Music Success Classification
Project Details

1. Project Overview

This project aims to predict the popularity of a song based on its lyrics using advanced Natural Language Processing (NLP) and machine learning techniques. By analyzing lyrical content, the model will classify whether a song is likely to be a hit or not.

1.1 Business Objective:

1. Supporting Music Producers & Record Labels

- Helps production houses and music labels decide whether to invest in a song before production.
- Determines the level of promotion needed for maximizing returns on a song.

2. Enhancing Marketing & Promotion Strategies

- Predicts the likelihood of a song becoming a hit, allowing for optimized marketing efforts.
- Suggests adjustments in lyrics based on past hit patterns to improve success rates.

3. Personalization & Recommendation Systems

- Can be integrated into music streaming platforms (Spotify, Apple Music) to refine song recommendations.
- Aids in creating AI-powered personalized playlists based on lyrical themes.

4. Reducing Financial Risk

- Reduces the uncertainty in song investments by providing data-driven insights.
- Helps independent artists and record labels focus on songs with high hit potential.

2. Dataset Overview

Kaggle Link :

https://www.kaggle.com/datasets/thedevastator/billboard-hot-100-audio-features

The dataset, sourced from Kaggle, contains detailed information about songs that have appeared on the Billboard Hot 100 charts. It provides metadata, including the song title, artist, release year, and chart performance, along with audio features extracted from Spotify. These features include acousticness, danceability, energy, instrumentalness, liveness, loudness, speechiness, tempo, and valence.

The dataset is particularly useful for analyzing trends in song success, as it contains:

- Popularity Scores: An indicator of how well a song is performing, derived from Spotify's popularity metric, which takes into account play counts and recency of listens.
- Weekly Chart Positions: A track's movement on the Billboard Hot 100 over time, reflecting its peak rank, consistency, and longevity.

2.1 Defining Success:

Definition 1:

To objectively determine the success of a song, the dataset is aggregated at the SongID level, considering three key metrics:

1. **Average Weeks on Chart:**
   This measures how long, on average, a song remains on the Billboard Hot 100. A higher value indicates sustained popularity and a lasting impact.

2. **Average Spotify Popularity Score:**
   This provides an indicator of the song's continued streaming performance on Spotify. A higher value suggests that the song

remains relevant and widely listened to.

3. **Best Weekly Position (Min Week Position):**
   This captures the peak ranking a song has achieved on the Billboard Hot 100. Lower values (closer to 1) indicate that the song reached a higher chart position.

Interpreting the Success Metrics

- **Highly successful songs** tend to have:
  - **High average weeks on the chart** → Long-term presence in the industry.
  - **High Spotify popularity scores** → Active streaming even beyond initial release.
  - **Low best weekly position (closer to 1)** → Strong chart performance.
- **One-hit wonders or viral trends** may:
  - Have a high peak position but low weeks on chart, indicating a short-lived spike.
  - Show moderate popularity scores if streaming trends decline after initial hype.

Success in the music industry is subjective and can be measured using multiple factors, but for this project, popularity and weekly chart positions serve as primary indicators.

Using Billboard chart entry as a prerequisite for success has several limitations:

1. **Bias Toward Already Charted Songs:**
   - This definition excludes songs that were commercially or culturally successful but never entered the Billboard Hot 100.
   - Many successful songs gain traction through streaming platforms, social media, or niche markets but might never make it to the Billboard charts due to industry gatekeeping, marketing efforts, or genre preference biases.

2. **Underestimating Non-Hit Songs with Longevity:**
    - Some songs might debut on the Hot 100 but drop off quickly, while others may not peak high but stay on the chart for months.
    - This definition tends to favor songs with a high peak ranking over those with steady long-term appeal.

Definition 2:

To address these limitations, success can also be defined purely by the song's average Spotify popularity score. This method has the following characteristics:

1. **Includes Non-Charting Hits:**
    - Unlike Billboard-based success, this definition considers songs that never charted but are widely streamed.
    - For example, an indie song with a Spotify popularity score of 80+ might be just as successful as a Billboard No. 1 hit.
2. **Reflects Modern Listening Trends:**
    - Streaming better represents ongoing user engagement and song relevance, whereas Billboard rankings often weigh initial industry push.
    - This approach removes the bias of radio play and physical sales, making it more relevant in the digital age.
3. **May Overestimate Viral Trends:**
    - A short-lived viral song (e.g., TikTok trends) may achieve a high popularity score but fade quickly.
    - Without considering chart longevity, this metric alone may not fully capture long-term impact.
4. **Does Not Capture Classic Longevity Well:**
    - Older songs often see declining Spotify popularity but remain iconic (e.g., a 90s hit with a low popularity score today).
    - A song that was a massive hit decades ago may not get enough streaming traction to register as "successful" in this model.

