Week 13 Writeup - Final Report

Langley, Jack, Chan, Ryan

Github Repo: jmetz1313/spotify-project

Introduction

With millions of new songs released weekly, listeners often get stuck in repetitive loops of familiar tracks. Popular streaming recommendations tend to focus on genre or trends, missing deeper emotional and lyrical connections. Our project builds a system that recommends songs based on semantic similarity—capturing mood, lyrics, tempo, and energy—to help users discover music that truly resonates, beyond genre boundaries.

We compiled a cleaned dataset of 4,000+ English songs from Spotify playlists, including detailed audio features and full lyrics. This rich mix supports nuanced recommendations that reflect how people experience music. By blending lyrical and audio data, our system aims to offer more meaningful, mood-aware discovery, boosting engagement and giving emerging artists better exposure.

Exploratory Data Analysis

To understand the characteristics and structure of the dataset, we performed a detailed exploratory data analysis focusing on the distributions, relationships, and quality of the features.

Data Cleaning and Partitioning

The dataset underwent several cleaning steps including the removal of duplicates, non-English songs, and tracks with low popularity (below 20 on a 0-100 scale) to focus recommendations on relevant and listenable music. Genre tags were standardized and simplified to primary and secondary categories to improve classification accuracy. A 70-15-15 split was chosen for train-validation-test partitions, facilitating classifier training and evaluation of genre-aware recommendations in the absence of explicit user feedback data.

Feature Distributions and Summary Statistics

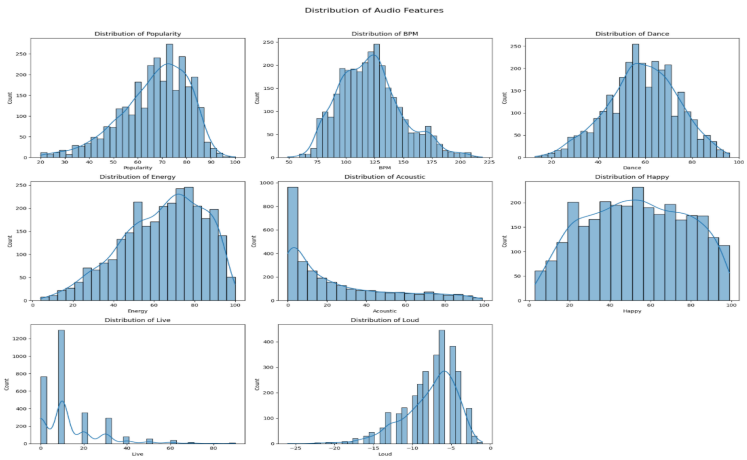

Figure 1: Distribution histograms for each of the 8 numeric audio features (popularity, BPM, dance, energy, acoustic, happy, live, loud).

	Popularity	BPM	Dance	Energy	Acoustic	Happy	Live	Loud	Time
count	2919.000000	2919.000000	2919.000000	2919.000000	2919.000000	2919.000000	2919.000000	2919.000000	2919.000000
mean	66.835218	121.072628	58.758136	64.303529	23.637205	52.902021	13.614251	-7.727304	234.816033
std	13.897719	27.444159	14.979713	20.097220	26.541940	24.532573	14.308000	3.454340	64.207000
min	20.000000	49.000000	13.000000	4.000000	0.000000	3.000000	0.000000	-26.000000	75.000000
25%	58.000000	101.000000	49.000000	50.000000	2.000000	33.000000	0.000000	-10.000000	195.000000
50%	69.000000	120.000000	59.000000	67.000000	13.000000	53.000000	10.000000	-7.000000	226.000000
75%	77.000000	137.000000	69.000000	80.000000	38.000000	73.000000	20.000000	-5.000000	264.000000
max	100.000000	219.000000	96.000000	100.000000	99.000000	99.000000	90.000000	-1.000000	685.000000

Table 1: Descriptive summary statistics for the Spotify song recommendation dataset.

Key audio features exhibit a mix of distribution patterns:

- Popularity shows a left-skewed distribution concentrated around moderately popular songs.
- BPM spans a wide range with a right skew, averaging around 121 beats per minute.
- Danceability presents a roughly normal distribution centered near 59 out of 100.
- Energy and acousticness are notably skewed, reflecting the diversity between electronic and acoustic tracks.
- Other features like loudness and live-performance confidence also show skewed patterns, suggesting potential transformations or binarization for modeling.

Correlation Analysis

Correlation matrices revealed intuitive relationships such as a strong positive correlation between loudness and energy (+0.73), and negative correlations between acousticness and

both energy (-0.63) and loudness (-0.50). Interestingly, energy and danceability showed little to no correlation, indicating that danceability is influenced by factors beyond sheer energy levels.

