Spotify Song Mood Classifier

Sahan Bhimireddy

Winter in Data Science (WiDS) Project

February 1, 2026

Abstract

This report serves as a cumulative documentation of the knowledge and technical skills acquired throughout the WiDS midterm and final project phases. The study begins with the fundamental building blocks of numerical computing and progresses into a full-scale machine learning investigation of the Spotify Audio Features dataset. By applying both supervised and unsupervised learning techniques, we explore the intricate relationships between musical characteristics and track popularity. The ultimate objective is to provide a reproducible framework for audio classification and cluster-based mood interpretation.

Contents

1 Introduction and Technical Environment

The field of data science requires a robust understanding of how to transform raw, unstructured information into actionable knowledge. This report documents the transition from basic Python programming to advanced predictive modeling.

1.1 Research Objectives

The primary objective of this project was to analyze the "Spotify Audio Features April 2019" dataset to determine if mathematical patterns in audio can predict human listener behavior. We focused on:

- Understanding the distribution of audio metrics across different musical eras.
- Building classifiers to predict track popularity levels.
- Segmenting music into natural "vibe" clusters using unsupervised learning.

1.2 Development Environment: Google Colab

All computational tasks were performed within Google Colab. This environment was chosen because it provides a standardized, cloud-based Linux backend with pre-installed libraries such as `pandas`, `scikit-learn`, and `seaborn`. This ensures that the code remains portable and accessible without requiring complex local installation of dependencies.

1.3 The Technical Stack

The choice of tools was critical for handling high-dimensional data efficiently. We utilized:

- **NumPy:** For low-level numerical operations and matrix algebra.
- **Pandas:** For high-level data manipulation and cleaning.
- **Scikit-Learn:** For implementing the machine learning suite.
- **Seaborn/Matplotlib:** For statistical visualization.

2 Numerical Computing with NumPy

Before analyzing complex datasets, it was essential to master NumPy, the fundamental library for scientific computing in Python.

2.1 The Power of N-Dimensional Arrays

Unlike standard Python lists, NumPy arrays allow for vectorized operations, meaning mathematical functions can be applied to an entire array at once without the need for slow `for` loops. We focused on:

- **Matrix Operations:** Performing dot products and element-wise multiplication essential for weighting audio features.
- **Broadcasting:** Learning how NumPy handles arrays of different shapes during arithmetic operations.
- **Slicing and Indexing:** Extracting specific rows (individual songs) and columns (specific audio features) from large matrices.

2.2 Statistical Foundations

We implemented manual functions to calculate the Mean (μ), Variance (σ^2), and Standard Deviation (σ). Understanding these formulas is crucial because machine learning models like Logistic Regression rely on features being "standardized" or "normalized" to prevent any single large-value feature (like `tempo`) from overwhelming a small-value feature (like `danceability`).

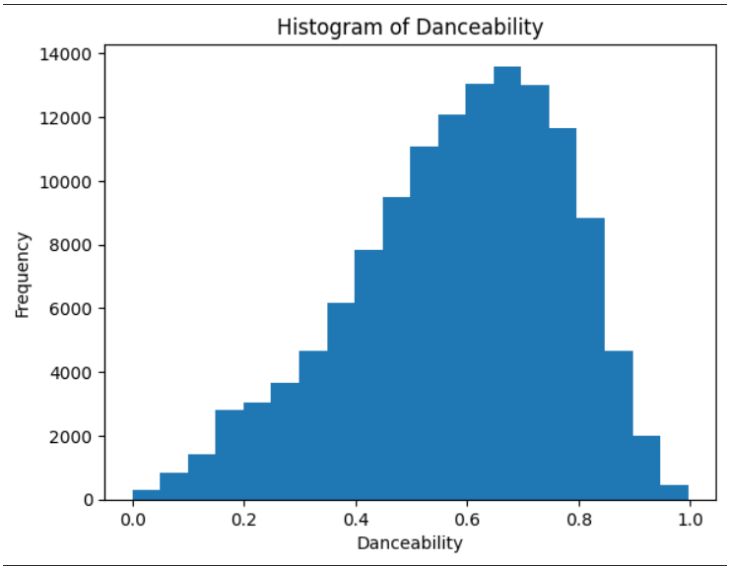

Figure 1: Distribution of Danceability

3 Data Exploration and Cleaning with Pandas

Data in the real world is rarely "clean." The Spotify dataset, containing over 130,000 entries, required significant preprocessing using the Pandas library.