Definition	% Hits	% Non Hits
Success 1	9.17	90.83
Success 2	35.58	64.42

Why We Chose Definition 2: Average Spotify Popularity as the Measure of Success

After evaluating both definitions, we opted to define success based on average Spotify popularity rather than relying on Billboard chart performance. This decision is driven by several key reasons:

1. Broader Inclusion of Successful Songs

Definition 1 only considers songs that have entered the Billboard Hot 100, which introduces bias against successful songs that never charted.

- Many independent, viral, or niche songs gain widespread recognition without ever appearing on Billboard.
- Billboard rankings are influenced by radio airplay, label promotions, and industry factors, which may not always reflect true audience-driven success.
- Streaming platforms like Spotify provide a more direct representation of real-time user engagement.

By using Spotify popularity, we ensure that both mainstream charting songs and organic streaming hits are included in the definition of success.

2. Better Representation of Modern Music Consumption Trends

By choosing average Spotify popularity, we ensure that success is measured holistically, including both traditional hits and modern streaming-driven songs. This approach reflects real audience engagement,

avoids industry bias, and captures long-term relevance beyond peak chart performance.

3. Methodology

3.1 Data Exploration

For this project, we focus on the following key columns from both datasets to analyze song success:

1. Billboard Chart Positions
    - Week Position: The song's ranking on the Billboard Hot 100 for a specific week. A lower value (closer to 1) indicates higher success.
    - Peak Position: The highest position the song has ever reached on the chart. This helps measure how well a song performed at its peak.
    - Weeks on Chart: The total number of weeks the song has stayed on the Billboard Hot 100. A longer duration suggests sustained success rather than a short-lived hit.
2. Song Information
    - Song: The title of the song, used to match records between the two datasets.
    - Performer: The name of the artist, which helps analyze trends across different musicians.
    - SongID: A unique identifier for each song, ensuring accurate aggregation of multiple weekly Billboard entries.
3. Spotify Popularity

- spotify_track_popularity: A numeric score (0-100) representing how frequently the song is streamed on Spotify. This serves as an alternative metric for success beyond traditional chart rankings.

2. Data Aggregation

Upon identifying duplicate records for the same song with different popularity scores, we implemented data aggregation to ensure each song has a single representative entry. The aggregation was performed as follows:

1. Grouping Criteria

The dataset was grouped by:

- Song: To ensure each unique track is considered.
- Performer: Since different artists may cover the same song.
- SongID: A unique identifier for each track, avoiding mismatches.

2. Aggregation Strategy

To handle multiple records per song, the following aggregation techniques were applied:

Feature Type	Aggregation Method	Reason

Numeric Features (e.g., Popularity, Weeks on Chart, Tempo, Danceability, etc.)	Average (Mean)	Ensures consistency by smoothing variations across different entries.
Categorical Features (e.g., Track ID, Mode, Key, Genre, Time Signature)	Mode (Most Frequent Value)	Captures the most representative value for categorical fields.

This approach ensures:

- Stability in numeric values by averaging across all occurrences.
- Consistency in categorical attributes by selecting the most common entry.
- Reduction of redundant data points while preserving meaningful insights.

Since the Billboard dataset contains multiple weekly records for the same song, we grouped the data by Song, Performer, and SongID to consolidate the records and retain the best performance indicators.

1. Grouping Criteria

- Song → Ensures each track is analyzed as a single entry.
- Performer → Accounts for different artists performing the same song.
- SongID → Ensures accurate grouping, avoiding mix-ups with similar song titles.

2. Aggregation Strategy for Billboard Data

To capture the best performance, the following rules were applied:

Feature	Aggregation Method	Reason

Week Position	Min (Best Position Achieved)	A lower value (closer to 1) indicates a higher Billboard ranking.
Weeks on Chart	Max (Longest Time on Chart)	Captures the song's staying power and sustained success.
Peak Position	Min (Best Peak Ranking)	Ensures the highest rank the song achieved is retained.

Impact on Data Quality

- Avoids over-counting songs that charted for multiple weeks.
- Ensures a single entry per song while keeping its best Billboard performance metrics.
- Allows better comparisons between long-term charting songs and short-lived hits.

After aggregating both datasets separately, the final step involved joining them using an inner join on `SongID`, ensuring only common records were retained.

