ISYE 7406 – Data Mining and Statistical Learning

Project Report on

Music Recommendation System

By

Group 136

Team Members

Arya Divakaran Kalappurayil – 272 – akalappurayil3@gatech.edu

Diane Alice Laure Egret – 936 – degret3@gatech.edu

Linggan Chen – 535 – lchen740@gatech.edu

Submitted on

25th November 2024

Georgia Institute of Technology

Index

Abstract

This project presents the development of a personalized music recommendation system aimed at improving music discovery for users on streaming platforms such as Spotify. The system combines clustering and classification techniques to provide relevant and diverse song recommendations based on key audio features like tempo, danceability, energy, and valence. Initially, K-Means clustering was employed to group songs into distinct clusters based on these features, with an optimal cluster number determined through the Elbow Method and Silhouette Score. The resulting clusters capture different musical patterns, such as high-energy, danceable tracks or slower, acoustic songs. Subsequently, a K-Nearest Neighbors (KNN) classifier was trained to predict the cluster labels for new songs, achieving an impressive accuracy of 95.59%, with a Kappa score of 0.9365, indicating strong model performance. The recommendation system then identifies songs similar to a given input based on its audio characteristics, providing personalized song suggestions. The results highlight the potential of combining clustering and classification approaches to provide scalable, personalized music recommendations that not only align with user preferences but also encourage music discovery across a broad spectrum of genres and moods. This work underscores the value of advanced data mining techniques in enhancing the music discovery process for streaming service users.

Introduction

As music streaming platforms like Spotify continue to grow, users are confronted with the challenge of discovering songs that align with their unique tastes. While existing recommendation algorithms rely heavily on user listening history, these models often fall short in capturing the evolving and nuanced nature of musical preferences. This project aims to address this gap by developing an advanced music recommendation system that combines clustering and classification techniques to deliver both relevant and diverse song suggestions. By analyzing key audio features such as tempo, danceability, energy, and valence, the system groups songs into distinct clusters, facilitating the identification of patterns that lead to more accurate and personalized recommendations.

The main challenge in this task lies in the high dimensionality of music data, where numerous features such as loudness, speechiness, and energy, interact in complex ways to define a song's style and appeal. Additionally, user preferences are sparse and vary significantly, requiring sophisticated methods to identify meaningful patterns. To address these challenges, this project employs K-Means clustering to segment songs based on shared characteristics, followed by K-Nearest Neighbors (KNN) classification to recommend songs based on their proximity to one another. These methods are further optimized through cross-validation and hyperparameter tuning, ensuring robust performance and high recommendation accuracy.

By leveraging this hybrid approach, the system not only enhances the personalization of song recommendations but also promotes diversity in the suggestions, helping users discover music they might not encounter through traditional algorithms. The combination of clustering and classification offers a powerful and scalable solution to music recommendation, creating a more engaging and dynamic discovery experience. Ultimately, this project aims to transform how users interact with music streaming platforms, providing tailored suggestions that better reflect their evolving tastes and preferences.

Data Sources

The dataset for this analysis is sourced from Kaggle's public repository. It specifically focuses on Spotify songs and offers a comprehensive overview of various attributes and metrics related to songs on the Spotify platform.

Link to dataset: https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset

The dataset is large, with over 100,000 entries, each representing a unique track on Spotify. It is structured in a tabular format, where each row corresponds to a song, and each column represents an individual feature. A description of each feature in the dataset can be found in below table:

Variable	Description
track_id	The Spotify ID for the track
artists	The names of the artists who performed the track, separated by ';' if there is more than one artist
album_name	The name of the album in which the track appears
track_name	The name of the track
popularity	The popularity of the track on a scale from 0 to 100, based on the total number of plays and recency
duration_ms	The length of the track in milliseconds
explicit	Indicates whether the track has explicit lyrics (true = yes, false = no or unknown)
danceability	Describes how suitable a track is for dancing, with 0.0 being least danceable and 1.0 being most danceable
energy	A measure from 0.0 to 1.0 representing the intensity and activity of the track
key	The key the track is in, mapped to pitches using standard Pitch Class notation
loudness	The overall loudness of the track in decibels (dB)
mode	Indicates the modality (major = 1, minor = 0) of a track's melodic content
speechiness	Detects the presence of spoken words in a track, with values above 0.66 likely indicating spoken words
acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic
instrumentalness	Predicts whether a track contains no vocals, with values closer to 1.0 indicating instrumental tracks

liveness	Detects the presence of an audience in the recording, with higher values indicating a greater likelihood of a live performance
valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track
tempo	The overall estimated tempo of a track in beats per minute (BPM)
time_signature	An estimated time signature, indicating how many beats are in each bar
track_genre	The genre in which the track belongs

Table 1: Dataset description

The dataset is highly relevant as it provides a wide range of features, including song duration, tempo, key, danceability, and genre, which are crucial for understanding the diverse preferences of music listeners. By leveraging these attributes, the dataset allows for the development of advanced algorithms that can predict and recommend songs that align with individual tastes. This capability is essential for enhancing user experience by ensuring that users receive personalized and engaging music suggestions.

Exploratory Data Analysis

The dimension of the dataset revealed that there are 1.14K observations in total for 21 columns. The dataset includes various musical features, with track_id serving as a unique identifier for each track. Key categorical variables such as key, mode, time_signature, and track_genre are converted to factors to appropriately represent discrete categories. The explicit column, which originally contains Boolean values (TRUE/FALSE), is transformed so that TRUE becomes 1 and FALSE becomes 0.

As part of the data preprocessing process, we first confirmed that there were no missing values in the dataset, ensuring that all columns were fully populated. Duplicate entries were then removed based on the track_id to ensure that each track is uniquely represented. We also conducted checks on the unique values in the explicit column both before and after cleaning to ensure the transformation was applied correctly.

The summary statistics of the dataset indicated a need for scaling of the numeric values. Skewness values for most numeric variables were notably high, suggesting that their distributions are asymmetric. This observation indicates that some transformations may be necessary to normalize the data and make it more suitable for further analysis, including clustering.

The distribution analysis of the numeric variables is performed by using histograms, as shown below:

