Natural Language and Text Processing

Do Words Make Hits? An Application of NLP

Keywords: Spotify, BERT, Random Forest, Naive Bayes, Neural Networks
Type: Written assignment
Number of characters: 31,131
Number of pages: 15
Github link: https://github.com/JurajSeptak/nlp-final

Beatriz Alessandra Tomasoni Jorquera (#175869), *beto24ac@student.cbs.dk*
Sara Recio Cristobal (#176185), *sare24ac@student.cbs.dk*
Juraj Septak (#172693), *juse24ab@student.cbs.dk*

MSc Business Administration and Data Science
May 30, 2025

Contents

Abstract

This study explores whether a combination of structured audio attributes, lyrical linguistic features, and lyrical text can effectively predict chart success on U.S. Billboard rankings. We constructed a dataset by merging 955,320 Spotify tracks with Billboard chart entries (Hot 100, Radio, Streaming, and Digital Sales) spanning from 2013 to 2024. Using Spotify-provided acoustic metrics along with engineered features, including language agnostic structure, and content representations such as TF-IDF and dense embeddings, we trained multiple classification models, namely, Logistic Regression, Naive Bayes, Random Forest, and Multilayer Perceptron (MLP). To counter class imbalance in hit vs. non-hit labels, we applied oversampling methods such as SMOTE and ADASYN. With the best achieving model being Random Forest on feature set A_L_T with an F_2 score of 0.32. These findings offer valuable implications for industry stakeholders involved in A&R decisions and automated content promotion.

1 Introduction

The competition for audience attention in today's music industry has become more intense than ever. With thousands of tracks released daily and featured on different streaming platforms, the ability to identify the potential of a track becoming a "hit" offers a vital advantage to artists, record labels, and digital curators. Traditionally, Artists and Repertoire (A&R) teams rely on expert analysis and manual listening sessions to tackle song popularity. [1] However, these methods are time-consuming, subjective, and challenging to scale. Simultaneously, the rise of digital distribution of songs on platforms such as Spotify or Apple Music, allows for global access to datasets containing detailed, rich audio features and metadata. Such widespread access opens the door for systematic and data-driven analysis.

Previous studies demonstrate the ability to predict song chart success using structured audio descriptors, however, our present study extends such an approach by incorporating the power of Natural Language Processing (NLP) techniques to analyze lyrical content in depth, and therefore demonstrate the role of lyrics in hit potential.

The United States Billboard charts have been reported to have a correlation of 0.89 with global rankings, therefore, we reasonably use Billboard chart data to model our hit prediction task, with the U.S. market serving as a proxy for global success. [2]

To achieve such a task, we have built a framework that approaches predictions as a binary classification task, leveraging a wide range of feature sets ranging from audio features, lyrical features, and textual embeddings, and various combinations of these. A methodology like this could serve as an insightful tool for practitioners in the music industry to understand the characteristics behind popular music.

Such framework therefore aims to address our central research question:

"Is it possible to predict Billboard song hits in the United States?"

With a sub-question *"How do lyrics influence the identification of hits?"*.

2 Motivation

Predicting music popularity is often referred to as "Hit Song Science", and has become increasingly important in today's music industry. [3] With a growing volume of data generated from streaming platforms, social media, and the music itself, stakeholders are leveraging such information to guide strategic decisions. Hence, the integration of Natural Language Processing (NLP) offers a deeper

understanding of how both musical and linguistic elements influence a song's success.

Our framework is motivated by the need to transform raw streaming, audio, and lyrical data into actionable insights. These insights can support business decisions such as artist scouting, marketing investments, content strategy, and pre-release benchmarking.

In an era where viral trends, especially on platforms like TikTok and Instagram, shape the global music landscape, understanding the drivers behind song breakthroughs is more relevant than ever. As artists and labels navigate ecosystems dominated by digital-first engagement and rapid distribution, data-driven prediction models bridge the gap between machine learning research and real-world application in the music industry by providing interpretable models and strategic recommendations for stakeholders.

3 Related Work

To strengthen our framework, we have conducted prior research on three related projects similarly aimed at predicting hits, as well as tackling the problem of class imbalance - a challenge we also encountered in our own dataset (see Section 4.3.4).

A notable early work by Schedl et al. [4] applied machine learning classifiers to predict hit songs using features derived solely from song lyrics. The authors of this text extracted sentiment, lexical diversity, rhyme density, and repetitive patterns to train models including Support Vector Machines (SVMs) and Random Forests. The lyric features by themselves produced a moderate performance, with accuracy ranging between 60%-68% and F_1 scores around 0.60. Hence, they advocated that instead, a combination of modalities including lyrics, metadata, and audio features would be more promising, which they witnessed to reach an accuracy slightly above 70% and F_1 scores around 0.7.

Furthering such suggestion, Castelli [5] proposed a sophisticated hit song prediction framework that integrated both audio and lyrical embeddings using the models ResNet-50 for audio features, and Sentence-BERT for lyrics. The findings collected from this framework demonstrated that transformer-based lyrical embeddings outperform classification methods, and that a multimodal approach that combined audio and lyrical features substantially improved predictive accuracy on hit prediction, achieving validation accuracy up to 0.82 and F_1 scores around 0.80. This work provided direct evidence for the importance of NLP techniques and feature combinations that are at the center of our own approach.

A critical problem that is experienced in hit song prediction is the skewed distribution of hits in comparison to non-hits. Therefore, Brandt & Lanzen [6] provide empirical analysis of two oversampling techniques, namely, SMOTE and ADASYN, which are tested across various datasets and classifiers. Both methods improve minority class prediction, but it was difficult to distinguish

which method is best, as both have tradeoffs, such as creating more noise or over-smoothing complex data.

4 Methodology

We framed our task as a binary classification problem to predict hit songs by using a rich subset of our data, containing solely English tracks (which made up around 75% of the whole dataset). In addition to analysis of audio characteristics, we conducted an in-depth NLP analysis on song lyrics to uncover deeper patterns. To analyze predictive performance, we trained and evaluated multiple models, including Logistic Regression, Random Forest, Naive Bayes, and Multilayer Perceptron (MLP) across distinct feature modalities, namely, audiological, lyrical structure, and lyrical text, using classic and modern NLP techniques. Further, we trained three advanced fusion models to assess whether combined features and hybrid architectures could improve performance on a realistic and highly imbalanced dataset.

4.1 Definition of a "hit"

As there is no common formal definition of a "hit", we were faced with a challenge in relation to this ambiguity. However, after careful consideration and considering the application of our project, we opted to define a hit as a *song that appears on any of the Billboard charts at any point in the past decade (between 2013 and 2024)*. This simple definition was spread throughout our entire modeling process to ensure standardized performance. Moreover, the temporal limitation to this specific decade was intended to enhance the relevance of the analysis to contemporary music trends.

4.2 Description of Dataset

The core dataset consisted of 955,320 songs from a Kaggle Spotify collection [1] containing audio descriptors (see Table 1) and raw lyrics and was created by collecting data from six external sources, including Spotify's API. We merged this with four major Billboard chart datasets [7] (Hot 100, Radio Songs, Streaming Songs, and Digital Song Sales) to determine which songs could be regarded as "hits" using standardized song and artist names in both datasets. These were normalized via lowercasing, punctuation removal, and other standardization. Although *fuzzy* string matching (i.e., approximate matching that accounts for minor differences in spelling or formatting) was tested, its performance gains were marginal relative to computational cost.

Merging these datasets allowed us to capture a reliable spread of U.S. market data. However, these datasets brought limitations, including chart operating inconsistencies, a recent reduction in paid downloads due to online streaming, heavy biases towards Western music preferences, and a lack of sentiment in user reviews by simply focusing on streaming volume and lyrics.

4.3 Pre-processing of Data

As our dataset was integrated from multiple sources, data pre-processing was crucial to ensure quality, coherence, and compatibility with supervised learning models, and to effectively handle linguistic features. Our preprocessing consisted of a multi-step pipeline including dataset integration, target definition, language normalization, text cleaning, and feature engineering from lyrics. We also fixed relevant features like loudness, mode, and key, because they presented a wide range of data types, which would be difficult for our models to process. Songs with no information on lyrics were dropped completely. The resulting dataset combined audio features, language-agnostic features, and text features for over 713,620 English-language songs.

4.3.1 Dataset Feature Engineering

From the Billboard datasets, we engineered features including a binary indicator to determine whether a song was a hit or not (hit = 1, non-hit = 0), best position among all charts, a normalized version of best chart position, total weeks on charts, and a "long-running hit" feature for songs that were for more than 10 weeks in total. However, to avoid label leakage, Billboard-derived features were used only for exploratory data analysis, not as predictors in model training.

Feature Name	Description
`danceability`	Suitability for dancing.
`energy`	Song intensity or activity level.
`key`	Musical key (e.g., C, F#, A).
`loudness`	Average volume (dB).
`mode`	Major (1) or minor (0) key.
`speechiness`	Spoken word content.
`acousticness`	Likelihood of acoustic sounds.
`instrumentalness`	Likelihood of no vocals.
`liveness`	Presence of a live audience.
`valence`	Musical positivity.
`tempo`	Speed in beats per minute.
`duration_ms`	Song length in milliseconds.

Table 1: Spotify audio features used for hit prediction.

4.3.2 Lexical Features

Since the goal of our project was to predict whether a song was a hit based on its features and lyrics, we first ensured linguistic consistency. All lyrics were processed using a language detection module (`langdetect`), which filtered the dataset to include only English songs. Although this reduced the dataset size, it improved the quality and interpretability for NLP tasks, practices aligned

with standard preprocessing guidelines in text classification. [8] As an additional benefit, this preprocessing step also contributed to a slight reduction in class imbalance.

The remaining lyrics were then processed using text normalization procedures, lowercased, stripped of extra whitespace, and cleaned of irregular characters. As Jurafsky and Martin mention [8], this preprocessing is crucial for minimizing noise, particularly when constructing statistical models or applying regular expressions for tokenization.

To test which features help our machine learning models predict "hits", we needed to give our analysis several approaches, one of them being the use of language-agnostic, structural features extracted from the lyrics (see Table 2). These features were designed to abstract meaning or grammar away, instead capturing the stylistic and phonetic patterns commonly associated with commercial appeal (such as high levels of word repetitions). This mirrors the strategy described in Bird et al. [9], where structural patterns are abstracted as features, allowing models to perform effectively without relying on deep linguistic analysis.

Feature Name	**Description**
`ly_num_chars`	Total number of characters in the lyrics.
`ly_vocab_size`	Count of unique word tokens in the lyrics.
`ly_ttr`	Type–Token Ratio; a measure of lexical diversity.
`ly_top5_ratio`	Proportion of lyrics made up by the top 5 most frequent words.
`ly_ngram_repetition`	Max frequency of any repeating 4-word phrase (4-gram).
`ly_rhyme_like_ratio`	Ratio of adjacent word pairs with matching suffixes (proxy for rhyme).
`hit`	Binary label to identify if the song appeared on any Billboard chart and it is considered a hit or not.

Table 2: Custom features used for hit prediction.

