% of popularity score
- Neutral Popularity (Class 1): Middle 25% of popularity score
- High Popularity (Class 2): Top 25% of popularity score

The key advantages of this method were that we achieved a semi-balanced class distribution with 25% of the songs categorized as “Low”, 50% as “Neutral”, and 25% as “High”. This method focused on using relative popularity instead of absolute popularity. This ensured that the classification has the inherent distribution of the dataset.

This distribution strategy was the prime source for six different models training and to get the best performing ML model.

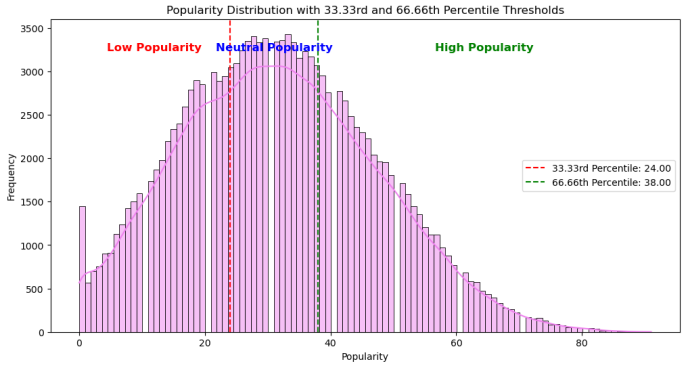

Fig. 11. Popularity Distribution with 33rd and 66th Percentile

```
popularity_class
0    47682
1    45890
2    45861
Name: count, dtype: int64
```

Fig. 12. Distribution of Popularity Classes in Equal Percentile Based Approach

3) Equal Split using Probability Based Classification : The Equal Split approach was applied to balance the dataset by creating an equal distribution across the three classes. The reasoning behind this method was to address the class imbalance issue observed in previous attempts, ensuring that each class had a similar number of samples.For this the following logic was used:

- Low Popularity (Class 0): Bottom 33% of popularity score
- Neutral Popularity (Class 1): Middle 33% of popularity score
- High Popularity (Class 2): Top 33% of popularity score

By equally distributing the samples, the model is less biased toward the majority class, leading to better generalization. This balanced approach usually helps the model learn effectively from each class. With an equal number of samples in each class, the model is less likely to make predictions based on the skewed representation of the data.

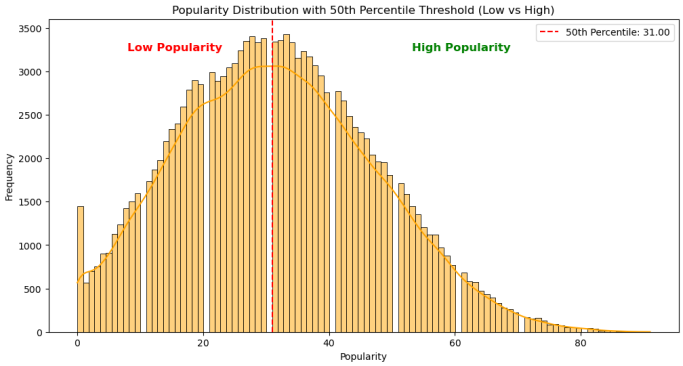

Fig. 13. Popularity Distribution with 50th Percentile

```
popularity_class
0    70819
1    68614
Name: count, dtype: int64
```

Fig. 14. Distribution of Popularity Classes in Binary Split

4) Binary Classification: While the percentile-based classifications made the dataset balanced, a simpler binary classification approach was also tested. Here, the dataset was split into two classes:

- High Popularity: Top 50
- Low Popularity: Bottom 50

This approach created a balanced split. This makes it easier to interpret the results while improving the model performance All approaches offered unique insights into the target variable’s distribution.

B. Feature Selection

The process of feature selection is important to ensure that models focus on the most relevant attributes. The combination of domain knowledge and exploratory data analysis guided the selection process. The selected features were grouped into the following primary categories.

1) Lyrical Features: Lyrical features were used to capture the textual complexity and structure of the song lyrics. The word count represents the total number of words in lyrics. This represents the lyrical density. The number of parts gave the count of distinct lyrical sections such as verses and choruses. The Lyric Page Counter is a measure of the views of the song by people on the Genius website.

2) Sentiment Features: Sentiment analysis was performed on the lyrics to measure the emotional tone of the songs. Based upon the sentiment analysis the new features were added using One-Hot Encoding technique. The engineered features were positive sentiment, negative sentiment and neutral sentiment.

3) Encoded Categorical Features: Certain Categorical attributes were converted into numerical formats to ensure compatibility with Machine Learning models. Artist Name was encoded using label encoding and Genre was also transformed to numerical values.

