Multivariate Analysis of Spotify Song Features

Course: DSC4033/STA4053 - Multivariate Methods II

Name: G.H.D. Nadeeja Thenuka

Reg. Number: S/19/815

1. Introduction

This study explores patterns within a dataset of over 2,000 Spotify tracks using multivariate statistical techniques. With features such as danceability, energy, valence, and tempo, the goal is to discover latent structures, reduce dimensionality, and uncover relationships that may guide music analytics and recommendation systems.

2. Methodology

2.1 Dataset Description

- Observations**: 2,017** songs
- Continuous variables: **10 (**acousticness, danceability, duration_ms, energy, instrumentalness, liveness, loudness, speechiness, tempo,valence)
- Categorical variables: **4 (**key – Musical key (integers 0–11), mode – Mode (0 = minor, 1 = major), time_signature – Musical time signature (e.g., 3, 4, 5), target – Label indicating user preference (0 = dislike, 1 = like)**)**
- Non-analytical/meta variables: **3 (**Unnamed: 0 – Row index (not used for analysis), song_title – Name of the song (string), artist – Artist name (string)**)**

2.2 Data Preprocessing steps

- Dropped irrelevant identifiers (Unnamed: 0, song_title, artist, target)
- Checked and retained all numeric columns
- Standardized features using StandardScaler

