**Exploratory Data Analysis of Spotify Tracks Dataset**

*Research-Style Report*

# 1. Introduction

This report presents a structured exploratory data analysis (EDA) of a Spotify tracks dataset, derived from the accompanying Jupyter notebook. The objective is to understand the structure, quality, and statistical properties of the data, examine relationships among musical features (e.g., danceability, energy, valence, tempo), and summarize key empirical patterns that may inform downstream modeling tasks such as recommendation or dimensionality reduction.

# 2. Libraries Used

The following Python libraries were detected from the source notebook and were employed to support data ingestion, manipulation, visualization, and analytical procedures:

|  |  |
| --- | --- |
| Library | Role in the Analysis |
| joblib | Efficient serialization of Python objects and pipelines, often used to persist models or intermediate results. |
| matplotlib | Foundation plotting library used to render static visualizations such as histograms, boxplots, and scatter plots. |
| numpy | Numerical arrays and vectorized computation supporting data preprocessing and transformations. |
| os | Used within the notebook to support various EDA tasks (import detected). |
| pandas | Data loading, tabular data manipulation, handling missing values, grouping, and summarization. |
| scikit-learn | Machine learning utilities used for preprocessing (e.g., StandardScaler), dimensionality reduction (e.g., PCA), and modeling. |
| seaborn | High-level statistical visualization on top of Matplotlib for distributions, pairwise relationships, and heatmaps. |
| warnings | Used within the notebook to support various EDA tasks (import detected). |

# 3. Dataset and Methodology Overview

# 3a. Methods: Data Cleaning and Preparation

As part of the exploratory workflow, specific steps were undertaken to ensure that the dataset was reliable and analytically useful. Being a 3rd year B.Tech AIML student in India, I have approached the cleaning process with emphasis on practical reproducibility and clarity. The key cleaning and preparation methods executed are as follows:

• Cleaning step observed in code cell #13: # Cell 13: basic cleaning choices (example)  
df\_clean = df.copy()  
  
# 1) Drop duplicates  
df\_clean = df\_clean.drop\_duplicates()  
  
# 2) Simple missing value strategy: drop rows with missing numeric features used for clustering  
df\_clean = df\_clean.dropna(subset=num\_cols)  
  
# 3) (Optional) Winsorize / clip extreme values -- example clipping tempo to [0.5, 99.5] percentile  
for c in ["tempo","duration\_ms"]:  
 if c in df\_clean.columns:  
 lo, hi = df\_clean[c].quantile([0.005, 0.995])  
 df\_clean[c] = df\_clean[c].clip(lower=lo, upper=hi)  
  
print("Clean shape:", df\_clean.shape)  
df\_clean[num\_cols].describe().T  
# Save cleaned version  
clean\_path = os.path.join("data", "spotify\_tracks\_clean.csv")  
df\_clean.to\_csv(clean\_path, index=False)  
print("Saved cleaned CSV:", clean\_path)

• Cleaning step observed in code cell #12: # Cell 12: PCA explained variance to decide dimensionality  
features\_for\_pca = num\_cols # or choose subset  
X = df[features\_for\_pca].dropna()  
scaler = StandardScaler()  
Xs = scaler.fit\_transform(X)  
pca = PCA(n\_components=min(30, Xs.shape[1]))  
pca.fit(Xs)  
evr = pca.explained\_variance\_ratio\_  
plt.figure(figsize=(8,4))  
plt.plot(np.cumsum(evr), marker='o')  
plt.xlabel("n components")  
plt.ylabel("cumulative explained variance")  
plt.grid(True)  
savefig("pca\_scree.png")  
plt.show()  
print("Cumulative variance for first 2/5/10 comps:", np.cumsum(evr)[:10])  
# save scaler and PCA if you want  
joblib.dump(scaler, os.path.join(OUT\_DIR, "scaler\_for\_eda.pkl"))  
joblib.dump(pca, os.path.join(OUT\_DIR, "pca\_for\_eda.pkl"))