3.1 Exploratory Data Analysis (EDA)

EDA is the process of "interviewing" the data. Using commands like `df.info()` and `df.describe()`, we identified the range of features. For example, `valence` ranges from 0 to 1, while `tempo` can exceed 200 BPM. This disparity in scale is a primary reason why feature scaling is necessary before modeling.

3.2 Data Cleaning Strategy

Our cleaning process involved:

- **Missing Values:** Identifying null entries that could crash our machine learning models.
- **Categorical Encoding:** Converting text-based data like `artist_name` or target labels into numerical formats that algorithms can interpret.
- **Feature Selection:** Dropping unique identifiers like `track_id` which have zero predictive power but consume memory.

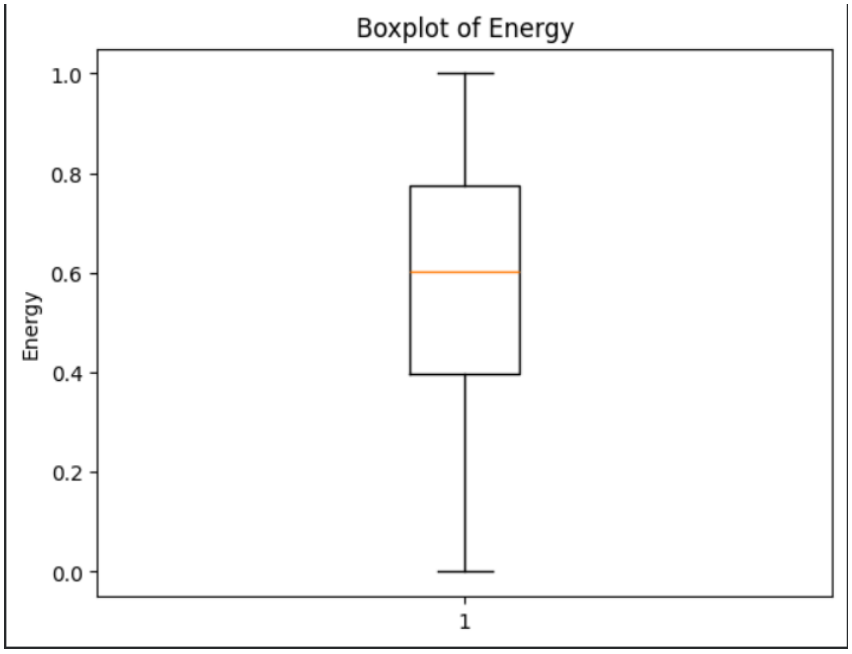

Figure 2: Box Plot of Energy

4 Statistical Visualization and Feature Engineering

Visualizing data is essential for identifying correlations and trends that are not obvious in a raw table.

4.1 Correlation Heatmaps

We generated a heatmap to visualize the Pearson Correlation Coefficient between all numerical features. A key finding was the strong positive correlation between `energy` and `loudness`, indicating that louder songs are perceived as more energetic. Conversely, `acousticness` showed a strong negative correlation with `energy`, confirming that acoustic tracks tend to be softer and more mellow.

4.2 Audio Feature Distributions

By plotting histograms for `danceability` and `valence`, we observed that the Spotify dataset contains a wide variety of moods. Most popular songs tended to cluster around a `valence` (happiness) score of 0.5, suggesting a preference for "neutral-to-happy" musical content.