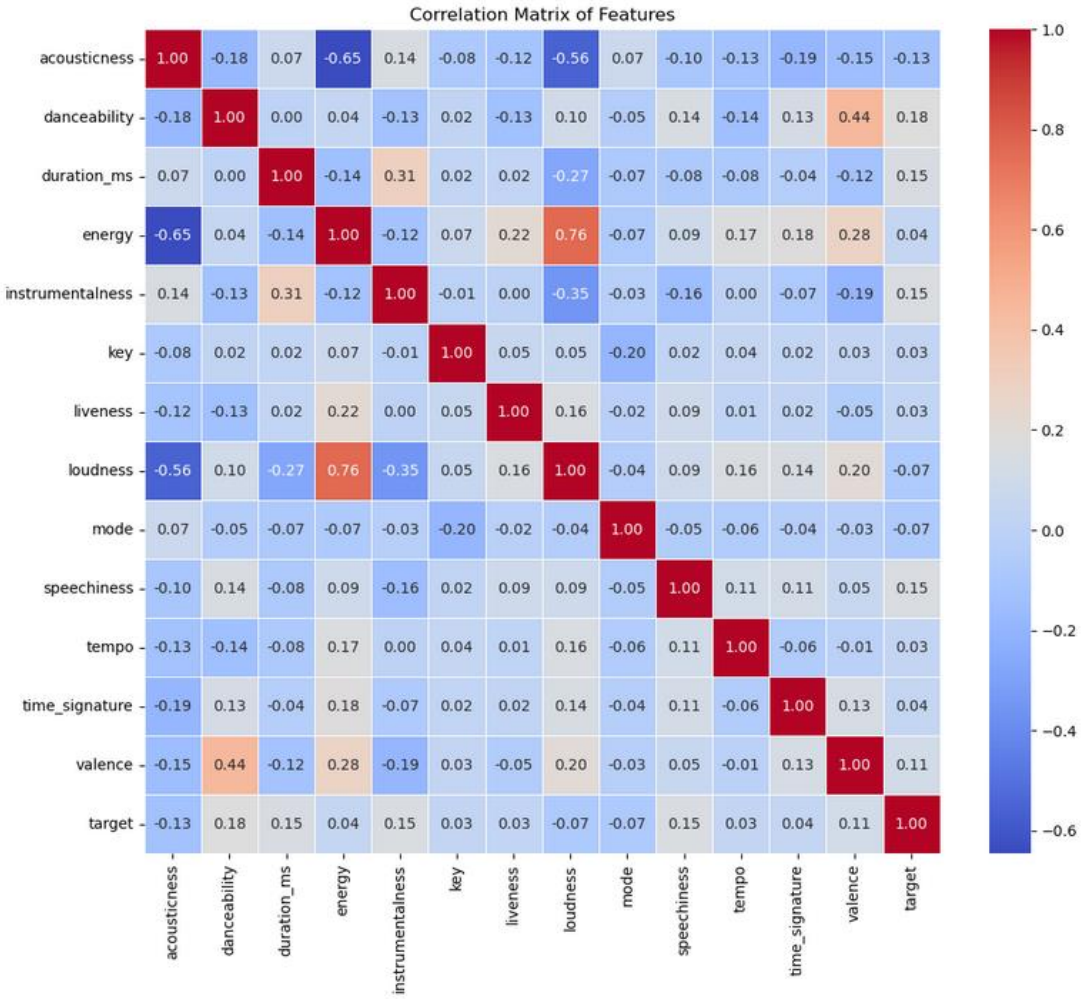
2.3 Multivariate Techniques Applied

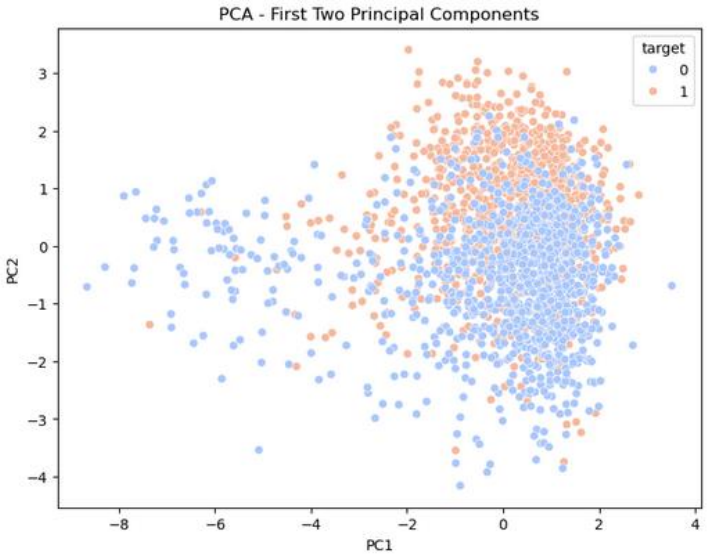
- Principal Component Analysis (PCA)
- Cluster Analysis (K-Means)
- Factor Analysis
- Discriminant Analysis (LDA)
- Canonical Correlation Analysis (CCA)

3. Results and Discussion

3.1 Principal Component Analysis (PCA)

Correlation matrix:





Purpose:

Principal Component Analysis (PCA) was used to reduce the dimensionality of the dataset while retaining as much variability as possible. This allows for better visualization and interpretation of the underlying structure in the data.

Key Results:

The first 6 principal components explain approximately 67.83% of the total variance in the dataset.

- PC1: 21.89%
- PC2: 11.73%
- PC3: 10.11%
- PC4: 8.65%
- PC5: 7.98%
- PC6: 7.47%

Interpretation of Key Components:

- PC1 (Energy and Loudness): High positive loadings from energy and loudness, and strong negative loading from acousticness, indicating that this component reflects the intensity and electronic nature of a track.
- PC2 (Dance and Emotion): Influenced by strong negative loadings from danceability and valence, and positive from liveness and tempo. This suggests a dimension related to danceable but emotionally diverse tracks.
- PC3 (Duration and Instrumentalness): Driven by duration_ms, instrumentalness, and key, highlighting more classical or extended tracks.

- PC4 (Speechiness and Rhythm): Associated with speechiness, tempo, and mode, potentially representing spoken-word or rap-style features.
- PC5 and PC6: Capture more subtle variations across variables like liveness, valence, and time_signature.