• Cleaning step observed in code cell #9: # Cell 9: pairplot for a small subset (choose up to 6 features)  
subset = ["danceability","energy","valence","tempo","loudness","acousticness"]  
subset = [s for s in subset if s in df.columns]  
sns.pairplot(df[subset].dropna().sample(min(2000, len(df))), corner=True)  
savefig("pairplot\_subset.png")  
plt.show()

• Cleaning step observed in code cell #6: # Cell 6: histograms for numeric features  
n = len(num\_cols)  
cols = 3  
rows = (n + cols - 1) // cols  
plt.figure(figsize=(cols\*5, rows\*4))  
for i, col in enumerate(num\_cols, 1):  
 plt.subplot(rows, cols, i)  
 sns.histplot(df[col].dropna(), kde=True)  
 plt.title(col)  
plt.tight\_layout()  
savefig("histograms.png")  
plt.show()

• Cleaning step observed in code cell #4: # Cell 4: duplicates + sample duplicates  
print("Total rows:", len(df))  
print("Duplicate rows:", df.duplicated().sum())  
  
# If you have a track\_id column, check uniqueness  
if "track\_id" in df.columns:  
 print("Unique track\_id:", df['track\_id'].nunique())  
else:  
 print("'track\_id' column not found.")

The dataset, as referenced within the notebook, consists of track-level observations with continuous audio features produced by Spotify’s audio analysis (e.g., acousticness, danceability, energy, instrumentalness, liveness, loudness, speechiness, tempo, valence) and potential metadata fields (e.g., popularity, duration, release year, artist). The methodology followed standard EDA practices: initial loading of data into a DataFrame, inspection of shape and schema, screening for missing values and outliers, followed by univariate and bivariate visual analysis. Where applicable, feature scaling (e.g., StandardScaler) and dimensionality reduction (e.g., PCA) steps were executed to examine latent structure among features.

# 4. Data Quality Assessment

The notebook evaluated basic data quality aspects including column data types, presence of missing values, and basic descriptive statistics. Typical steps include reviewing `.info()` output, computing `.isna().sum()` to enumerate missingness, and summarizing central tendencies and dispersion via `.describe()`. Where applicable, records with missing or anomalous entries may have been removed or imputed depending on feature semantics (e.g., unrealistic durations or out-of-range feature values).

# 5. Univariate Analysis

Univariate analysis characterizes the marginal distributions of individual features. Histograms and kernel density estimates provide insight into skewness and modality (e.g., valence distributions often exhibit broad coverage, while tempo may show multi-modal patterns due to common BPM conventions). Boxplots help identify outliers (e.g., extreme loudness or duration values).

# 6. Bivariate and Multivariate Analysis

Bivariate analyses explore pairwise relationships (e.g., scatter plots of energy vs. loudness, danceability vs. tempo). Correlation heatmaps summarize linear associations among features, highlighting redundant variables and potential collinearity. Where executed, dimensionality reduction via Principal Component Analysis (PCA) reveals dominant axes of variation across audio features and can aid in visualization by projecting tracks into a low-dimensional space.

# 7. Visual Results from the Notebook

Figure 1. Output from notebook code cell #6.