Join Method: Inner Join on `SongID`

- Why Inner Join?
  - It removes songs that appear only in one dataset, ensuring that all remaining records have both Billboard chart performance and Spotify popularity data.
  - This avoids incomplete records where key attributes like popularity score or chart position might be missing.
- Impact:

- Songs that were on Billboard but not available on Spotify were dropped.
- Songs that exist on Spotify but never charted on Billboard were also removed.

Total Songs in the Data	13815

3. Lyrics Scrapped

To enrich the dataset with textual features, we scraped song lyrics using the API from Lyrics.ovh. This step aimed to extract textual data for NLP-based analysis in later stages.

1. Scraping Methodology

- The scraper iterated through each song-performer pair from the merged dataset.
- API Endpoint used:

```
https://api.lyrics.ovh/v1/{artist}/{song}
```

Those records for which lyrics weren't available were removed.

4. Data Cleaning

The lyrics cleaning process consists of three functions:

1. `clean_lyrics(text)` – Handles structural and formatting inconsistencies.
2. `expand_contractions(text)` – Expands common contractions for better text processing.
3. `remove_stopwords(text) - Removes stop words`

Handle Missing Values – Replace missing lyrics with an empty string.

Convert to Lowercase – Standardize text for uniform processing.

Remove Section Markers – Eliminate `[Chorus]`, `[Verse]`, and other metadata.

Remove Numbers – Get rid of numeric annotations.

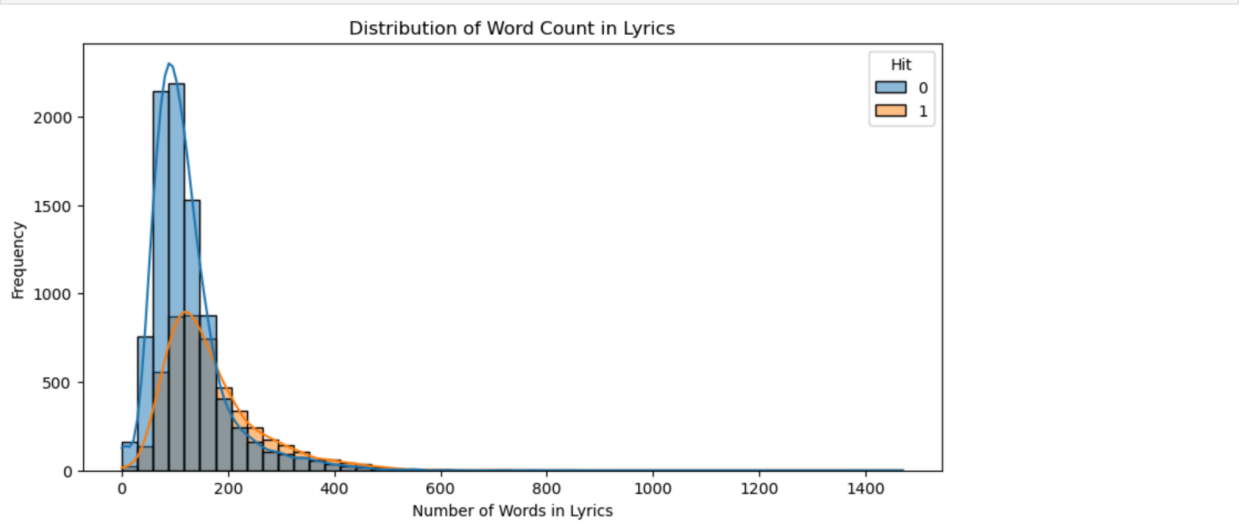
Remove Special Characters – Keep only words and apostrophes, discarding punctuation.

Normalize Spaces – Collapse multiple spaces and trim extra spaces.

Expand Contractions – Convert words like `"can't"` to `"cannot"` for better NLP interpretation.