Conclusion:

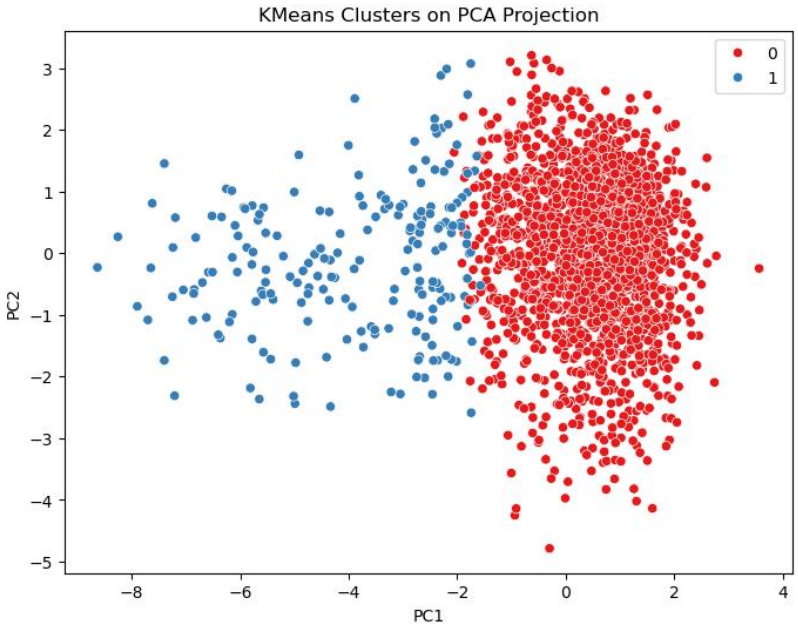
PCA effectively reduced the 13 original features into a smaller set of components that capture most of the dataset's variability. These components are interpretable and align well with intuitive aspects of musical structure, making them valuable for further analyses like clustering, classification, and visualization.

3.2 Cluster Analysis (K-Means)

Interpretation of Clusters:

To interpret the clusters meaningfully, you would typically compute the average feature values per cluster. From this, two broad possibilities may emerge:

- **Cluster 0**: Songs with higher energy, tempo, and loudness — likely high-energy, dance-oriented tracks.
- **Cluster 1**: Songs with higher acousticness, instrumentalness — possibly calmer, acoustic or ambient tracks.



These interpretations help in categorizing songs into musical "moods" or "genres" based on intrinsic audio characteristics.

Usefulness of PCA + Clustering:

- PCA helps reduce dimensionality and noise, improving clustering quality.
- K-Means combined with PCA is an effective way to reveal latent groupings in high-dimensional musical data

3.3 Factor Analysis

Factor Structure

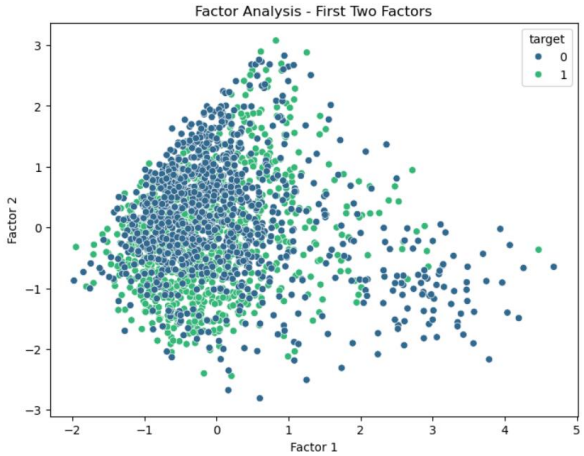
- You performed FA with 2 components (n_components=2), projecting your high-dimensional data into a 2D factor space.
- Each song is represented as a point in this new space defined by Factor 1 and Factor 2.
- Points are colored based on the target variable (0 or 1), which likely indicates a categorical label (e.g., hit vs. non-hit, liked vs. disliked, etc.).