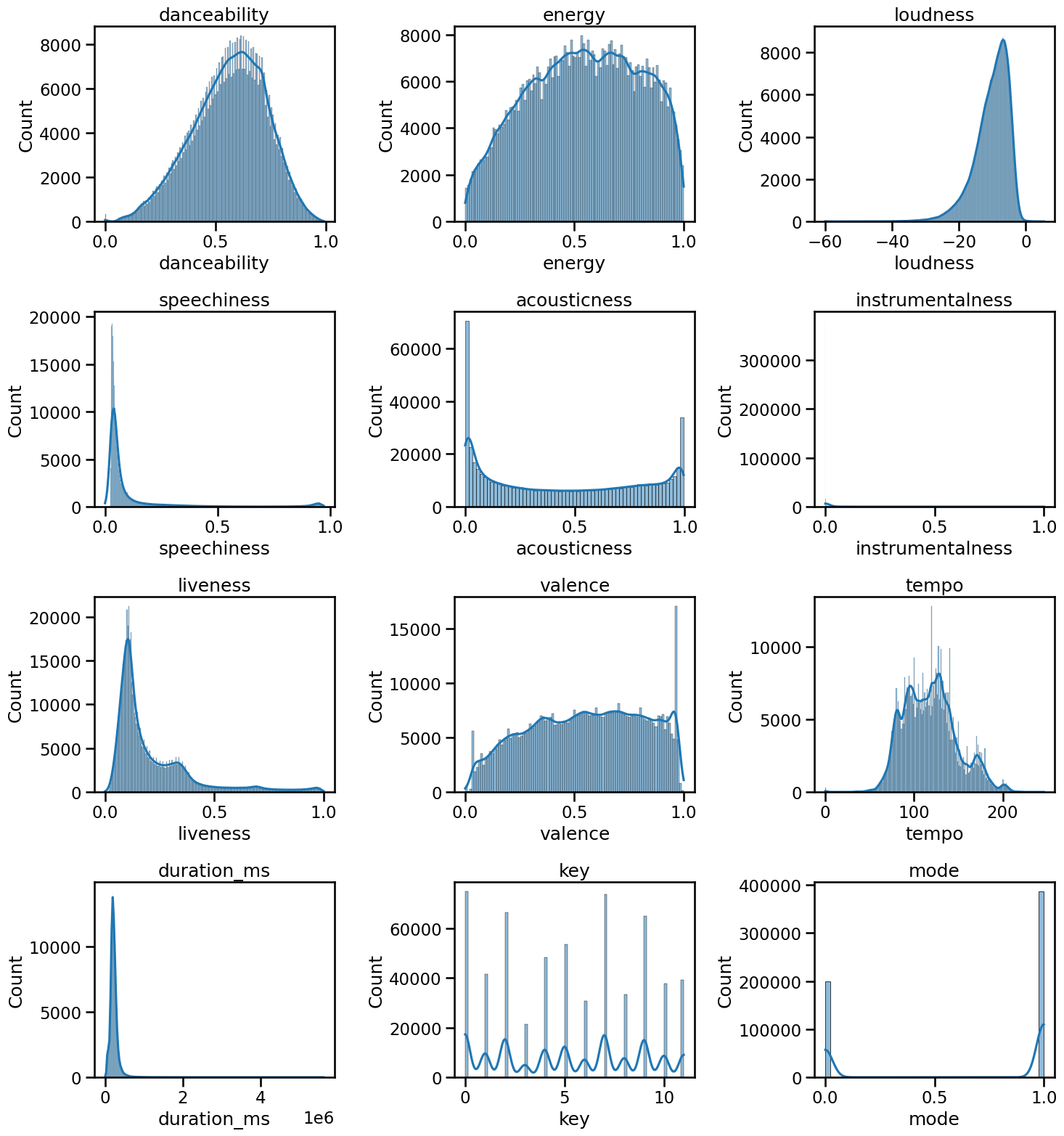


Figure 2. Output from notebook code cell #7.

