Keyword-Based Clustering and Analysis of Music Genre Dataset

Agrim Tripathi,
Indian Institute of Technology, Kharagpur

Overview— **This project explores the clustering and classification of songs into genres based on descriptive keywords (instrument, mood, style) using Bag of Words and TF-IDF for vectorization, PCA for dimensionality reduction, and K-means clustering to uncover patterns in the data.**

It also covers various bases in terms of analysing the data provided, forming insights and drawing useful conclusions on the dataset using the results from the algorithm and the various metrics used to judge its performance.

1. INTRODUCTION

This project leverages textual keywords—describing instruments, mood, and style—to group songs into meaningful clusters. Using traditional methods of vectorization for feature extraction, Principal Component Analysis (PCA) for dimensionality reduction, and K-means clustering, this project aims to uncover patterns in song genres and evaluate how well unsupervised learning techniques align with true genre labels. It uses metrics such as Purity and Silhouette Score to quantify the pipeline's performance and helps us judge the alignment of the genres with the clusters they were assigned to.

2. THE PIPELINE

Below is a brief outline of the notebook, what it does step-by-step and what each step accomplishes.

1) **Decide which libraries to use:** In this case, we limited the use to NumPy, Matplotlib, and Pandas. Pre-built algorithms, such as those provided by Scikit-Learn, were not used. A large part of the computational heavy lifting was done by NumPy. Python (3.13) was the preferred language for this project due to the substantial documentation available for the above-mentioned libraries and extensive resources and support for such problems in Python.
2) **Fetch the dataset:** The dataset contains 147 unique songs, each with a different song ID. There are three keyword columns: `keyword_1`, `keyword_2`, and `keyword_3`, which carry information about the instrument, mood, and style of the song, respectively. Additionally, there is a "genre" column that serves as the ground truth label to help us judge the clusters with an extrinsic element.
3) **Vectorization:** This step involves converting textual data (in this case, keywords) into numerical vectors. The methods tested were TF-IDF and Bag of Words vectorization, both yielding different results.
4) **Dimensionality Reduction:** In this step, the high-dimensional vectors created in the previous step were reduced to lower dimensions while retaining as much original information as possible. The chosen method was PCA.
5) **Combining the vectors:** The three vectors (one for each keyword of each song) were combined using weighted coefficients.
6) **Clustering using K-Means:** The well-known K-Means algorithm was applied to find clusters that the pre-processed data could be separated into. These clusters provided important insights into the relationships between genres.
7) **Plotting an elbow curve:** An elbow curve with various values of `k` was plotted to help choose the best value of `k` for fitting the data.
8) **Plotting the cluster scatter plot:** A cluster scatter plot was generated using the chosen value of `k`.
9) **Evaluation techniques:** Two evaluation metrics were used: Purity and Silhouette Score, both of which provided useful insights.
10) **Classifying new songs:** Given a new song and a corresponding set of keywords, it is possible to perform the above steps to obtain the embeddings of the new song and assign it to a cluster, indicating the genre it is most likely to belong to.

A. *The Dataset*

song_id	keyword_1	keyword_2	keyword_3	genre
74	guitar	happy	distorted	rock
103	brass	energetic	melodic	classical
201	banjo	happy	acoustic	country
194	synth	energetic	heavy	hip-hop
184	synth	energetic	slow	hip-hop

The dataset contains 147 rows and 5 columns:

- `song_id`: Unique identifier for each song.
- `keyword_1, keyword_2, keyword_3`: Descriptive keywords representing the instrument, mood, and style of each song.
- `genre`: The ground truth genre label (used only for final evaluation).

Dataset Characteristics:

- Keywords Distribution: The keywords include terms such as "funky," "danceable," "calm," and "distorted," which capture the song's attributes across different features. Each song has exactly three keywords, ensuring a uniform feature structure.
- Genre Distribution: The dataset contains multiple genres, with varying keyword patterns. Genre labels are primarily used for post-clustering evaluation.