4.3.3 Text representations

Furthermore, in addition to Spotify's audio features and our language-agnostic lyric metrics, we also processed the raw lyrics using Bag of Words (BoW), TF-IDF, and SBERT embeddings. These representations were included not only to assess whether incorporating semantic or frequency-based information could enhance predictive performance, but also to test the hypothesis that lyrical content beyond structure alone may carry additional signals of commercial success. This allowed us to compare their effectiveness against relying solely on structural and audio features.

4.3.4 Approaching Class Imbalance

The extreme class imbalance exhibited in our dataset is due to the very small number of hits (accounting for 1.15% of all songs). This extreme skew created a significant challenge for classification,

as most models tend to be biased toward the majority class. For example, a naïve baseline that predicts every song as a non-hit would achieve an accuracy of 99.4% just by predicting every song as a non-hit, but that would not help with our objective of identifying actual hits. To mitigate this imbalance and improve model sensitivity to the minority class, we applied two oversampling techniques to the training set - SMOTE and ADASYN (see Table 3).

Technique	Description
SMOTE	This method generates synthetic samples for the minority class by interpolating between existing hit songs and their nearest neighbors in feature space.
ADASYN	A method built upon Smote, that adaptively generates more synthetic samples for minority class instances that are harder to learn—particularly those located near decision boundaries. This targeted oversampling improves the model's ability to generalize by focusing on ambiguous cases, rather than assuming uniform distribution across the minority class.

Table 3: Overview of oversampling techniques

4.3.5 Working subset

We initially ran our analysis and model training on the full dataset. However, this quickly proved computationally infeasible. The memory and processing demands, especially when using pre-trained models like SBERT, resulted in system instability and extreme hardware stress, spiking the CPU temperature to nearly 100°C. To address this, we reduced the dataset to a 5% stratified random sample to preserve class balance while significantly reducing processing time. This allowed us to continue implementing text embedding techniques, without overwhelming our computational resources.

4.4 Exploratory Data Analysis

Before developing our models, we carried out a detailed exploratory analysis to compare characteristics of hits and non-hits, examining both lyrical and audiological features. Graphic visualizations of our EDA performance can be seen in the Appendix.

4.4.1 NLP Analysis

Across both hit and non-hit songs, personal pronouns including “i”, “you”, “me”, and “my” were among the most frequently used. However, hits contained repetitive hooks such as "la la", "oh oh", "yeah yeah". Repetition advocates for chantability, which is repeatedly evidenced to be a key to hit songwriting. Additionally, the top trigrams included emotional expressions such as "don t wanna"

or "i don t", as well as "oh oh oh", which suggests both emotional resonance and performative energy are important.

Part-of-speech tag analysis revealed that hits rely more heavily on pronouns, punctuation, and interjections, while relying less on adjectives and adverbs. This supports the idea that successful lyrics tend to be more emotionally expressive, rather than detailed or with a narrative density.

Interestingly, hit lyrics had slightly higher median Flesch Reading Ease scores in comparison to non-hits, which indicates that hits benefit from being simpler and more accessible in language, making it easier for listeners to sing along.

We decided to embed lyrics into a high-dimensional space using a transformer model, and projected them into two dimensions using both PCA and t-SNE. However, visualizations of these techniques revealed that hit songs are scattered throughout the embedding space, without a distinct clustering that would separate hits from non-hits. This could suggest that lyrical semantics captured by sentence embeddings may not be a strong standalone predictor of commercial success. We also applied K-Means clustering. While this method clearly formed distinct clusters, hits were still sparsely distributed across all of them.

To explore latent thematic structure, we integrated topic modeling using Latent Dirichlet Allocation (LDA). The model extracted ten coherent topics. Some emphasized repetition (such as "yeah, do, what, gonna"), others captured emotional expressions (such as "love, down, heart"), and others captured religious and spiritual references (such as "god, lord, will"). Finally, some topics focused on social relationships (such as "baby, girl, come" and "she, he, her").

4.4.2 Audio Feature Analysis

The variables *danceability, energy, valence, loudness*, and *tempo* showed right-skewed distributions, with higher values more common in hits. Both violin and box plots stratified by hit status also revealed consistent trends, namely, that hit songs tend to have higher medians and tighter distributions across *danceability, loudness, energy, valence*, and *tempo*. Furthermore, high outlier presence suggests that some hits were pushing features to extremes. Hits experienced faster tempos, with more tracks exceeding 140 BPM, and exhibited rare acoustic or instrumental qualities.

Further, hits were written in a major key approximately 78% of the time, and were more likely to be composed in C, G, or D. However, the key was quite broadly distributed and may not necessarily offered useful insights.

For language-agnostic features, hit songs used fewer characters, smaller vocabularies, lower type-token ratios, and slightly more repetition. This suggests that simpler and more repetitive lyrics are common among popular tracks.

The scatterplots and pairplots (see Appendix) showed that hits tend to fall into higher energy danceability regions, however, the low hit rate makes patterns difficult to visually separate. Additionally, we found a strong correlation between energy and loudness, a strong negative correlation between energy and acousticness, a strong negative correlation between loudness and acousticness, and a moderate correlation between valence and danceability, indicating potential redundancy in training.

To confirm features used for training had a significant impact on hits, we ran statistical tests, including Welch's t-test and Mann-Whitney U tests. Features showing statistically significant differences between hits and non-hits included *danceability, loudness, acousticness, instrumentalness, vocabulary size, valence,* and *type-token ratio* (TTR). However, energy, acousticness, and speechiness were marginally significant. Contrastingly, *tempo* and *n-gram repetition* were not significantly different between the two groups.

Finally, a PCA on all features revealed that the first two principal components explained about 30% of the variance. The main drivers were *loudness, vocabulary size, TTR,* and *danceability*. However, PCA confirmed that hits and non-hits are not linearly separable, implying the need for complex models.

4.5 Model Training

After an initial exploratory data analysis, we removed non-predictive features such as song IDs and album names, along with features likely to cause data leakage. To ensure that the model learned from meaningful and generalizable signals, we retained only intrinsic attributes such as Spotify's audio features and feature-engineered lyric metrics designed to capture language-independent patterns in textual structure and sentiment.

4.5.1 Feature and model selection

To determine whether the inclusion of lyrics enhances hit prediction performance, and assess the predictive power of different modalities (audio, lyrical features and raw lyrics) in a systematic way, we have trained and evaluated our machine learning models on seven different feature sets (see Table 4).

These feature combinations were grouped into two categories: single-modality baselines and multi-modal combinations (fusions). Our baseline feature sets focused on individual modalities to provide interpretable references for what can be learned from audio, structural lyric metrics, or raw text alone. These served as essential benchmarks for later comparisons with more complex, fused models. To investigate whether combined modalities improve predictive performance, we trained models on fused features integrating both audio and lyric-based representations.

We evaluated the performance of several machine learning models - Logistic Regression, Random Forest, and a Multilayer Perceptron (MLP). Logistic Regression was chosen for its interpretability and to serve as a simple reference point for establishing whether more complex models add predictive value. To extend this baseline, we employed Random Forests (RF) and Multilayer Perceptrons (MLP). Random Forest was chosen for its ability to model non-linear relationships, robustness to noise, and inherent feature importance ranking. Its ensemble nature made it a strong candidate for structured, tabular features like audio metrics and engineered lyric features. Meanwhile, MLPs offered greater modeling flexibility due to their capacity to learn complex, non-linear interactions and deal with fused feature sets and dense vector representations such as word embeddings.

ID	Feature set	Used for (models)
Baselines (one modality each)		
A	Audio features only (≈ 13 Spotify floats)	LogReg, RF, MLP
L	Lyric metrics only (3–5 numeric)	LogReg, RF, MLP
T1	Lyrics only → BoW / TF-IDF	NB
T2	Lyrics only → Dense Embeddings (sBERT)	LogReg, RF, MLP
Advanced Fusions		
L_T	Lyrics (embeddings) + Lyric metrics	LogReg, RF, MLP
L_A	Lyrics (embeddings) + Audio metrics	LogReg, RF, MLP
A_L_T	Lyrics (embeddings) + Lyric metrics + Audio metrics	LogReg, RF, Hybrid MLP

Table 4: Baseline and Advanced Fusion Models with Different Feature Sets

We also tested Naive Bayes, but limited its application to sparse textual representations (Bag-of-Words and TF-IDF), where its strong independence assumptions tend to perform well. Due to its inability to effectively model dense or continuous input data, Naive Bayes was not applied to audio features or embedding-based representations.

Additionally, we explored the use of Decision Trees, XGBoost, CatBoost, and LightGBM. However, preliminary assessment revealed that these models did not yield substantial performance improvements over Random Forest or MLP in our use case. Given that training these models would require an increase in time and computational complexity, we chose to exclude them from the final model suite.

4.5.2 Stratified Splitting

All feature sets were split into training (70%), validation (15%), and test (15%) sets, with stratification to maintain the 1.15% hit ratio in each subset. This ensured that evaluation would not be skewed by class imbalances.

4.5.3 Model training

As mentioned, extreme class imbalance was addressed at the data level before any model was fit. Three variants of the data (original, ADASYN, SMOTE) were passed independently through every grid search so that the algorithm itself determined whether oversampling was beneficial.

All model choices and hyperparameter decisions were made with a single guiding metric, the F_2 score (recall weighted four times more heavily than precision), as missing a potential hit was determined to be more costly than flagging extra false positives. The artificially inflated accuracy due to class imbalance was therefore not used for model selection.

Classical models

All classical baseline models (Naive Bayes, Logistic Regression, and Random Forest) were implemented using the scikit-learn library, ensuring consistent data preprocessing, training, and evaluation across models. For each model, we wrapped data standardization (`StandardScaler`) and fitting within a `GridSearchCV` pipeline, allowing hyperparameters to be tuned via 3-fold cross-validation with the F_2 score as the optimization target. This design ensured that every model saw identical data splits and evaluation logic, and enabled seamless comparison across different algorithms and resampling variants.

Model	**Hyperparameter Grid**
Naive Bayes	`alpha` $\in$ {0.1, 0.5, 1.0, 2.0}
Logistic Regression	`C` $\in$ {0.01, 0.1, 1, 10, 100} $\times$ `solver` $\in$ {"lbfgs", "liblinear"}
Random Forest	`n_estimators` $\in$ {50, 100, 200} $\times$ `max_depth` $\in$ {5, 10, 20, None} $\times$ `min_samples_split` $\in$ {2, 5, 10} $\times$ `class_weight` $\in$ {None, "balanced"}

Table 5: Grid search hyperparameter spaces for baseline models

The search spaces explored for our retained baselines can be seen in Table 5. For Naive Bayes, we tuned the smoothing strength (`alpha`). For Logistic Regression, we explored a range of regularization strengths (`C`) with both `lbfgs` and `liblinear` solvers. Random Forest was tuned more extensively, with grid sweeps over the number of trees, maximum depth, minimum split size, and optional class balancing via `class_weight`.