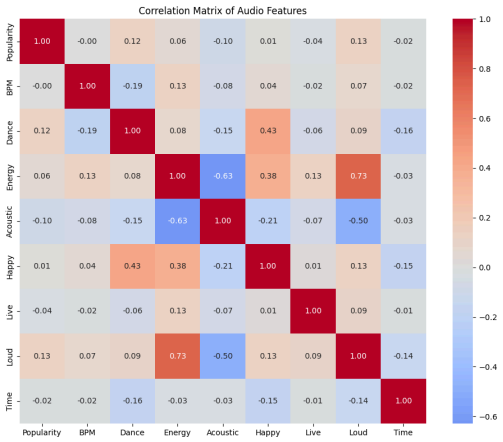

Figure 2: Correlation matrix of all audio features. Red signals a positive correlation, blue signals a negative correlation, and gray signals no correlation.

Genre Distribution

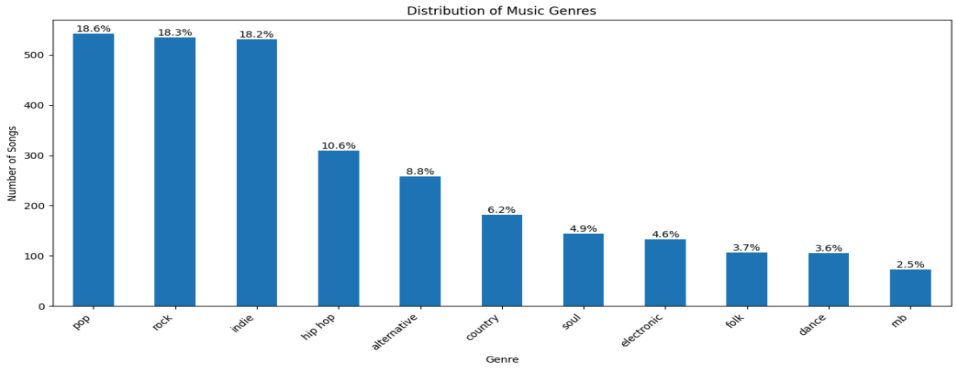

Figure 3: Bar chart showing the distribution of the different primary genres in the dataset.

The dataset includes a realistic representation of popular genres, dominated by pop, rock, and indie, accounting for over half the songs. Less common genres are also present but with fewer examples. This natural genre distribution mirrors real-world listening habits and provides the model with a balanced perspective on music diversity while acknowledging some bias toward mainstream tastes.

Data Challenges and Preprocessing Considerations

Several challenges emerged, including the complexity of multiple genre tags per song, skewed feature distributions requiring normalization, and the need to convert acousticness and

live features into binary indicators for clarity. Standardizing feature scales is essential to avoid bias during model training, particularly for deep learning approaches.

Data Preparation and Feature Engineering

As mentioned above the cleaning techniques began in Weeks 2 and 3, starting with 8,600 songs collected from 17 public Spotify playlists. We removed duplicates, remixes, non-English songs, and tracks with incomplete metadata or low popularity, narrowing the dataset to roughly 4,095 songs with clean genre and lyric data. Dropped ‘Live’ and ‘Time’ features due to imbalance and low predictive value.

Genre

Songs originally had up to 5 genre tags, which we made into one primary genre and one subgenre. We identified 11 main genres and 17 subgenres. We consolidated genres into primary and subgenre labels, merging rare subgenres to reduce sparsity and noise.

Normalization

We applied Min-Max normalization (scaling features between 0 and 1) to all numeric features - popularity, BPM, dance, energy, acoustic, happy, and loud - using parameters computed only from the training set to prevent data leakage. This normalization mitigated weight bias in models, especially deep learning and distance-based algorithms, caused by heterogeneous feature scales or negative values (for example, loudness in decibels).

We explicitly avoided using features that could leak future or outcome information, such as user ratings, playlist co-occurrences, or collaborative filtering signals. By focusing on intrinsic song attributes scraped independently, we ensured model generalizability and unbiased performance estimates.

Lyric Tokenization and Embedding

To leverage lyrical content, we enhanced text cleaning by addressing contractions (for example, changing "can't" to "cannot") to improve semantic consistency. Using NLTK, we tokenized lyrics into individual words, preparing the text for embedding.

FastText embeddings were chosen for vectorizing tokens due to their subword n-gram representation, enabling robust semantic capture of rare or misspelled words common in lyrics. Each song's lyric embedding was generated by averaging the FastText vectors of its tokens,

creating a fixed-size, semantically meaningful vector representation ready for inclusion in downstream modeling.