Removed Stopwords

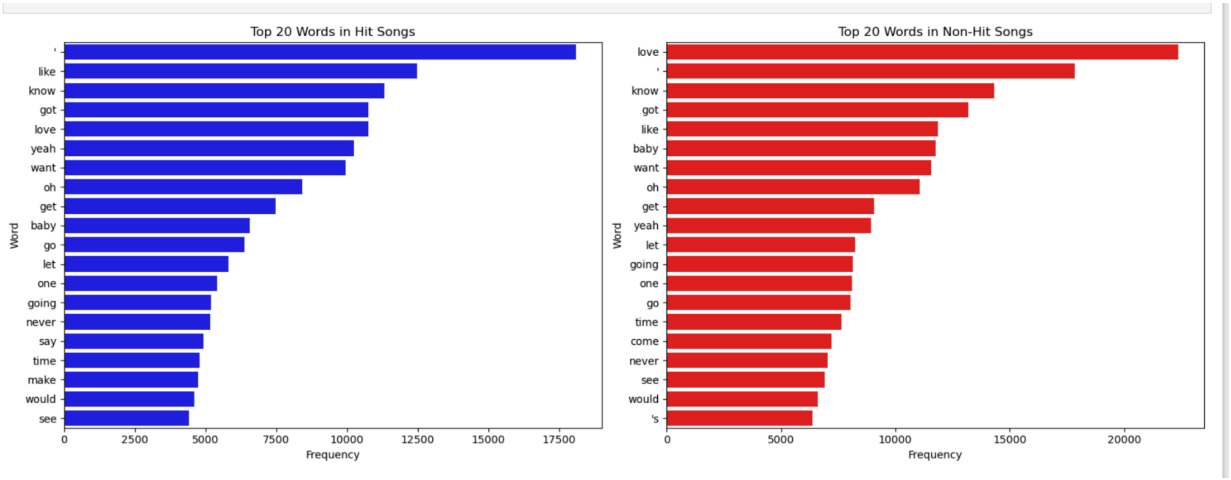
5. Analysing Trends



This histogram shows the **distribution of word counts in song lyrics**, separated by whether the song is classified as a **hit (1) or non-hit (0)**.

Key Observations:

- The majority of songs have **fewer than 500 words**, with the highest concentration around **100-300 words**.
- The distribution is **right-skewed**, meaning there are fewer songs with extremely long lyrics.
- **Hit songs (orange density curve) and non-hit songs (blue density curve) follow a similar pattern**, but hit songs seem slightly more distributed across different word counts.
- **There is no strong indication that longer lyrics correlate with hit status**, but further statistical testing would be required to confirm.



This visualization compares the **top 20 most frequently used words** in hit songs (left, blue) and non-hit songs (right, red).

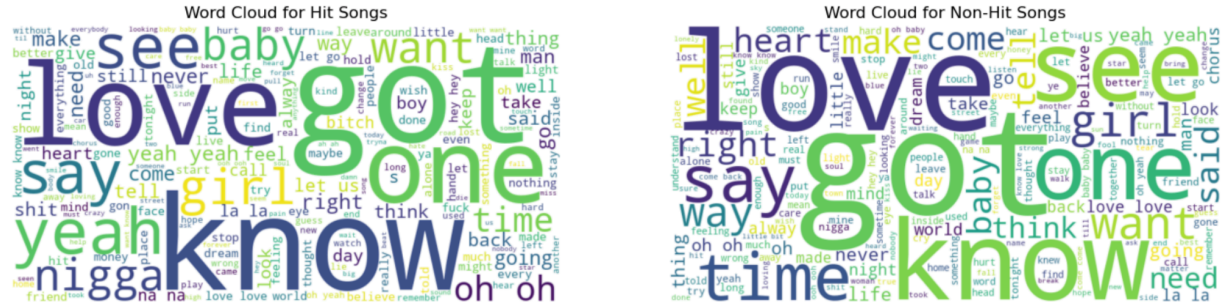
Key Observations:

- **Common Words Across Both Categories**
  - Words like **"love," "know," "got," "yeah," "want," "oh," and "baby"** appear frequently in both hit and non-hit songs, suggesting they are commonly used across all lyrics.
- **Differences in Frequency**
  - **"Love" is the most frequent word in non-hit songs**, while it appears slightly lower in the hit songs ranking.
  - **"Like" appears more frequently in hit songs**, indicating that its usage might correlate more with successful songs.
  - **Non-hit songs show a higher frequency of "time" and "come"**, which could indicate differences in lyrical themes.
- **Punctuation Handling Issue**
  - The apostrophe (') appears as one of the most frequent "words," suggesting further cleaning might be needed in text preprocessing.

Conclusion:

- There is **a lot of overlap** in frequently used words between hit and non-hit songs, meaning lyrics alone may not be the sole determinant

of success, but we will train the models, first solely on lyrics and then experiment with other features.