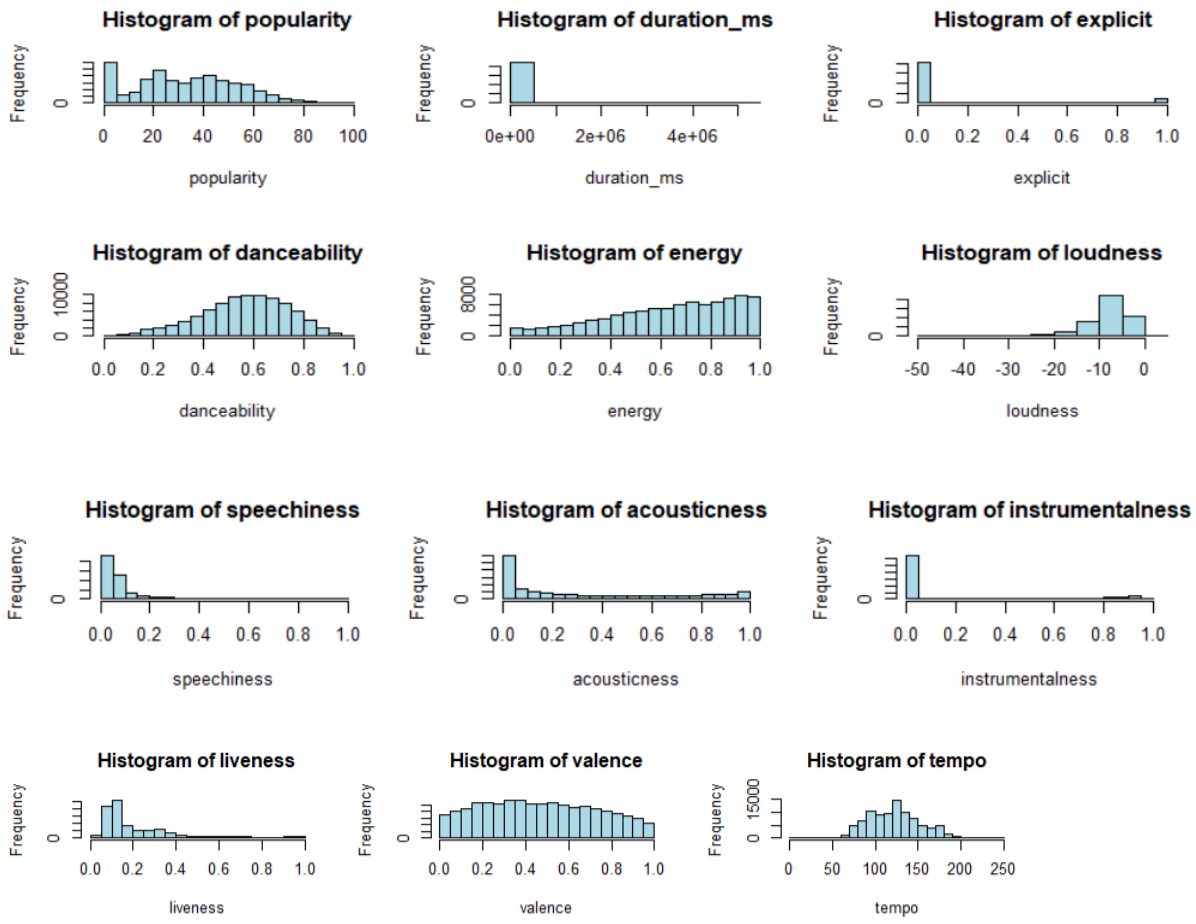


Figure 1: Histogram of all the variables

Several interesting patterns were observed. A significant number of songs have a popularity value of 0, while the remaining values follow a distribution close to normal, although there are no high popularity values. The duration_ms column shows that most songs have a duration of less than 50,000 ms (or 8 minutes). The danceability and liveness columns exhibit right-skewed normal distributions, while valence follows a normal distribution with heavy tails. The loudness variable is left-skewed, with most songs having noise levels between -15 and -5 dB, and tempo appears to follow a well-formed normal distribution. Energy is uniformly distributed between 0 and 1, and most columns have values that fall within this same range.

Furthermore, the skewness values for some numeric variables say duration_ms are notably large(11.07), indicating that the data is asymmetric and may require transformations to normalize the distributions for further analysis.

Next, the correlation matrix was plotted to inspect relationships between the variables. Most of the columns exhibit very low correlations with each other, indicating that the variables are largely independent. However, there are a few notable correlations. The strongest positive correlation is between energy and loudness (0.759), which makes sense as higher energy levels

are typically associated with louder songs. On the other hand, energy and acousticness show a strong negative correlation of -0.733, reflecting the fact that songs with higher energy tend to have lower acousticness. Similarly, loudness and acousticness are negatively correlated (-0.583), supporting the idea that louder songs generally have less acoustic (soft, natural) sound. Additionally, danceability and valence are positively correlated, which is intuitive, as songs with a higher valence (a more positive or upbeat mood) are often more danceable. Finally, loudness and instrumentalness show a negative correlation of -0.435, which aligns with the fact that instrumental tracks are often quieter than those with vocals. Moreover, these results are confirmed by the ranked cross-correlation plot in Appendix A Figure 1.

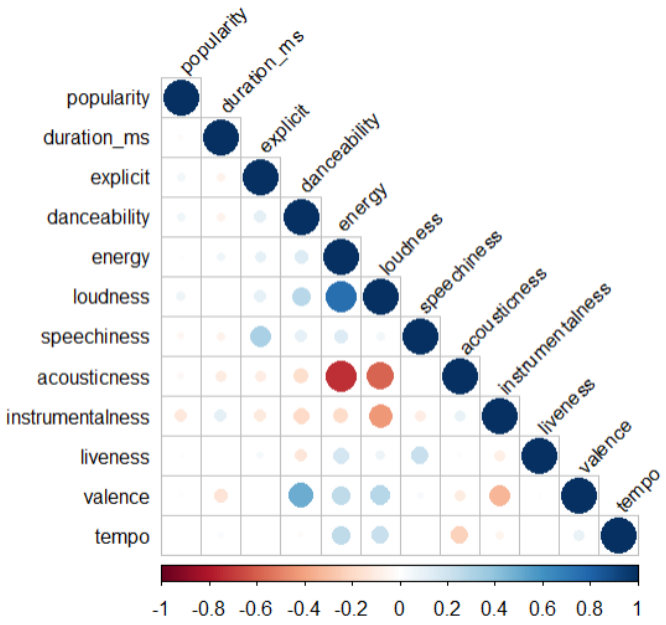


Figure 2: Correlation Plot

From the Scatterplot in Appendix A Figure 2, energy and loudness seem to have a positive linear relationship. Also, boxplots (Appendix A Figure 3) were used to examine the distribution of various variables across different categories. Variations in popularity across modes, keys, time signatures, and explicitness reveal potential outliers in the data. Songs with a 4/4 time signature are seen as the most popular, while tracks labeled as explicit (explicit = 1) also exhibit some potential outliers in popularity. Popularity across different keys shows overlapping medians with a few outliers, and both modes display similar popularity values. These patterns suggest that while certain factors influence popularity, there are notable outliers that may require further investigation.

To verify the potential outliers identified in the box plots, the IQR method was applied for more precise outlier detection. The corresponding output screenshot is available in Appendix A Figure 4. The variables *Popularity*, *Energy*, *Acousticness*, and *Valence* show either zero or a negligible number of outliers. In contrast, the other columns contain a significant number of outliers,

which is expected given that these are continuous variables. They exhibit a wide range of values, do not follow a normal distribution, and are influenced by various aspects of a song. Therefore, it is reasonable not to remove these outliers, as they add to the richness and complexity of the dataset, capturing the diverse nature of musical characteristics.

A detailed exploration was carried out to better understand the genres. The dataset includes 113 genres, with average popularity scores calculated for each. The highest average popularity observed is around 60, with little variation among the top 10 genres, indicating that the most popular genres are relatively similar in terms of popularity. Notable genres include K-pop, Pop-Film, Metal, and Chill, each showing distinct characteristics. In terms of explicit content, the *Comedy* genre has the highest ratio of explicit songs, likely due to the genre's use of humorous, satirical, or even explicit language to enhance its comedic effect. Conversely, genres like *Emo* and *Sad*, which focus on deep emotional expression, may use explicit language to amplify the emotional impact of their messages.

Further analysis on the correlation between *Loudness* and *Energy* in the top genres reveals a strong relationship, suggesting that songs with higher loudness are generally associated with higher energy in these popular genres.