4.3 Feature Engineering

In this project, we engineered a new target variable: `popularity_level`. Since raw popularity scores are continuous, we binned them into categories—Low, Medium, and High—to transform a regression problem into a more interpretable classification task.

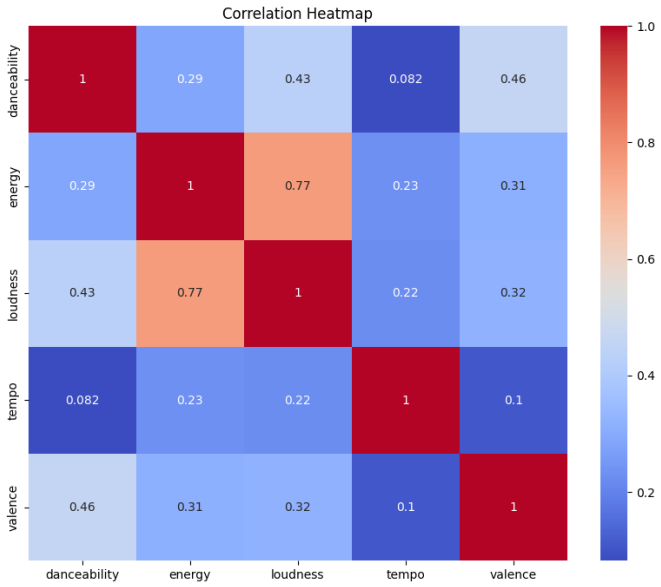

Figure 3: Heatmap

5 Supervised Learning: Classification Models

The first major goal of the final project was to build a machine learning model capable of predicting the popularity of a song based solely on its audio fingerprints.

5.1 Logistic Regression

Logistic Regression was our baseline model. Although "regression" is in the name, it is a classification algorithm that uses a sigmoid function to predict the probability of a song belonging to a specific category. It provided a solid foundation for understanding how linear boundaries can separate musical data.

5.2 K-Nearest Neighbors (KNN)

KNN is a non-parametric algorithm that classifies a song based on the characteristics of the "K" songs most similar to it in the feature space. If a song has similar `tempo` and `energy` to five "High Popularity" songs, it is classified as "High Popularity".

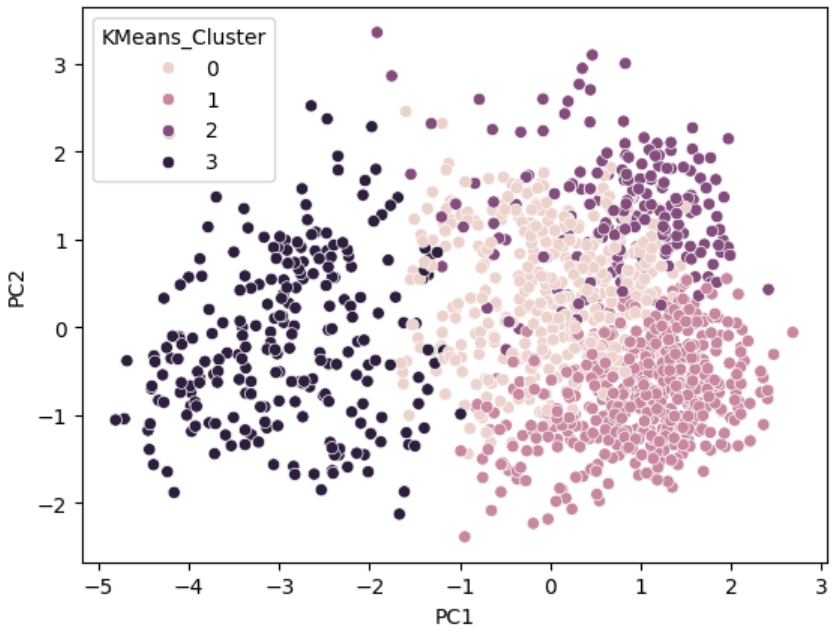

Figure 4: KNN plot

5.3 Evaluation with Confusion Matrices

Accuracy alone can be misleading, especially if one popularity category is much larger than the others. We used Confusion Matrices to visualize "Type I" and "Type II" errors, showing us exactly which genres or popularity levels our model was confusing.