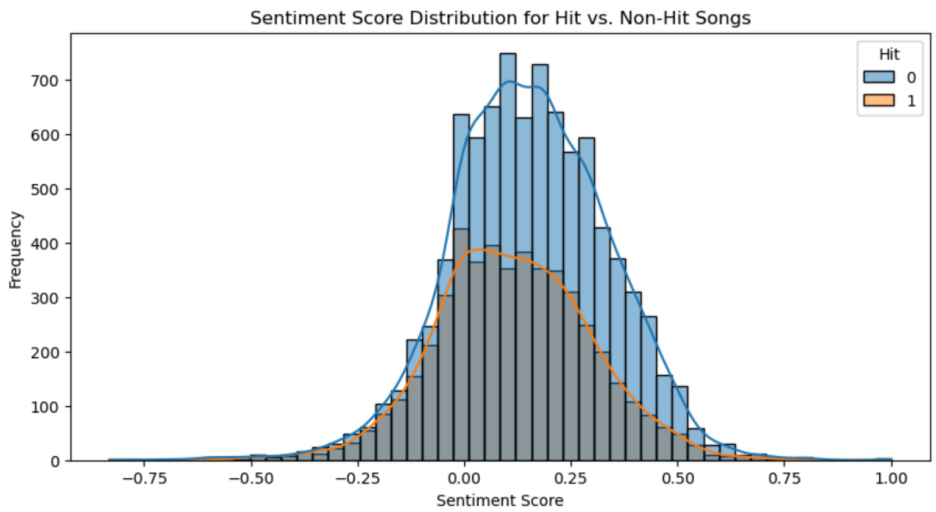


This visualization presents a word cloud comparing the most frequently used words in hit songs (left) and non-hit songs (right).

Key Observations:

- Common Words Across Both Categories
    - Words like "know," "got," "one," "love" appear prominently in both hit and non-hit songs, indicating their widespread usage in song lyrics.
- Hit Songs (Left Word Cloud)
    - "got" appears the largest, suggesting it is the most frequently used word in successful songs.
    - "know," "one," and “love" also appear frequently, reinforcing common themes of relationships and emotions.

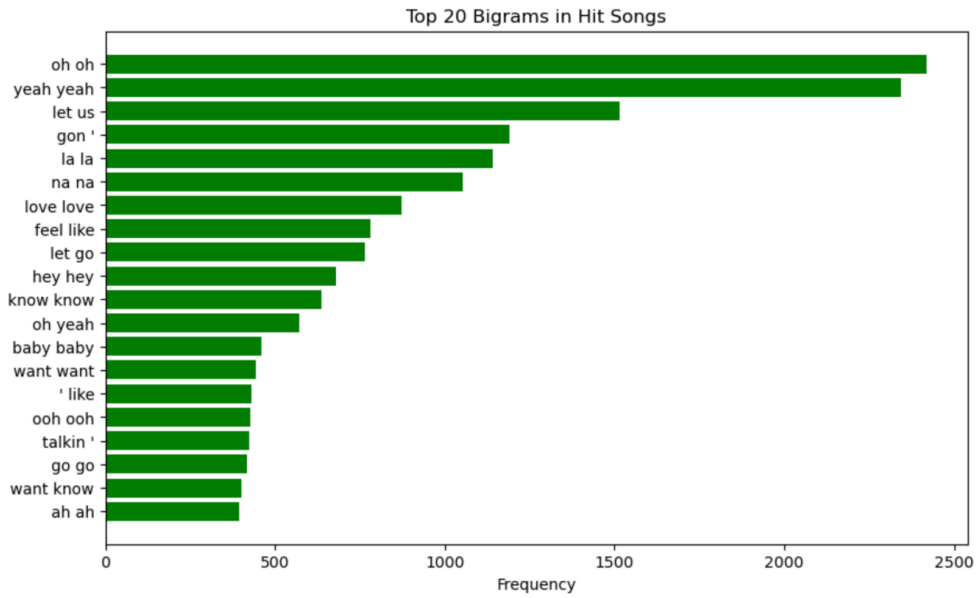
- Non-Hit Songs (Right Word Cloud)
    - "got" is the most prominent.
    - Similar words like "know" and "want" appear but with slightly different emphasis than in hit songs.



This histogram compares the **sentiment scores** of hit songs (orange) and non-hit songs (blue), showing how positivity or negativity in lyrics relates to a song's success.

Key Observations:

- **Most songs have neutral sentiment scores (close to 0)**, meaning lyrics tend to be balanced rather than extremely positive or negative.
- **Hit songs (orange) and non-hit songs (blue) follow a similar distribution**, suggesting sentiment alone is not a strong differentiator of success.
- **Slightly higher concentration of hit songs in the positive sentiment range (0.0 to 0.4)**, indicating hit songs may have a **small tendency towards more positive lyrics**.
- **Few songs have extreme sentiment values (< -0.5 or > 0.5)**, meaning highly negative or overly positive lyrics are rare in both categories.