- Vocabulary Size: The keyword_1 column contains 6 unique terms, keyword_2 has 9, and keyword_3 has 10.
- Data Quality The dataset contains no missing values, making it clean and ready for analysis. The uniform three-keyword structure per song facilitates straightforward vectorization.

B. *Vectorisation*

The first step in effectively processing this data to prepare it for clustering is **vectorisation**, for which we have 2 methods to choose from **TF-IDF and Bag of Words(BoW)**. Both methods result in vectors of the song's keywords which help us visualise these terms numerically/mathematically.

A brief dive into both these vectorisation methods:

- **BoW:** Bag of Words creates a vector where each dimension corresponds to a unique word in the entire corpus. The value in each dimension represents the word count in the given text.
  Let the "document" contain a vocabulary of size V. Given a document d, the BoW representation is:
  $$\text{BoW}(d) = [f_1, f_2, \ldots, f_V]$$
  where f_i is the frequency of the i^{th} word in the vocabulary appearing in document d.
  Dimensionality: $N \times V$, where N is the number of documents and V is the vocabulary size.

- **TF-IDF:** TF-IDF (Term Frequency-Inverse Document Frequency) creates a weighted vector where each dimension corresponds to a unique word, with values representing the importance of the word in the document relative to the entire corpus.
  Let the "document" contain a vocabulary of size V. Given a document d and a term w, the TF-IDF representation is:
  $$\text{TF-IDF}(w, d) = \text{TF}(w, d) \times \text{IDF}(w)$$
  where:
  $$\text{TF}(w, d) = \frac{\text{Frequency of } w \text{ in } d}{\text{Total number of words in } d}$$
  and
  $$\text{IDF}(w) = \log\left(\frac{N}{1 + \text{D}(w)}\right)$$
  where N is the total number of documents and $\text{D}(w)$ is the number of documents containing the word w. Adding 1 to $\text{D}(w)$ prevents division by zero.
  Dimensionality: $N \times V$, where N is the number of documents and V is the vocabulary size.

For the music keyword dataset, the following points hold true:

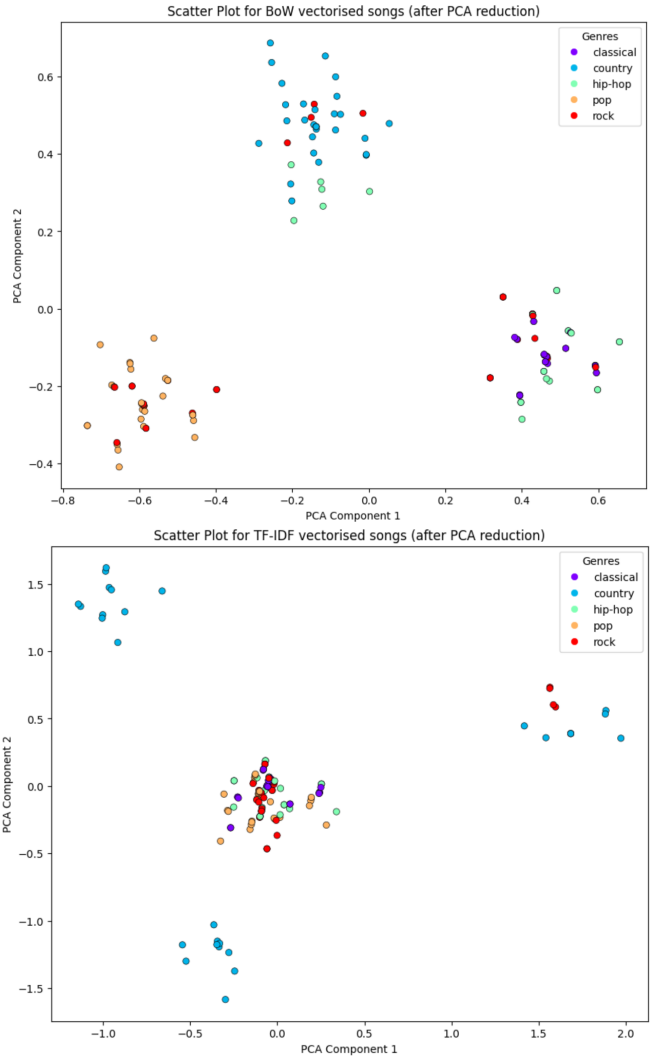
- "document" is equivalent to a song and its columns in our context.
- We will vectorise each column of keywords separately.
- The "vocabulary" of each column is :- 6 for keyword_1, 9 for keyword_2, and 10 for keyword_3. So we end up with a 6D, 9D and 10D vector for each song.
- There are 147 documents(songs)
- For BoW, since there is only one word per keyword column, $f_i = 1$ for the keyword present in the song. This means that our BoW vector is no different from a binary vector.
- Similarly, $TF = 1$ holds true for all keywords of all songs.