4) Temporal Features: The feature year of release was included to account for the temporal evolution of song popularity, as trends and listener preferences often change over time.

By combining these features, the models were trained on a diverse set of inputs. These included lyrical content, emotional tone, categorical metadata, and temporal trends. This was done so that models could make accurate predictions.

C. Data Scaling and Splitting

To prepare the selected features for model training, standardization was applied to ensure that all features were on a similar scale. This is important when working with features that have a variety of ranges such as word counts, sentiment scores and numerical encodings.

After feature scaling, the dataset was split into training and testing subsets to evaluate models performance. The split was conducted where 80

VII. Model Selection

In the initial phase of this project, we trained multiple machine learning models on the first classification approach with 25-50-75 popularity thresholds. This approach divided songs into three classes Low, Neutral, and High Popularity which allowed us to explore model performance across an imbalanced, multi-class problem.A combination of traditional machine learning algorithms, ensemble methods, and deep learning models was chosen to capture both linear and non-linear relationships in the dataset.Models were selected not only for their predictive power but also for their ability to provide interpretable insights into the features contributing to song popularity. The models selected for this phase are explained in detail below:

A. Random Forest Classifier

Random Forest is an ensemble learning algorithm that combines multiple decision trees to make robust and generalized predictions.Given the diverse nature of our features which had lyrical, sentiment, numerical, and categorical features, Random Forest was an ideal choice as it works well with mixed feature types.It provides feature importance scores, which are valuable for identifying which features contribute most to predicting song popularity. For example, it can help determine if tempo or positive sentiment has a stronger influence on classifying songs as "High Popularity." Random Forest served as our baseline ensemble model and was expected to perform well due to its versatility across a variety of datasets.

B. K-Nearest Neighbors (KNN)

KNN is a simple algorithm that classifies data based on the majority vote of its nearest neighbors in the feature space. While it is not typically used for large datasets due to computational overhead, it provides an understanding of how songs with similar features are grouped. The inclusion of KNN allowed us to explore a clustering-based approach to classification. KNN's simplicity made it a useful benchmark to evaluate the performance of more complex models. By testing KNN, we could observe whether songs with similar attributes consistently fell into the same popularity class.

C. Logistic Regression Classifier

Logistic Regression is a widely used linear classification model that predicts the probability of a class label. It is efficient and performs well on datasets where the relationship between features and the target is linear. Logistic Regression served as our baseline linear model to assess whether simple linear relationships existed between features and the popularity classes.The model's simplicity enabled faster training and provided a benchmark for comparing the performance of more advanced methods like ensemble and deep learning models.Logistic Regression's inclusion ensured that we had a reference point for linear models, offering a clear contrast to non-linear methods such as Random Forest and deep learning approaches.

D. XGBoost Classifier

Extreme Gradient Boosting is an ensemble method based on gradient boosting. It builds multiple weak learners sequentially, with each tree attempting to correct the errors of the previous one. XGBoost was chosen for its ability to handle imbalanced classes, which was particularly relevant for our first approach, where the "High Popularity" class was significantly underrepresented. By incorporating XGBoost, we aimed to explore the benefits of boosting algorithms in capturing subtle patterns in the data that simpler models like Logistic Regression or KNN might miss.

E. Multi-Layer Perceptron (MLP)

Multi-Layer Perceptron (MLP) is a type of feedforward neural network capable of capturing complex, non-linear relationships between input features and the target variable. Deep learning models like MLP are well-suited for datasets with patterns that traditional machine learning methods may fail to identify .By including MLP, we aimed to determine whether deep learning could uncover hidden patterns within the dataset that were not apparent using traditional approaches. Its inclusion allowed us to explore the potential advantages of deep learning for predicting song popularity.

F. Linear Support Vector Machines (SVM)

Support Vector Machines (SVM) are effective classification algorithms that find the optimal hyperplane to separate classes in the feature space. Linear SVM assumes that the classes are linearly separable and offers a computationally efficient

solution. Linear SVM was chosen for its simplicity and interpretability in solving multi-class problems. Given that we transformed the target variable into discrete classes (Low, Neutral, High Popularity), Linear SVM provided a quick and efficient method for classifying songs while serving as a useful benchmark. SVM performs well on datasets with clear class boundaries, and testing it allowed us to evaluate whether the popularity classes were linearly separable in the feature space.