Multilayer perceptron (MLP)

To extend our suite of classical machine learning models, we included a neural network to explore whether it could better capture complex non-linear relationships, especially when dealing with

fused feature sets and high-dimensional text embeddings. Given the inclusion of numeric audio and lyric features, as well as high-dimensional dense vectors from SBERT, we required a network architecture flexible enough to model complex nonlinear interactions, yet regularized enough to generalize from a sharply imbalanced dataset (with hits representing <1.5% of examples).

We selected the scikit-learn `MLPClassifier` for its simplicity and tunability. Architectural depth and width were both treated as hyperparameters. Specifically, we swept over three hidden-layer configurations - (128, 64), (256, 128, 64), and (384, 128, 32) - to assess the effects of both shallow and deeper compression paths. The latter layers were sized to accommodate the size of SBERT embeddings (384-d).

In total, the grid explored:

- **Hidden layers**: 3 variants
- **Activations**: ReLU and Tanh (to compare stability vs. expressivity)
- **L2 regularization**: `alpha` ∈ {1e-4, 1e-3, 1e-2, 1e-1}
- **Initial learning rates**: {0.001, 0.01}

This yielded 48 hyperparameter combinations, each trained using early stopping (with a 10% validation split and patience of 10 epochs). Dropout and batch normalization were not used, as they are not supported in `MLPClassifier`; instead, early stopping and L2 served to reduce overfitting.

Hybrid MLP

To fully leverage the multi-modal structure of our richest feature set (audio + lyric stats + dense lyric embeddings), we designed a Hybrid MLP architecture implemented in Keras. This model processed the SBERT embedding and numeric features in parallel through separate input branches: the embedding vector passed through two dense layers ($128 \rightarrow 64$ units) with dropout, while the handcrafted audio and lyrical features were compressed through a single dense layer. The outputs of both branches were then concatenated and passed through additional dense layers before a sigmoid classification head. Early stopping (patience = 5) was applied to prevent overfitting, and dropout served as the main form of regularization. Models were trained with the Adam optimizer and evaluated using the same training/validation/test splits and F_2 score protocol as the classical models. For fairness, the Hybrid MLP was only trained on the combined `A_L_T` feature set (with and without SMOTE/ADASYN) to isolate its performance under full information.

5 Results

To assess the ability to predict Billboard hit songs, we evaluated our models based on both F_1 and F_2 scores on the validation sets, reflecting a balance between precision and recall, with an emphasis

on minimizing false negatives. The top 5 performing models are shown in Table 6. Please refer to our Appendix to see the results visually: Appendix C, Appendix D.

We evaluated Naive Bayes, Logistic Regression, Random Forest, and MLP using original, SMOTE, and ADASYN resampling. In our baseline models, Naive Bayes underperformed, with F_1 scores below 0.08. Logistic Regression showed slight gains from resampling, reaching 0.09. Random Forest performed best among baseline models, achieving 0.22 with SMOTE. MLP showed high variance, but surpassed 0.20 with SMOTE or ADASYN. F_2 scores reflected similar trends. Random Forest with Sentence BERT features and ADASYN reached 0.26, while MLP with the same setup achieved 0.20.

Type	Feature Set	Embedding	Resampling	Model	Val F1	Val F2
Advanced	A_L_T	sbert	ADASYN	RandomForest	0.1869	**0.3195**
Advanced	L_T	sbert	ADASYN	RandomForest	0.2143	0.2564
Baseline	T	sbert	original	RandomForest	0.1283	0.2465
Advanced	A_L_T	sbert	original	RandomForest	0.1248	0.2440
Advanced	L_T	sbert	original	RandomForest	0.1114	0.2197

Table 6: Top 5 Models by Validation F2 Score

In the advanced fusions, we merged three different types of feature sets mixing between fusions of artist, lyric, and embedding features (T_A, L_T, A_L_T) and added a HybridMLP model. Random Forest remained strong, with F_1 scores of 0.24 - 0.26. MLP with ADASYN achieved the top F_1 of 0.27. Top F_2 results came from Random Forest with A_L_T and ADASYN (0.32), followed by MLP with T_A and ADASYN (0.22), and HybridMLP-A_L_T with SMOTE (0.21).

The Hybrid MLP model exhibited stable yet lower performance compared to Random Forest, with an F_2 score of 0.21 versus 0.32. This gap may be attributed to the limited size of the training set, especially after downsampling to 5% of the original dataset to manage computational constraints. Neural networks, and especially hybrid architectures with multiple input streams, typically require large volumes of data to generalize effectively. The small number of positive samples (hits), even after oversampling, likely reduced the model's ability to learn meaningful patterns without overfitting. Additionally, while the hybrid architecture was designed to handle multi-modal inputs, its added complexity may not have yielded proportional gains in a task where structured tabular models like Random Forest were already well-suited to the feature space.

Through the application of various analytical approaches, we were able to uncover several key insights. Firstly, SBERT embeddings consistently outperformed BoW approaches, demonstrating the value of including transformer-based representations of lyrics. Also, models that used all three types of data consistently outperformed those based on lyrics alone, despite increased complexity. Regarding specific models, Random Forest was a robust choice across multiple configurations, offering strong results with relatively simple tuning and architecture. As for metrics, the use of F_2

was essential, given the highly imbalanced nature of the data. In tasks like hit prediction, where missing a hit is more costly than incorrectly predicting one, recall-heavy evaluation ensured the models were optimized toward practical impact.

6 Discussion

While our framework highlights factors contributing to a song's commercial success, it also raises concerns regarding fairness and bias. First, because our hit labels are based on Billboard charts, which predominantly reflect Western preferences and consumption behavior, there is a clear bias favoring mainstream genres, English language songs, and heavily marketed tracks. This creates challenges for artists whose music performs well in non-Western contexts but remains underrepresented in Billboard rankings. Although our merged dataset initially included all Spotify tracks, we limited our analysis to English language songs to leverage advanced NLP models, introducing a significant language bias into the study.

While the tracks in our dataset theoretically span multiple years, and the hit labels cover the past decade, the lack of precise release dates in the Spotify portion of the merged dataset limits our ability to conduct detailed temporal analysis, including identifying recurring trends or the resurgence of older songs. For instance, track duration features reflect evolving listening behaviors such as recent trends toward shorter songs driven by platforms like TikTok. [3] Because our analysis does not incorporate this temporal dimension, it misses critical insights, and hence, is something that future work should aim to address.

Furthermore, cultural and regional variations complicate any universal definition of a "hit." Since our dataset is U.S.-centric, the resulting models may not generalize well to other musical cultures, where different aesthetic, social, or industry factors determine success.

Lastly, the Billboard charts themselves employ varying criteria to define a "hit," including chart longevity, airplay, or raw streaming counts. A lack of clarity around these definitions influences both how labels are assigned and which features the models come to prioritize, potentially skewing the predictions.

7 Conclusion & Future Work

This research project has developed an extensive framework aimed at addressing our central research question, namely, can we predict Billboard song hits in the United States? With a particular sub focus on the role of lyrics in such predictions.

By framing the problem as a binary classification task, predicting hits vs. non hits, we explored a

range of models from traditional classifiers (Naive Bayes, Logistic Regression) to ensemble methods (Random Forest) and neural architectures (MLP, HybridMLP). These were evaluated across various feature sets and resampling techniques.

In the baseline model performance, the highest achieving model was Random Forest with the T_sbert configuration on both the ADASYN resampling and the original set. Both of their performances reached an F_2 score of 0.25, demonstrating the model's robustness in handling imbalanced data and capturing semantic nuance from lyrics. In our advanced fusion model performance, on the other hand, the best performing model, and overall best model, was Random Forest with ADASYN resampling on the feature set A_L_T, with an F_2 score of 0.32.

Regarding our sub question "do lyrics contribute meaningfully to hit prediction?", our results indicate that lyrics alone provide some predictive power, especially when enhanced through sophisticated embeddings like Sentence BERT. However, lyrics in isolation are not sufficient to reach high performance, indicating that other modalities such as musicality and additional metadata such as artist popularity or genre play critical roles in chart success.

For future work, we recommend expanding the feature space to incorporate multimodal and temporal data including the following concepts.

- **External signals:** Including artist history, social media performance, and release dates.
- **Metadata features:** Such as genre and label.
- **Convolutional Neural Networks (CNNs):** Detecting even deeper structural patterns in lyrics.
- **Time Series Modeling:** Implementing architectures like LSTM (Long Short Term Memory) or GRU (Gated Recurrent Unit) to capture release trends and lyrical sequences.

Such future work would help move toward a more holistic and realistic model of what makes a song successful, bridging quantitative analysis with cultural phenomena.

References

[1] BwandoWando. *960k Spotify songs with lyrics data*. Kaggle dataset. Feb. 2025. URL: https://www.kaggle.com/datasets/bwandowando/spotify-songs-with-attributes-and-lyrics/data.

[2] N. Koenigstein and Y. Shavitt. *Predicting Billboard Success Using Data-Mining in P2P Networks*. ResearchGate. 2009. URL: https://www.researchgate.net/publication/221558655_Predicting_Billboard_Success_Using_Data-Mining_in_P2P_Networks.

[3] G. P. Oliveira, A. P. Couto da Silva, and M. M. Moro. "What makes a viral song? Unraveling music virality factors". In: *Proceedings of the 16th ACM Web Science Conference*. May 2024, pp. 181–190.

[4] David G. Brown et al. "Can Song Lyrics Predict Hits?" In: *Proceedings of the 2015 International Conference on Music and Music-Related Research (CMMR)*. 2015. URL: https://cs.uwaterloo.ca/~browndg/CMMR15data/CMMR2015paper.pdf.

[5] Elisa Castelli. "Hit Song Prediction system based on audio and lyrics embeddings". Master's Thesis. Politecnico di Milano, 2023. URL: https://www.politesi.polimi.it/handle/10589/210808.

[6] Erik Broman. "Predicting Hit Songs with Machine Learning". Master's Thesis. University of Gothenburg, 2021. URL: https://www.diva-portal.org/smash/get/diva2:1519153/FULLTEXT01.pdf.

[7] *Billboard Hot 100 & More*. https://www.kaggle.com/datasets/ludmin/billboard. (Visited on 05/28/2025).

[8] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. Online manuscript released January 12, 2025. 2025. URL: https://web.stanford.edu/~jurafsky/slp3/.

[9] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. https://www.nltk.org/book/. Accessed: 2025-05-27. 2009.

Please note: This project is a continuation of the project for the Machine Learning and Deep Learning (KAN-CDSCO2004U) we submitted on May 14, 2025, aiming to enhance and expand its core approaches. Substantial efforts were made to tailor the feature engineering, model selection, and discussion to suit each course's specific learning objectives. This note is included for full transparency and to avoid concerns of duplicate or self-plagiarized work.

8 Appendix

The appendix includes supplementary materials that support the content of this document. These materials provide additional information and references to enhance the understanding of the aspects discussed.

A Reproducibility

To ensure the reproducibility of our work, we followed rigorous documentation and modularization of all code and data processing steps. The following measures were implemented:

Data Sources & Preprocessing: All scripts for cleaning, standardizing, and merging datasets—including language filtering, text normalization, and feature engineering—are contained in a Jupyter notebook titled *1. Data Cleaning & Feature Engineering*.