This bar chart shows the **most frequently occurring two-word phrases (bigrams) in hit songs**.

Key Observations:

- **Repetitive Bigrams Are Common**
  - Many bigrams consist of **repetitive words** such as **"oh oh," "yeah yeah," "na na," "love love," "baby baby," "hey hey," and "ooh ooh."**
  - This suggests that **repetition is a significant feature of hit songs**, possibly enhancing their catchiness.
- **Conversational and Emotional Expressions**
  - Phrases like **"feel like," "let go," "want know," and "talkin'"** indicate **expressive language**, which may contribute to relatability.
- **Common Slang and Filler Words**
  - Phrases such as **"gon'," "talkin'," and "go go"** reflect **colloquial speech** often found in popular music.

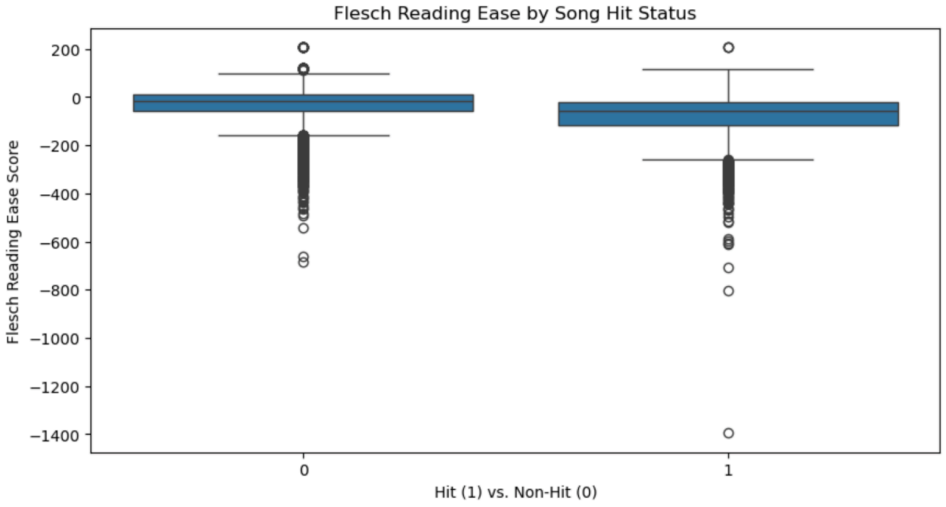
The table below summarizes the key topics identified in the song lyrics using **Latent Dirichlet Allocation (LDA)**. Each topic consists of a set of high-weighted words, representing different lyrical themes.

Topic	Top Words	Possible Interpretation
0	la, hey, oh, dance, ah, baby, shake, que, come, de	Dance and party-themed lyrics with a mix of Spanish words (e.g., "que", "de").
1	yeah, got, na, get, want, like, oh, know, ooh	Conversational and casual language, often seen in pop and hip-hop tracks.
2	like, get, got, know, nigga, ya, shit, bitch, want, go	Slang-heavy and expressive, possibly linked to rap and hip-hop genres.
3	love, know, want, oh, baby, like, never, time, one, got	Romantic and emotional themes commonly found in love songs.
4	's, little, like, got, man, going, night, back, said, old	Storytelling and reflective themes, possibly seen in folk or country music.

Topic modeling identified **five distinct thematic clusters** in song lyrics.

- **Topic 0** highlights **dance-oriented** words, suggesting **party or club music**.
- **Topic 1** contains **common filler words**, reflecting a **casual, conversational** tone.
- **Topic 2** includes **slang-heavy expressions**, indicative of **hip-hop or rap lyrics**.
- **Topic 3** focuses on **love and emotions**, commonly associated with **romantic ballads**.
- **Topic 4** suggests **narrative-driven lyrics**, often found in **storytelling genres** like **country or folk music**.

This analysis provides insights into how **lyrical themes vary across music genres** and their potential influence on song success.



his boxplot compares the **Flesch Reading Ease scores** of hit songs (1) and non-hit songs (0), assessing the readability of lyrics.

Key Observations:

- **Similar Distributions:**
  - Both hit and non-hit songs have **a comparable median readability score**, suggesting that hit status does not strongly depend on how easy the lyrics are to read.

- **Presence of Outliers:**
    - Some songs have **extremely low readability scores**, possibly due to **complex vocabulary, long sentences, or unconventional lyrical structures** (e.g., rap or spoken-word lyrics).
    - There are also a few **high readability** outliers, indicating **simplistic and repetitive** lyrics.
- **Majority of Scores Cluster Around the Same Range:**
    - The **main distribution for both hit and non-hit songs is concentrated around a similar range**, implying that lyrical complexity does not significantly impact a song's commercial success.