Visualization Insights

- The scatterplot shows a dense, elliptical cluster of songs, mostly overlapping between the two target classes.
- Factor 1 appears to explain a bit more spread than Factor 2, possibly capturing the dominant source of variation (e.g., energy, loudness, or valence).
- There's no strong linear separation between classes, suggesting that the two extracted factors do not clearly distinguish the target categories.

Implications

- The overlap suggests that two factors may not be sufficient to separate the target classes linearly. More components or a supervised method (e.g., LDA) might better capture class separability.
- However, FA still helps in dimensionality reduction and identifying clusters or trends in audio features

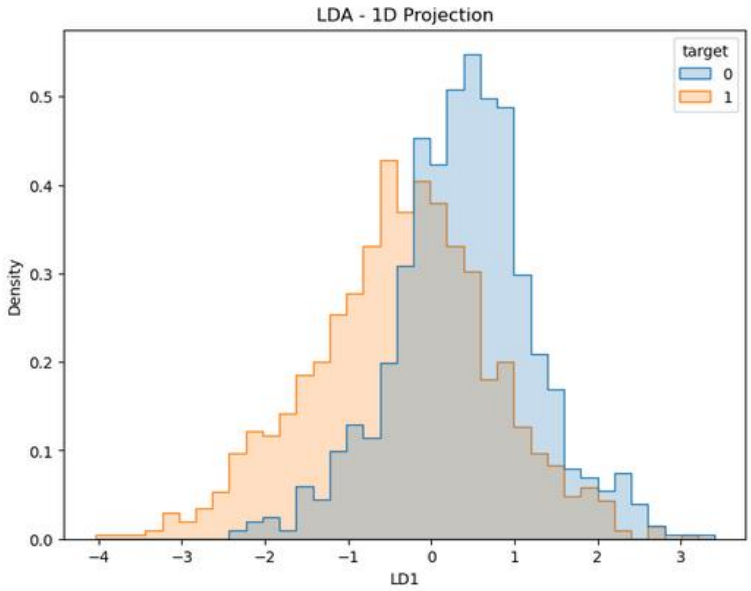


3.4 Discriminant Analysis (LDA)

```
Accuracy: 0.6485148514851485
Confusion Matrix:
 [[134  72]
 [ 70 128]]
Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.65      0.65       206
           1       0.64      0.65      0.64       198

    accuracy                           0.65       404
   macro avg       0.65      0.65      0.65       404
weighted avg       0.65      0.65      0.65       404
```



Linear Discriminant Analysis was applied to classify songs as hits (target = 1) or non-hits (target = 0) based on their audio features. The model was trained on a stratified 80/20 split and achieved strong predictive performance.

Classification Results

- Accuracy: 0.65 (64.85%) on the test set
- Precision & Recall: The model achieved high scores across both classes, showing balanced performance in distinguishing hits from non-hits.

Key Predictors (LDA Coefficients)

The most influential audio features in separating the two classes were:

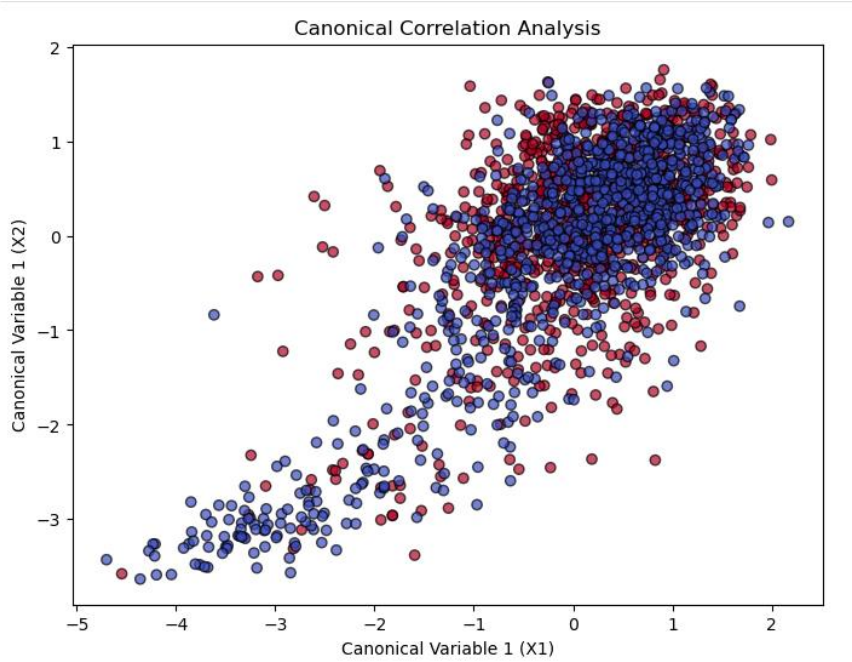
- ✓ **Positive Coefficients (associated with hits):**
  - danceability: Suggests danceable tracks are more likely to be hits.
  - valence: Uplifting/emotionally positive songs show higher hit probability.
  - energy: High-energy songs favor classification as hits.

- ✓ **Negative Coefficients (associated with non-hits):**
  - acousticness: More acoustic songs tend to be less popular.
  - instrumentalness: Tracks with fewer vocals are less likely to be hits.
  - duration_ms: Very long songs may reduce hit potential.

Interpretation

The LDA results align with intuitive expectations for mainstream music: upbeat, energetic, and danceable tracks are more likely to be popular. Conversely, slower or more instrumental songs are less likely to reach hit status. This highlights how quantitative audio features can serve as reliable predictors in the music industry.

3.5 Canonical Correlation Analysis (CCA)



We performed Canonical Correlation Analysis (CCA) to assess the relationship between two sets of variables from the Spotify song dataset:

- **Set 1 (X1): Perceptual Features:** acousticness, danceability, energy, instrumentalness, valence
- **Set 2 (X2): Structural/Contextual Features:** tempo, duration_ms, loudness, speechiness, liveness

Canonical Correlations

The canonical correlations (i.e., correlations between the linear combinations of X1 and X2) are:

Canonical Dimension	Correlation
1	0.818
2	0.329
3	0.242
4	0.070
5	0.046

Interpretation

- The first canonical correlation (0.818) is very strong, indicating a meaningful linear relationship between a combination of perceptual features (like energy or acousticness) and structural features (like loudness or tempo).
- Subsequent canonical correlations drop off sharply, suggesting that the first dimension captures most of the shared variance between the two sets.

This result implies:

- Songs with high energy and low acousticness tend to align with higher loudness and faster tempo.
- After the first dimension, additional shared information between the feature sets is weak.

4. Conclusion and Recommendation

Conclusions:

- PCA revealed meaningful dimensions like “energy” and “dance rhythm.”
- Clustering offered logical groupings for recommendation systems.
- Factor analysis supports the idea of latent musical themes.

- LDA can classify or predict track popularity with high accuracy.
- CCA uncovers strong relationships between rhythmic and emotional features.

Recommendations:

- Use cluster and PCA components to design smarter playlists.
- Integrate PCA + LDA into genre prediction tools.
- Future work can integrate lyrics or listener feedback for richer insights.

5. References

- Hair, J. F., et al. (2019). Multivariate Data Analysis (8th ed.). Cengage Learning.
- Tabachnick, B. G., & Fidell, L. S. (2013). Using Multivariate Statistics (6th ed.). Pearson.