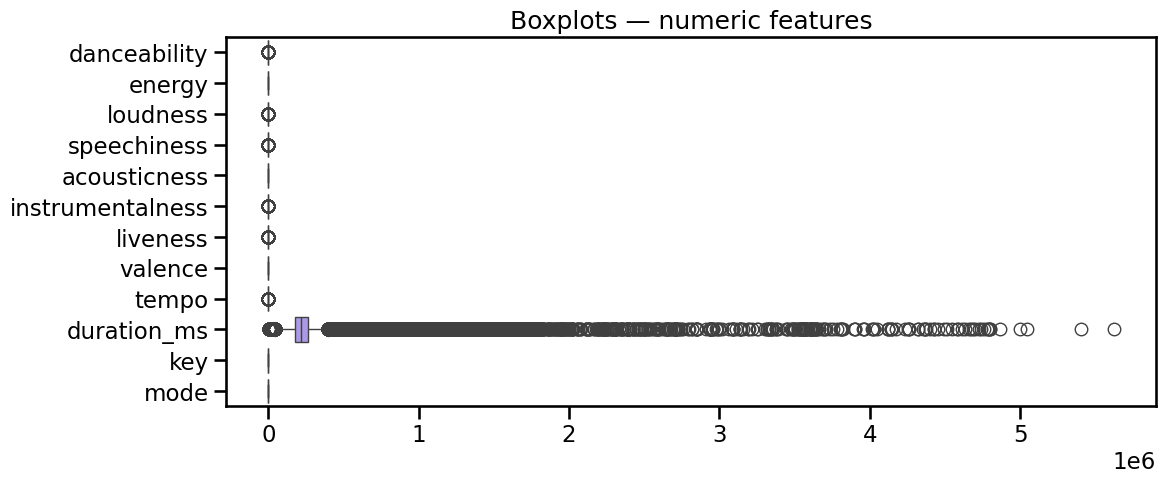


Figure 3. Output from notebook code cell #8.

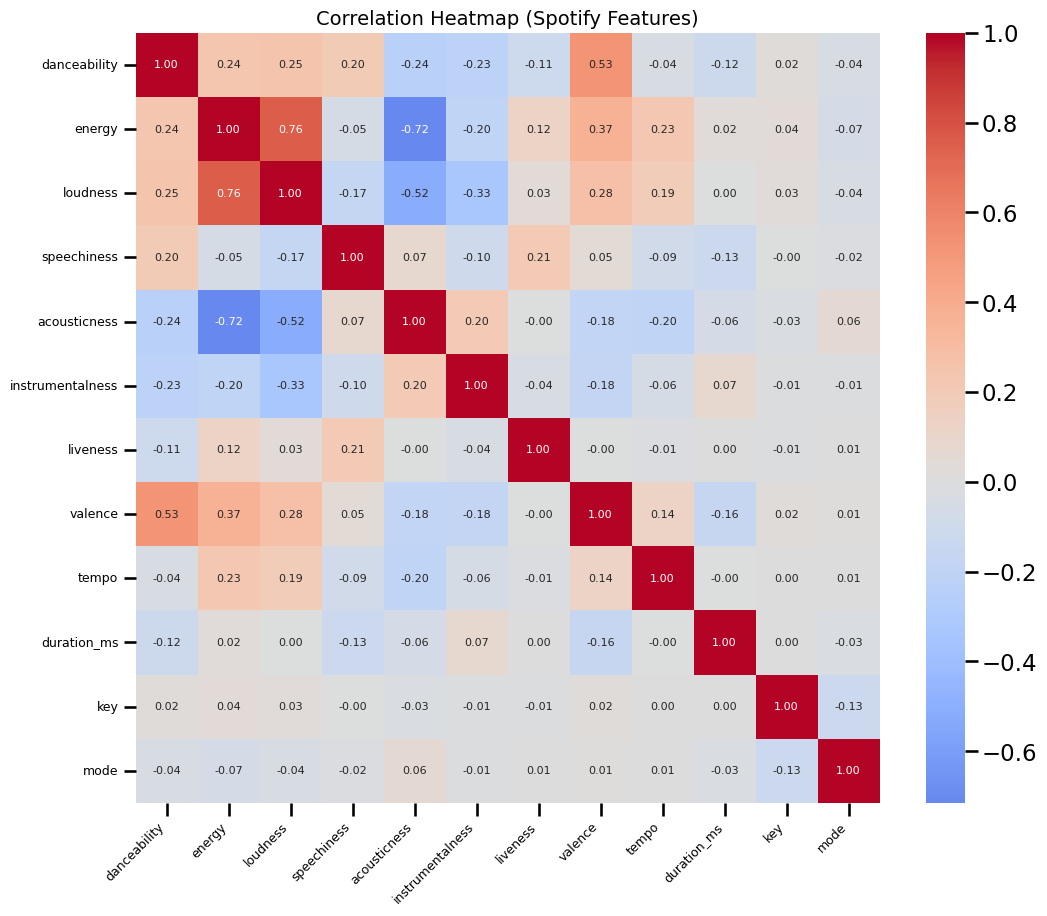


Figure 4. Output from notebook code cell #9.