Choosing between the two: After comparing the scatter plots obtained from PCA-reduced embeddings, we opted for BoW due to the following observations:

The scatter plot generated from BoW vectors showed clearer and more distinct clusters with more meaningful genre populations compared to TF-IDF. This suggested that BoW better captured meaningful patterns in the data for the purpose of clustering.

Given that each song contains exactly three keywords, assigning equal importance to each keyword (as BoW does) appeared more effective than the weighted representation provided by TF-IDF.

Moreover, BoW vectors resulted in a more stable representation with less noise in the PCA-reduced space, improving visualization and clustering outcomes.



C. *Principal Component Analysis (PCA)*

The songs, which are now vectorised into vectors of higher dimensions (6, 9 and 10) are not yet suitable for clustering. To be suitable for clustering, they need to be reduced to vectors of lesser dimensions. Dimensionality reduction comes into play. Here we will reduce each keyword's vector into 2 dimensions, while at the same time trying to retain as much of the original patterns and variations as possible. Our method of choice is **PCA**, The process behind this implementation relies heavily on NumPy to calculate the eigenvectors and eigenvalues of the desired axes.
The steps are as follows:
1. **Mean Centering:** Each feature was centered by subtracting its mean to ensure the dataset was properly aligned for PCA.
2. **Covariance Matrix Computation:** We computed the covariance matrix to capture the relationships between features.It contains the pairwise covariances between features (in this case, the BoW values of the three keywords)

$$\Sigma = \frac{1}{n-1} X^T X$$

In PCA, the covariance matrix helps identify the directions in which the data varies the most. These directions are captured by the eigenvectors of the covariance matrix, with the corresponding eigenvalues representing the magnitude of variance along those directions.
3. **Eigen Decomposition:** Eigenvalues and eigenvectors were extracted from the covariance matrix to identify the principal components. Only the eigenvectors having the 2 highest eigenvalues are fit to be our axes.
4. **Projection:** The original data was projected onto the top two principal components, reducing its dimensionality to two.

$$X_{reduced} = X_{standard} W$$

Where W is the eigenvector along which X_standard is projected.

Transformations so far:
Hence, for example, if a particular keyword pertaining to a song's mood is first vectorised and then reduced to 2 dimensions, here are the transformations it goes through:

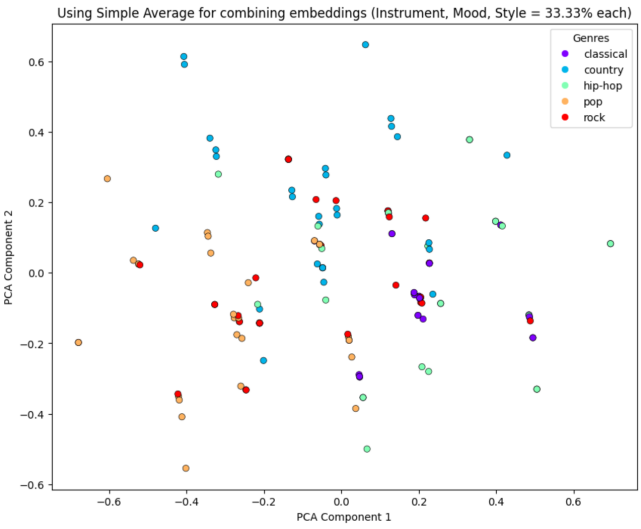
$$\text{Word} \xrightarrow{\text{BoW}} \mathbf{v}_{\text{BoW}} = [f_1, f_2, \ldots, f_V] \xrightarrow{\text{PCA}} \mathbf{v}_{\text{PCA}} = [p_1, p_2, \ldots, p_k]$$

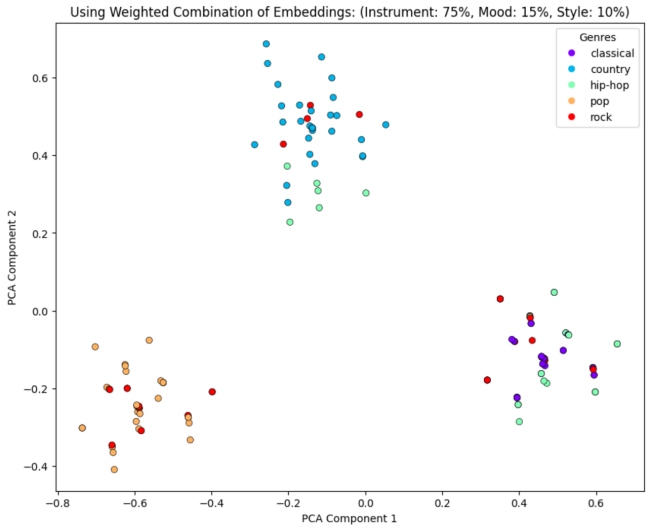
- *Word(mood):* "energetic"
- *BoW vector:* $\mathbf{v}_{\text{BoW}} = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]$
- *After PCA:* $\mathbf{v}_{\text{PCA}} = [-0.27745219, 0.74898762]$

.
Each song is now composed of three 2D vectors pertaining to each keyword type. For our purposes, we need to combine these three vectors into one 2D vector, making it easier for plotting and clustering applications.

D. *Combination of Feature Vectors*

This is the last step that involves data pre-processing, after which we move to clustering.
Initially, we used a simple averaging approach where the final 2D vector for each song was computed by taking the mean of the corresponding components of the three keyword vectors. **This method assumes that all keywords contribute equally to the overall representation of the song**. While simple and intuitive, this approach led to overlapping clusters and noise, as it failed to account for the relative importance of the different keywords.
To address this, we experimented with a weighted combination of the three 2D vectors. We assigned different weights to each keyword's PCA components based on their perceived significance.
Mathematically this means that in the final embedding for a song $emb = aV_1 + bV_2 + cV_3$, $a, b, \&c$ represent the weights of keywords 1, 2 and 3 respectively.
By emphasizing more meaningful keywords(such as "instrument" in this case), the weighted combination better represented the unique attributes of each song, allowing the clustering algorithm to group similar songs more effectively.



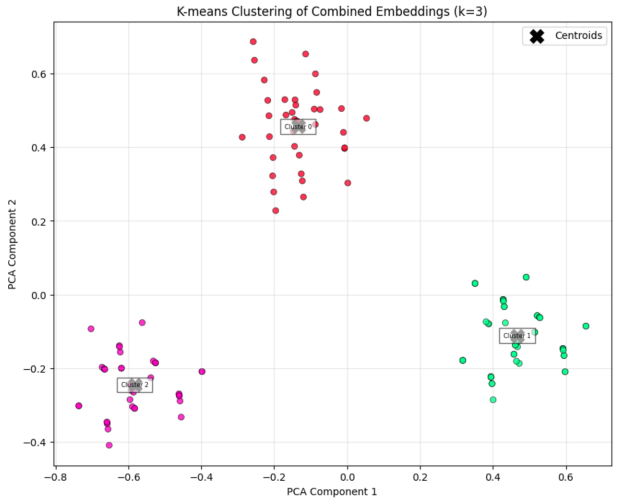


E. *Clustering*

Now we find suitable clusters in the data and analyse how well these clusters align with ground truth genre labels. We will be using the KMeans algorithm to achieve this

