MOODIFY: Personalized Music Recommendation System

Welcome to MOODIFY, a personalized music recommendation system designed to understand your emotional state and musical preferences. This presentation will delve into the machine learning techniques that power MOODIFY, showcasing a sophisticated blend of algorithms to deliver tailored music experiences.

Project Objective: Addressing a Complex ML Problem

The core objective of MOODIFY is to provide personalized music recommendations by understanding the user's emotional state and musical taste. This isn't a singular machine learning problem but rather a **synthesis of multiple ML approaches**.

Natural Language Processing (NLP)	Unsupervised Learning	Recommendation Systems
Estimating mood from user-provided text.	Clustering songs into natural groups based on audio features.	Learning user preferences to generate personalized suggestions.

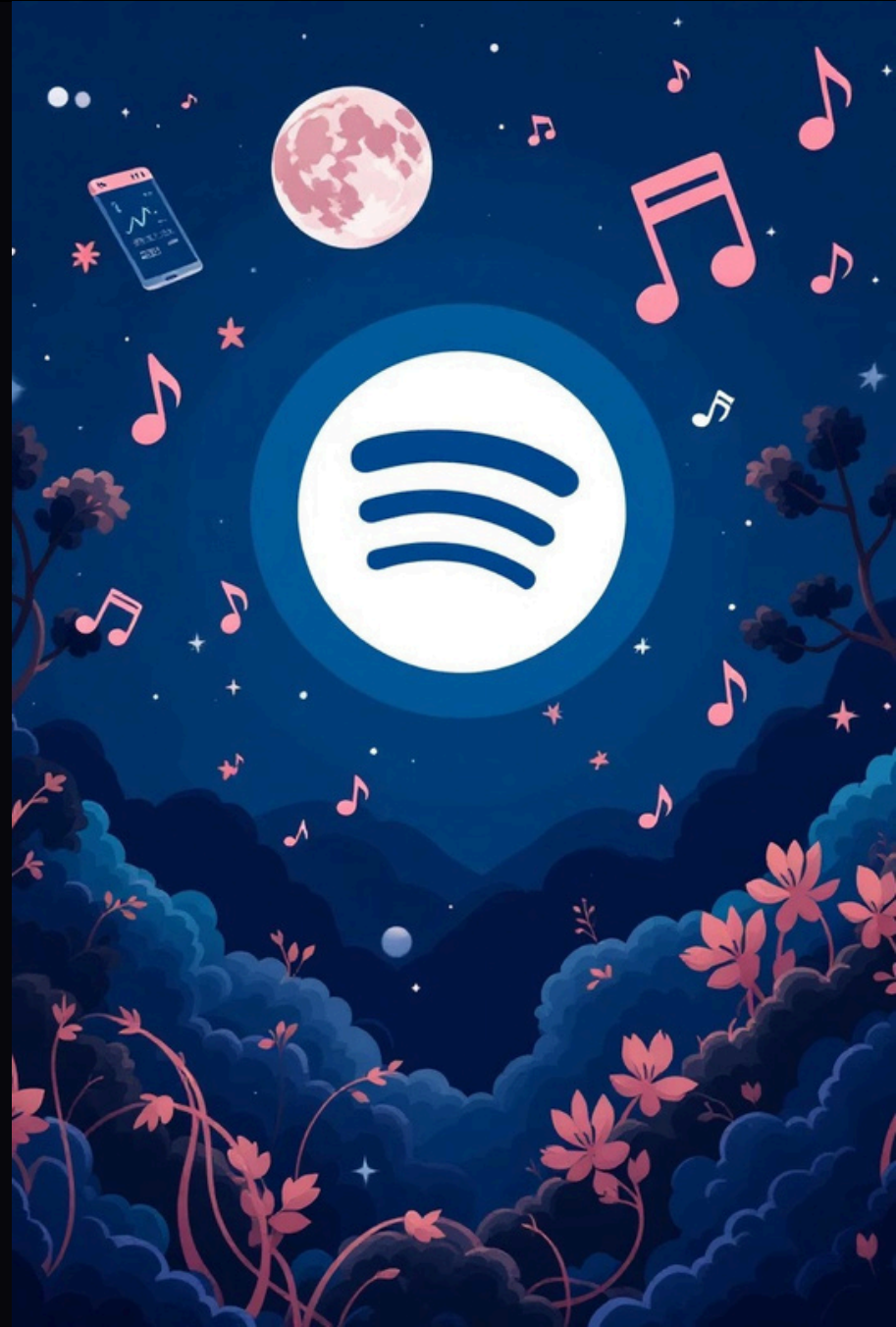
This multi-faceted approach classifies MOODIFY as a **Hybrid ML System**.

