|  |  |
| --- | --- |
| **Project Title** | **Amazon Music Clustering** |
| **Skills take away From This Project** | **Data exploration, Data cleaning, Feature selection, Data normalization, K-Means clustering, Elbow method, Silhouette score, PCA (optional), Cluster visualization, Genre inference, Python (pandas, NumPy), scikit-learn ,Data storytelling** |
| **Domain** | **Music Analytics / Unsupervised Machine Learning** |

**Problem Statement:**

With millions of songs available on platforms like Amazon, manually categorizing tracks into genres is impractical. The goal of this project is to automatically group similar songs based on their audio characteristics using clustering techniques. By analyzing patterns in features such as tempo, energy, danceability, and more, learners will develop a model that organizes songs into meaningful clusters, potentially representing different musical genres or moods—without any prior labels.

**Technical Tags:**

Python, Pandas, scikit-learn, KMeans, DBSCAN, Hierarchical Clustering, EDA, Clustering, PCA, Recommendation, Unsupervised Learning

**Business Use Cases:**

* **Personalized Playlist Curation:** Automatically group songs that sound similar to enhance playlist generation.
* **Improved Song Discovery:** Suggest similar tracks to users based on their preferred audio profile.
* **Artist Analysis:** Help artists and producers identify competitive songs in the same audio cluster.
* **Market Segmentation:** Streaming platforms can use clusters to analyze user listening patterns and optimize recommendations or promotions.

**Data Set:**

* **File Name:** single\_genre\_artists
* **Features Include:**
  + danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo, duration\_ms, etc.
* **Text Fields:** track\_id, track\_name, artist\_name (used for reference only).

**Data Set Explanation:**

This dataset provides **audio characteristics** of Amazon Music songs that define how a song "sounds." These include rhythm, mood, intensity, and instrumentation.

**Approach:**

The core idea is to group similar songs based on their **audio characteristics**, using **unsupervised machine learning**. The approach is broken into logical phases that learners can follow step-by-step:

### **1. Data Exploration & Preprocessing**

**Goal:** Understand the structure of the dataset, handle missing data, and prepare features for clustering.

**Steps:**

* Load the single\_genre\_artists.csv dataset into a Pandas DataFrame.
* Explore the dataset:  
  + View column names, datatypes, and shape.
  + Check for missing values or duplicates.
* Drop unnecessary columns:  
  + Remove track\_name, artist\_name, and track\_id as they are not needed for clustering.
* Visualize distributions of key features to understand variation in values.
* Normalize the data:  
  + Use StandardScaler or MinMaxScaler to bring all features to the same scale. Clustering is distance-based, so feature scaling is crucial.

### **2. Feature Selection**

**Goal:** Select features that represent the sound of a song and are useful for clustering.

**Recommended features:**

* danceability
* energy
* loudness
* speechiness
* acousticness
* instrumentalness
* liveness
* valence
* tempo
* duration\_ms

**Why:** These features describe the rhythm, mood, instrumentation, and energy level of a track — perfect for grouping similar songs.

### **3. Dimensionality Reduction (Optional but Recommended for Visualization)**

**Goal:** Reduce the dataset to 2 or 3 components to **visualize clusters** later.

**Options:**

* **PCA (Principal Component Analysis):** Helps reduce dimensions while preserving variance.
* **t-SNE:** Captures complex relationships better but is computationally heavier.

**Use:** Only for visualization and analysis — **not** for clustering input.

### **4. Clustering Techniques**

**Goal:** Apply one or more clustering algorithms to group the songs.

#### **🔸 Option A: K-Means Clustering**

* Start with **K-Means** — simple and effective.
* Determine the best number of clusters k:  
  + Use the **Elbow Method**: Plot SSE (inertia) vs. k and find the "elbow."
  + Use **Silhouette Score** to evaluate how well clusters are formed.
* Apply KMeans(n\_clusters=k) on the scaled feature matrix.
* Add the predicted cluster label as a new column to the original dataset.

#### **🔸 Option B: DBSCAN (for discovering arbitrary-shaped clusters)**

* Good for detecting noise/outliers.
* Use after tuning parameters eps and min\_samples.

#### **🔸 Option C: Hierarchical Clustering (Agglomerative)**

* Can create a **dendrogram** to visualize merging of songs into clusters.
* Does not require specifying number of clusters upfront.

### **5. Cluster Evaluation and Interpretation**

**Goal:** Assess how well the clustering algorithm worked and understand the nature of each cluster.

**Evaluation Techniques:**

* **Silhouette Score:** Measures how close points are to their own cluster vs. others.
* **Davies-Bouldin Index:** Lower value means better separation.
* **Inertia (for K-Means):** Measures cluster compactness.

**Interpretation:**

* Use the mean values of each feature per cluster to **profile clusters**.  
  + E.g., Cluster A = high danceability, high energy → “Party tracks”
  + Cluster B = low energy, high acousticness → “Chill acoustic”

### **6. Visualization**

**Goal:** Create clear visualizations to show how songs are clustered.

**Ideas:**

* **2D scatter plots** using PCA/t-SNE with color-coded clusters.
* **Bar charts** showing average feature values per cluster.
* **Heatmaps** comparing features across clusters.
* **Distribution plots** for features like danceability, tempo, etc. within each cluster.

### **7. Final Analysis and Export**

**Goal:** Wrap up results in a format useful for decision-making or recommendation.

* Add cluster labels to the original DataFrame.
* Sort or group by clusters and show top tracks per cluster.
* Export the final dataset to CSV.
* Write a **summary report** explaining each cluster and its characteristics.

**Results:**

By the end of the project, learners should be able to:

* Generate distinct clusters of songs that reflect underlying similarities in audio features.
* Visualize and interpret what each cluster represents in terms of musical characteristics.
* Use the resulting clusters to support use cases like recommendation or playlist generation.