Feature Engineering

Encoding Categorical Variables

Genre and Subgenre: One-hot encoded to transform categorical genre labels into binary feature columns, ensuring the model could interpret them numerically.

Camelot (Musical Key): Represented using circular encoding, converting the Camelot Wheel's 24 key-mode combinations into two continuous features (camelot_sin and camelot_cos). This approach respects the circular adjacency of musical keys (ex, 12B adjacent to 1A) and reduces dimensionality compared to one-hot encoding.

Adding Interaction Terms

An interaction feature power was created by multiplying normalized Energy and Loudness. This variable captures the combined effect of acoustic intensity, highlighting songs like heavy rock or dance music with both high energy and volume. Some data transformation steps were also taken, including:

- BPM was square-root transformed to reduce right skew.
- The acoustic feature was also square-root transformed to mitigate extreme right skew.
- Other features were left untransformed to preserve interpretability given their mild skew.

Dimensionality Reduction

At this stage, dimensionality reduction was deferred to retain maximum information for our recommendation system. The circular encoding of Camelot helped control dimensionality growth without losing relational nuance.

Modeling Approach, Evaluation, and Future Work

Our recommendation system is built on the K-Nearest Neighbors (KNN) algorithm, which we chose for its non-parametric nature, flexibility, and ability to capture complex, nonlinear relationships within high-dimensional feature spaces. KNN suits our task of identifying songs most similar to a given some without relying on explicit labels, making it great for the unsupervised nature of music recommendation.

In early modeling stages, we used normalized numerical data and one-hot encoded categorical features extracted in Week 5 to train the KNN model. A key hyperparameter is K, the

number of neighbors considered. However, given the subjective nature of recommendations and absence of explicit labels, we prioritized similarity metrics such as average cosine similarity (cohesion) and neighborhood density (compactness) over traditional accuracy. These metrics guided tuning to balance neighborhood tightness without overfitting.

An initial challenge involved preparing the lyric embeddings, which were generated via pre-trained FastText models but stored in incompatible formats. Using NumPy stacking, we converted these into fixed-length vectors suitable for KNN input. We then applied cosine similarity as the distance metric to find nearest neighbors by lyrical content.

Despite promising qualitative recommendations, the lyrics-only model revealed limitations: recommendations often crossed genres in unintuitive ways (e.g., country songs paired with rap), and some genres like EDM performed poorly due to sparse lyrics. Rap songs, by contrast, showed more coherent clusters likely due to distinct vocabulary and longer lyrics.

Incorporating Additional Features and Weighted Models

Recognizing that lyrics alone did not capture the full musical context, Week 8 introduced more complexity by incorporating musical metadata such as popularity, BPM (square-root transformed), danceability, acousticness, happiness, and power features. To maintain balance between metadata and lyrics, we applied scaling and normalization, including square-root transformations, and concatenated the feature blocks.

We tested three KNN variations; (V1) Metadata only (numerical audio features), (V2) Lyric embeddings only (200-dimensional GloVe embeddings), and (V3) A hybrid model combining both metadata and embeddings, weighted equally (50% each) in the cosine similarity calculation.

For this stage, K was fixed at 1 since we were generating single top recommendations for evaluation. The weighted hybrid model (V3) showed notable gains, achieving a manual accuracy of 55/100 on the test set. It improved recommendations particularly for genres like electronic music, which had previously suffered due to limited lyrical content. Country music remained problematic, with suggestions often deviating into different genres such as rock or indie. The team proposed lightly integrating genre labels to nudge such edge cases.

Expanding Feature Integration and Tuning

In Week 9, we went further by assessing nine different KNN models, blending feature types and tuning their influence. A weighted distance strategy emerged as optimal, assigning:

- 45% influence to lyric embeddings

- 10% to genre encoding (one-hot)
- 45% to metadata

This distribution balanced feature dominance and dimensional disparities, yielding the lowest validation error and consistent test performance.

This final model achieved the highest accuracy yet - 60% on the test set - and notably enhanced recommendations for previously difficult genres like country and electronic music. While genre weighting improved accuracy for those cases, it slightly lowered recommendation quality for others, suggesting future iterations might apply genre adjustments selectively.

Final Model Evaluation Table

Model #	**KNN (n=)**	**Accuracy**	**Description**
Model 1	5	*The accuracy for the beginning models was not calculated as it wasn't determined how we would do it with our model	Lyrics-only model using FastText embeddings. Tested with varying n-gram sizes (n=5, 15, 50).
Model 2	15	"We evaluated our model in a more general sense as we are not currently clustering or using any sort of labels in this iteration of our model building process." By gathering the top 5 recommended songs for many different input songs and seeing how they match.	
Model 3	50	"while there are certainly recommendations that make a lot of sense, you can tell that there is a reason songs cannot be suggested purely based on the merit of their lyrics."	