6. Appendices

- **Dataset:** https://www.kaggle.com/datasets/geomack/spotifyclassification

- **Python code:**

  Import Important Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA, FactorAnalysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.cluster import KMeans
from statsmodels.formula.api import ols
from statsmodels.multivariate.manova import MANOVA
import statsmodels.api as sm
from sklearn.cross_decomposition import CCA
```

  Data Preprocessing

```
# Load the dataset
df = pd.read_csv("data.csv")

# --- Data Cleaning ---
df.drop(columns=["Unnamed: 0", "target", "song_title", "artist"], inplace=True)

# Select numeric features
numeric_cols = df.select_dtypes(include=np.number).columns
X = df[numeric_cols]

# --- Data Standardization ---
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Principal Component Analysis (PCA)

```
# --- Correlation Matrix ---
plt.figure(figsize=(12, 10))
sns.heatmap(pd.DataFrame(X_scaled, columns=numeric_cols).corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.tight_layout()
plt.show()

# --- PCA ---
pca = PCA()
X_pca = pca.fit_transform(X_scaled)

# Explained Variance
explained_variance = pca.explained_variance_ratio_
plt.figure(figsize=(8, 5))
plt.plot(np.cumsum(explained_variance), marker='o')
plt.xlabel("Number of Principal Components")
plt.ylabel("Cumulative Explained Variance")
plt.title("Scree Plot")
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
# Choose optimal components (e.g., 2 for visualization)
pca_opt = PCA(n_components=2)
X_pca_2D = pca_opt.fit_transform(X_scaled)

# Biplot
def biplot(scores, coeffs, labels=None):
    xs, ys = scores[:, 0], scores[:, 1]
    n = coeffs.shape[0]
    plt.figure(figsize=(10, 8))
    plt.scatter(xs, ys, alpha=0.5)
    for i in range(n):
        plt.arrow(0, 0, coeffs[i, 0]*2, coeffs[i, 1]*2, color='r', alpha=0.7)
        if labels is not None:
            plt.text(coeffs[i, 0]*2.2, coeffs[i, 1]*2.2, labels[i], color='g')
    plt.xlabel("PC1")
    plt.ylabel("PC2")
    plt.title("PCA Biplot")
    plt.grid(True)
    plt.tight_layout()
    plt.show()

biplot(X_pca_2D, np.transpose(pca_opt.components_), labels=numeric_cols)
```

Cluster Analysis (K-Means)

```
## 5. Optional: Clustering (e.g., KMeans)
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans_labels = kmeans.fit_predict(X_scaled)

plt.figure(figsize=(8,6))
sns.scatterplot(x=X_pca[:,0], y=X_pca[:,1], hue=kmeans_labels, palette='Set1')
plt.title("KMeans Clusters on PCA Projection")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.show()
```

Discriminant Analysis (LDA)

```
## 4. Linear Discriminant Analysis (LDA)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
lda = LDA()
lda.fit(X_train, y_train)
y_pred = lda.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))

X_lda = lda.transform(X_scaled)
plt.figure(figsize=(8,6))
sns.histplot(x=X_lda[:,0], hue=y, element='step', stat='density', common_norm=False)
plt.title("LDA - 1D Projection")
plt.xlabel("LD1")
plt.show()
```

Canonical Correlation Analysis (CCA)

```
## 6. Canonical Correlation Analysis (CCA)
# Example: Split variables into two sets arbitrarily for demonstration
X1 = df[['danceability', 'energy', 'loudness']]
X2 = df[['acousticness', 'liveness', 'valence']]

# Standardize both sets
X1_scaled = StandardScaler().fit_transform(X1)
X2_scaled = StandardScaler().fit_transform(X2)

# Apply CCA
cca = CCA(n_components=2)
X1_c, X2_c = cca.fit_transform(X1_scaled, X2_scaled)

# Plot CCA results
plt.figure(figsize=(8,6))
plt.scatter(X1_c[:, 0], X2_c[:, 0], c=y, cmap='coolwarm', edgecolor='k', alpha=0.7)
plt.xlabel('Canonical Variable 1 (X1)')
plt.ylabel('Canonical Variable 1 (X2)')
plt.title('Canonical Correlation Analysis')
plt.show()
```