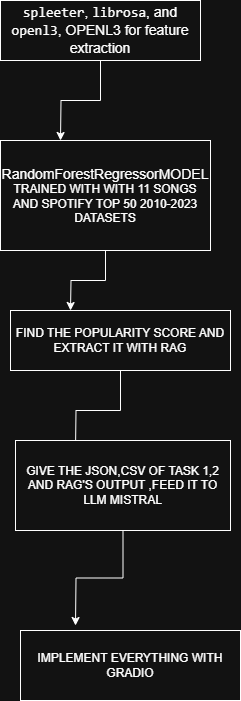
**Title:** Smart Audio Analysis & Viral Track Prediction with AI

Colab: https://colab.research.google.com/drive/1bTWqKOauz5yyI1CvKuWktY-8mvPiMifd?usp=sharing

### **Project Purpose**

The goal of this project is to merge advanced audio analysis with AI-generated feedback to assess the viral potential of a song. Built using Gradio and powered by a local instance of the Mistral language model (via Ollama), this tool is ideal for playlist curators, music producers, or anyone looking to understand what makes a song stand out.I’ve also read some papers to have the right pipeline for the project.

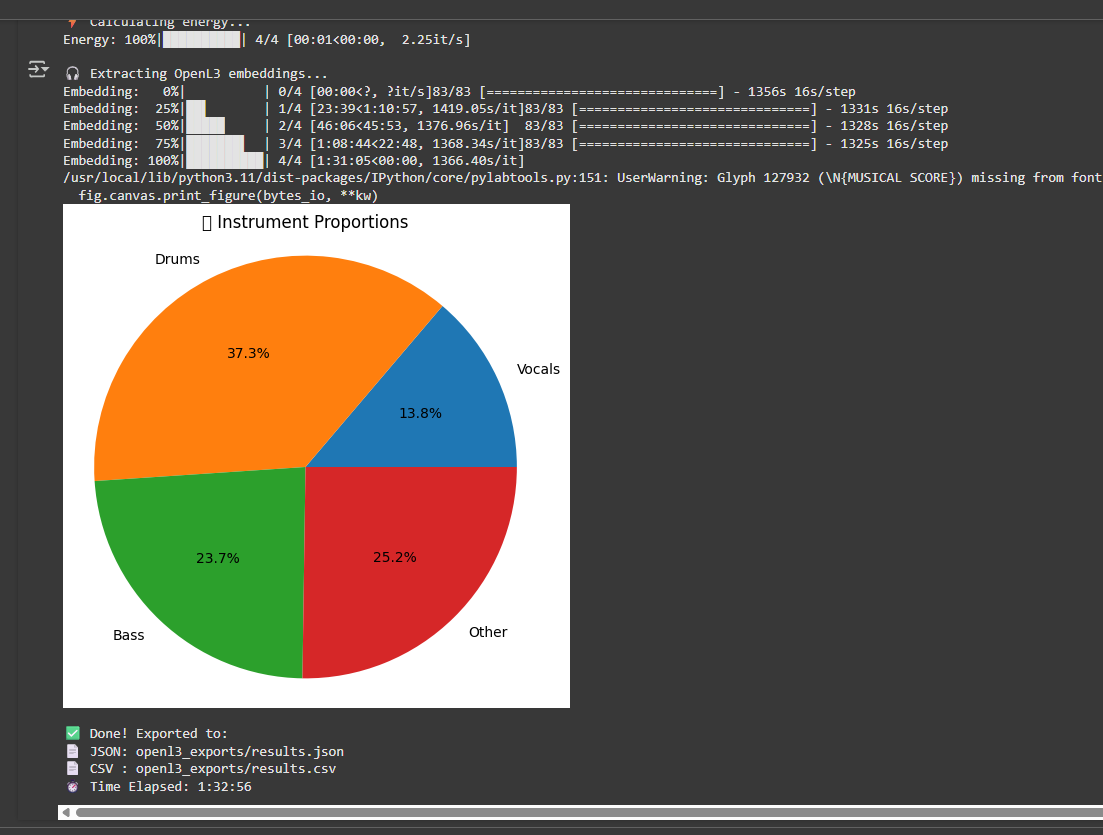


### 

### 

### **How the System Works**

TASK 1:

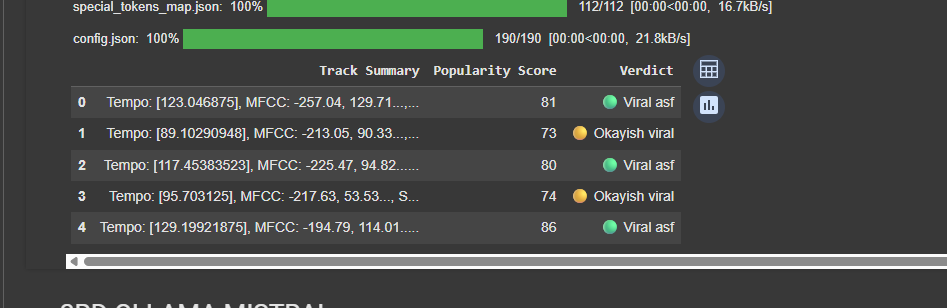


This Python script is a complete audio analysis pipeline designed to extract detailed musical features from an uploaded audio file. The process begins by installing essential dependencies like spleeter, librosa, and openl3, which are responsible for audio stem separation, feature extraction, and deep audio embeddings, respectively. Once an audio file is uploaded, the script uses spleeter to separate the track into four distinct stems: vocals, drums, bass, and other instruments. Each stem is then analyzed using librosa to compute its raw energy, loudness in decibels (dB), and duration in seconds. These measurements help determine the relative contribution of each instrument to the overall mix, which is later visualized in a pie chart showing the energy proportions.Here’s the pic of pipeline given below.In addition to signal-level analysis, the script leverages OpenL3, a pre-trained deep learning model, to generate a 512-dimensional embedding for each stem. These embeddings provide a rich, high-level representation of the audio content that can be used in downstream tasks such as genre classification, similarity comparison, or machine learning model input. All data — including raw energy metrics, stem durations, dB levels, proportions, and embeddings — are compiled into structured formats. The results are saved as both a JSON file (for structured data analysis) and a CSV file (for spreadsheet or ML pipeline usage). The script is built for Google Colab and includes automatic file downloads to make data export seamless.

Overall, this code provides a robust and efficient framework for analyzing the sonic composition of music tracks. It combines traditional digital signal processing with modern deep learning embeddings to produce both interpretable and machine-usable outputs, making it suitable for music information retrieval, audio dataset creation, or educational use in audio engineering and AI-based music research.

TASK:2

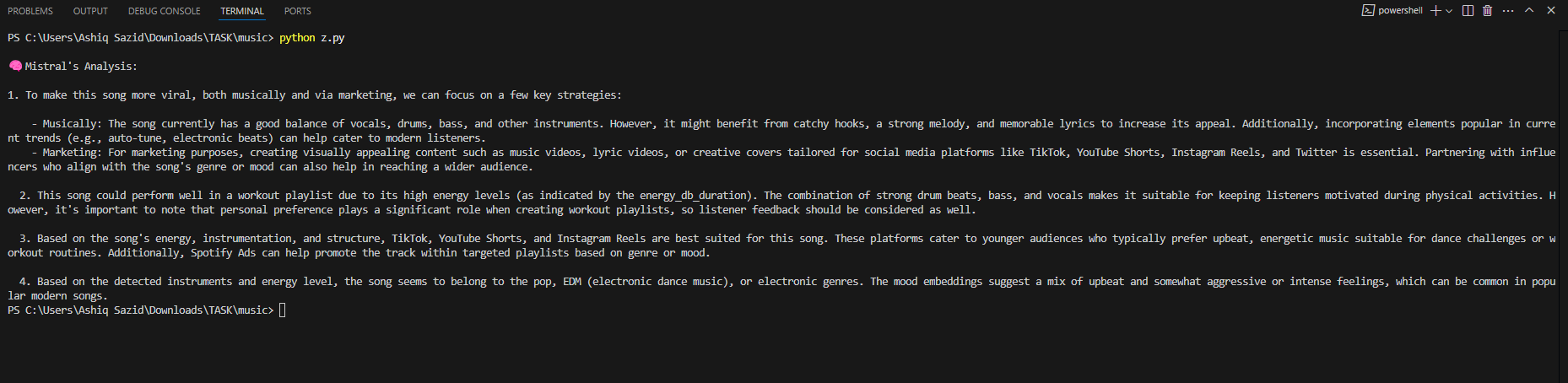
This Python script builds a predictive machine learning model that estimates the popularity of songs based on their audio features and associated metadata. It uses the librosa library to extract meaningful audio characteristics such as tempo, MFCCs (Mel-Frequency Cepstral Coefficients), spectral contrast, and chroma vectors from audio files. Each song is matched to its corresponding metadata from a CSV file, which includes details like the track name, artist name, artist popularity, year of release, duration, and the track's actual popularity score. The audio features are then combined with these metadata attributes to form a comprehensive feature set for each track.