Model 4	5	The manual accuracy score for this model was 49 ± 5, reflecting a 44–54% relevant recommendation accuracy range after accounting for subjectivity in scoring.	Lyrics-only model using Twitter embeddings, same n-gram tuning as Model 1.
Model 5	15	-	-
Model 6	50	-	-
Model 7	5	43/100	Used all features except genre to build a content-based recommendation model.
Model 8	5	49/100	Same as Model 7, but with adjusted feature weights—lyrics were weighted more heavily.
Model 9	5	55/100	-
Model 10	6	60/100	Used all available data, including genre and other metadata, for full-feature modeling. Because we used all data, K=6 was essentially K=5 because the first recommendation was the original song

Final Considerations and Model Selection

Our best-performing KNN model weights features as 45% lyrics embeddings, 45% metadata, and 10% genre, reflecting a balance between lyrical content and musical context. The lyrics embeddings remain the most influential individual feature, capturing thematic and emotional similarities across songs. Metadata features like Power (energy × loudness), danceability, tempo, acousticness, and valence provide crucial complementary signals that help maintain recommendations within a consistent sonic and rhythmic space.

Additional metadata such as artist and subgenre enrich the model's ability to capture listener preferences and niche musical styles, promoting stronger local recommendations.

Example recommendations, such as matching “The Night We Met” with “High Road”, demonstrate how the model balances lyrical emotion and musical mood to produce satisfying results. This practical behavior validates the weighted combination’s effectiveness.

Bias and Variance

While our dataset excludes explicit protected attributes like race, gender, or age, some features (ex, artist metadata and genre) may indirectly correlate with cultural and demographic aspects, raising potential proxy bias concerns. Music genres reflect rich cultural and historical backgrounds - rap and hip-hop from African American communities, classical music rooted in European traditions etc.

Our model’s reliance on lyrics and genre weighting helps preserve these authentic cultural nuances, which are essential for meaningful music recommendation. At the same time, this introduces bias that must be carefully managed. For instance:

- Linguistic bias: Variations in vernacular, slang, and dialect (common in rap or regional music) might overweight certain lyrical patterns.
- Genre bias: Over- or underrepresentation of genres in training data can skew recommendations, possibly marginalizing emerging or niche artists.

We deliberately reintroduced a small genre weighting (10%) after initial exclusion to reduce mismatches and improve recommendation quality while avoiding excessive rigidity that could stifle musical discovery. Potential mitigation strategies like balanced datasets or post-recommendation genre checks were considered but ultimately rejected due to their risk of distorting authentic listening patterns.

Risks and Ethical Considerations

Two key risks came up during our evaluation:

- Representation risk: Underrepresented artists, especially from marginalized or indie communities, may receive fewer recommendations, limiting exposure and potential growth. This is a critical concern for platforms committed to fairness and diversity in music discovery.
- User engagement and trust risk: Recommendations perceived as repetitive, overly mainstream, or biased could cause user disengagement. Stakeholders may question fairness if the system appears to favor major labels or popular genres, particularly when monetization depends on streaming volume.

Deployment and Monitoring

Our KNN-based model, leveraging a cosine-distance metric with balanced feature weighting, is well-suited for music streaming services that aim to deliver contextually rich song recommendations based on both lyrics and sonic attributes. It can effectively support playlist curation tools by balancing lyrical themes with musical energy and mood, as well as discovery platforms seeking to expose listeners to thematically similar yet diverse tracks while preserving cultural authenticity. In terms of scalability and performance, KNN offers fast training and manageable query costs on datasets comprising a few thousand songs, and techniques such as approximate nearest neighbor (ANN) methods and dimensionality reduction can facilitate scaling to tens or hundreds of thousands of songs for production environments.

The model is designed to support low-latency, real-time recommendations through efficient feature storage and computation. Looking ahead, future iterations could incorporate explicit user preference data and feedback loops to refine recommendations and address cold-start issues, while increasing the number of neighbors (K) and introducing controlled randomness can enhance the diversity and novelty of recommendations. Finally, regular retraining is essential to keep the model current with evolving music trends, and ongoing bias and fairness audits are recommended to detect and mitigate any unintended disparities in recommendation outputs.

Our progression from simple lyric-based models to a nuanced, weighted KNN approach highlights the value of combining textual, musical, and genre data for effective music recommendations. By balancing feature contributions and considering cultural and bias factors, we've created a model that delivers accurate and meaningful suggestions - with monitoring and learning from real users. Although challenges with representation and fairness persist, this model lays a strong groundwork for practical deployment and ongoing improvement.