Version Control: Python packages used include `scikit-learn`==1.4.2, `nltk`==3.8.1, `spacy`==3.7.4, `sentence-transformers`==2.6.1, and `torch`==2.1.2+cpu. All installations were pinned to specific versions to avoid compatibility issues (see notebook *2. EDA* for full environment logs).

Random Seeds: All model training and data splits were done with a fixed random seed (`seed\_value` = 42) to ensure consistent results across runs.

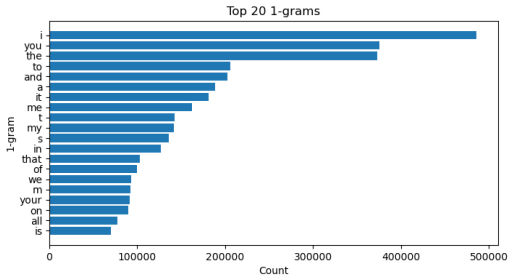
Train/Val/Test Splits: The dataset was split into training, validation, and test sets using stratified sampling. Separate CSV files were saved for each feature set (audio, lexical, text, and combinations thereof) and are stored in the training_sets/directory for ease of reuse.

Embeddings & Feature Representations: For text-based features, we used three techniques: Bag-of-Words, TF-IDF, and SBERT (with model `all-MiniLM-L6-v2`). All embeddings were precomputed and cached for each dataset split. Code for this is provided in *3. Model Training*.

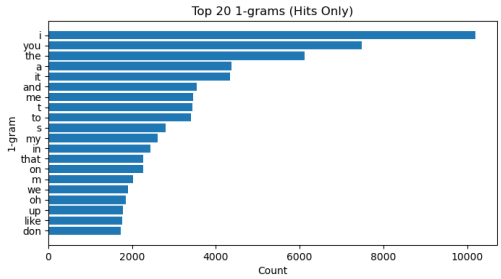
Availability: All code, notebooks, and processed datasets have been made available in a GitHub Repository

Trained models: Trained models with the best hyperparameters were saved as `.pkl` files per embedding type and feature split. These are available in the `saved_models/` directory in the repository.