6. Undersampling

Undersampling was performed to balance the dataset by reducing the number of majority-class samples (likely non-hit songs) to match the minority class (hit songs).

Why Was Undersampling Needed?

- **Class Imbalance Issue:** The dataset had significantly more non-hit songs than hit songs, leading to a biased model favoring the majority class.
- **Avoiding Overfitting to the Majority Class:** Without undersampling, the model could learn to predict most songs as non-hits, resulting in **high accuracy but poor predictive power** for hit songs.
- **Ensuring a Balanced Training Set:** Reducing the non-hit songs helped create a **fair representation** of both classes, improving model generalization.

7. Vectorization

To convert lyrics into a numerical format for machine learning, we used **Word2Vec**, a word embedding technique that represents words as dense vectors in a high-dimensional space.

Key Parameters and Their Relevance

Parameter	Value	Explanation
max_len	1001	The sequence length was set to 1001, slightly above the dataset's average word count of 850, to accommodate most songs while limiting excess padding.
dimensional size	235	The embedding size was set to 235, chosen randomly but sufficiently large to capture contextual word relationships.
window size	5	The model considered **5 words before and after** each target word, ensuring it learns both **local context** and **semantic relationships** within lyrics.

Why Word2Vec?

- **Captures Semantic Meaning**: Unlike simple frequency-based methods (e.g., TF-IDF), Word2Vec places similar words closer in the vector space, improving **context-aware learning**.
- **Reduces Sparsity**: Traditional **bag-of-words** models create **high-dimensional sparse vectors**, whereas Word2Vec provides **compact and dense** word representations.

- **Improves Model Performance**: By learning relationships between words, the model can better identify **patterns in hit vs. non-hit song lyrics**.

7. Model Training

1. LSTM

To predict whether a song is a **hit or non-hit**, we trained a **Recurrent Neural Network (RNN) using an LSTM-based architecture**. This model processes song lyrics as sequential data, leveraging **word embeddings and deep learning** to capture contextual relationships.

Why LSTM (Long Short-Term Memory) Networks?

- **Handles Sequential Data**: Lyrics are a **sequence of words**, where order matters; LSTM effectively captures **long-term dependencies**.
- **Avoids Vanishing Gradient Problem**: Unlike standard RNNs, LSTMs retain information across longer sequences, making them suitable for processing **long lyrics**.
- **Context-Aware Learning**: LSTMs consider **previous words when predicting**, improving the understanding of recurring lyrical patterns.

Model Architecture Breakdown

Layer	Details	Purpose
Embedding Layer	Input dimension = vocabulary size, Output dimension =	Converts words into dense vector

	235, Pre-trained Word2Vec weights, Non-trainable	representations, capturing **semantic meaning** while keeping embeddings fixed.
LSTM Layer	128 units, Dropout 0.2, Recurrent Dropout 0.2, Return sequences = True	Captures sequential dependencies in lyrics, prevents overfitting with dropout.
Global Average Pooling	Pooling across sequence dimension	Reduces dimensionality while retaining essential features.
Dense Layer	64 neurons, ReLU activation	Learns higher-level features, enhancing non-linear learning.

Dropout Layer	50% dropout	Further prevents overfitting by randomly deactivating neurons.
Output Layer	1 neuron, Sigmoid activation	Outputs a probability score for song success classification (hit/non-hit).

Compilation & Optimization

- **Loss Function**: **Binary Crossentropy** → Since it's a **binary classification problem** (hit vs. non-hit).
- **Optimizer**: **Adam** → Adaptive learning rate ensures efficient convergence.
- **Metric**: **Accuracy** → Evaluates model performance on predicting hits correctly.

Conclusion

This **LSTM-based neural network** effectively learns patterns in song lyrics, combining **word embeddings, sequential modeling, and dropout regularization** to optimize hit song prediction.

2. **Random Forest**

   Ensemble learning method using multiple decision trees.

   Why Random Forest?

- Handles high-dimensional data well, making it effective for analyzing a mix of lyrical, audio, and metadata features.
- Reduces overfitting by averaging multiple decision trees.

3. **Support Vector Machine**

   A classifier that finds the best decision boundary between two classes.

   Why SVM?

- Effective for text classification tasks when combined with vectorized lyrics (e.g., TF-IDF, Word2Vec).
- Works well in high-dimensional spaces, making it robust for lyric-based feature vectors.
- Less prone to overfitting when using kernel tricks (e.g., radial basis function (RBF) kernel).