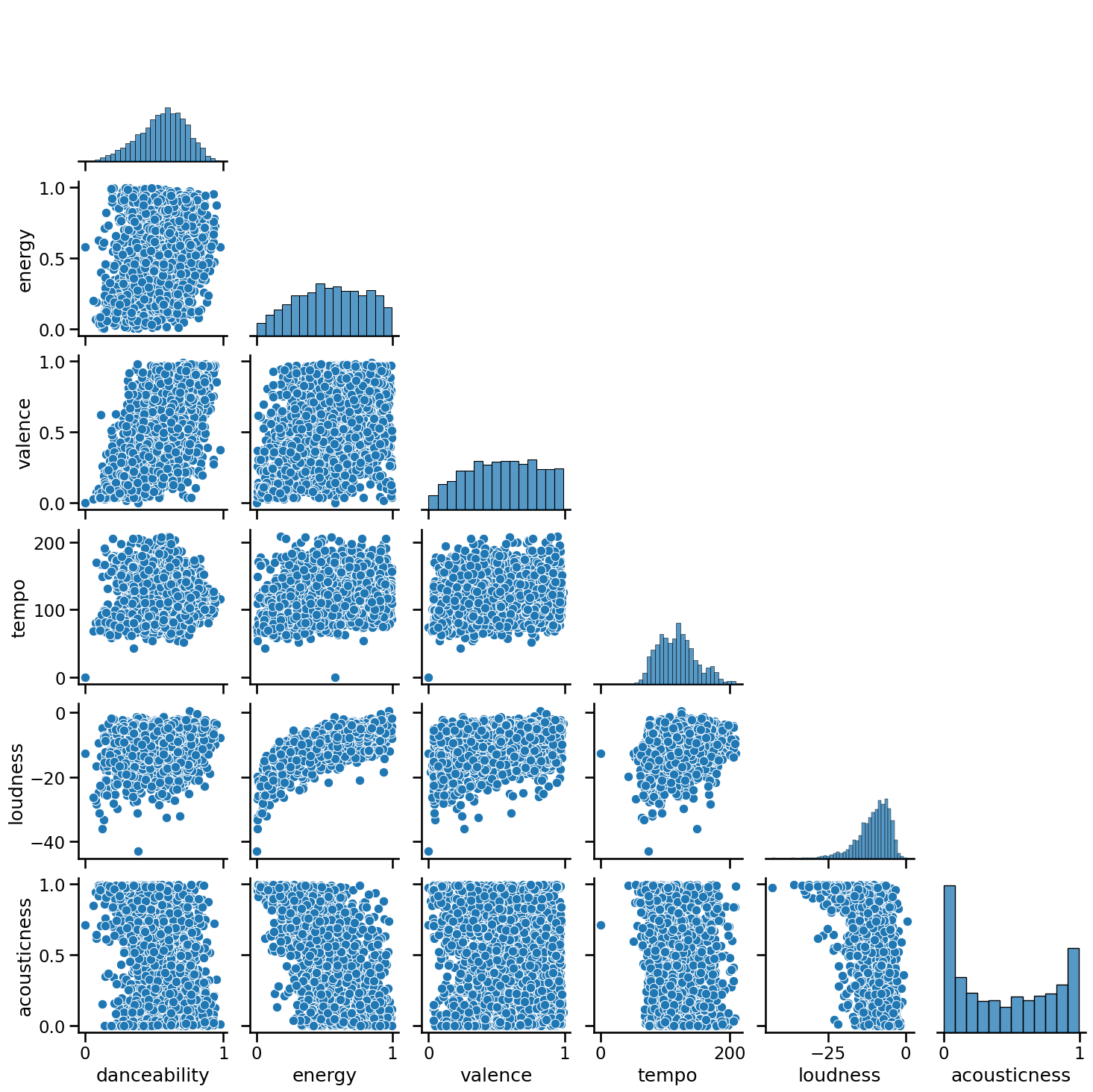
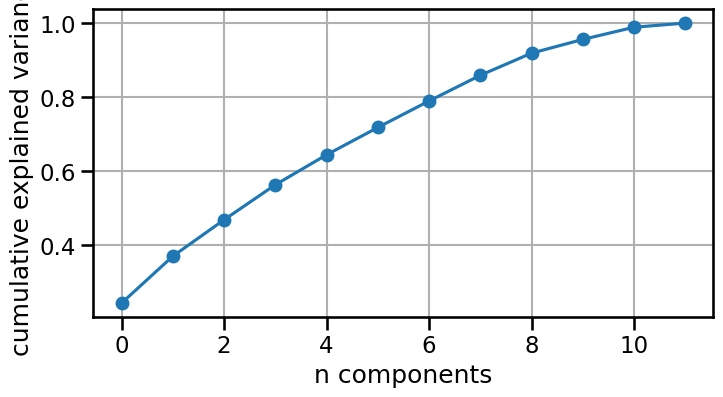