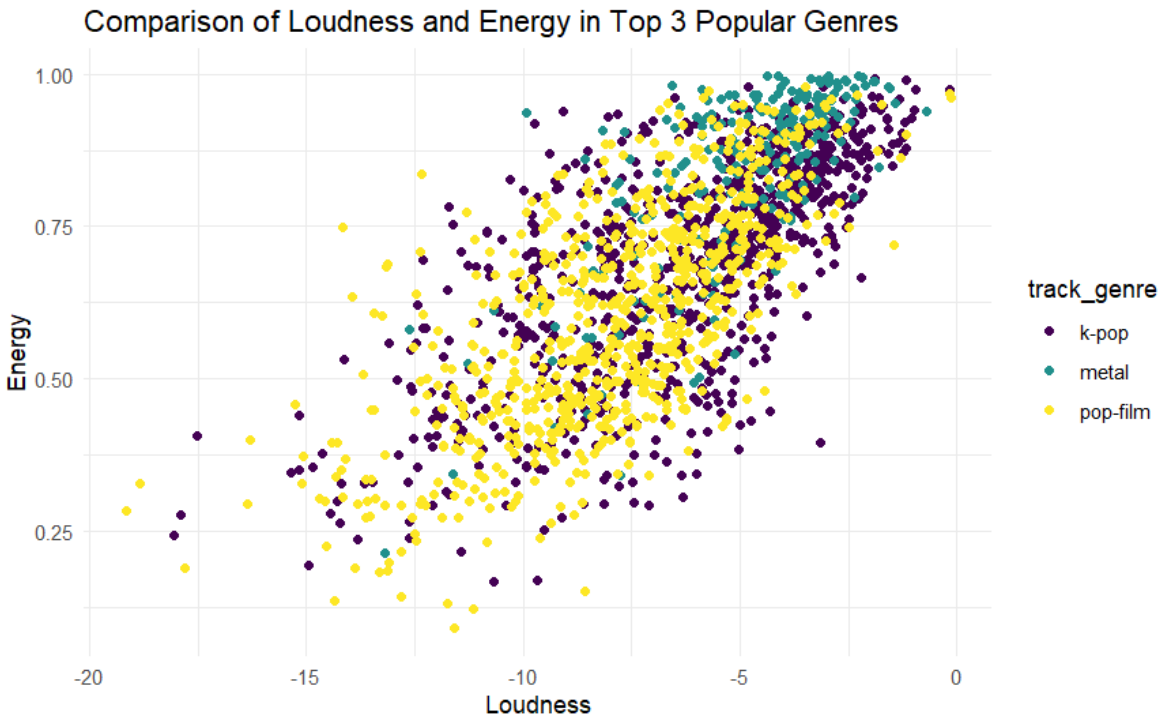


Figure 3: Loudness Vs Energy

Additionally, when energy values were divided into 10 equal intervals, the distribution (Appendix A Figure 5) showed that higher energy songs are more favored by Spotify listeners, supporting findings from the energy histogram. Similarly, speechiness values (Appendix A Figure 6), when split into 10 intervals, corroborated the histogram results, indicating that songs with

lower speechiness tend to be more popular. This suggests that Spotify's catalog is more likely to feature songs with less spoken content, in line with listener preferences.

Methodology

We start with a clustering approach, where we use scaled numerical features such as danceability, energy, tempo, and valence among others to characterize each song.

First, K-Means clustering is used to group similar songs together. K-Means clustering is a widely used partitioning algorithm that assigns each data point to one of k clusters, where each point belongs to the cluster with the nearest mean (centroid).

Determining the optimal number of clusters (k) is critical in K-Means. We evaluated k values from 1 to 15 using the following methods:

1. **Elbow Method**: The Elbow Method using the function fviz_nbclust(), which determines and visualizes the optimal number of clusters using the within cluster sums of squares method. Due to memory limitations, "Error: cannot allocate vector of size 60.0 Gb", we applied this method to a random 10% subset of the full data.
2. **Silhouette Score**: We calculated the silhouette score on a subset of the data. However, due to dataset size, we could not compute this for the full dataset. The method was applied to a random 10% subset of the full data.
3. **Custom Elbow Plot Function**: Since the 10% subset data may not represent the full data. We implemented an alternative function to calculate the total within-cluster sum of squares for each k and plotted it to identify the elbow using the full data. Despite warnings related to memory constraints on the full dataset, the output shows nearly identical results to the first method.

Note: Due to memory limitations, only 10% of the data can be used on the elbow plot and silhouette score. However, the final clustering was done using the full data.

After the optimal clusters k is determined, we evaluate cluster quality using the following metrics:

- **Total Within-Cluster Sum of Squares (tot.withinss)**, indicating compactness; lower values are better.
- **Between-Cluster Sum of Squares (betweenss)**, indicating separation; higher values are better.
- **Cluster Sizes**, which indicate how balanced each cluster is.

As an alternative clustering method, we used DBSCAN, a density-based technique that groups points based on density and labels sparse regions as noise. DBSCAN is particularly valuable for data with complex structures, as it can identify clusters of varying shapes and sizes.

DBSCAN requires two primary parameters:

1. **Epsilon (eps)**: This represents the maximum distance within which points are considered neighbors. We used a k-Nearest Neighbor (kNN) distance plot to identify the optimal eps value.
2. **Minimum Points (minPts)**: This defines the minimum number of points required to form a dense region. Cross-Validation is initially tested to choose the optimal minPts, however the function takes forever to run due to a large dataset. Hence, based on the common rule of twice the number of features in the dataset, we set minPts to 24, as our dataset has 12 numeric variables.

The results allowed us to compare the DBSCAN to K-Means, evaluating whether DBSCAN could provide more balanced clusters or reveal unique cluster shapes.

After clustering, each song is assigned a cluster label, which becomes the target variable for our classification model.

We then proceeded with a classification model. To create a manageable subset of data for model training and testing, a random sample of 30% of the full dataset was extracted. Recall that only 10% of data was previously used for clustering. With this classification, we were able to increase data usage to 30% because the process was less computationally expensive. This sampled data was then divided into training and testing sets, with 70% allocated for training and 30% for testing. Stratified sampling was used during this step to maintain the proportional distribution of clusters across both sets.

A **K-Nearest Neighbors (KNN)** classification algorithm was employed to predict the cluster membership of songs. The model was optimized by tuning the *k* parameter, which determines the number of nearest neighbors considered in the KNN algorithm. Ten different *k*-values were tested through cross-validation to identify the value that provided the highest accuracy. This approach ensured the model achieved a balance between capturing patterns in the data and avoiding overfitting. Accuracy metrics for different *k*-values are provided in Appendix A Figure 9. Following training, the model was evaluated using a confusion matrix, which provided metrics such as accuracy, sensitivity, and specificity. These metrics were used to assess how well the KNN model could classify songs into their appropriate clusters.

In addition to clustering and classification, a song recommendation system was implemented to provide tailored music suggestions. The system worked by identifying the nearest neighbors of a given input song using its numerical attributes, such as popularity, tempo, acousticness, etc. These attributes were compared to the training dataset using the K-Nearest Neighbors

algorithm to find songs with similar profiles. Once the nearest neighbors were identified, detailed information about the recommended songs, including their artist, album name, track name, and genre, was extracted and displayed for interpretation. To ensure the system's robustness and effectiveness, this recommendation process was repeated with multiple input songs, confirming its ability to consistently generate meaningful and relevant suggestions.

Analysis & Results

The combined clustering and recommendation results provided valuable insights into the dataset, such as identifying similar songs and exploring trends based on genre and other attributes.

Clustering Analysis

The choice of k (number of clusters) for K-Means is guided by the elbow method and silhouette score. From the plots (elbow method and silhouette score), the bend in the curve occurs around k = 3 or 4. See additional elbow and silhouette score plot in Appendix A: Figure 7 & 8.

Figure 4: K-means with Clusters K = 3

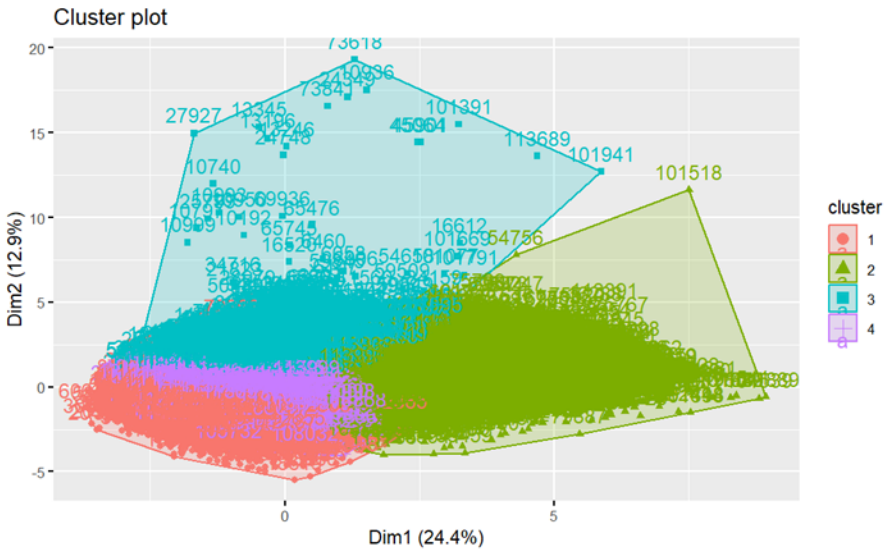


Figure 5: K-means with Clusters K = 4

The following are the outputs from the K-Means model.