B Exploratory Data Analysis

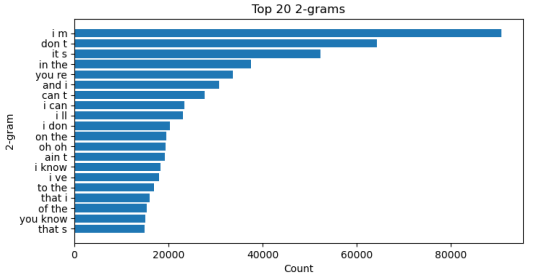


(a) Top Unigrams - Non Hits

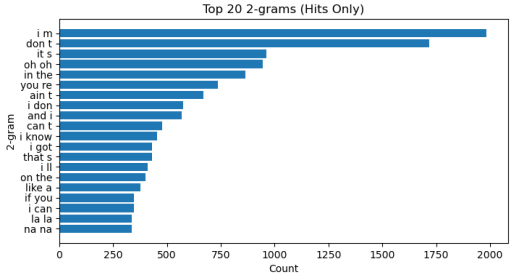


(b) Top Unigrams - Hits

Figure 1: Top Unigram Frequency Distributions

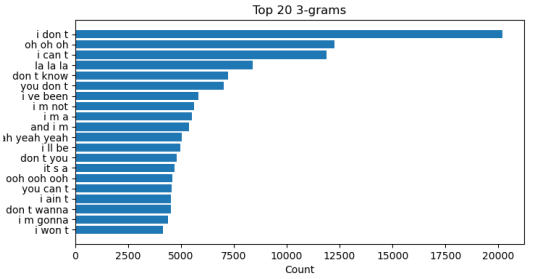


(a) Top Bigrams - Non Hits

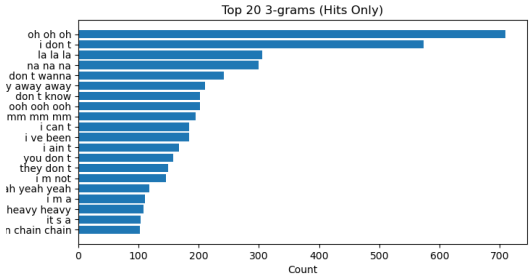


(b) Top Bigrams - Hits

Figure 2: Top Bigram Frequency Distributions

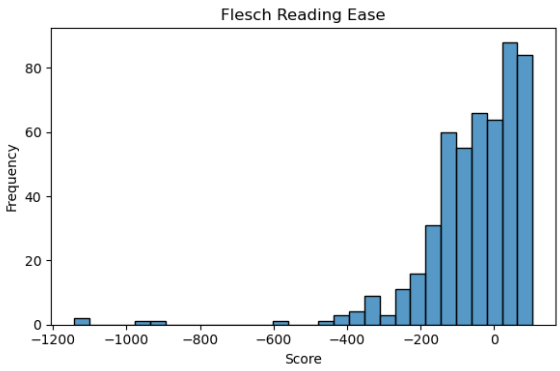


(a) Top Trigrams - Non Hits

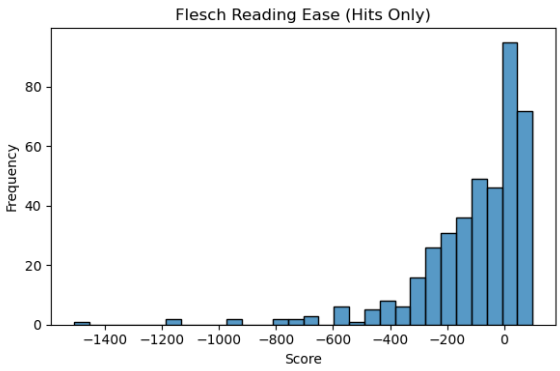


(b) Top Trigrams - Hits

Figure 3: Top Trigram Frequency Distributions

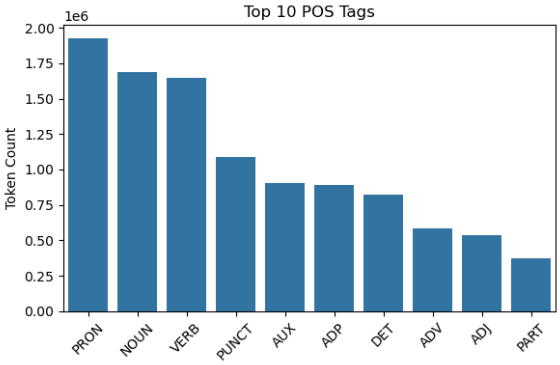


(a) Flesch Reading - Non Hits

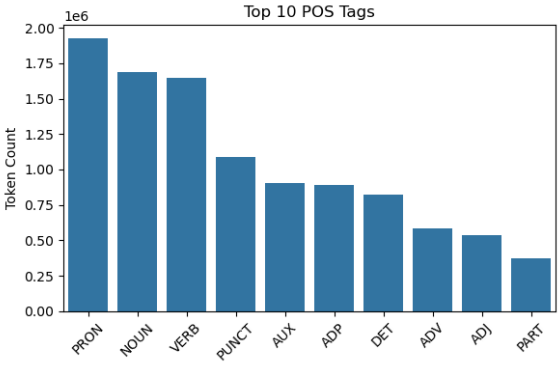


(b) Flesch Reading - Hits

Figure 4: Flesch Reading Distributions

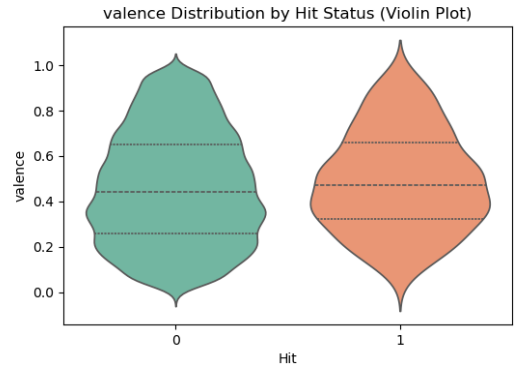


(a) POS Tags - Non Hits