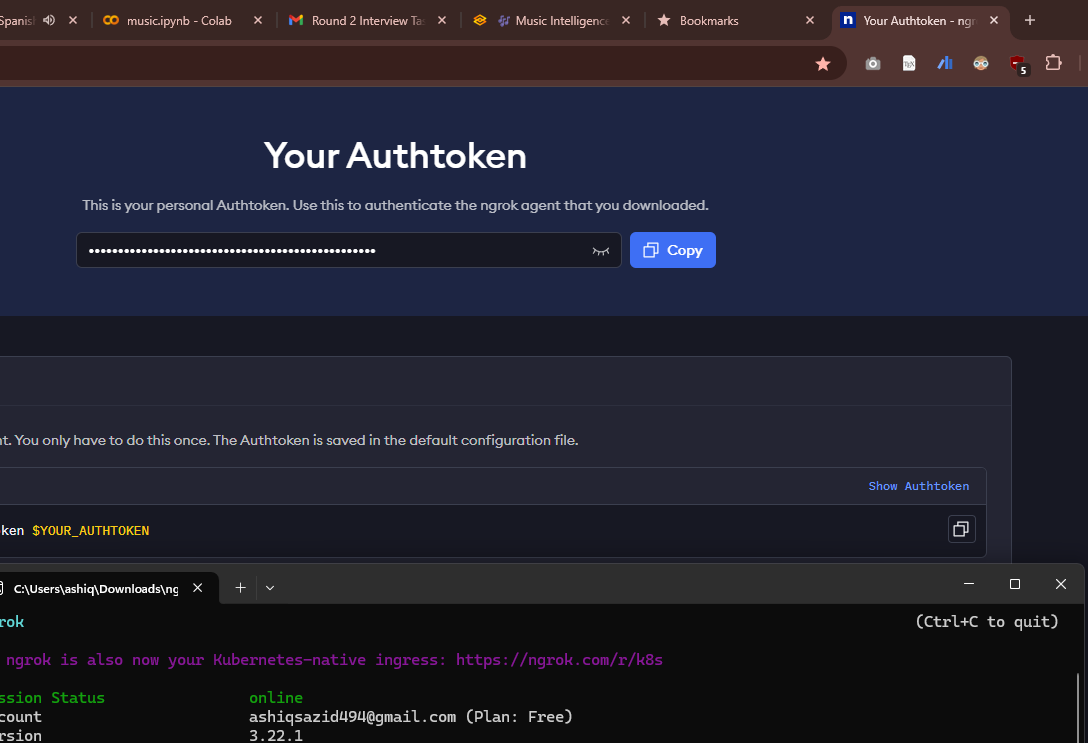
The script uses StandardScaler to normalize the input features and trains a RandomForestRegressor model from the scikit-learn library. This ensemble-based model is well-suited for capturing complex, non-linear relationships in the data. It is configured with 200 decision trees (n\_estimators=200) and a fixed random seed for reproducibility. After training, the model's performance is evaluated using three standard regression metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the R-squared (R²) score. MAE provides the average absolute difference between predicted and actual values, RMSE penalizes larger errors more heavily, and R² indicates how well the model explains the variance in song [popularity. We](http://popularity.we)’ve also rag and faiss vector to extract jscon and csv file.