- **Initialization:** Randomly select k unique data points from the dataset as the initial centroids.
- **Assignment Step:** For each data point, calculate its distance to all centroids and assign it to the nearest one.
- **Update Step:** Compute the mean of all points assigned to each cluster to update the centroids. If a cluster receives no points, reinitialize its centroid randomly.
- **Convergence Check:** Compare the new centroids with the previous ones. If the centroids remain unchanged (within a small threshold), the algorithm stops. Otherwise, repeat the assignment and update steps.
- **Final Output:** The algorithm returns the final cluster centroids and the cluster labels for all data points.

The clustering results highlighted interesting patterns — certain clusters predominantly contained genres with similar mood or instrumental features, suggesting that the embeddings effectively captured meaningful relationships between songs. While the clusters didn't perfectly align with the genre labels, the overlaps often made intuitive sense, hinting at genre crossovers. This highlighted the complex nature of song classification, where rigid genre boundaries are sometimes difficult to maintain.



Choosing the Optimal Value of k:
The K-Means clustering algorithm requires specifying the number of clusters k beforehand. Choosing the best value of k is crucial because an inappropriate choice can lead to poorly defined clusters, either by underfitting (too few clusters) or overfitting (too many clusters). To identify the optimal value of k, we used the **Elbow Method** based on the **Inertia Cost Parameter**.

Inertia: The inertia measures how tightly the points are grouped around their centroids and is computed as the sum of squared distances from each data point to its assigned centroid. It serves as the cost function which we aim to minimise during KMeans. Mathematically, it is defined as:

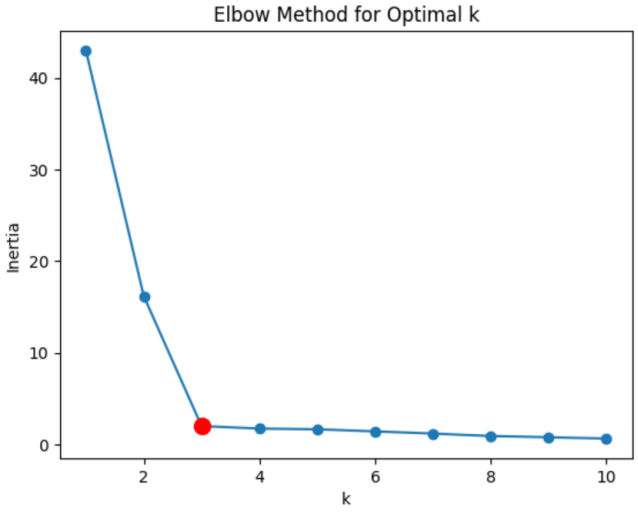
$$\text{Inertia} = \sum_{i=1}^{n} \|x_i - c_i\|^2$$

where:

- n is the number of data points,
- x_i is the i-th data point,
- c_i is the centroid of the cluster assigned to x_i,
- $\|\cdot\|$ denotes the Euclidean distance.

Lower inertia values indicate that points are closer to their centroids, suggesting better clustering quality.

The Elbow Method: To determine the best value of k, we plotted the inertia values against different values of k. The plot typically shows a sharp decrease in inertia initially, followed by a point where the decrease becomes gradual, forming an "elbow" shape. This "elbow" represents the optimal number of clusters, where adding more clusters no longer significantly reduces inertia.