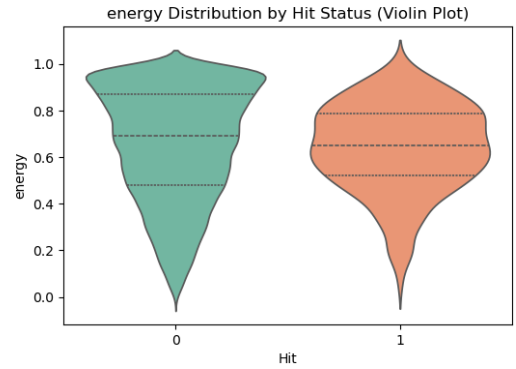


(b) POS Tags - Hits

Figure 5: POS Tags

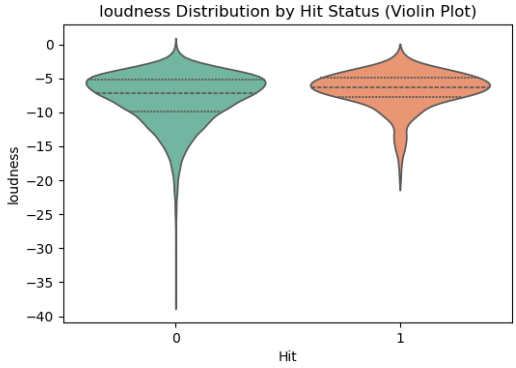


(a) Valence

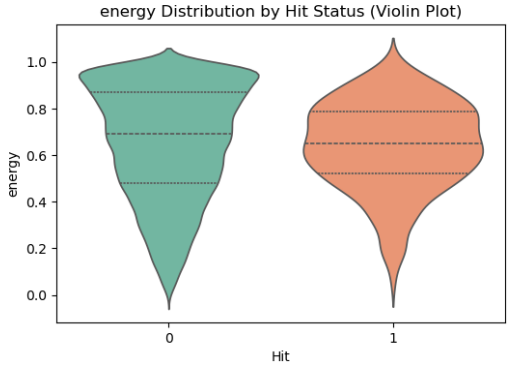


(b) Tempo

Figure 6: Violins - Valence, Tempo

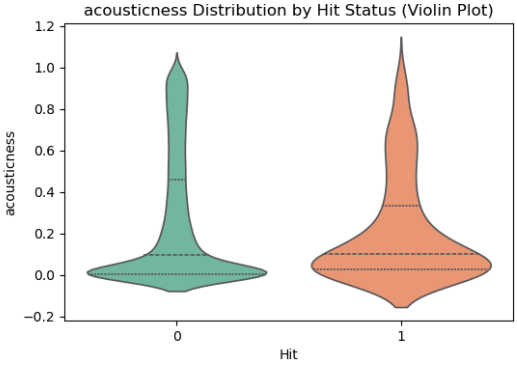


(a) Loudness

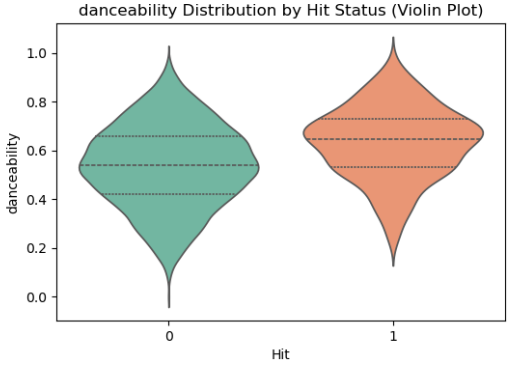


(b) Energy

Figure 7: Violins - Loudness, Energy

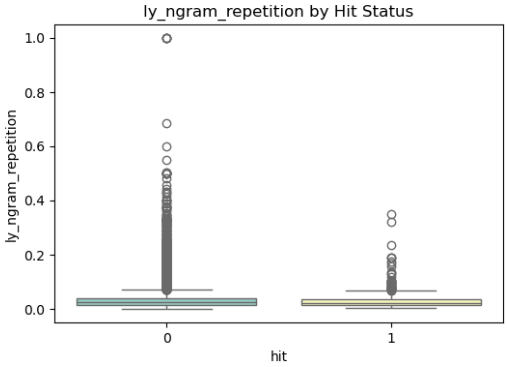


(a) Acousticness

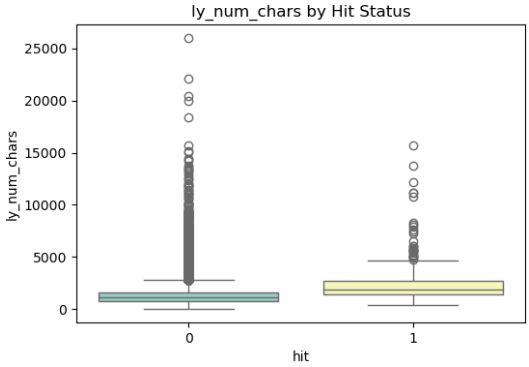


(b) Danceability

Figure 8: Violins - Acousticness, Danceability

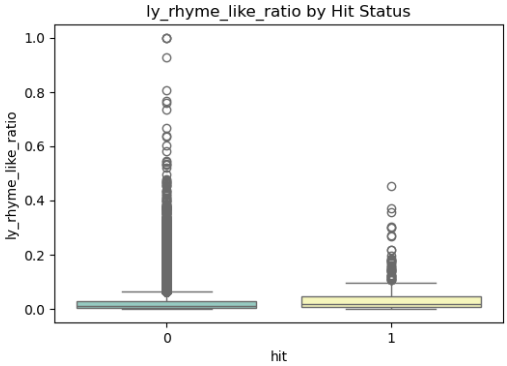


(a) Ngram Repetition

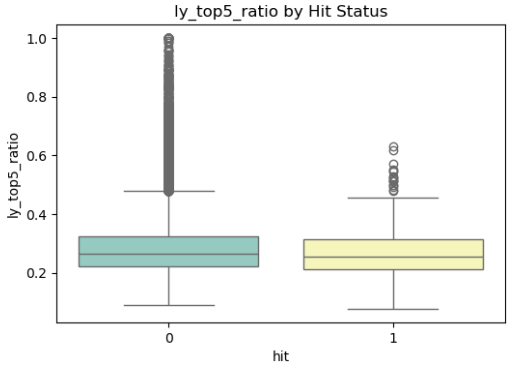


(b) Num Chars

Figure 9: Lyric Distributions - Ngram Repetition, Number of Characters

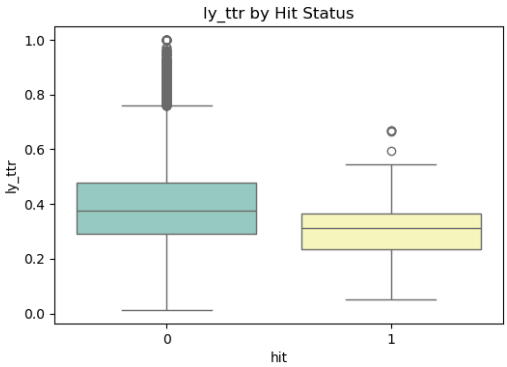


(a) Rhyme

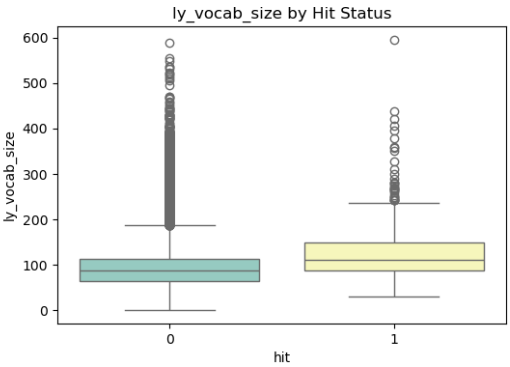


(b) Top5 Ratio