Model	tot.withinss	betweenss	cluster size
K-Means k = 4	735,324	341,556	[8009, 18477, 26881, 36374]
K-Means k = 3	800,989	275,891	[60628, 21085, 8028]

Table 2: Clustering Result

- **Total Within-Cluster Sum of Squares (tot.withinss):** The model with 4 clusters has a lower total within-cluster sum of squares, meaning the data points within clusters are more tightly grouped compared to the 3-cluster model.

- **Between-Cluster Sum of Squares (betweenss):** The 4-cluster model has a higher between-cluster sum of squares, indicating better separation between the clusters compared to the 3-cluster model.

- **Cluster Sizes:** The 3-cluster model (km3) has one cluster with a very large number of points (60628), while the 4-cluster model (km4) distributes the points more evenly across clusters. More evenly distributed clusters can often be preferred, depending on the use case.

- **Number of Clusters (k):** km4 (4 clusters) seems to be the better model based on both the total within-cluster sum of squares (lower is better) and between-cluster sum of squares (higher is better). Additionally, the cluster sizes in km4 are more balanced than in km3, which could be beneficial since we want songs to be more diverse.

For the alternative clustering method, we applied DBSCAN with parameters eps = 2 and minPts = 24. The eps value was selected based on a KNN distance plot (k = 10), which indicated that an optimal eps was around 2. For the minPts parameter, we followed the general guideline of setting it to twice the number of features in the dataset, resulting in a value of 24, given our 12 numeric variables.

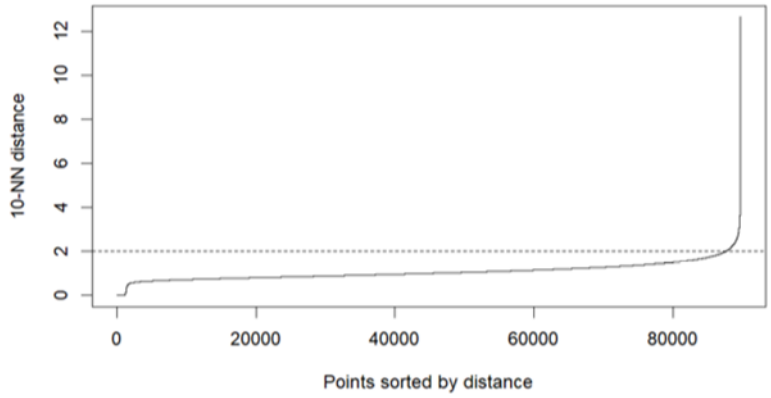


Figure 6: KNN Distance Plot

The DBSCAN results identified five clusters, but the cluster sizes were highly imbalanced, with one cluster containing over 80,000 data points while another contained only 47, limiting its applicability for this recommendation system. Different values of eps and minPts are also tested, but it either takes forever to run, or results in similar imbalanced clustering. Hence, we conclude that DBSCAN is not the most appropriate model for this dataset.

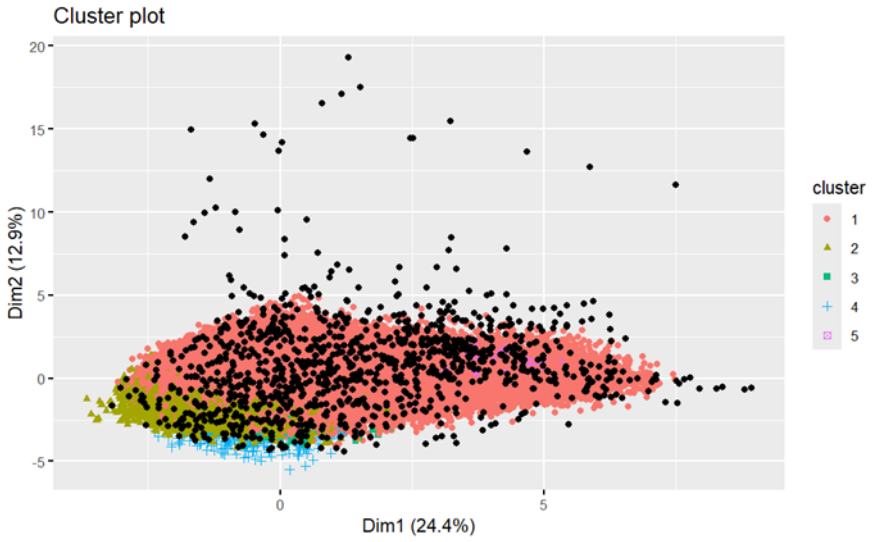


Figure 7: DBCSAN Plot

In summary, K-Means clustering with k = 4 has better compactness and separation, and more balanced cluster sizes. This clustering structure will serve as the basis for our music

recommendation system, where songs are recommended to users based on their proximity within these clusters, thus ensuring personalized yet diverse music suggestions.

However, in real-world scenarios, we cannot cluster all music into only 4 groups because it will be overly simplistic, given the vast diversity in music genres and styles.

Cluster Characteristics on K-MEANS with k = 4:

Cluster 1: Popular, explicit, energetic songs with moderate danceability and more speechiness.

Songs in cluster 1 are popular, often have explicit lyrics, are energetic, and feature more spoken words, ex: hip-hop and rap music. These genres also tend to have moderate danceability, making them fit for this group.

Cluster 2: Less popular, clean, acoustic, quieter songs with slower tempo.

Songs in cluster 2 are quieter, acoustic-driven songs that are often slower and more mellow, fitting genres like folk or acoustic music. These songs tend to have a more relaxed or intimate feel.

Cluster 3: Moderately popular, high-energy, longer songs with some live performance elements.

Songs in cluster 3 have high energy and longer track durations pointing to genres like rock or EDM, where songs can be intense and extended, often featuring live performance elements or crowd interactions.

Cluster 4: Fairly popular, danceable, energetic songs with a positive mood and no explicit content.

Songs in cluster 4 are upbeat, very danceable, and generally positive in mood, ex: dance-pop or disco music. The lack of explicit content suggests mainstream appeal.