Observations for Our Dataset: Our plot clearly showed an "elbow" at $k = 3$, indicating that three clusters were sufficient to capture the natural structure in the data. Choosing $k = 3$ ensured well-separated and meaningful clusters without introducing unnecessary complexity. This observation validated the structure of the dataset and highlighted distinct patterns within the embeddings.

F. *Evaluation and Analysis*

Now that we have clustered the data, we can proceed to the final part of the project: **Analysis**. In the next section, we will discuss various evaluation metrics to gain insights into the performance of our pipeline. Additionally, we will examine the dataset to observe how different keywords contribute to defining a "song." These observations will help justify our chosen vectorization method and the assigned weights used when combining embeddings.

3. ANALYSIS

In this section, we will discuss the following:

- **Justification for Vectorization Method:** Validation of why the Bag of Words (BoW) vectorization approach performed better than TF-IDF, based on clustering outcomes.
- **Embedding Combination Strategy:** Explanation of why assigning weights to the embedding components improved clustering performance compared to simple averaging.
- **Number of Clusters vs Number of Genres:** Analysis of the relationship between the chosen number of clusters and the number of unique genres in the dataset to assess whether clustering captures genre diversity effectively.
- **Evaluation Metrics:** Silhouette score and Purity.
- **Purity Table:** Presentation of a purity table to quantify how well the clusters match the genre labels.
- **Cluster Characteristics:** Analysis of the keywords present in each cluster to identify dominant patterns and align them with intuitive song groupings.
- **Classification of New Songs:** Method to assign a genre to new songs based on their keywords by identifying the nearest cluster.

A. *Justification for Vectorisation Method*

Observations: Our initial plots (page 2) compared the performance of BoW and TF-IDF for vectorizing song keywords. The clustering results using BoW produced clearer and more well-separated clusters than TF-IDF. This was evident from the scatter plots generated after PCA, where BoW embeddings displayed more distinct and meaningful groupings.

In the plot for TF-IDF, a specific genre (country) formed three small, isolated clusters on the exterior, while the remaining genres were densely intermixed in the center without clear boundaries. This lack of separation provided little meaningful insight into genre relationships. In contrast, the BoW plot revealed three well-defined clusters with noticeable genre diversity, offering a far more interpretable representation, which will be analyzed in greater detail later.

Dataset Characteristics: The dataset contained exactly three descriptive keywords per song, making frequency information less significant. TF-IDF's weighting mechanism unnecessarily penalized common keywords that were often contextually important. In contrast, BoW's simple binary presence-absence representation treated all keywords equally, resulting in cleaner and more meaningful embeddings.

Keyword Semantics: Since the dataset keywords inherently carried equal importance, TF-IDF's weighting skewed the significance of certain keywords, leading to noisier embeddings. BoW treated all keywords uniformly, better capturing the relationships between songs for clustering.

B. *Embedding Combination Strategy:*

To determine the best embedding combination strategy, we experimented with assigning higher weights to each keyword type individually. The results were as follows:

- Simple Averaging: This approach produced the noisiest and least defined clusters. The scatter plot showed significant overlap between genres, making it difficult to derive meaningful insights.
- Instrument Weighted Higher (75%): Assigning a higher weight to the instrumental keyword yielded the best clustering results. This approach produced more distinct clusters with better genre alignment, as reflected in the purity table. The clusters showed some partiality to specific genres, indicating that **instrumental characteristics played a major role in defining song similarities.**
- Mood or Style Weighted 75%: When mood or style keywords were given higher weights, the clusters became poorly defined, and the purity table showed significantly degraded results. The generic and subjective nature of these keywords likely contributed to this outcome, making them less reliable for clustering.