Figure 10: Lyric Distributions - Rhyme, Top5 Ratio

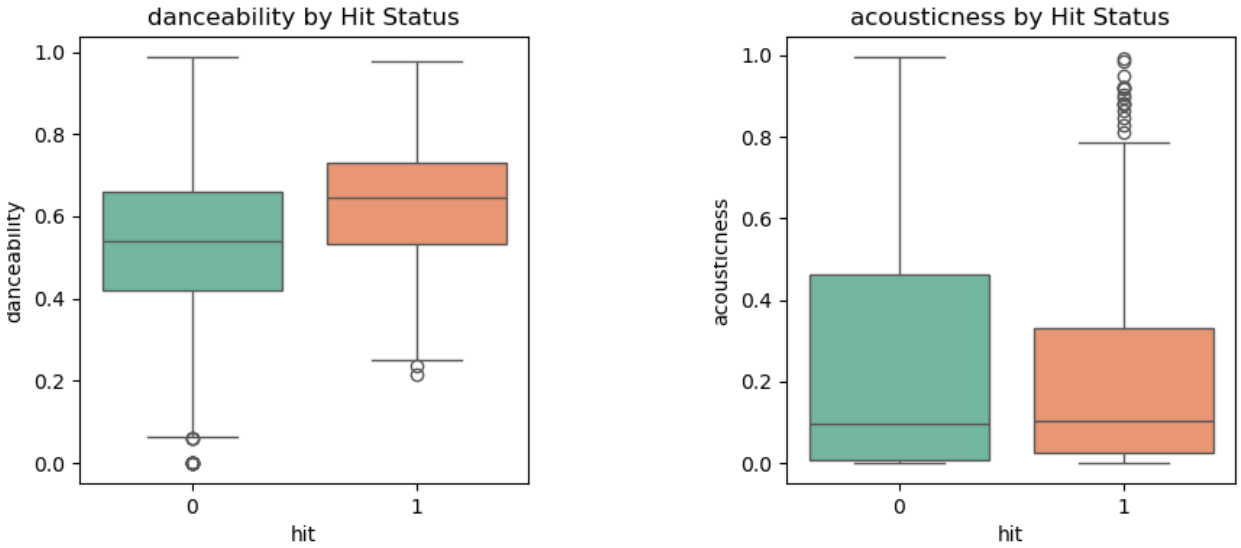


(a) TTR



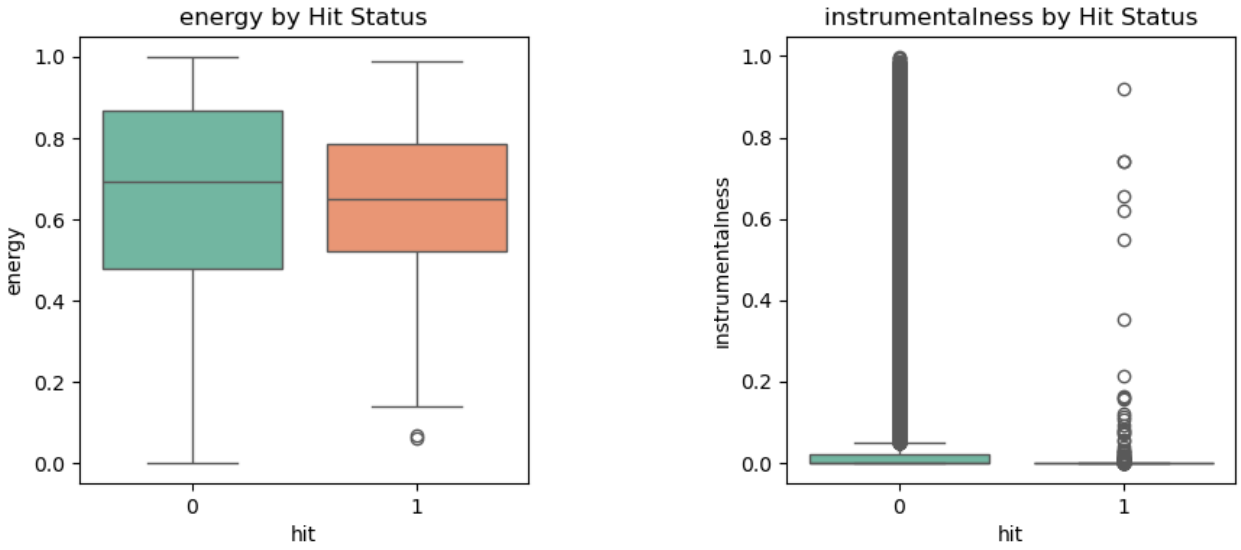
(b) Vocab Size

Figure 11: Lyric Distributions - TTR, Vocab Size



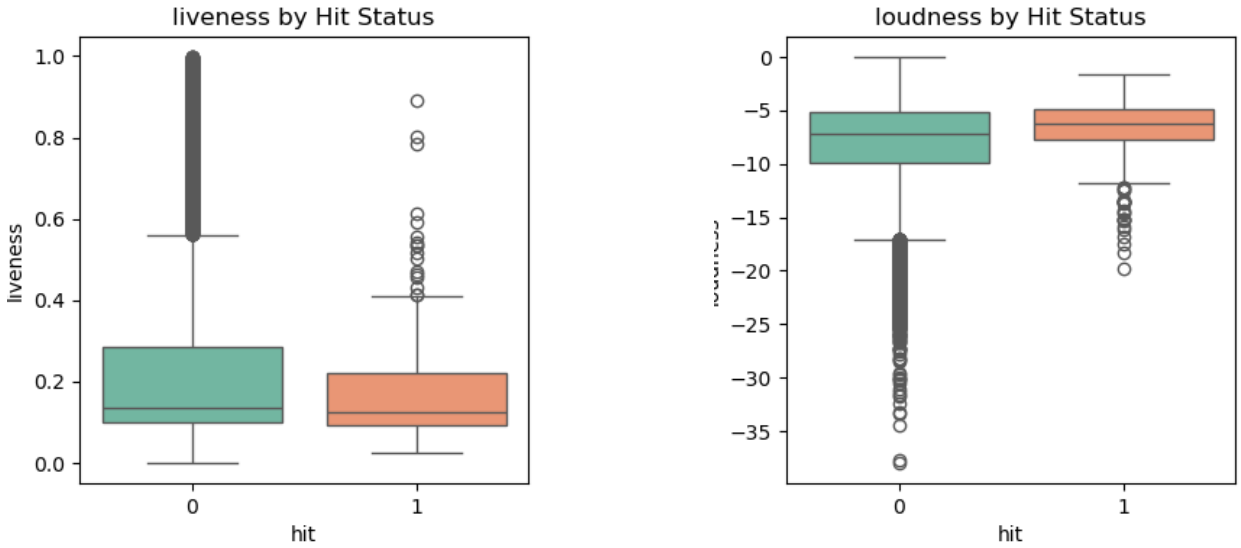
(a) Danceability **(b)** Acousticness

Figure 12: Audio Distributions - Acousticness, Danceability



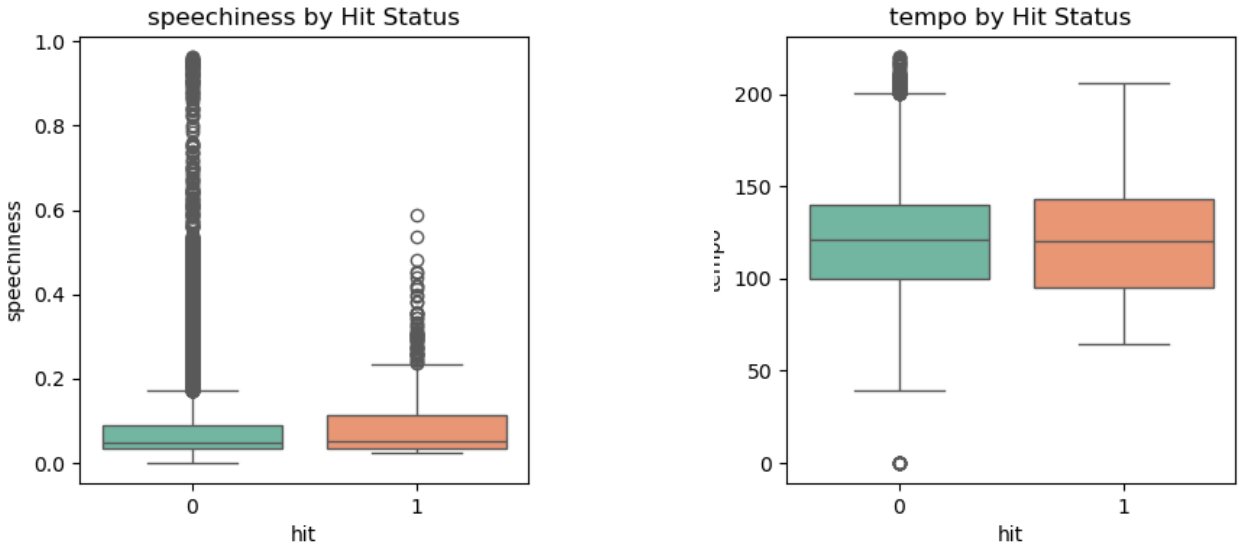
(a) Energy **(b)** Instrumentalness

Figure 13: Audio Distributions - Energy, Instrumentalness



(a) Liveness **(b)** Loudness

Figure 14: Audio Distributions - Liveness, Loudness



(a) Speechiness **(b)** Tempo

Figure 15: Audio Distributions - Speechiness, Tempo

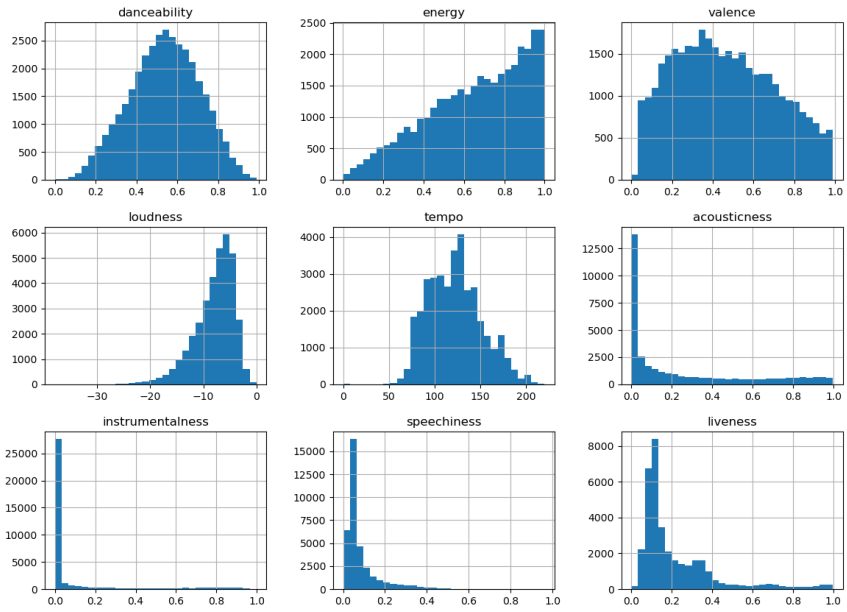


Figure 16: Audio Feature Distributions

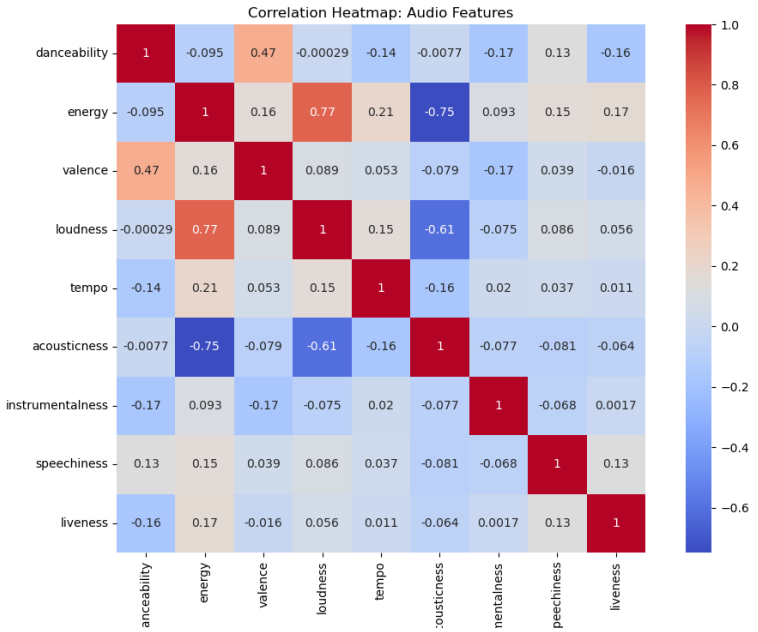


Figure 17: Correlation Heatmap