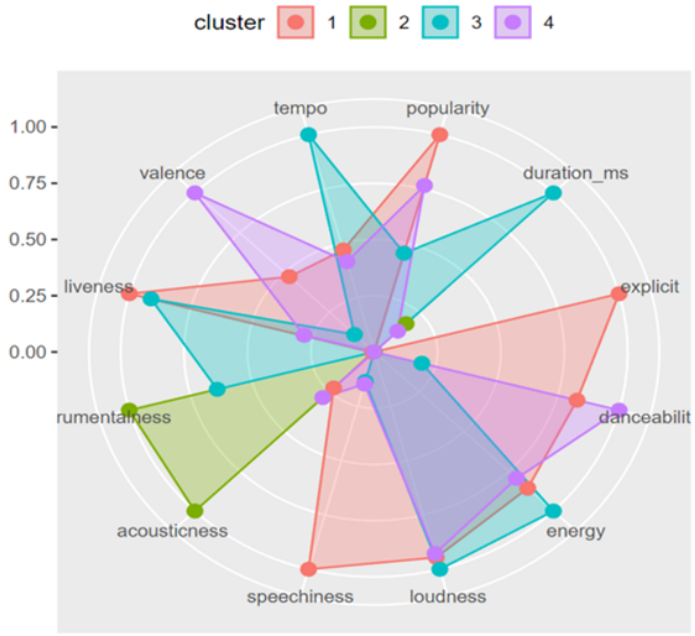


Figure 8: Cluster Characteristics

Classification Analysis

The k-Nearest Neighbors (KNN) model was developed to predict song clusters based on their musical attributes. The model was tuned across different values of *k* (number of neighbors) to identify the optimal configuration, which was k = 23. As seen in Table 3, k = 23 did not maximize raw accuracy, but did have the highest Kappa score, lowest accuracy standard deviation (SD) and an accuracy only slightly lower than other values of k. Its selection aligns with the model tuning goal of balancing performance with stability and robustness. This choice ensures that the model generalizes well to unseen data and avoids being overly sensitive to specific subsets of the training data.

k	Accuracy	Kappa	Accuracy SD	Kappa SD
5	0.9334966	0.9042925	0.0022872	0.0033806
7	0.9382576	0.9111416	0.0025896	0.0037505
9	0.9415755	0.9159021	0.0023667	0.0034995
11	0.9446543	0.9203260	0.0020896	0.0031032
13	0.9466835	0.9232398	0.0025822	0.0037879
15	0.9486369	0.9260552	0.0030338	0.0043568
17	0.9498671	0.9278194	0.0024295	0.0034901
19	0.9500267	0.9280448	0.0022911	0.0032855
21	0.9510298	0.9294865	0.0026162	0.0037789
23	0.9518353	0.9306437	0.0021450	0.0031300

Table 3: KNN Parameter Tuning Metrics across 10 values of k

The model was evaluated on a separate test dataset using a confusion matrix, yielding the following key results shown in Table 4 and Figure 9:

- Overall Accuracy: The model achieved an impressive accuracy of 95.59%, meaning it correctly classified most songs into their respective clusters.
- Class-Specific Metrics:
    - Sensitivity (the proportion of correctly identified cluster members) was very high across clusters, ranging from 93.34% to 99.86%.
    - Specificity (the ability to exclude songs that do not belong to a cluster) ranged from 95.87% to 99.96%.
    - Precision (Positive Predictive Value) and recall metrics were similarly strong, ensuring a balanced performance across all clusters.

These results highlight that the KNN model effectively learns the underlying patterns in the dataset and can accurately classify songs into clusters.

Metrics	Class 1	Class 2	Class 3	Class 4
Sensitivity	0.99862	0.9465	0.9334	0.9677
Specificity	0.99959	0.9941	0.9794	0.9587
Pos Pred Value	0.99588	0.9765	0.9504	0.9413
Neg Pred Value	0.99986	0.9862	0.9720	0.9775
Prevalence	0.09003	0.2062	0.2973	0.4064
Detection Rate	0.08991	0.1952	0.2775	0.3933
Detection Prevalence	0.09028	0.1999	0.2920	0.4178
Balanced Accuracy	0.99911	0.9703	0.9564	0.9632

Table 4: KNN Model Statistics by Class with Best Tune

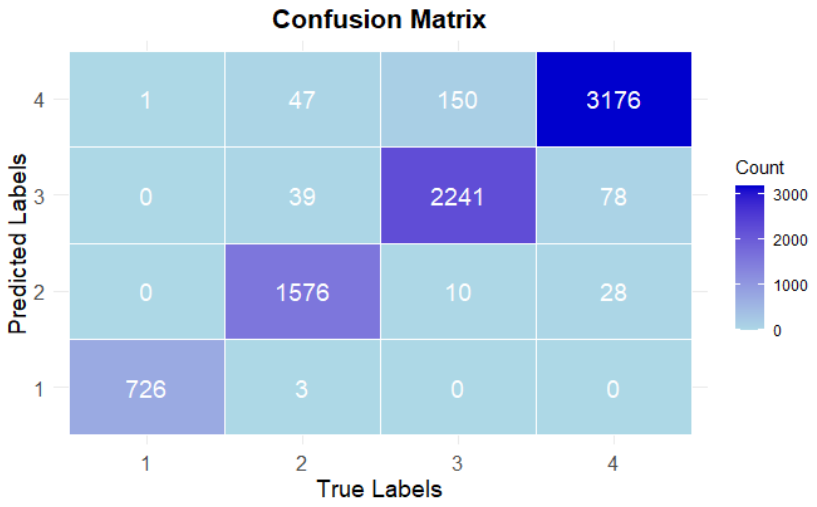


Figure 9: KNN Model Confusion Matrix

Song Recommendation

To demonstrate the model's application, a song from the test dataset was selected as the input (Table 5). Note that additional recommendation examples are available in Appendix B. Here, we chose Johann Sebastian Bach's Suite No. 3, a beautiful piece of classical music that I recommend you give a listen. Using the KNN algorithm, 10 similar songs were identified from the training dataset based on their proximity to the input song in the feature space. The values for all numerical variables of these recommended songs are displayed in Table 6; along with the naming details of these in Table 7.

The recommended songs shared high similarity in key attributes like low energy, low danceability and low loudness, but high acousticness. Additionally, all recommended songs belonged to the same cluster as the input song, confirming that the KNN model effectively identifies and retrieves relevant songs.

artists	album_name	track_name	track_genre	cluster
Johann Sebastian Bach; Daniel Hope; Lorenza Borrani; Lucy Gould; Stewart Eaton; William Conway; Enno Senft; Stefan Maass; Stefan Rath	Classical Christmas	Suite No. 3 in D Major, BWV 1068: II. Air	classical	2

popularity	duration_ms	explicit	danceability	energy	loudness	speechiness	acousticness	instrumentalness	liveness	valence	tempo
-1.613	0.639	-0.306	-2.225	-2.282	-2.856	-0.432	1.855	-0.483	-0.344	-1.482	0.465

Table 5: Example Input Song – Johann Sebastian Bach

index	popularity	duration_ms	explicit	danceability	energy	loudness	speechiness	acousticness	instrumentalness	liveness	valence	tempo	cluster
7721	-1.2243649	0.8732462	-0.3064438	-1.585624	-1.997069	-2.844207	-0.3393628	1.7489641	-0.4264763	-0.5848058	-1.6433521	1.0769370	2
17020	-1.4673095	0.9601271	-0.3064438	-1.217750	-2.012657	-2.430915	-0.4046894	1.8878855	-0.3465005	-0.5078371	-0.8083182	0.1235720	2
11785	-1.6130762	0.0784273	-0.3064438	-1.755411	-1.958099	-1.635931	-0.4099861	1.8731066	-0.3715122	-0.4462621	-1.6007445	0.3279712	2
12369	-1.6130762	0.1068741	-0.3064438	-1.421495	-1.389131	-2.456961	-0.4108689	1.8967528	-0.4317257	0.1130439	-1.0860288	0.0166243	2
23203	-1.6130762	-0.3500894	-0.3064438	-1.715794	-1.950305	-1.973382	-0.4700159	1.9263106	-0.0043646	-0.6391970	-1.3789564	0.3513463	2
16377	-0.4955311	1.0910284	-0.3064438	-1.636560	-1.997069	-2.618792	-0.2643256	1.8671951	-0.5251954	-0.7874901	-1.5079206	-0.1441126	2
10542	-1.6130762	1.2615054	-0.3064438	-1.155495	-1.891849	-2.109167	-0.4196968	1.3262884	-0.5135233	-0.5488871	-1.3561309	0.3057582	2
5209	-1.6130762	1.2618596	-0.3064438	-2.027072	-2.363001	-3.597632	-0.3790884	1.9410894	-0.5282833	-0.5283621	-1.4116730	-0.8229199	2
16795	-0.9328313	-0.0516683	-0.3064438	-1.347921	-2.115149	-2.083313	-0.4329387	1.7253179	-0.5327978	0.0360751	-1.5607998	-0.0394560	2
17218	-1.6130762	0.5350367	-0.3064438	-2.659248	-2.182568	-2.922154	-0.3490735	0.9538609	-0.5347462	-0.2512747	-1.4729215	1.7868225	2

