Project 2: Spotify Songs’ Genre Segmentation

1. Introduction

Music recommendation systems play a crucial role in modern streaming platforms such as Spotify. These systems analyze audio features of songs and user listening behavior to suggest relevant music. In this project, unsupervised machine learning techniques are used to group Spotify songs into clusters based on their audio features. These clusters help in understanding similarities among songs and form the basis of a recommendation system.

The objective of this project is to preprocess Spotify song data, perform exploratory data analysis, visualize relationships among features, apply clustering algorithms, and demonstrate how the resulting clusters can be used for music recommendation.

---

2. Dataset Description

The dataset consists of Spotify song-level data containing various audio features such as: - Danceability - Energy - Loudness - Speechiness - Acousticness - Instrumentalness - Liveness - Valence - Tempo

It also includes metadata like playlist genre and playlist name, which are used later for interpretation and validation of clusters.

---

3. Data Preprocessing

Data preprocessing is a critical step to ensure that the clustering algorithm works effectively.

Steps performed: 1. Loaded the dataset using Pandas 2. Checked for missing values and handled them using mean imputation 3. Selected relevant numerical audio features for clustering 4. Applied feature scaling using StandardScaler to normalize the data

---

4. Exploratory Data Analysis (EDA)

Exploratory Data Analysis was performed to understand the distribution and relationships among audio features.

Visualizations included: - Distribution plots for danceability, energy, and valence - Scatter plot between energy and danceability - Boxplot of loudness across playlist genres

These visualizations provide insights into how different musical properties vary across songs and genres.

---

5. Correlation Analysis

A correlation matrix was generated to analyze relationships among numerical features.

Key observations: - Energy shows strong positive correlation with loudness - Acousticness is negatively correlated with energy - Valence shows moderate correlation with danceability, indicating mood-related patterns

Understanding these correlations helps justify feature selection for clustering.

---

6. Clustering Methodology

K-Means clustering was chosen due to its simplicity and effectiveness in grouping numerical data.

Steps followed: 1. Used the Elbow Method to determine the optimal number of clusters 2. Applied K-Means clustering on scaled features 3. Assigned cluster labels to each song

---

7. Cluster Visualization

Since clustering was performed in a high-dimensional feature space, Principal Component Analysis (PCA) was used to reduce dimensions to two for visualization.

Clusters were visualized in 2D space, clearly showing separation among different song groups.

---

8. Genre and Playlist Analysis

Clusters were analyzed with respect to playlist genres and playlist names.

Findings: - Certain clusters were dominated by high-energy genres such as EDM and Rock - Other clusters contained acoustic and low-energy genres - Overlap among genres was observed, reflecting real-world music diversity

---

9. Recommendation System Logic

The clustering results can be used to build a basic recommendation system.

Recommendation approach: - Identify the cluster of a song a user listens to - Recommend other songs belonging to the same cluster

This ensures that recommendations share similar audio characteristics, improving user satisfaction.

10. Conclusion

In this project, Spotify song data was successfully clustered using unsupervised learning techniques. Through preprocessing, visualization, correlation analysis, and clustering, meaningful song groupings were obtained. These clusters provide a strong foundation for building a music recommendation system. The project demonstrates how machine learning can be applied to real-world audio data to extract insights and drive intelligent recommendations.

11. Complete Python Implementation

```python
# Spotify Genre Grouping Project
# Complete, clean, and submission-ready code

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA

# -----------------------------
# 1. Load Dataset
# -----------------------------
df = pd.read_csv('spotify dataset.csv')

# -----------------------------
# 2. Data Preprocessing
# -----------------------------

# Select numerical audio features
features = [
    'danceability', 'energy', 'loudness', 'speechiness',
    'acousticness', 'instrumentalness', 'liveness',
    'valence', 'tempo'
]

# Handle missing values
for col in features:
    df[col].fillna(df[col].mean(), inplace=True)

# Feature scaling
scaler = StandardScaler()
```

```python
scaled_features = scaler.fit_transform(df[features])

# -----------------------------
# 3. Exploratory Data Analysis
# -----------------------------

# Distribution plots
for col in ['danceability', 'energy', 'valence']:
    plt.figure()
    sns.histplot(df[col], kde=True)
    plt.title(f'Distribution of {col}')
    plt.show()

# Scatter plot
plt.figure()
sns.scatterplot(x='energy', y='danceability', data=df)
plt.title('Energy vs Danceability')
plt.show()

# Boxplot
plt.figure(figsize=(10, 5))
sns.boxplot(x='playlist_genre', y='loudness', data=df)
plt.xticks(rotation=45)
plt.title('Loudness across Playlist Genres')
plt.show()

# -----------------------------
# 4. Correlation Matrix
# -----------------------------

plt.figure(figsize=(10, 8))
sns.heatmap(df[features].corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix of Audio Features')
plt.show()

# -----------------------------
# 5. Elbow Method
# -----------------------------

wcss = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_features)
    wcss.append(kmeans.inertia_)

plt.figure()
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()

# -----------------------------
# 6. K-Means Clustering
# -----------------------------

kmeans = KMeans(n_clusters=5, random_state=42)
df['Cluster'] = kmeans.fit_predict(scaled_features)

# -----------------------------
# 7. PCA Visualization
# -----------------------------

pca = PCA(n_components=2)
pca_features = pca.fit_transform(scaled_features)

plt.figure()
plt.scatter(pca_features[:, 0], pca_features[:, 1], c=df['Cluster'])
plt.title('PCA Visualization of Song Clusters')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()

# -----------------------------
# 8. Cluster vs Genre Analysis
# -----------------------------

cluster_genre = pd.crosstab(df['Cluster'], df['playlist_genre'])
print(cluster_genre)

# -----------------------------
# 9. Recommendation Example
# -----------------------------

def recommend_songs(song_index, n_recommendations=5):
    cluster = df.loc[song_index, 'Cluster']
    recommendations = df[df['Cluster'] == cluster].sample(n_recommendations)
    return recommendations[['track_name', 'playlist_genre']]

# Example usage
print(recommend_songs(0))
```