4. **XGBoost**

   Boosted ensemble learning method using gradient boosting.

   Why XGBoost?

- Optimized for performance and speed, handling large datasets efficiently.
- Captures complex feature interactions, making it suitable for metadata-driven predictions (e.g., Spotify popularity, chart positions).
- Regularization techniques (L1 & L2) help reduce overfitting.

5. Autogluon

AutoGluon is an automated machine learning (AutoML) framework that efficiently trains and optimizes multiple machine learning models with minimal manual intervention. It handles tasks such as feature engineering, hyperparameter tuning, model selection, and ensemble learning to improve predictive performance.

Models Used in AutoGluon:

1. **CatBoost_BAG_L2** – A gradient boosting algorithm specifically optimized for categorical features and structured data. Known for handling imbalanced datasets well.
2. **RandomForestGini_BAG_L2** – A bagged version of a random forest using the Gini impurity criterion to split trees, focusing on feature importance.
3. **RandomForestEntr_BAG_L2** – Similar to the Gini-based random forest but uses entropy for splitting, prioritizing information gain.
4. **LightGBM_BAG_L1** – A fast and efficient gradient boosting framework optimized for high-dimensional data with large feature spaces.
5. **WeightedEnsemble_L2** – A combination of multiple models that enhances generalization by averaging predictions, often improving stability and accuracy.

8. Performance Comparision

Model	Accuracy	Precision	Recall	F1-Score
LSTM	**0.6409**	0.6313	**0.6560**	**0.6434**
Random Forest	0.6053	0.61	0.60	0.61
SVM	0.6236	0.62	**0.65**	0.62

XGBoost	0.6109	0.61	0.62	0.61

Model	Test Accuracy	Validation Accuracy	Prediction Time (Test)
CatBoost_BAG_L2	**0.7223**	0.7139	39.38s
RandomForestGini_BAG_L2	0.7208	**0.7215**	39.28s
RandomForestEntr_BAG_L2	0.7205	0.7228	39.10s
LightGBM_BAG_L1	0.7186	0.7232	**3.67s**
WeightedEnsemble_L2	0.7186	0.7232	3.70s

9. Result Metrics

1. Uniqueness Ratio

- **Definition**: Measures the proportion of **unique words** in a song compared to the total word count.

A **higher uniqueness ratio** suggests **rich and varied vocabulary**.

Hit songs may have a **lower uniqueness ratio** if they rely on **repetitive, catchy lyrics**.

2. Repetition Ratio

- **Definition**: Calculates how often words are repeated within a song.

A **higher repetition ratio** indicates **more lyrical repetition**, often used in **choruses and hooks**.

Hit songs may **lean toward repetition** to enhance **memorability and engagement**.

3. Lexical Diversity

- **Definition**: Measures the **variety of words** used in a song, considering how often new words appear relative to total words.

Higher lexical diversity = **more complex vocabulary** (e.g., storytelling lyrics).

Lower lexical diversity = **simpler, repetitive lyrics** (e.g., pop and dance music).

4. Rhyme Density

- **Definition**: Measures the **frequency of rhyming words** within lyrics.

High rhyme density is often found in **hip-hop and rap**, where intricate rhyme schemes are common.

Pop songs may have **moderate rhyme density**, while storytelling genres (e.g., folk, country) may have lower values.

5. LSTM Probability of Classifying as a Hit

- **Definition**: The probability output from the **LSTM neural network** that a song belongs to the **hit class (1)**.
- **Interpretation**:
  - Values closer to **1** indicate **strong confidence** in predicting a hit.
  - **Lower values suggest non-hit classification**.
  - The model may struggle with borderline cases where probability is near **0.5**.

6. AutoGluon Probability of Classifying as a Hit

- **Definition**: Similar to LSTM, this metric represents the **probability from AutoGluon's model** that a song is classified as a hit.
- **Interpretation**:
  - A high probability means AutoGluon strongly predicts **hit status**.
  - A comparison with **LSTM's probability** helps assess which model is more aligned with ground truth labels.

7. Entropy of Both Models' Classifications

- **Definition**: Measures the **uncertainty/confidence** of model predictions using entropy.
- **Lower entropy** → Model is **more confident** (probabilities closer to 0 or 1).
- **Higher entropy** → Model is **less confident** (probabilities closer to 0.5).
- Comparing LSTM and AutoGluon's entropy helps determine **which model is making stronger, more decisive predictions**.