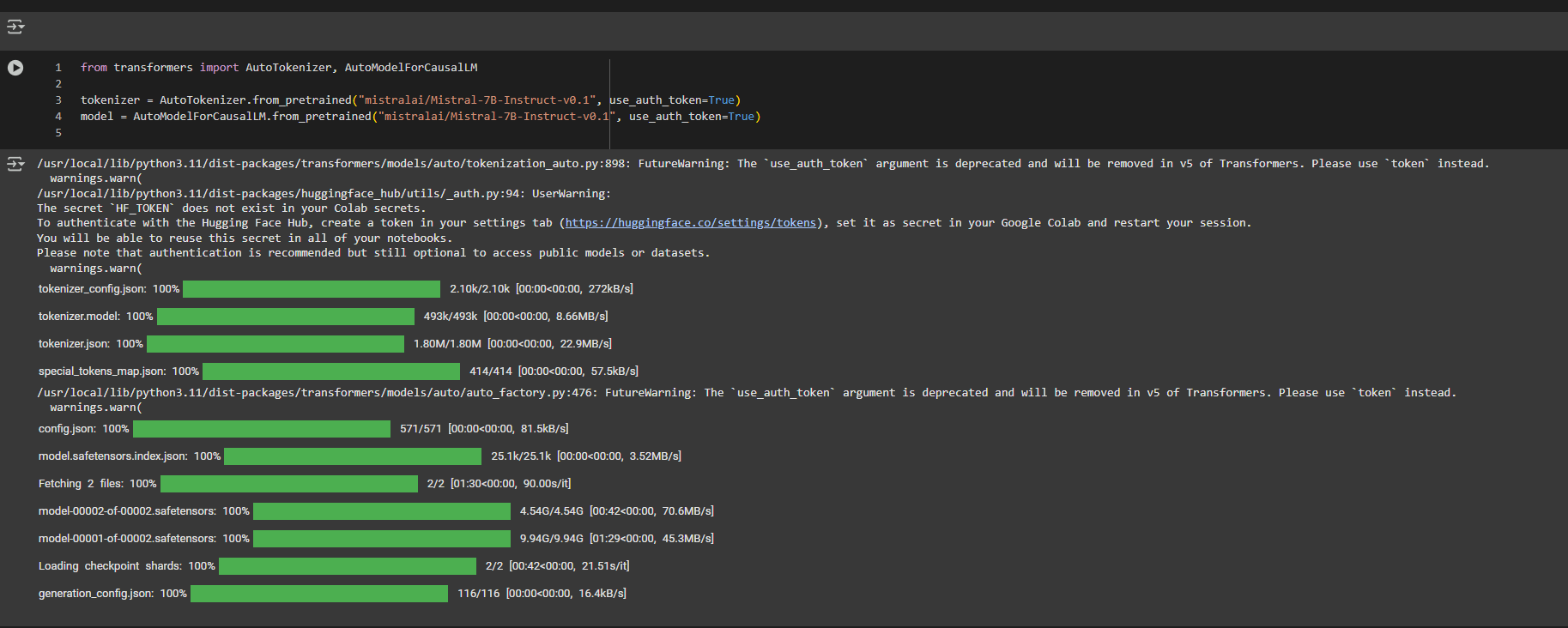
Finally, the script saves the trained model, the scaler, and the processed dataset into a local folder named virality\_audio\_model. It also exports a JSON file containing the evaluation metrics and a sample of the top 100 predictions, along with a CSV file containing the full feature set. This setup makes it easy to reload the model for future predictions or integrate it into a larger music analytics pipeline. Overall, the code provides a foundational workflow for predicting music virality using machine learning and signal processing techniques.