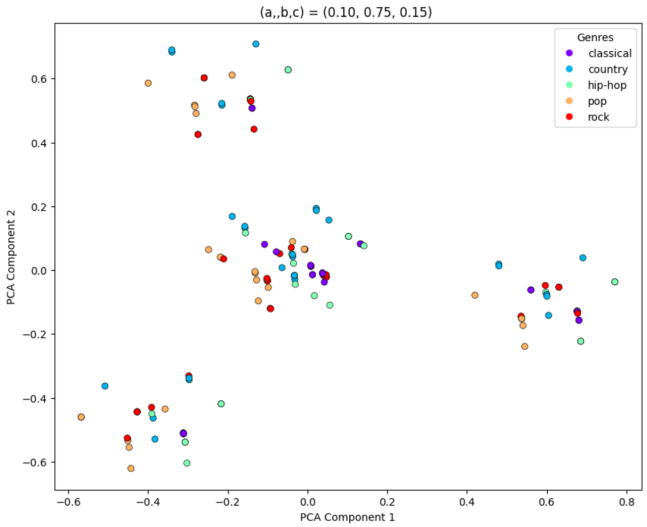
Hence the chosen coefficients for song embeddings:

$$(a, b, c) = (0.75, 0.15, 0.10)$$

Where,

$$emb = aV_{instrument} + bV_{mood} + cV_{style}$$

Below is the scatter plot when mood is assigned a high 75% weight, and its corresponding purity table. Notice how no conclusions about genre relations can be drawn from these clusters as no one genre maintains a noticeable majority in any cluster, making this combination of coefficients undesirable.



	classical	hip-hop	rock	country	pop
0	17.948718	20.512821	20.512821	19.658120	21.367521
1	35.714286	14.285714	14.285714	21.428571	14.285714
2	6.250000	25.000000	25.000000	18.750000	25.000000

C. *Number of clusters vs genres*

The dataset contains five distinct genres, but the optimal number of clusters determined using the elbow method was three. This discrepancy suggests that genres are not perfectly separable based on the given keywords. Instead, some genres likely share similar characteristics, causing them to group naturally into fewer clusters. Despite this, the three-cluster solution captured meaningful relationships between songs and provided better-defined groupings compared to higher values of k.

D. *Evaluation Metrics*

The Silhouette Score evaluates the quality of clustering without requiring ground truth labels. It measures how well each data point fits within its assigned cluster compared to other clusters. The score is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

where:

- $a(i)$: The average distance between point i and all other points in the same cluster.
- $b(i)$: The average distance between point i and all points in the nearest different cluster.

The Silhouette Score ranges from -1 to 1, where higher values indicate better-defined and well-separated clusters. Since this metric is unsupervised, it does not rely on genre labels.

```
Silhouette Score = 0.8250
```

Purity Table: A Purity Table evaluates how well clusters align with ground truth labels (genres), making it an extrinsic evaluation method that relies on external information. Each row represents a cluster, and each column represents a genre, with table entries showing the number of songs from each genre in each cluster. Unlike intrinsic methods such as the Silhouette Score, which assess clustering based on internal data structure, the Purity Table provides insights by comparing clusters to predefined categories.

E. *Purity Table*

CLUSTER-TO-GENRE PURITY TABLE (%)

Cluster	Classical	Country	Pop	Rock	Hip-Hop
0	69.23%	15.38%	15.38%	0.00%	0.00%
1	0.00%	35.38%	18.46%	46.15%	0.00%
2	0.00%	0.00%	30.23%	0.00%	69.77%

- **Cluster 0:** Dominated by classical music (69.23%), with minor contributions from country and pop (15.38% each). The absence of rock and hip-hop indicates a clear separation for more traditional or instrumental genres.
- **Cluster 1:** Shows a more balanced composition, with rock (46.15%) as the dominant genre, followed by country (35.38%) and pop (18.46%). This cluster likely represents genres with overlapping characteristics or hybrid styles.
- **Cluster 2:** Strongly associated with hip-hop (69.77%), followed by a smaller presence of pop (30.23%). The absence of other genres suggests a distinct grouping for modern, high-energy tracks.