VIII. Evaluation and Results

Models	Accuracy	Precision	Recall	F1 Score
Random Forest	71.61%	0.73	0.69	0.71
K-Nearest Neighbors	64.19%	0.65	0.63	0.63
Logistic Regression	66.99%	0.69	0.64	0.66
XGBoost	**74.28%**	**0.75**	**0.74**	**0.74**
Multi-Layer Perceptron	68.75%	0.70	0.67	0.68
Linear SVM	66.93%	0.69	0.66	0.64

Fig. 15. Evaluation Table

In the evaluation phase we assessed the performance of various models and identified the most effective solution for our case. The key metrics used were Accuracy, Precision, Recall, and F1 Score. Also Classification Reports and Confusion Matrices were also tested on some specific models. The models were initially trained using the first classification approach based on thresholds of 20-50-75%. The results of this approach are shown in the following approach

XGBoost was the most effective model with the highest accuracy of 74.28% along with the best precision, recall and F1 scores. The result confirmed that XGBoost's ability enabled it to generalise to the dataset. Random Forest also demonstrated strong performance with an accuracy of 71.61% but it falls short compared to XGBoost. Multi-Layer Perceptron performed decently with an accuracy of 68.75%. Logistic Regression and Linear SVM are relatively simpler models. These delivered similar results with accuracies 66.99% and 66.93%, respectively. This suggests that linear models were not fully capable of capturing the complex variations in the dataset. KNN exhibited the weakest performance. This model achieved an accuracy of 64.19%. This indicated that clustering based approaches struggled with the nature of the dataset. This can be caused by the high dimensionality and overlapping classes.

A. Detailed Analysis of XGBoost

A deeper evaluation of the XGBoost model was conducted to get a better understanding of the model's strengths and weaknesses. The XGBoost model gave good results across all key metrics. However, there were some minor inconsistencies and misclassifications particularly in distinguishing between Neutral Popularity and Low Popularity classes.

The confusion Matrix provided further insights into the misclassifications. It was observed that a portion of Neutral

```
XGBoost Accuracy: 0.7428
              precision    recall  f1-score   support

           0       0.77      0.72      0.74      7199
           1       0.72      0.80      0.75     13700
           2       0.78      0.66      0.72      6988

    accuracy                           0.74     27887
   macro avg       0.76      0.73      0.74     27887
weighted avg       0.75      0.74      0.74     27887
```

Fig. 16. Classification Report of XGBoost on Initial Approach

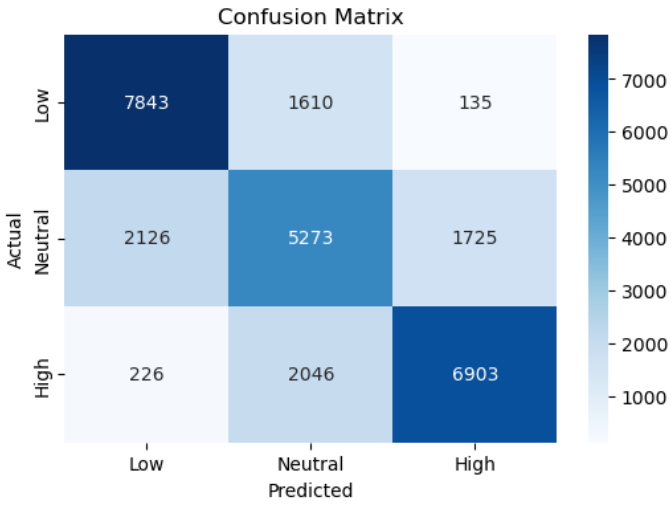

Fig. 17. Confusion Matrix of XGBoost on Initial Approach

samples were misclassified as high and similarly, a subset of Low Popularity was misclassified as neutral. These observations suggested that the boundaries between the two classes were not clearly defined. This could be possible due to the overlapping features in the dataset.

B. Hyper Parameter Tuning Experiment

To further optimise the XGBoost model, we conducted hyperparameter tuning by adjusting to the key parameters such as Learning rate, Maximum tree depth and Number of boosting estimators. Minor improvements were observed in the process but the overall progress was not significant.

C. XGBoost on Balanced Dataset

We applied the XGBoost model on the created balanced dataset with 33% distribution of popularity in each class. Here the dataset was balanced and had an equal number of samples. The rationale behind this step was that reducing the imbalance would enable the model to differentiate the three classes more effectively and to avoid overfitting to the majority class with "Neutral" popularity.

