🎵 Emotion-Based Music Recommendation System using DeepFace and Deep Learning

🧠 Project Overview

This project integrates **facial emotion recognition** with **music mood classification** to recommend Spotify songs that match a user's detected emotion.

It uses **DeepFace** to analyze facial expressions and a **Deep Learning classification model** (trained using **Altair AI Studio (RapidMiner)**) to map songs to emotional categories.

The system flow is as follows:

> 🎥 ***Input Image*** → 🧠 ***DeepFace Emotion Detection*** → 🎶 ***Music Classification Model*** → 🔗 ***Join Results*** → 🎧 ***Spotify Song Recommendation***

It supports three major emotional states:

- 😊 **Happy**
- 😢 **Sad**
- 😐 **Neutral**

Whenever a user's facial image is given as input, the system automatically identifies the emotion and recommends a Spotify track that reflects that mood.

🗂️ Files Included

File Name	Description
`278k_labelled_uri.csv`	The original **Kaggle dataset** with 278,000 labeled tracks containing musical and mood attributes.
`songs.csv`	A **refined dataset** derived from the Kaggle dataset, keeping only the essential features for emotion classification.
`Test.csv`	Contains **three unlabeled song samples** used for testing the trained deep learning model.
`Output.csv`	Shows the **output results** when a "happy" image is given as input.
`Happy.jpg`, `Sad.jpg`, `Neutral.jpg`	Sample facial images used to test the **DeepFace emotion recognition**.
`emotion_music_recommender.rvg`	The **Altair AI Studio (RapidMiner)** process file that integrates both the DeepFace and the Deep Learning classification components.

File Name	Description
`README.md`	This documentation file containing setup, process, and execution details for the judges.

⚙️ Requirements & Setup

🧩 Software

- **Altair AI Studio (RapidMiner)**
- **Python 3.12 environment**

🐍 Python Libraries

This project relies on specific Python library versions to avoid dependency conflicts (especially in the TensorFlow/Keras ecosystem). Judges **must** follow these steps to ensure a smooth run:

1. **Create and Activate a Virtual Environment** (`deepface_env`):

```
python -m venv deepface_env
# Windows:
deepface_env\Scripts\activate
# macOS/Linux:
source deepface_env/bin/activate
```

2. **Install Required Libraries** (`requirements.txt`): (Place the list below into a file named `requirements.txt` in the project directory, then run the install command.)

```
tensorflow==2.20.0
tf-keras==2.20.1
deepface
pandas
opencv-python-headless
```

```
pip install -r requirements.txt
```

3. **Configure Altair AI Studio Python Path:** Point the Altair Interpreter to your new environment's executable:

`Altair Studio → Preferences → Python → Interpreter Path` should be set to:

```
C:\Users\YourUser\Path\To\deepface_env\Scripts\python.exe
```

▶️ How to Run the Project

1. **Open Altair AI Studio (RapidMiner).**
2. **Import the process file:** `File → Import Process → emotion_music_recommender.rvg` .

3. Ensure all required datasets (`songs.csv` , `Test.csv` , `Output.csv`) and image files (`Happy.jpg` , `Sad.jpg` , `Neutral.jpg`) are stored **in the same directory** as the `.rvg` file.
4. Open the **`Create ExampleSet`** operator and point it to one of the test images (`Happy.jpg` , `Sad.jpg` , or `Neutral.jpg`).
5. Click the **Run button** in Altair to execute the workflow.

🧠 System Execution Flow

1. 🖼️ **Image Input** → Provided through `Create ExampleSet` .
2. 🔍 **Emotion Detection** → **DeepFace** identifies facial emotion (happy, sad, neutral) inside the `Execute Python` block.
3. 🎶 **Song Classification** → Deep Learning model predicts the mood category of each song.
4. 🔗 **Join Operation** → Combines detected emotion with classified songs.
5. 🎧 **Output Display** → Shows the emotion and corresponding Spotify track link.

📊 Dataset and Model Details

Dataset Description

The Kaggle dataset used contains detailed audio features such as:

Feature	Description
Danceability	How suitable the track is for dancing (0–1).
Energy	Perceptual measure of intensity and activity.
Loudness	Overall volume in decibels (dB).
Acousticness	Confidence measure of whether a track is acoustic.
Instrumentalness	Likelihood that the track contains no vocals.
Valence	Positiveness of the song's mood.

Emotion Mapping: The initial labels are simplified: `0 → Sad, 1 → Happy, 2 → Energetic, 3 → Neutral` .

- To align with DeepFace output, **Energetic (2)** is treated as **Happy (1)**.

Model Training Details

Parameter	Description
Environment	Altair AI Studio (RapidMiner)
Dataset Used	`songs.csv` (preprocessed using Turbo Prep)
Model Type	Deep Learning Classifier (via Auto Model)
Best Performing Model	Random Forest with **79% accuracy**

Parameter	Description
Overall System Accuracy	**97.35%** (enhanced by DeepFace precision)

💾 Example Output Format

The final output is joined and formatted for readability:

Row No.	Spotify Link	Prediction (Label)	Prediction	Image Path
1	`https://developer.spotify.com/documentation/web-api/concepts/spotify-uris-ids4`	1	Happy	`Happy.jp`

💡 Use Cases & Conclusion

Use Cases

- 🚗 **Driver Alertness System** — Plays energetic music if the driver appears sleepy or sad.
- 🎧 **Mood-Based Playlist Generator** — Suggests playlists matching the detected mood.
- 🧘 **Therapeutic Assistant** — Plays calm or joyful tracks for stress relief.

Conclusion

The Emotion-Based Music Recommendation System demonstrates how AI and Deep Learning can merge to interpret human emotion and provide a personalized music experience. With a total system accuracy of **97.35%**, the project bridges the gap between human emotion and technology—turning facial expressions into melodies. **"The system doesn't just play music—it understands you."**

✍️ **Author** Nirlesh Saravanan Sri Sairam Engineering College, Chennai