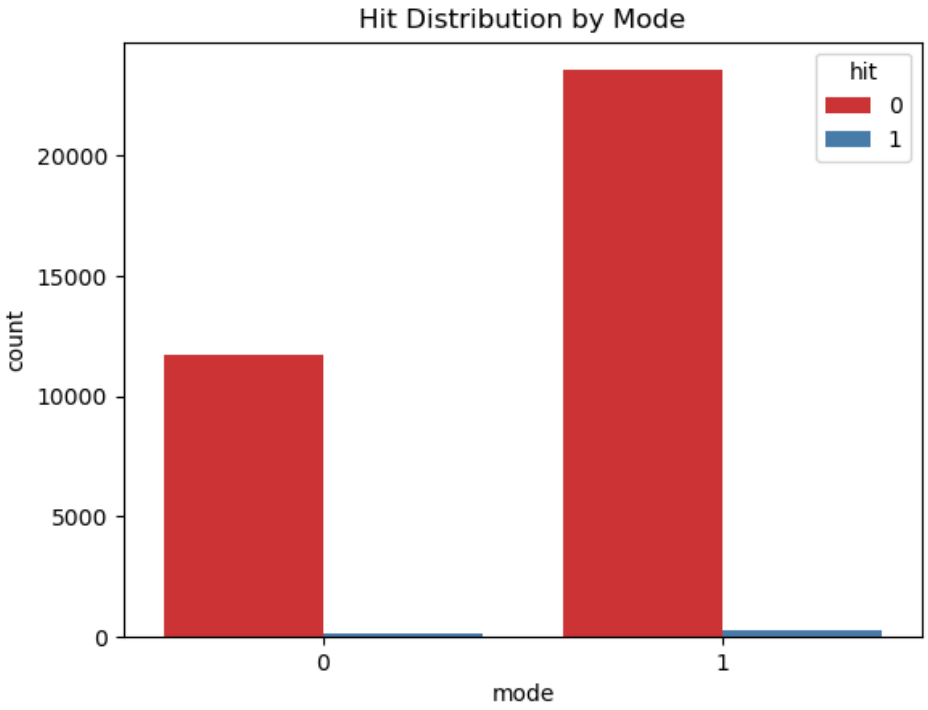


Figure 18: Hit Distribution by Mode

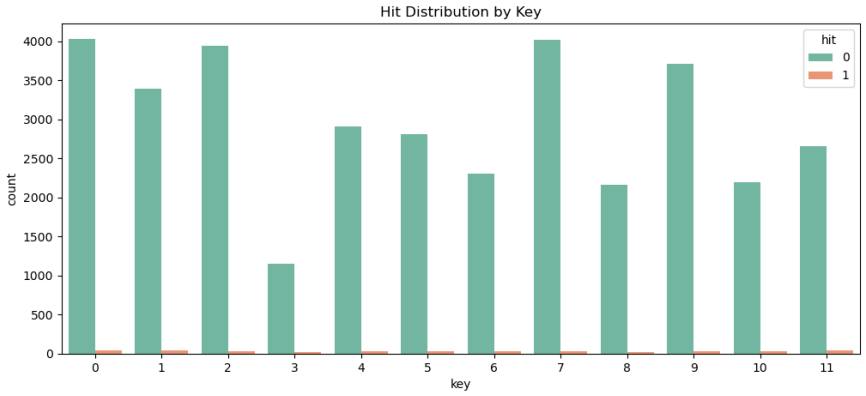


Figure 19: Hit Distribution by Key

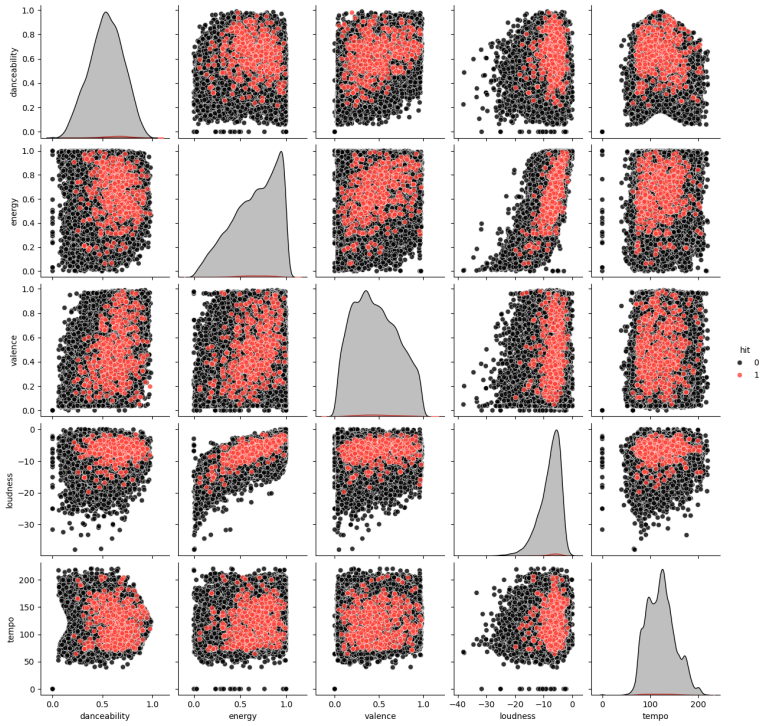


Figure 20: Pairplot

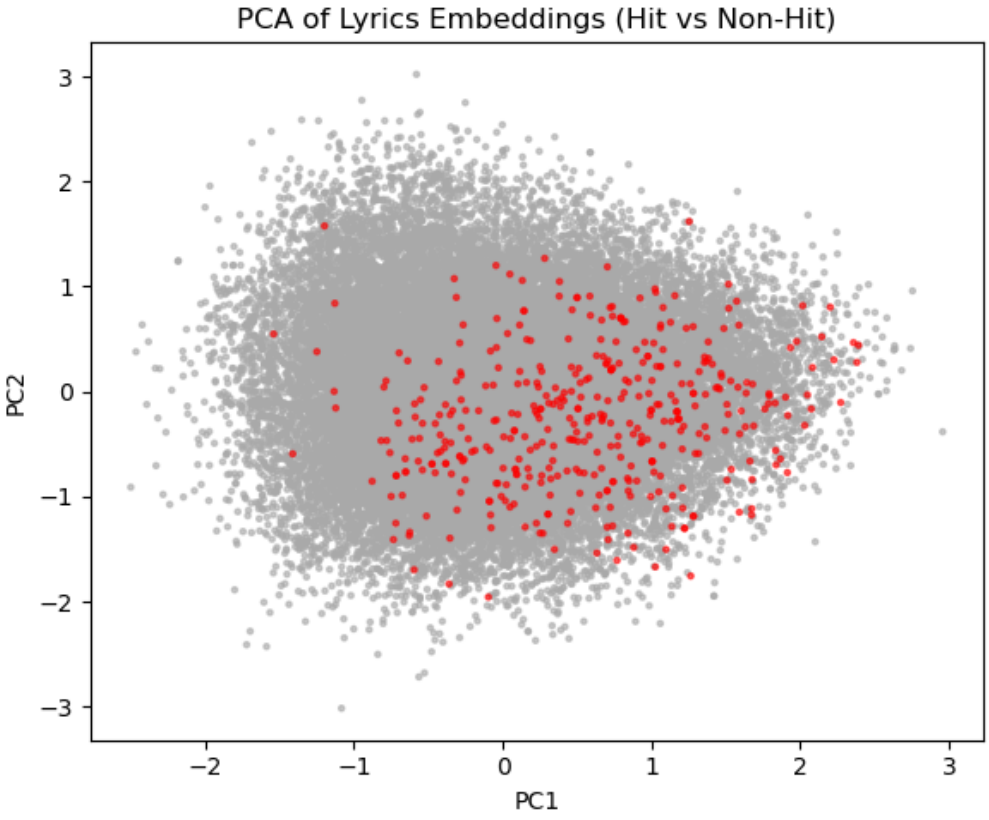


Figure 21: PCA Lyric Embeddings

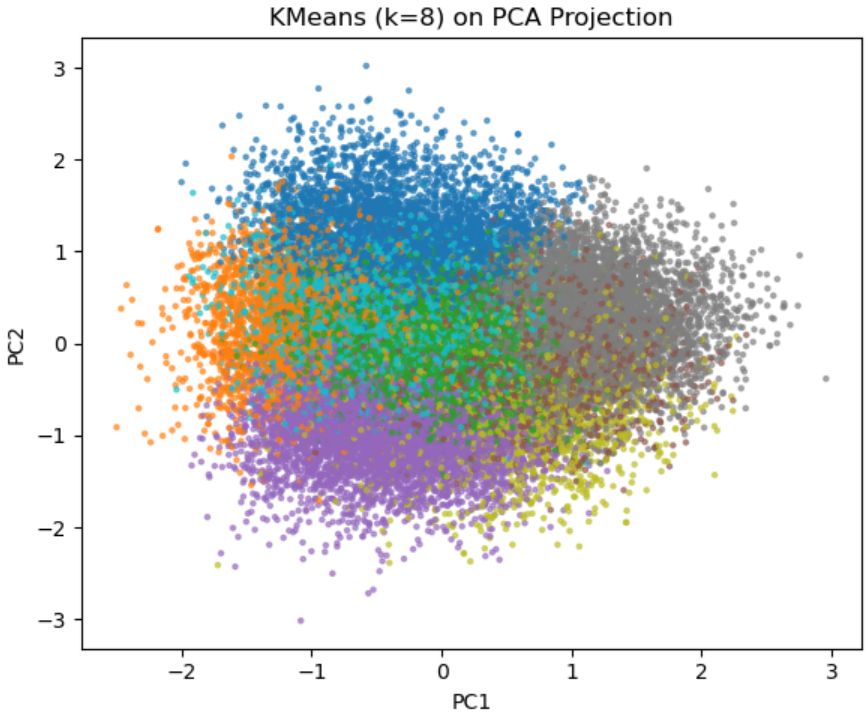


Figure 22: Kmeans

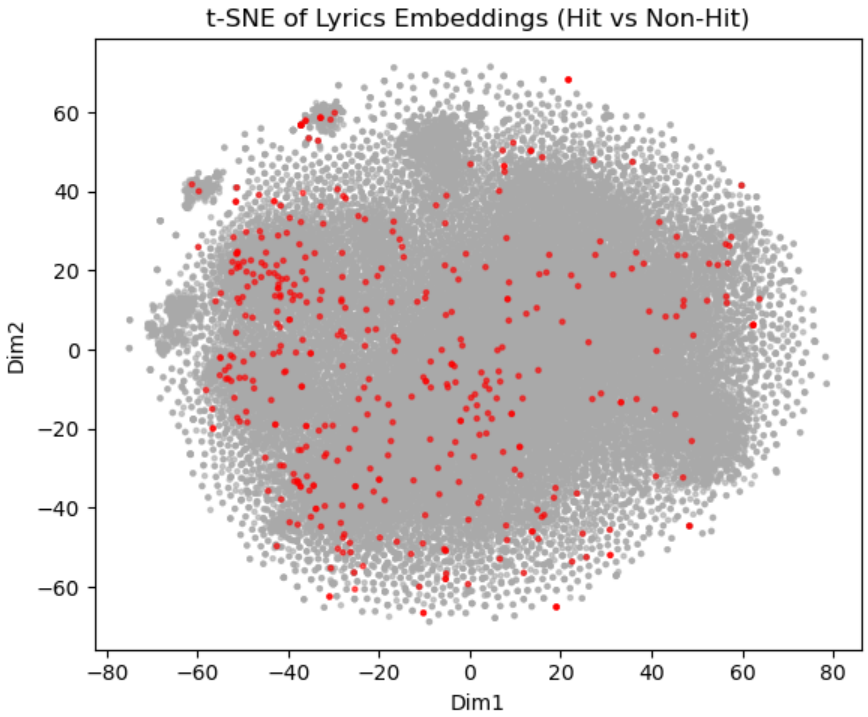


Figure 23: t-SNE Lyric Embeddings

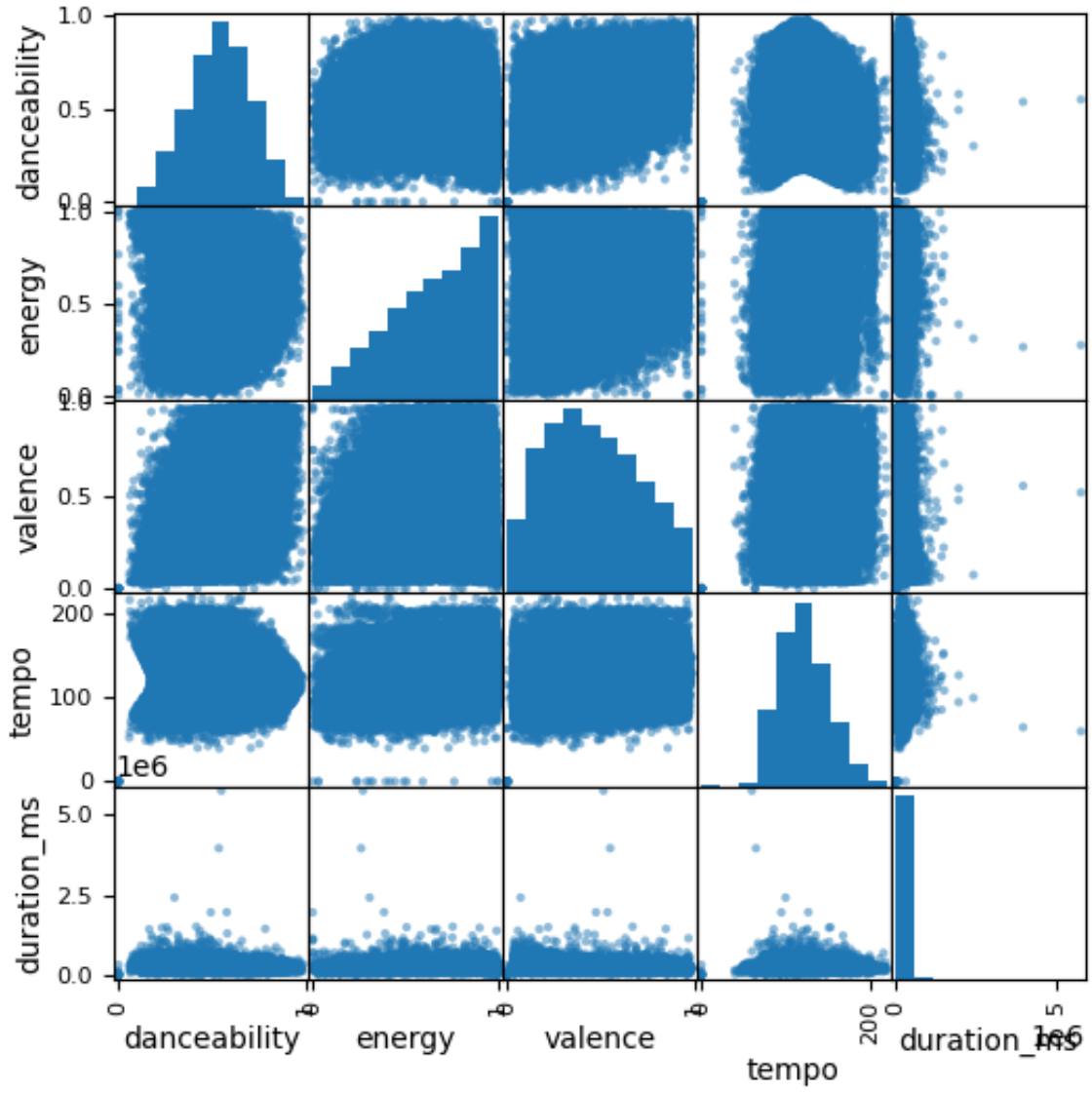


Figure 24: Scatter Matrix

C Baseline Models

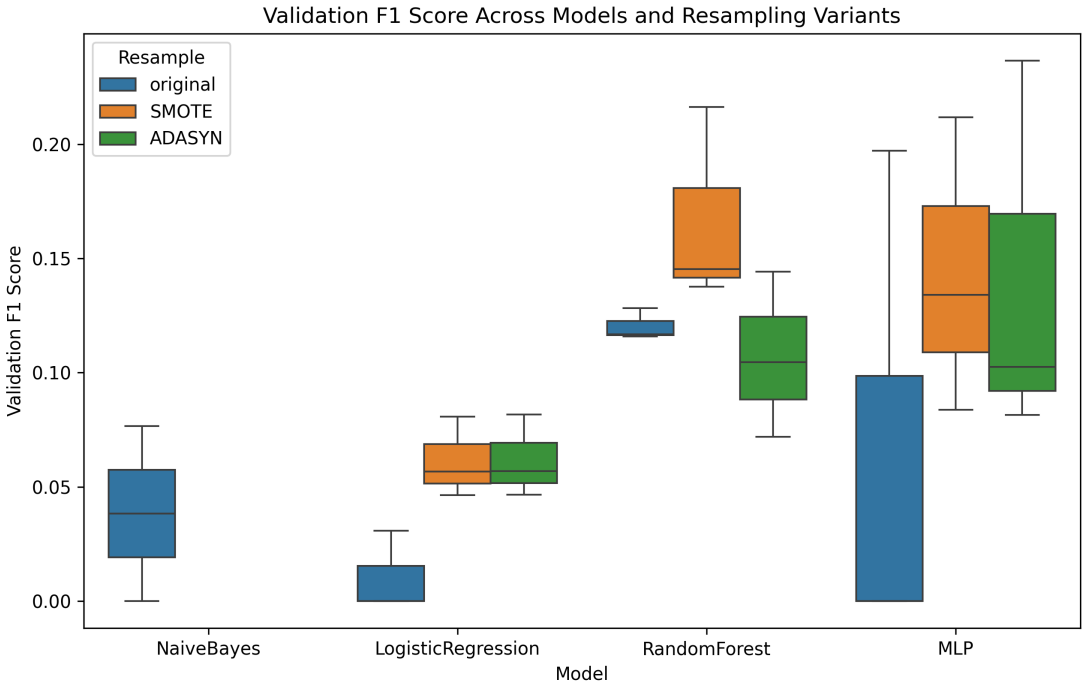


Figure 25: F1 Scores Baseline

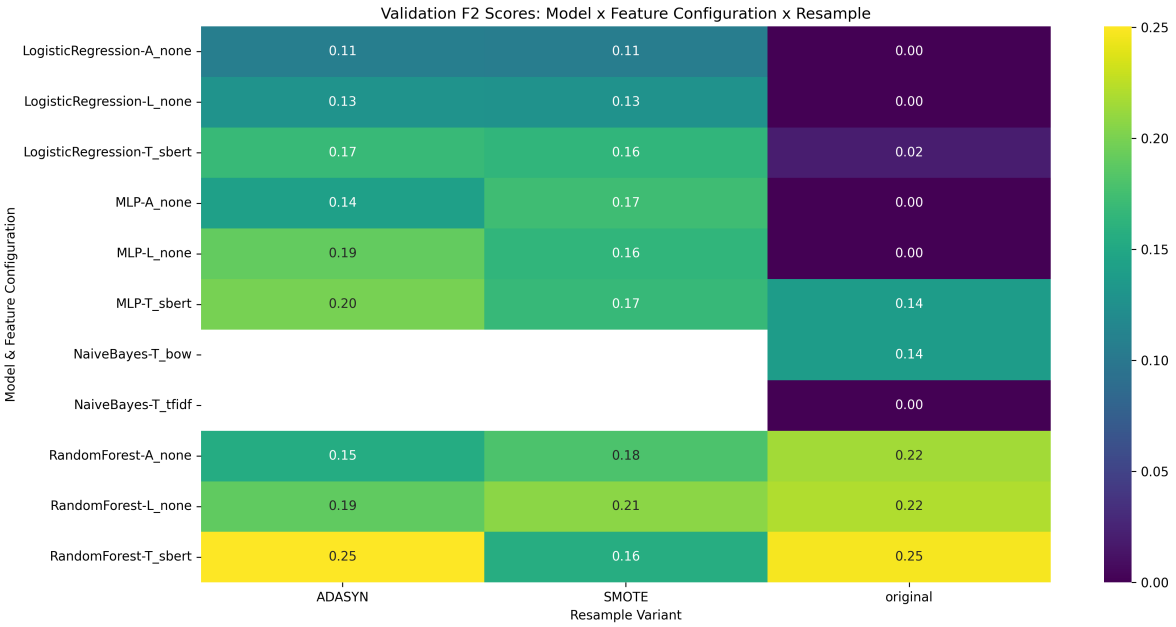


Figure 26: F2 Scores Baseline

D Advanced Fusions

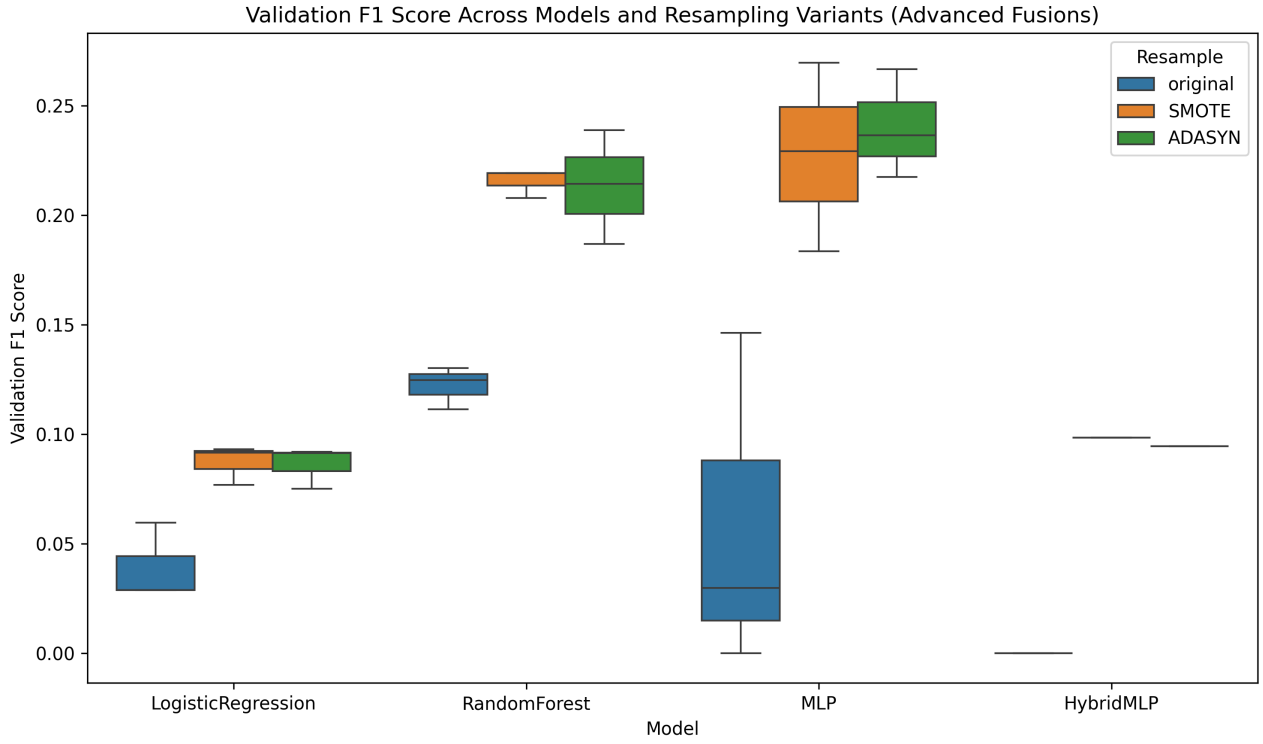


Figure 27: F1 Scores Advanced Fusions

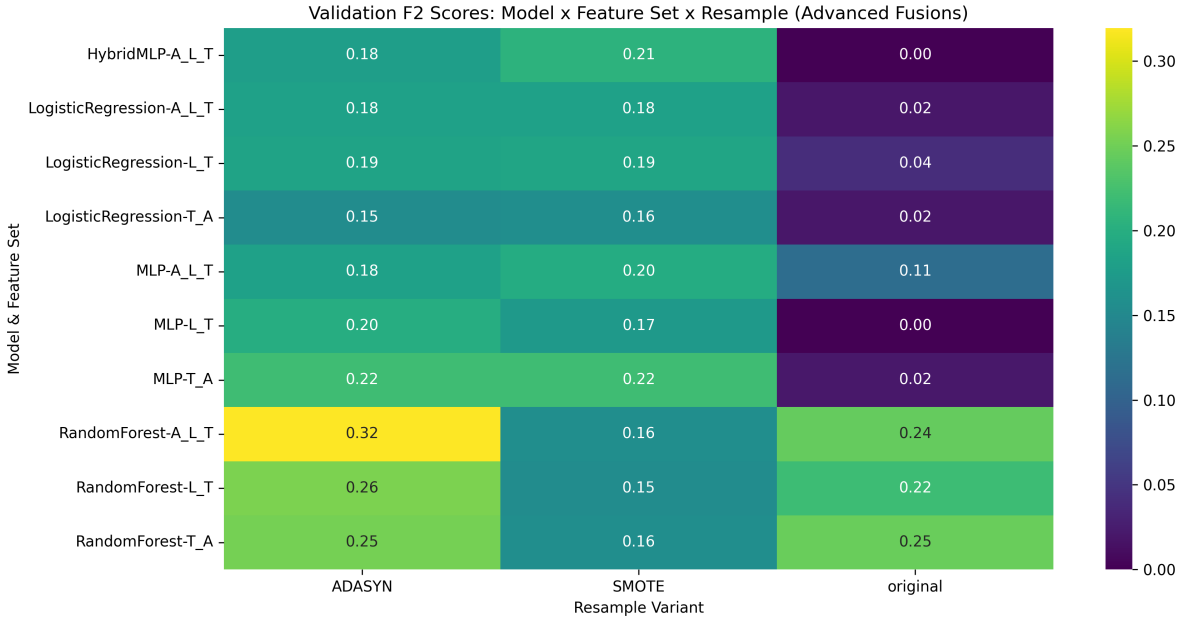


Figure 28: F2 Scores Advanced Fusions