However, the results were unexpected. After training XGBoost on the balanced dataset, the model's accuracy dropped to around 71% from the original 74.28%. While class balance was achieved, the loss of data reduced the ability of the

XGBoost Accuracy: 0.7179

	precision	recall	f1-score	support
0	0.77	0.82	0.79	9588
1	0.59	0.58	0.58	9124
2	0.79	0.75	0.77	9175
accuracy			0.72	27887
macro avg	0.72	0.72	0.72	27887
weighted avg	0.72	0.72	0.72	27887

Fig. 18. Classification Report of XGBoost on Balanced Dataset

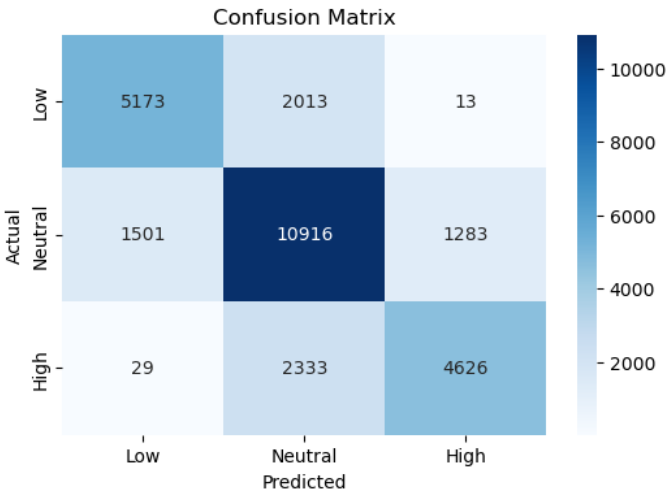

Fig. 19. Confusion Matrix of XGBoost on Balanced Dataset

model. This revealed a paradox that while balancing classes theoretically improves fairness, it can also negatively impact the model's overall performance.

D. XGBoost on Binary Distribution

XGBoost Accuracy: 0.8442

	precision	recall	f1-score	support
0	0.84	0.86	0.85	14177
1	0.85	0.83	0.84	13710
accuracy			0.84	27887
macro avg	0.84	0.84	0.84	27887
weighted avg	0.84	0.84	0.84	27887

Fig. 20. Classification Report of XGBoost on Binary Dataset

We also engineered a distribution with a 50-50 split for classifying popularity into two categories namely High and Low. Training XGBoost on this binary task generated a high performance boost. The model achieved an accuracy of 84.42% and all evaluation metrics showed substantial improvement. The Classification Report and Confusion Matrix indicated a significant reduction in misclassifications with the

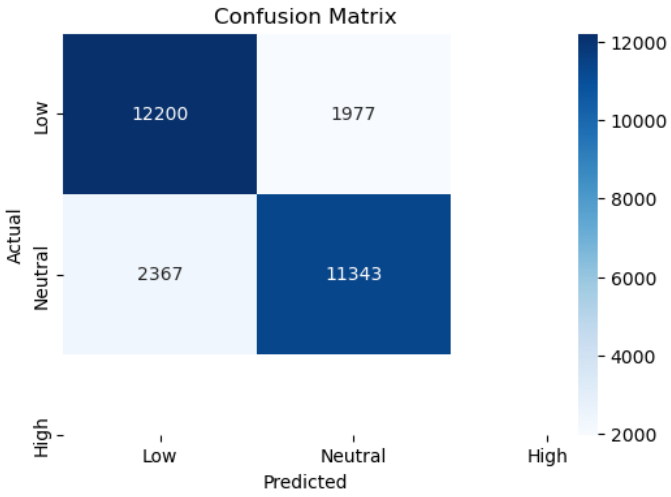

Fig. 21. Confusion Matrix of XGBoost on Binary Distribution

model showing great ability to distinguish between the two categories.

IX. Analysis

A. Understanding XGBoost Performance

XGBoost consistently gave best results in both accuracy and other performance metrics. It performs on this dataset because XGBoost is an advanced implementation of the gradient boosting framework. It combines the strength of multiple weak learners into a strong model. It is also excellent at handling complex relationships in the data specifically in cases like our dataset where features and targets likely have dependencies. XGBoost also incorporates L1 and L2 Regularization to reduce overfitting. Hence, It was the best model for our project.

B. Accuracy Drop with Class Balancing

The paradox can be explained by examining how balancing the dataset impacted the model. With a balanced dataset, the number of samples in each class became very close, which made it more difficult for the model to establish clear decision boundaries. In the imbalanced approach, the model relied on the majority class for accurate predictions, which helped achieve better overall performance.

However, in the balanced version, this reliance was removed. This exposes the model's inability to distinguish clearly between overlapping categories like Neutral and Low Popularity. The nature of the dataset inherently has some overlap between categories. Balancing the classes did not address this fundamental issue, as the features remained similar for these classes, leading to misclassifications.