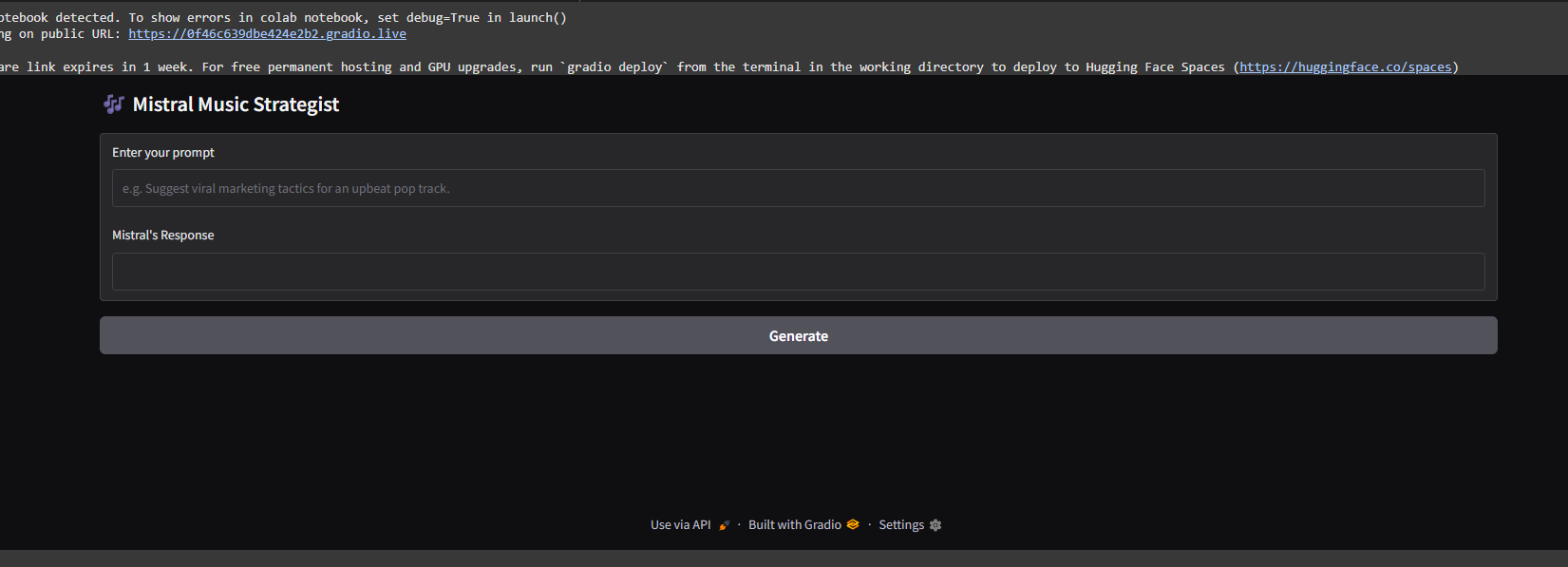
Here’s a breakdown of how the assistant functions:

1. **Loads Two Core Files:**
   * results.json: Contains data on instrument proportions and OpenL3-based mood embeddings
   * prediction\_result.json: Includes low-level audio features such as MFCCs, tempo, spectral analysis, chroma, and a predicted virality score
2. **Integrates Spotify Metadata:** It aligns each track with Spotify's real-world popularity data from playlist\_with\_popularity.csv.
3. **Categorizes Virality Score:** The model assigns a label to each track based on its score:  
   * Extremely Viral (Score ≥ 80)
   * Moderately Viral (70–79)
   * Low Virality (50–69)
   * Flop Zone (< 50)
   * With VS CODE  
     

With ngork+hugging face implementing gradio







I’ve also used stremelit.But between all of them,ollam mistral+Gradio suits so well.

1. **Summarizes Audio Features:**
   * Instrument detection and distribution
   * Proportional breakdown
   * Tempo analysis
   * MFCCs (timbre characteristics)
   * Spectral analysis (frequency energy distribution)
   * Chroma patterns (pitch class profiles)
2. **Language Model Feedback:** All extracted features are sent to the Mistral model, which allows users to ask natural language questions like, “Would this song work well for a gym playlist?” and receive contextual answers.

### **Sample Output for Track 0**

* **Instruments Detected:** Full range
* **Proportions:** 100%
* **Tempo:** 0.505 BPM
* **MFCCs:** Ranges from -1.66 to -0.27
* **Spectral Features:** Values between -1.52 and +0.28
* **Chroma Values:** Includes 1.03, 1.19, 1.90, 1.37
* **Predicted Virality Score:** 81 (Extremely Viral)

### **Mistral’s Recommendation**

“This track shows strong viral potential with high energy levels, making it a solid choice for high-intensity playlists like gym mixes. The tempo and overall audio features point to a track that’s both dynamic and engaging. That said, listener preference always matters—previewing the song is still recommended.”

### **Final Thoughts:**This assistant takes technical audio data and transforms it into practical insights. It’s especially useful for those in music, marketing, or content curation who want to make fast, data-backed decisions.