Data Understanding: Spotify Datasets

MOODIFY leverages two distinct datasets sourced from Spotify to ensure a comprehensive representation of musical diversity:

- **high_popularity_spotify_data.csv**
- **low_popularity_spotify_data.csv**

Merging these datasets ensures a more balanced representation of both popular and less popular songs, enriching the recommendation pool.
These features, derived from the Spotify Audio Features API, are numerical, making them directly suitable for machine learning algorithms.

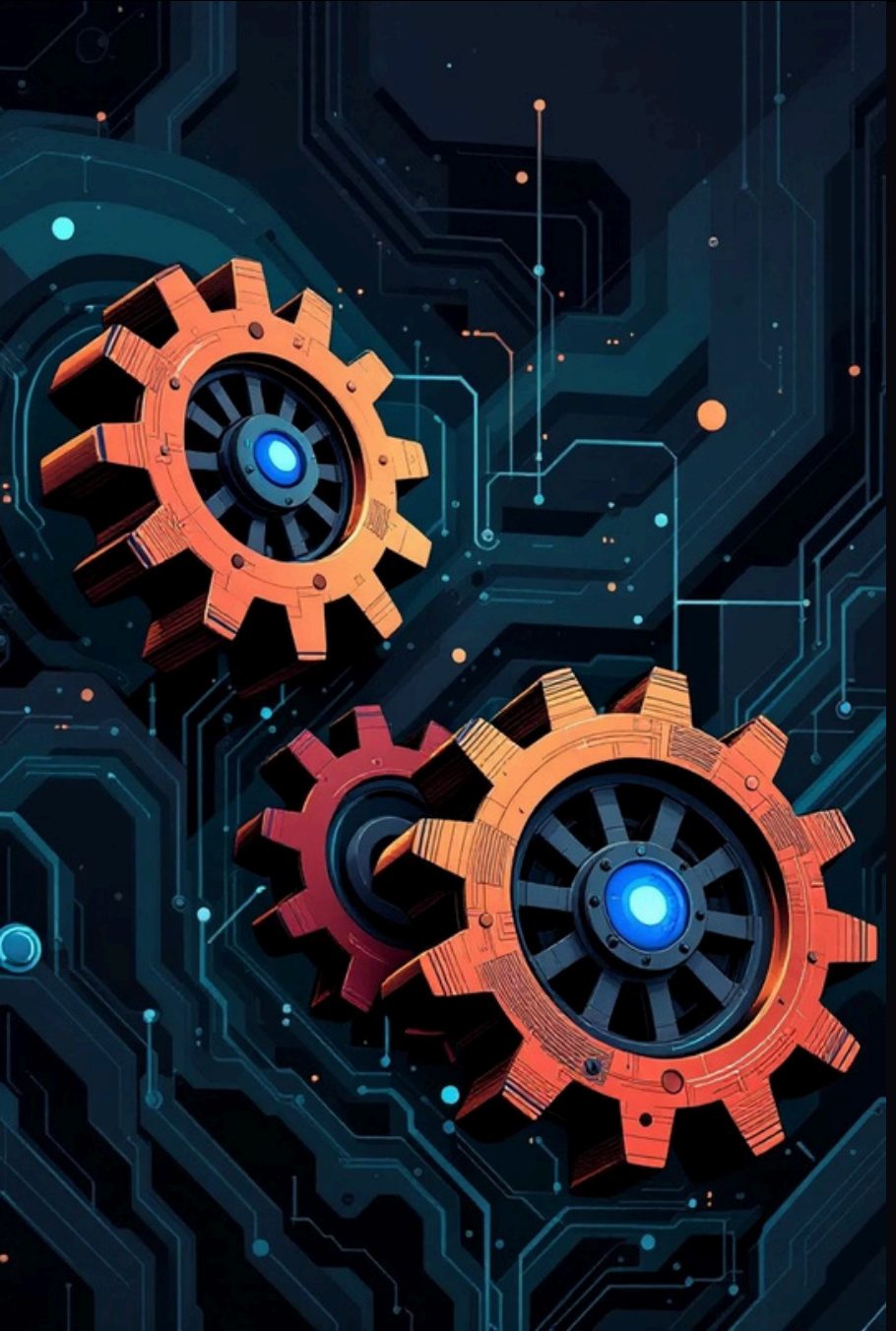


Critical Features for Machine Learning

For machine learning, understanding the core features of each song is paramount.These attributes provide the basis for clustering and recommendations:

Energy	Perceptual measure of intensity and activity.
Valence	Musical positiveness conveyed by a track (e.g., happy, cheerful vs. sad, angry).
Tempo	The overall estimated tempo of a track in beats per minute (BPM).
Danceability	How suitable a track is for dancing based on a combination of musical elements.
Acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic.
Instrumentalness	Predicts whether a track contains no vocals.
Mode	Indicates the modality (major or minor) of a track. (1= Major, 0 = Minor).

These features, retrieved directly from the Spotify Audio Features API, are numerical, making them ideal for ML algorithm processing.