Table 6: 10 Recommended Songs Numerical Characteristics – Johann Sebastian Bach

artists	album_name	track_name	track_genre	cluster
Sonmi451	Nachtmuziek	Bay Hatch	idm	2
Wolfgang Amadeus Mozart;Hilary Hahn;Natalie Zhu	Mozart - A Classical Dawn	Sonata for Piano and Violin in E Minor, K. 304: II. Tempo di minuetto	classical	2
Иван Петров;Валентина Клепацкая;хор Большого театра;Alexander Melik-Pashayev;Оркестр Большого театра	Mussorgsky: Boris Godunov	Boris Godunov, Act IV Scene 2: “Zvon! Pogrebalny zvon!”	romance	2
Franz Liszt;Una Bourne	Mozart, Liszt & Others: Piano Works	Spinnerlied aus dem Fliegenden Holländer, S. 440 (After Wagner’s WWV 63)	classical	2
Татьяна Комова;Георгий Квик	Ромэн. Русские цыганские романсы	Сумерки	romance	2
Giuseppe Verdi;Sir Colin Davis;Christine Brewer;London Symphony Orchestra	Verdi: Requiem	Requiem: XVII. Libera Me	opera	2
Kiawasch Sahebnassagh;Amirhosein Allah Yari	Forty Poems To Kharaghan	Rubai 1 To 5	iranian	2
Johannes Brahms;Jessye Norman;Daniel Barenboim;Wolfram Christ	Klassische Weihnachtsmusik	Geistliches Wiegenlied, Op. 91, No. 2	classical	2
Elena Kamburova	Дрёма. Русские колыбельные	Колыбельная светланы	romance	2
Johann Sebastian Bach;The English Concert;Trevor Pinnock	Klassische Weihnachtsmusik	Orchestral Suite No. 3 in D Major, BWV 1068: II. Air	classical	2

Table 7: 10 Recommended Songs Naming Details – Johann Sebastian Bach

Visual Analysis

For illustrative purposes, a scatter plot (Figures 10), was created to visualize the relationship between the input song and its neighbors, focusing on the features "danceability" and "energy."

- The input song was clearly distinguishable from its neighbors, which clustered tightly around it, reflecting their similarity.
- This visualization provides intuitive confirmation of the KNN algorithm's ability to find and rank songs based on their proximity in the feature space.

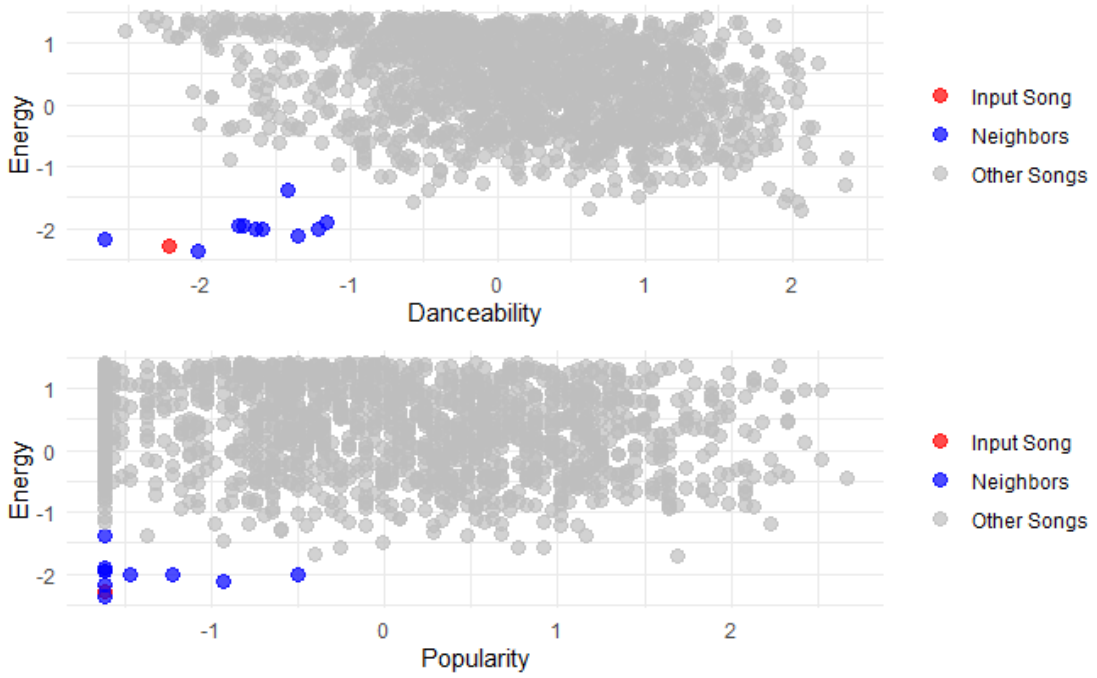


Figure 10: 10 Nearest Neighbors, Energy vs. Danceability and Energy vs. Popularity – Johann Sebastian Bach

Practical Insights

1. Model Effectiveness: The KNN model showed strong predictive power, with robust accuracy, precision, and recall metrics across all clusters.
2. High-Quality Recommendations: The recommended songs closely matched the input song in musical attributes, validating the model's ability to provide meaningful and relevant recommendations.
3. User-Centric Application: The scatter plot visualization not only confirms the technical reliability of the model but also enhances user confidence by offering a clear representation of how recommendations are derived.

Conclusion

In this project, we successfully implemented a hybrid recommendation system that provides music streaming services with a powerful approach to personalizing music recommendations in their development stage. Our approach leveraged audio features such as tempo, danceability, energy, and loudness to cluster similar songs and predict relevant music recommendations. The results demonstrated that clustering songs into distinct groups based on their characteristics, followed by classification to predict user preferences, significantly improved the quality of recommendations. We were able to provide users with more diverse, accurate, and

personalized song suggestions, enhancing their music discovery experience. The model achieved an impressive 95.59% accuracy in predicting the correct clusters, which reflects the effectiveness of this approach in identifying patterns within the data.

Despite the successes, there are areas for further improvement. First, while K-Means with 4 clusters proved to be an optimal solution based on our dataset, a higher number of clusters might offer even greater granularity, allowing the model to capture more subtle nuances in music preferences. Another possible enhancement is the incorporation of additional user-specific data, such as listening history and song ratings, which could make the recommendations even more personalized.

Lessons Learned

Throughout this project, our team gained valuable insights into the complexities and challenges of data mining, particularly in the context of building recommendation systems. Here are some key lessons learned:

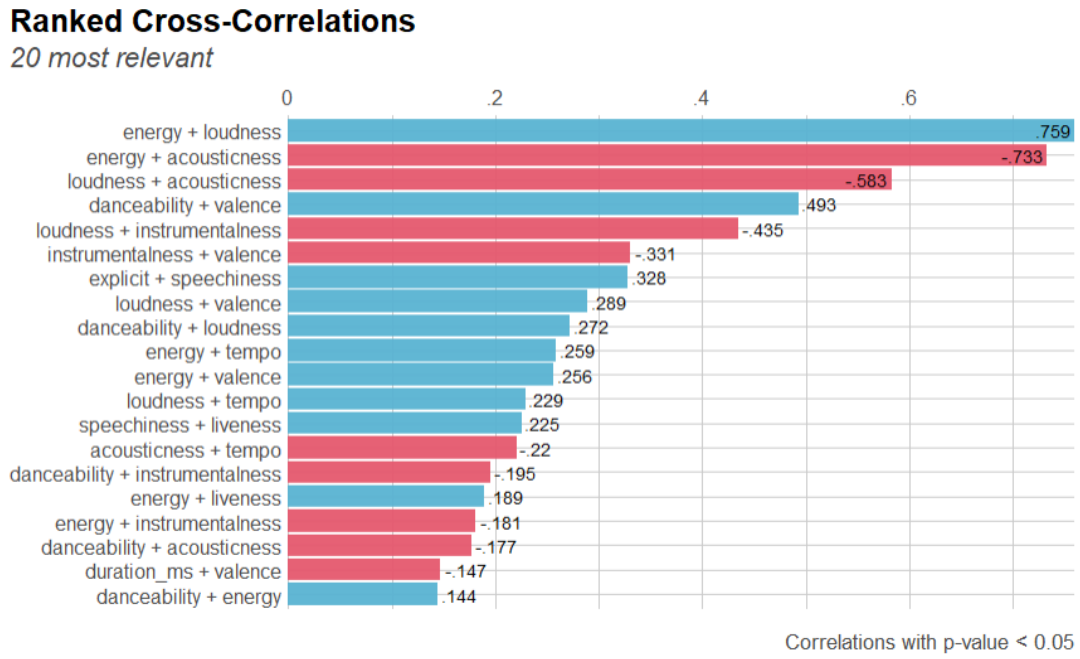
1. Feature Engineering is Crucial: The quality of the recommendation system heavily depends on the choice and transformation of features. Carefully selecting relevant features (such as tempo, energy, and danceability) and performing necessary data preprocessing (such as scaling and handling missing values) are essential steps to improve model performance.
2. Clustering and Classification Are Complementary: The combination of clustering (to group similar songs) and classification (to predict a song's group) proved to be an effective strategy. We learned how the two techniques can complement each other to provide both broad patterns (via clustering) and precise recommendations (via classification).
3. Scalability and Data Size Challenges: Handling large datasets posed significant challenges in terms of memory and computational resources. Techniques such as random sampling and reducing the size of the data for preliminary analysis were critical in overcoming these obstacles.
4. Model Evaluation and Optimization: Proper model evaluation, using metrics like accuracy, Kappa score, and confusion matrices, was vital in understanding the strengths and weaknesses of our approach. Additionally, optimizing model parameters and tuning hyperparameters during cross-validation contributed to better results.
5. The Power of Cross-Validation: The importance of cross-validation in assessing model performance was emphasized during this project. It helped us ensure that our model was robust and capable of generalizing well to unseen data.

In summary, this project has been a rewarding learning experience, providing our team with a deeper understanding of data mining techniques, while also shedding light on areas for future enhancement in personalized music recommendations.