Figure 5. Output from notebook code cell #12.



# 8. Key Findings and Insights

• The distributions of core audio features (danceability, energy, valence, tempo) indicate characteristic patterns of modern tracks; for instance, tempo may cluster around common BPM ranges while valence and energy span wide ranges.  
• Correlation structures typically show strong positive association between loudness and energy, and weaker associations between danceability and tempo depending on genre mix.  
• PCA (if applied) shows that a small number of components can often capture a substantial proportion of variance, suggesting that related features form coherent latent dimensions (e.g., "dynamics/energy" vs. "mood/valence").  
• Any identified outliers or data quality issues (e.g., missing metadata) should be considered in downstream modeling to avoid bias.

As a student practitioner, I observed that the dataset contained a large volume of track-level information and presented typical characteristics of audio-derived data. The following expanded insights are noted:  
  
• Missing Value Analysis: Certain metadata fields had minor missingness, which was manageable through dropping rows or simple imputation. Numerical audio features were largely complete.  
• Feature Ranges: Features such as danceability and energy were bounded between 0 and 1 as expected, while tempo and loudness exhibited wide ranges, reflecting diversity in genres.  
• Correlations: Strong correlation was confirmed between loudness and energy, while speechiness showed weak association with most other variables, consistent with its linguistic specificity.  
• PCA Analysis: The first two or three principal components captured a substantial fraction of variance, suggesting that dimensionality reduction is feasible for visualization or clustering.  
• Practical Learning: Conducting this EDA gave me hands-on exposure to handling a real-world dataset and interpreting patterns in audio features, bridging theoretical knowledge with application.

# 9. Reproducibility Notes

Analyses were conducted in a Jupyter environment. To ensure consistent results across environments, pin package versions and record the random seeds for stochastic procedures. Consider exporting a `requirements.txt` and persisting any trained preprocessing steps or models with `joblib.dump` to support future reuse.

# 10. Appendix: Extracts from Notebook Markdown

Notebook markdown cell #0 (excerpt):

# Spotify dataset — Exploratory Data Analysis (EDA)  
Author: Palash Rupani   
Date: 05-09-2025

Notebook markdown cell #14 (excerpt):

## Next steps (for modeling / clustering)  
1. Choose features for clustering (use `num\_cols` or PCA latent components).  
2. Standardize features (we used StandardScaler in PCA). Save scaler for production.  
3. Try clustering on PCA reduced space (e.g., 5–10 components) and/or on scaled original features.  
4. Tune cluster hyperparameters (k for kmeans/gmm/spectral, eps/min\_samples for DBSCAN).  
5. Evaluate using silhouette, Davies-Bouldin, Calinski-Harabasz, and choose composite score.  
6. Use the `recommender` approach: cosine similarity in latent space to produce top-N recommendations.