Feature Engineering: Data Preparation for ML

Feature Engineering is a critical step to ensure the robustness and accuracy of our machine learning models.

Data Cleaning

```
df = df.dropna(subset=features + ['track_name',  'track_artist',  'mode'])
```

- Missing values are removed.
- Noise in the dataset is reduced.

Feature Scaling (StandardScaler)

```
scaler  =  StandardScaler()
df_scaled_values = scaler.fit_transform(df[features])
```

Scaling is essential because distance-based algorithms, like K-Means, are sensitive to features with larger scales. For instance, 'tempo' (0-200) and 'valence' (0-1) are on different scales. StandardScaler transforms data to have a mean of and a standard deviation of 1, ensuring **fair contribution** from all features.

This meticulous preparation **ensures healthy model learning** and prevents skewed results.

Unsupervised Learning: K-Means Clustering

Our objective is to group songs based on their audio characteristics without predefined labels.

Algorithm Utilized

KMeans(n_clusters=8 , random_state=42)

Why K-Means?

- **Speed and Interpretability:** Efficient for large datasets and easy to understand.
- **Numeric Feature Compatibility:** Highly compatible with the numerical audio features.
- **Natural Cluster Formation:** Music characteristics naturally form distinct groups.

Learned Information

The model groups songs with similar energy, tempo, and valence values into the same cluster. This process assigns a unique **ML cluster ID** to each song:

```
df['cluster'] = kmeans.fit_predict(df_scaled_values)
```

This provides a foundational layer for understanding song similarities.

Moodify

ML HYBRID RECOMMENDER SYSTEM

✨ Explore Moods | Personal Palette

How are you feeling?

e.g., I'm feeling dark and gloomy

Regenerate

Target Mood Override:

Happy

Locked out of Heaven

Bruno Mars

Cluster: 1 | Similarity: 0.00

Locked out of Heaven
Bruno Mars
03:53

Save

Dejavu

Seyi Vibez

Cluster: 6 | Similarity: 0.00

Dejavu
Seyi Vibez
02:40

Save

Mood Labeling: Bridging ML with Human Perception

WhileK-Meanseffectivelygroupssongs,itdoesn'tinherentlyassignhuman-understandablelabelslike"Sad"or"Happy." To bridge this gap, we apply **rule-based post-processing**.

1

K-Means Output

Numerical features (e.g., low valence, low energy, minor mode).

2

Rule-Based Assignment

Translates numerical patterns into mood labels.

3

Mood Label

"Sad," "Happy," "Calm," "Energetic."

Example Rule:

```
def assign_refined_mood(row):
if row['valence'] < 0.25 and row['energy'] < 0.3 and row['mode'] == 0:
    return"Sad"
```

This approach is a prime example of **feature interpretation**, translating raw ML output into meaningful, human-centric mood categories.

Supervised Learning: Text-Based Mood Classification

Topredict a user'smoodfromtheirwritteninput,weemployasupervisedlearningapproach.

Problem Statement

Estimating the user's mood from text input.

Algorithm Used

Logistic Regression (Multiclass Classification)

LogisticRegression()

NLP Pipeline

1

1. Text Input

2

2. CountVectorizer

```
vectorizer = CountVectorizer(stop_words='english')
```

3

3. Numerical Representation (Bag of Words)

4

4. Classification

Target Classes & Probability Output

- Happy, Sad, Calm, Energetic
- The predict_proba() function provides not just a prediction but also a **confidence score**, a crucial detail for advanced ML understanding.

Recommendation System: Collaborative Filtering

MOODIFY learns and adapts to user preferences through a collaborative filtering mechanism.

Learning System Logic

As a user saves songs, the system dynamically updates their musical profile:

```
user_profile = df_scaled.loc[liked_indices].mean().values
```

This calculates the user's **average music DNA**, representing their cumulative taste.

Similarity Calculation

Song recommendations are generated based on their proximity to the user's profile:

```
np.linalg.norm(song - user_profile)
```

- **Smaller distance:** Higher similarity, more relevant recommendation.
- **Larger distance:** Lower similarity, less relevant recommendation.

This is a foundational **distance-based recommendation** approach.

Why MOODIFY is a Robust Machine Learning Project

MOODIFY is not merely a collection of algorithms; it's a comprehensive ML project integrating various components:

Real Data Utilization

Leveraging authentic Spotify datasets for realistic scenarios.

Extensive Feature Engineering

Preprocessing and scaling for optimal model performance.

Hybrid Learning

Combining Supervised and Unsupervised learning techniques.

Distance-Based Similarity

Core of the recommendation engine.

User-Adaptive System

Continuously learns from user interactions.

Prediction & Probability

Delivering nuanced outputs, including confidence scores.

Model Interpretability

Providing insights through features like the ML Inspector.

Online Inference

Capable of real-time predictions and recommendations.

This architecture reflects a **real-world ML system**, making it an academically strong and practical demonstration of machine learning principles.