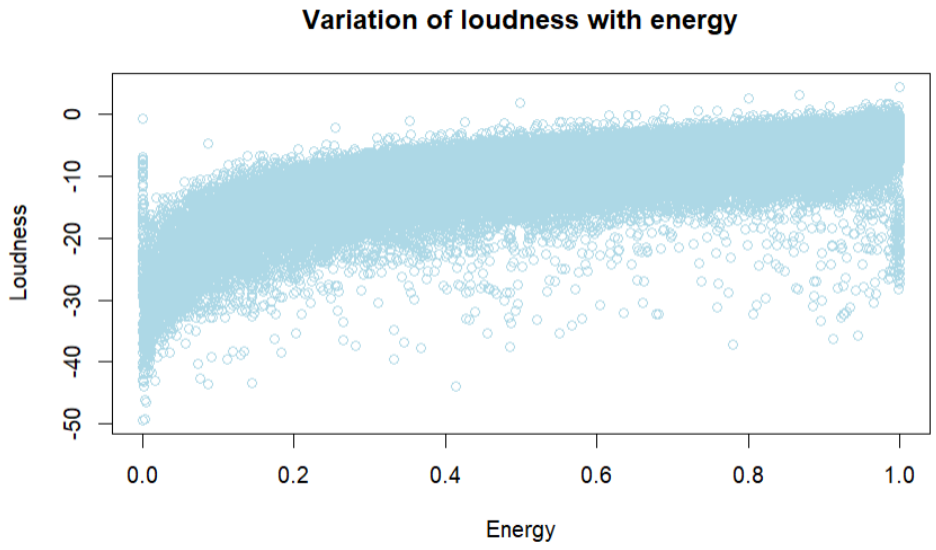
Appendix

Appendix A: Visualizations

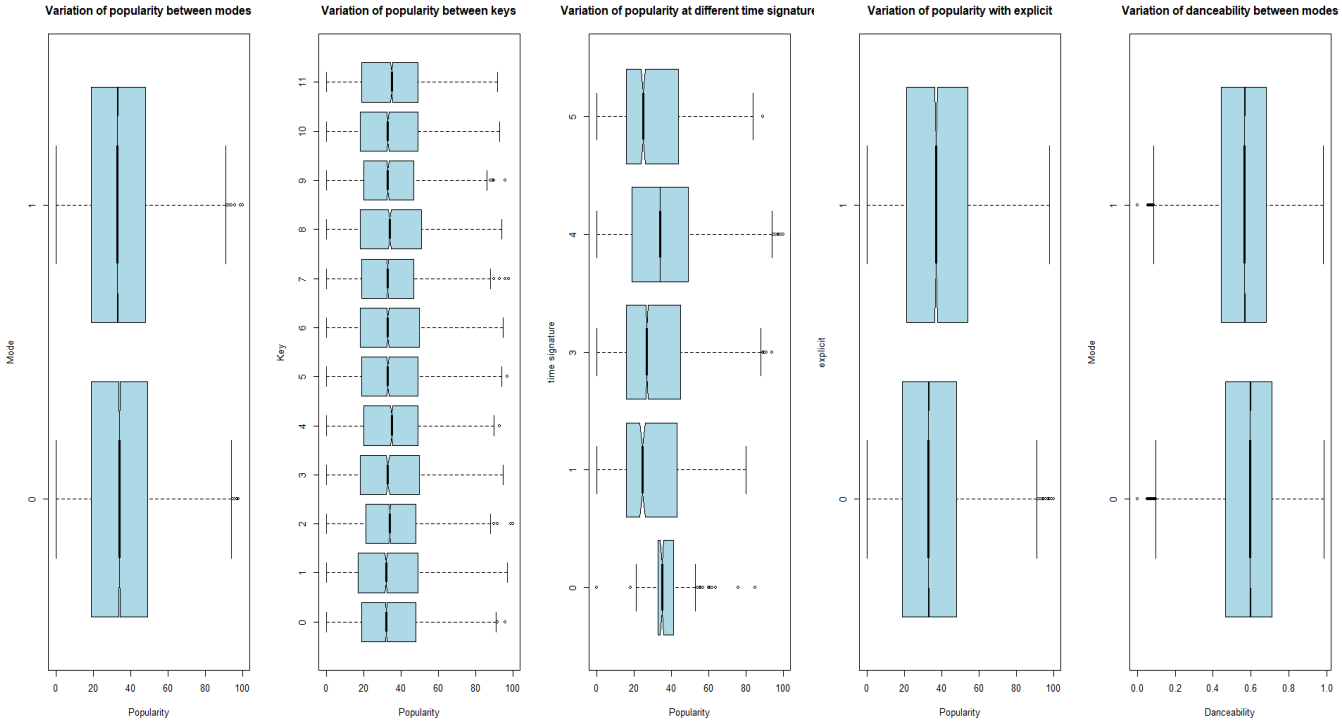
1. Ranked cross-correlation



2. Scatterplot



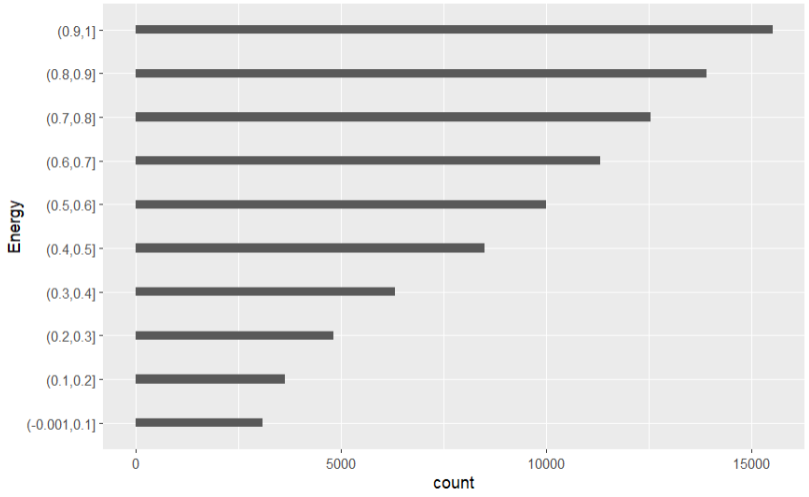
3. Box plots



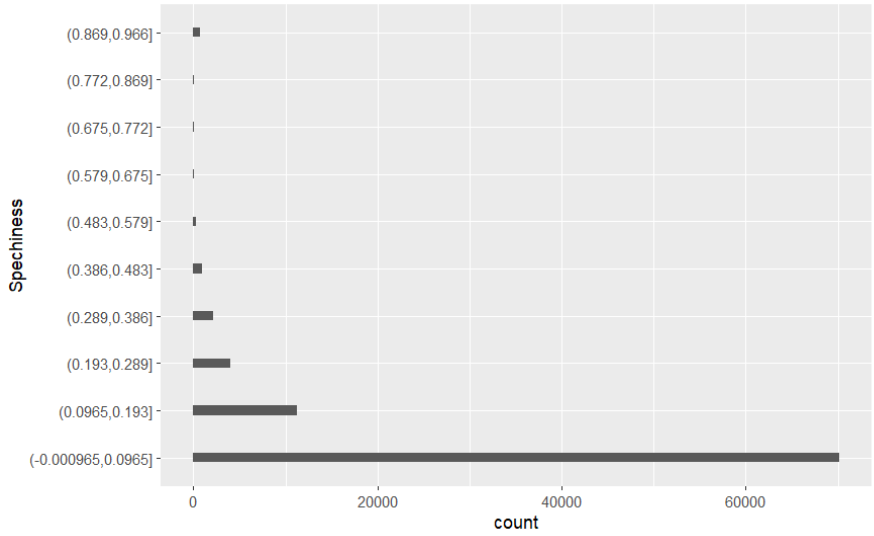
4. IQR Outlier detection output

```
Number of outliers popularity: 11
Number of outliers duration_ms: 4226
Number of outliers danceability : 474
Number of outliers energy : 0
Number of outliers loudness: 5026
Number of outliers speechiness: 10644
Number of outliers acousticness: 0
Number of outliers instrumentalness: 19613
Number of outliers liveness: 6981
Number of outliers valence: 0
Number of outliers tempo: 514
```

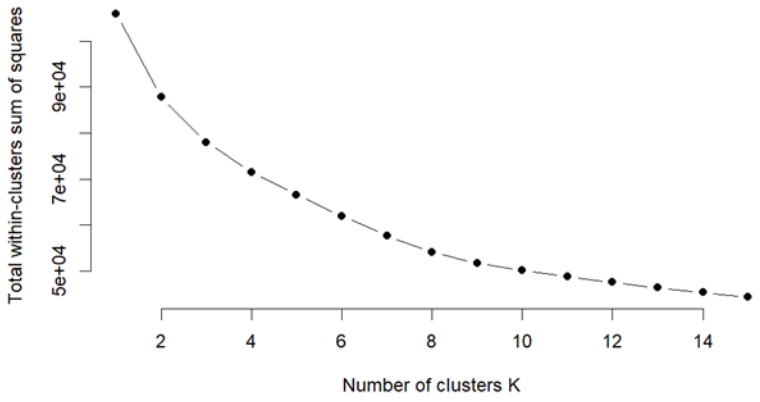
5. Energy distribution among songs



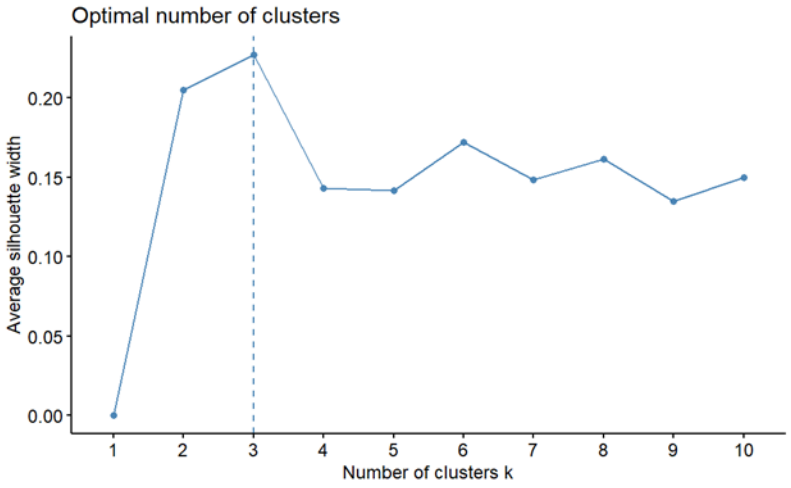
6. Speechiness distribution among songs



7. Elbow Plot for K-Means



8. Silhouette Score for K-Means



Appendix B: Recommendation Examples

1. Example Input Song – Billie Eilish

artists	album_name	track_name	track_genre
Billie Eilish	dont smile at me	party favor	electro

2. 5 Recommended Songs Naming Details – Billie Eilish

artists	album_name	track_name	track_genre	cluster
Lauren Spencer Smith	Back to Friends	Back to Friends	singer-songwriter	2
Kina;yaeow	Wish I Was Better	Wish I Was Better	sad	2
Cavetown	This Is Home	This Is Home	indie-pop	2
Moriarty	Gee Whiz But This Is a Lonesome Town	Jimmy	french	2
Videoclub	Amour plastique	Amour plastique	french	2

3. Example Input Song – Nirvana

artists	album_name	track_name	track_genre	cluster
Nirvana	Nevermind (Remastered)	Stay Away	grunge	3

4. 5 Recommended Songs Naming Details – Nirvana

artists	album_name	track_name	track_genre	cluster

Red	End Of Silence	Breathe Into Me	grunge	3
Breaking Benjamin	Phobia (Explicit Version)	Had Enough	grunge	3
Amaranthe	Manifest	Archangel	goth	3
Avenged Sevenfold	Avenged Sevenfold	Almost Easy	metal	3
Pitty	Anacrônico	Memórias	hard-rock	3

5. Example Input Song – Daddy Yankee

artists	album_name	track_name	track_genre	cluster
Daddy Yankee	Que Tire Pa Lante	Que Tire Pa Lante	latino	4

6. 5 Recommended Songs Naming Details – Daddy Yankee

artists	album_name	track_name	track_genre	cluster
TWICE	Taste of Love	Alcohol-Free	k-pop	4
Selena Gomez & The Scene	When The Sun Goes Down	Who Says	dance	4
Luis Fonsi;Demi Lovato	VIDA	Échame La Culpa	latin	4
Vickeblanka	Devil	Black Catcher	anime	4
Mohombi	MoveMeant (International)	Bumpy Ride	dance	4

Bibliography & Credits

1. Exploratory Data Analysis:
    a. https://r4ds.hadley.nz/eda
    b. https://www.kaggle.com/code/duongtruongbinh/spotify-eda-simple-songs-recommendation
2. Clustering:
    a. http://www.sthda.com/english/wiki/wiki.php?id_contents=7940
    b. https://uc-r.github.io/kmeans_clustering
3. Classification:
    a. https://cran.r-project.org/web/packages/FNN/FNN.pdf
    b. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761