**Documentation for Music Recommendation System Code**

**Overview**

This project processes and analyzes a dataset of Spotify tracks to explore correlations, visualize patterns, and build a predictive model to recommend music tracks based on similarity. The dataset contains various audio features, metadata, and popularity metrics.

**Key Libraries Used**

* **NumPy**: For numerical operations and array handling.
* **Pandas**: For data manipulation and analysis.
* **Matplotlib** and **Seaborn**: For data visualization.
* **Scikit-learn**: For machine learning tasks like model training and evaluation.

**Detailed Description**

**1. Setup and Import Libraries**

The script imports necessary libraries for data processing, visualization, and machine learning. It also ensures compatibility with the Kaggle notebook environment by setting paths for dataset access.

**2. Dataset Loading and Exploration**

* **Dataset Path**: The dataset is loaded from the Kaggle directory.
* **Preview Data**: data.head() displays the first five rows to understand its structure.
* **Columns**: Key columns include track\_name, artist\_name, popularity, danceability, energy, and language.
* **Shape**: The dataset contains 62,317 rows and 22 columns.
* **Missing Values**: Checked using data.isna().sum(). No missing values were found, ensuring data completeness.

**3. Data Cleaning**

* Dropped rows with missing values using data.dropna(). Although unnecessary here (as there are no missing values), it ensures robustness for future datasets.

**4. Visualization**

* **Popularity Distribution**:
  + Used sns.displot(data['popularity']) to plot a histogram showing the distribution of track popularity.
* **Year Distribution**:
  + Used sns.displot(data['year']) to show the number of tracks released over the years.
* **Popularity by Language**:
  + Boxplots were created using sns.boxplot() to compare track popularity and danceability across different languages.
* **Track Duration vs. Energy**:
  + Scatterplot showing the relationship between track duration and energy.
* **Track Count by Language**:
  + Countplot visualizing the number of tracks per language.

**5. Correlation Analysis**

* Selected numeric columns for correlation analysis using data.select\_dtypes(include=[np.number]).
* Computed correlations with popularity and plotted a bar chart showing how audio features (e.g., acousticness, danceability) correlate with popularity.

**6. Machine Learning Model**

**6.1 Feature Selection**

Selected relevant features for training the model:

* **Independent Variables (X)**:
  + acousticness, danceability, duration\_ms, energy, instrumentalness, liveness, speechiness, tempo, valence.
* **Target Variable (y)**:
  + popularity.

**6.2 Data Splitting**

Split the dataset into training and test sets using train\_test\_split():

* **Training Set**: 80% of the data.
* **Test Set**: 20% of the data.

**6.3 Model Training**

* Trained a **Random Forest Regressor** to predict the popularity of a track.
* Model parameters:
  + n\_estimators=100
  + random\_state=42

**6.4 Evaluation**

* Evaluated the model using mean\_squared\_error() on the test set (MSE metric calculation not shown explicitly in the snippet).

**Key Visualizations**

1. **Popularity Distribution**: Histogram showing how popularity is distributed across tracks.
2. **Popularity by Language**: Boxplot comparing track popularity across languages.
3. **Danceability by Language**: Boxplot comparing danceability scores for different languages.
4. **Track Duration vs. Energy**: Scatterplot showing the relationship between track duration and energy levels.
5. **Number of Tracks by Language**: Bar chart visualizing the number of tracks for each language.
6. **Feature Correlation with Popularity**: Bar chart showing correlations between audio features and track popularity.

**Improvements and Notes**

1. **FutureWarning**: Adjust the palette parameter in Seaborn plots to avoid deprecation warnings in future versions.
2. **Model Evaluation**: Include evaluation metrics like R-squared, Mean Absolute Error (MAE), or MSE for better model performance analysis.
3. **Feature Engineering**: Additional preprocessing (e.g., normalization or scaling) might improve model accuracy.

**How to Run**

1. Clone the repository and ensure all dependencies are installed.
2. Place the dataset in the appropriate directory.
3. Execute the notebook in Jupyter, Colab, or Kaggle.
4. Review visualizations and model predictions.

**Folder Structure**

* README.md: Project documentation.
* data/: Contains the dataset (spotify\_tracks.csv).
* notebooks/: Jupyter or Colab notebook files.
* src/: Scripts for data preprocessing and model training.

**Contact**

For queries or feedback, please contact:

* **Name**: Ameya Kalbande
* **Email**: thisisameya147@gmail.com