Thus, while balancing classes is generally expected to improve fairness in predictions, it paradoxically caused a performance drop here because the model struggled to identify clear decision boundaries with closely split classes.

C. Improved Performance on Binary Classification

When trained on a binary version of the dataset, XGBoost achieved a significant performance improvement, with accuracy increasing to 84.42%. This can be studied using proper reasoning. Binary classification is simple because the model only needs to differentiate between two classes instead of three. This reduces the complexity of decision boundaries, enabling the model to focus more clearly on distinguishing between High Popularity and Non-High Popularity categories. By merging the Neutral and Low Popularity classes, we effectively reduced the ambiguity and overlap between categories. This allowed the model to achieve better separation and fewer misclassifications, as the remaining two classes were more distinct.

Combining two classes provided the model with a larger and more balanced sample size for the Non-High Popularity category. Binary classification allowed the model to focus specifically on identifying patterns associated with the High Popularity class. As a result, the precision, recall, and F1-score for the High Popularity category improved significantly, contributing to the overall accuracy boost.

X. Future Work

While this project successfully evaluated multiple Machine Learning models and uncovered insights into song popularity prediction, there are other areas for further exploration and improvement. Future work can integrate more advanced techniques and expand the scope of analysis for predicting trends in song popularity.

A. Integrating Deep-learning and text processing models

Incorporating deep learning techniques with text preprocessing models can help achieve a deeper and understanding of the relationship between lyrical content and song popularity. Leveraging BERT to capture both the context and semantic meaning of words within a given text can help understand how certain phrases, themes, and word choices influence a song's popularity metrics.

B. Lyric Engineering

Future work will also explore the development of features derived from BERT-based embeddings to improve the predictive powers of our models. The combination of BERT with popularity metrics can create lyrical fingerprints of songs. This can help predict their performance on music charts directly. Lyric engineering can also help to identify themes, genre, and word usage to understand their consumers and listeners.

C. Weekly and Monthly Chart Analysis

By analyzing weekly and monthly top charts like Billboard Hot 100, Spotify Top 50, we can study the placement of songs on these charts change over time. This could help understand whether certain songs exhibit short term spikes like viral hits in popularity or long-term staying power with slow success. This can also help in identifying patterns of season that impact a song's success such as holidays and cultural events.

D. Trend Analysis

Trends like shifts in genre preferences, lyrical diversity, and artist collaborations can be analysed for determining the song success. For a more holistic approach, social media trends, streaming platforms, recommendation systems, and curated playlists can also be analysed. This could help us understand how certain songs dominate listener preferences over time.

XI. Conclusion

The project explained the effect of integrating audio and lyrical features to predict song popularity. Data preprocessing and feature engineering is necessary to ensure quality and extract insights. Preprocessing steps included cleaning lyrics and deriving new features such as sentiment scores and word counts. The merging of the Spotify 1 Million Tracks and Genius Song Lyrics datasets allowed for the combination of diverse attributes, enhancing the richness of the analysis.Out of all the machine learning models tested, XGBoost was the most effective for multi-class classification as this model achieved 74.28% accuracy. Binary classification simplified the problem, increasing accuracy to 84.42%. These results highlight the potential of engineered features and advanced models in addressing the complexities of song popularity prediction. This study provides valuable insights for artists, producers, and marketers, showcasing how well-processed data and diverse attributes can reveal patterns of success in the evolving music landscape.

References

[1] M. C. Mousmi *et al.*, "Large-Scale Music Genre Analysis and Classification Using Machine Learning with Apache Spark," *Electronics*, vol. 11, no. 6, p. 2567, 2022.

[2] K. Napier and L. Shamir, "Quantitative Sentiment Analysis of Lyrics in Popular Music," *Journal of Popular Music Studies*, vol. 30, no. 4, pp. 161–176, 2018.

[3] K. Li, "Predicting song popularity in the digital age through Spotify's data," *Theoretical and Natural Science*, vol. 39, pp. 68–75, 2024.

[4] J. Kamal, P. Priya, M. R. Anand, and G. R. Smith, "A Classification-Based Approach to the Prediction of Song Popularity," in *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, Chennai, India, 2021, pp. 1–5.

[5] A. Ara and R. V, "A Study of Emotion Classification of Music Lyrics Using LSTM Networks," in *2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)*, Lalitpur, Nepal, 2024, pp. 126–131.

[6] Y. Gao, J. Harden, V. Hrdinka, and C. Linn, "Lyric Complexity and Song Popularity: Analysis of Lyric Composition and Repetition Among Billboard Top 100 Songs," Paper 11500-2016, Oklahoma State University, 2016.