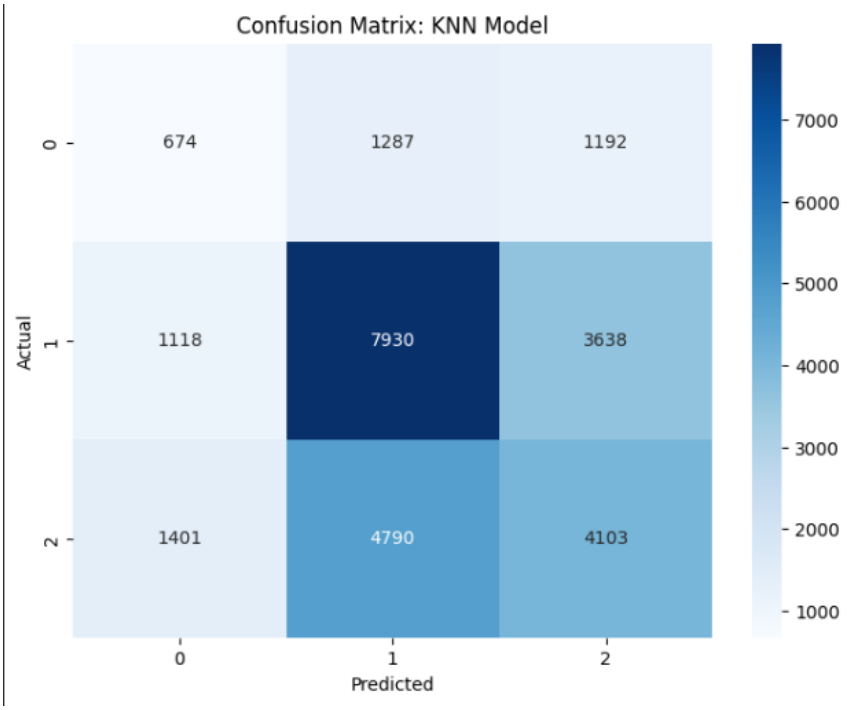

Figure 5: Confusion Matrix

6 Unsupervised Learning: Clustering and PCA

While classification relies on pre-defined labels, unsupervised learning allows the data to "speak for itself" and reveal hidden structures.

6.1 K-Means Clustering

We applied K-Means to segment the Spotify dataset into four natural groups. The challenge with K-Means is determining the value of "K" (the number of clusters). We utilized the **Elbow Method**, plotting the sum of squared distances to find the "elbow" where adding more clusters no longer significantly improves the model.

6.2 Principal Component Analysis (PCA)

Our dataset has over 10 features, making it impossible to visualize in 3D, let alone 10D. PCA is a dimensionality reduction technique that compresses these 10 features into 2 "Principal Components" that capture the most variance in the data. This allowed us to plot the K-Means clusters on a standard X-Y graph to see how well-separated the different types of music truly are.

