## **1. Title**

**Amazon Music Clustering**

## **2. Abstract**

**The music industry has witnessed rapid growth in digital platforms, creating vast datasets of songs, artists, and user preferences. This project presents an interactive clustering and recommendation dashboard that analyzes the Amazon Music dataset. Using machine learning algorithms (KMeans, DBSCAN, Hierarchical Clustering) and similarity measures (cosine similarity), songs are grouped based on features such as danceability, energy, tempo, and valence.**

**Dimensionality reduction techniques (PCA and t-SNE) are used to visualize clusters in 2D. The system provides song search, artist analysis, and personalized recommendations through an interactive Streamlit dashboard. This project demonstrates the power of unsupervised learning in music exploration and recommendation systems.**

## **3. Introduction**

**With millions of songs available on digital platforms, it is challenging for users to discover new music that matches their preferences. Recommendation systems and clustering algorithms can solve this problem by analyzing audio features and grouping songs with similar characteristics.**

**This project builds an interactive dashboard that combines data preprocessing, clustering, visualization, and recommendation techniques. It empowers users to explore patterns in Amazon Music data, discover hidden clusters, and receive song suggestions.**

## **4. Objectives**

* **To analyze Amazon Music songs using unsupervised learning techniques.**
* **To implement KMeans, DBSCAN, and Hierarchical clustering for grouping songs.**
* **To apply PCA and t-SNE for dimensionality reduction and visualization.**
* **To build a recommendation system using cosine similarity.**
* **To develop an interactive dashboard using Streamlit for user exploration.**

## 

## **5. A) Tools & Technologies Used**

* **Programming Language: Python**
* **Libraries: pandas, numpy, matplotlib, seaborn, plotly, scikit-learn**
* **Dashboard Framework: Streamlit**
* **Algorithms/Models:**
  + **KMeans**
  + **DBSCAN**
  + **Hierarchical Clustering (Agglomerative)**
  + **PCA**
  + **t-SNE**
  + **Cosine Similarity**

### **B) Scope of the Project**

## **What your project covers and does not cover.**

## **Example: ✅ Clustering songs, recommendations, visualizations. ❌ Real-time streaming integration (future work).**

## **c). System Requirements**

## **Hardware Requirements: (e.g., 4GB RAM, Intel i3 processor, 500MB storage).**

## **Software Requirements: Python 3.10+, Streamlit, scikit-learn, pandas, etc.**

## 

### **D) System Architecture / Flow Diagram**

## **A simple flowchart showing: Dataset → Preprocessing → Clustering → Visualization → Recommendations → Dashboard**

### **E)Testing & Validation**

* **Show how you tested your system:**
  + **Checking clustering with Silhouette Score.**
  + **Testing recommendations with sample queries.**

### **F). Limitations**

* **Large datasets may slow down t-SNE visualization.**
* **Real-time updates not supported (only static dataset).**

## **6. Methodology**

### **Step 1: Data Preprocessing**

* **Handle missing values (df.dropna()).**
* **Convert dates (pd.to\_datetime).**
* **Normalize features using StandardScaler.**

### **Step 2: Clustering**

* **KMeans → Partition dataset into k clusters.**
* **DBSCAN → Density-based clustering with outlier detection.**
* **Hierarchical Clustering → Builds cluster hierarchy.**

### **Step 3: Dimensionality Reduction**

* **PCA → Reduce features into 2D for plotting.**
* **t-SNE → Non-linear mapping for better cluster visualization.**

### **Step 4: Recommendation**

* **Use cosine similarity between feature vectors to suggest similar songs.**

### **Step 5: Dashboard Development**

* **Sidebar filters: genre, energy, tempo, danceability.**
* **Song & artist search functionality.**
* **Visualizations: bar charts, scatter plots, heatmaps, radar charts.**
* **Download/export options.**

## **7. A) Algorithms Used**

| **Algorithm** | **Purpose** |
| --- | --- |
| **KMeans** | **Partition songs into k groups** |
| **DBSCAN** | **Density-based clustering + outlier detection** |
| **Agglomerative Clustering** | **Hierarchical grouping** |
| **PCA** | **Linear dimensionality reduction** |
| **t-SNE** | **Non-linear dimensionality reduction for visualization** |
| **Cosine Similarity** | **Song recommendations** |

**B) Algorithm Explanation (Step by Step)**

* **Example for KMeans:**
  1. **Select number of clusters k.**
  2. **Initialize centroids randomly.**
  3. **Assign each song to the nearest centroid.**
  4. **Update centroid positions.**
  5. **Repeat until convergence.  
     (Same can be added for DBSCAN, Hierarchical, PCA, Cosine Similarity).**

## **8. Results & Screenshots**

* **Clusters of songs visualized in 2D (PCA/t-SNE).**
* **Cluster distribution (number of songs per group).**
* **Feature analysis (energy, danceability, valence comparisons).**
* **Search, filter, and recommendation functionalities in dashboard.**

***(👉 You can insert Streamlit dashboard screenshots here)***

## **9. Key Features**

* **🎵 Song Search & Artist Analysis**
* **🎯 Clustering with KMeans, DBSCAN, Hierarchical**
* **📊 PCA & t-SNE Visualization**
* **🔍 Similar Song Recommendations (Cosine Similarity)**
* **⚡ Interactive Streamlit Dashboard**
* **📥 Data Export Options**

## **10. Conclusion**

**This project successfully demonstrates the use of unsupervised machine learning in the music industry. By applying clustering techniques, dimensionality reduction, and similarity-based recommendations, the system provides an intuitive way to explore Amazon Music datasets. The dashboard allows users to search, analyze, and receive recommendations interactively.**

## **11. Future Scope**

* **Integration with Spotify/Amazon Music APIs for real-time data.**
* **Apply deep learning models for more accurate recommendations.**
* **Include lyrics-based sentiment analysis.**
* **User-based personalization for playlist generation.**

## **12. References**

* **Scikit-learn Documentation**
* **Streamlit Documentation**
* **Research papers on Music Recommendation Systems**
* **Amazon Music Dataset**