F. *Cluster Characteristics*

The purity table and the provided dataset led to the following observations:

- **Cluster 1: Rock and Country Overlap** Rock and country songs frequently share guitar as a primary instrument, which played a significant role in this cluster since instruments were given a higher weight during embedding combination. The overlap suggests that the guitar's prominence bridges the characteristics of these genres, especially in songs with energetic or twangy descriptors.
- **Cluster 0: Classical with Sparse Contributions from Pop and Country** Classical music is dominated by instruments like brass and violin, often accompanied by melodic and calm descriptors. This cluster maintained a strong separation due to these unique keywords. Sparse contributions from pop or country likely occurred in songs with overlapping descriptors like "melodic" or "slow."
- **Cluster 2: Hip-Hop Dominance with minor Pop overlap** This cluster had a clear dominance of hip-hop, defined primarily by synth as the primary instrument and energetic, rhythmic descriptors. There was minimal overlap, reinforcing the distinct identity of hip-hop within the dataset.
- **Observations on Mood and Style Contributions** Assigning higher weights to mood or style keywords produced noisy clusters since descriptors like "happy," "melodic," or "energetic" were present across multiple genres. **These findings underscore the importance of emphasizing instruments for better-defined clusters.**

G. *Classification of New Songs*

Classifying new songs based on the provided clustering model is challenging due to the lack of distinct genre boundaries in the dataset. The clusters formed during analysis show significant genre overlap, particularly between rock, country, and pop, as these genres often share similar instrument and mood descriptors. This suggests that assigning an explicit genre label to a new song may not always be feasible.

A more practical approach would be to identify the nearest cluster in the PCA-reduced space and infer genre tendencies based on the dominant genre within that cluster. This approach acknowledges the inherent complexity of genre classification, where songs often exhibit characteristics of multiple genres, emphasizing similarity-based grouping rather than rigid genre assignments.

4. CONCLUSIONS

In this project, we successfully implemented a music genre clustering pipeline using Bag of Words vectorization, dimensionality reduction with PCA, and K-Means clustering. Through careful analysis and evaluation, we demonstrated the effectiveness of weighted embedding combinations and highlighted the challenges of genre overlaps in the dataset.

Key insights include:

- Instrumental features played a crucial role in defining meaningful clusters, while mood and style descriptors contributed noise when overemphasized.
- The final cluster configuration aligned well with genre patterns, though perfect separability was limited by overlapping characteristics.
- The Purity Table and Silhouette Score highlighted the effectiveness of the approach, with distinct clusters and meaningful groupings.
- Classifying new songs remains challenging due to the lack of rigid genre boundaries; a similarity-based approach using the purity table proved more suitable.

Further improvements could involve exploring more sophisticated vectorization techniques or incorporating external metadata to better capture nuanced song characteristics.

The entire pipeline of this project is as follows:

$$\text{Raw Dataset} \xrightarrow{\text{Preprocessing}} X_{\text{keywords}} \xrightarrow{\text{Vectorization (BoW)}} X_{\text{vectors}}$$

$$X_{\text{vectors}} \xrightarrow{\text{PCA (Dimensionality Reduction)}} X_{\text{reduced embeddings}} = X_{PCA}$$

$$X_{\text{PCA}} \xrightarrow{\text{K-Means Clustering}} \{C_1, C_2, \ldots, C_k\} + labels \xrightarrow{\text{Evaluation}} \text{Purity Table, Silhouette Score}$$

Pipeline for New Keyword to Embeddings and Genre Tendencies:

$$\text{New Song Keywords} \xrightarrow{\text{Vectorization (BoW)}} W_{\text{new vector}}$$

$$W_{\text{new vector}} \xrightarrow{\text{PCA Projection}} W_{\text{new PCA}} \xrightarrow{\text{Nearest Cluster Assignment}} \text{Genre Tendencies}$$

5. GITHUB REPOSITORY

The complete project code, including the Jupyter notebook, dataset, and various plots, is available on GitHub. The repository can be accessed at the following link:

`https://github.com/brecon75/genre_clf`

This repository contains instructions on how to run the code and reproduce the results presented in this report.