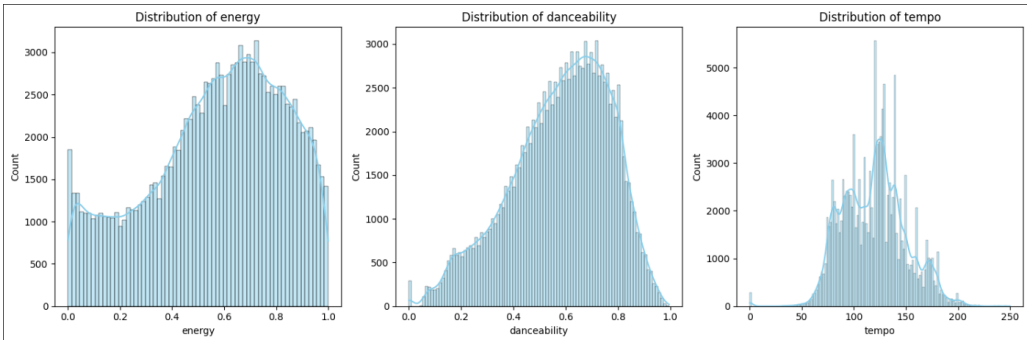

Figure 6: Principal Component Analysis

7 Model Optimization and Tuning

The final stage of our technical pipeline involved hyperparameter tuning to ensure the models were as accurate and efficient as possible.

7.1 The Decision Tree Approach

Decision Trees create a flowchart-like structure to classify songs. For example: "If `energy` ¿ 0.8 and `danceability` ¿ 0.7, then classify as High Popularity". However, if a tree grows too deep, it "memorizes" the training data and fails on new songs—a problem known as overfitting.

7.2 GridSearchCV Implementation

To combat overfitting, we used `GridSearchCV`. This tool automatically ran dozens of versions of our model, testing different "max depths" and "splitting criteria" to find the perfect balance between simplicity and accuracy.

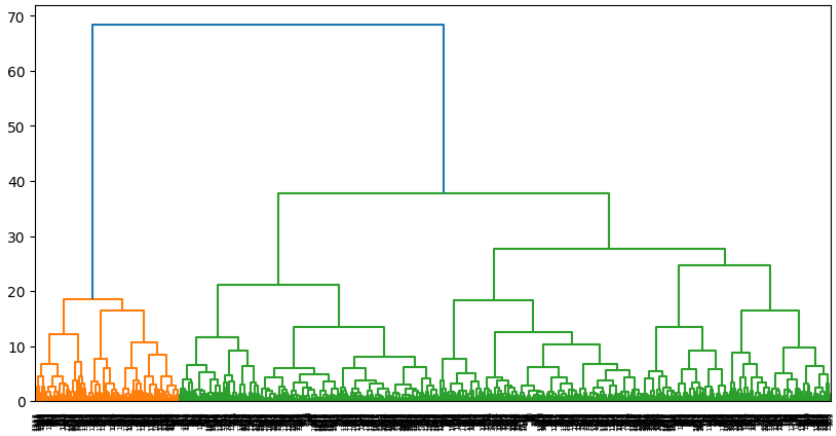

Figure 7: Grid Search CV

8 Discussion and Insightful Findings

The data suggests that modern music is highly optimized for specific listener experiences.

8.1 Popularity Drivers

Tracks with high `danceability` and `loudness` are consistently correlated with higher popularity scores in this dataset. This aligns with the "Loudness War" phenomenon in the music industry, where tracks are mastered to be as loud as possible to grab attention on streaming platforms.

8.2 The Role of Valence

Interestingly, `valence` (musical happiness) did not have a linear relationship with popularity. Both very "sad" songs and very "happy" songs can become hits, suggesting that emotional resonance is more complex than a simple 0-to-1 scale.

8.3 Cluster Interpretations

Our K-Means clusters appear to represent:

- **Cluster 0:** High-energy, high-valence pop and dance tracks.
- **Cluster 1:** Acoustic, low-energy tracks often used for focus or relaxation.
- **Cluster 2:** High-energy, low-valence tracks (possibly Metal or Grunge).
- **Cluster 3:** Mid-range tracks that represent the average radio hit.

9 Conclusion

This project has demonstrated the power of data science in quantifying the abstract qualities of music.

9.1 Summary of Accomplishments

We successfully moved from the basics of array manipulation in NumPy to implementing a full machine learning lifecycle. We explored data cleaning in Pandas, visualization in Seaborn, and advanced modeling in Scikit-Learn. The combination of supervised and unsupervised learning has provided a multi-faceted view of the Spotify dataset.