Project Title: Predicting Music Track Popularity

Short Description: This project guides students to predict the popularity of music tracks using Python, focusing on data analysis and simple machine learning techniques to estimate popularity scores.

Difficulty Level: Beginner

AI-ML Concepts Used: Linear Regression, Data Preprocessing

Learning Outcome: Data analysis, prediction, coding skills

Programming Skills Mastered: Pandas, Scikit-Learn

Introduction

Predicting the popularity of music tracks is valuable for artists, producers, and streaming platforms to understand audience preferences and enhance user engagement. In this project, we will use Python to predict music track popularity based on various audio features. This project is tailored for Class 9 and 10 CBSE/ICSE students to grasp the basics of data science and machine learning.

Dataset Description

We will use a dataset comprising 227 music tracks, each described by their audio features and metadata. The key features in the dataset include:

- **Track Name:** Name of the music track
- **Artists:** Name(s) of the artist(s)
- **Album Name:** Name of the album
- **Popularity:** Popularity score of the track
- **Duration (ms):** Duration of the track in milliseconds
- **Explicit:** Whether the track has explicit content
- **Danceability:** How suitable the track is for dancing
- **Energy:** Intensity and activity level of the track
- **Key:** Musical key of the track
- **Loudness:** Overall loudness of the track
- **Mode:** Modality of the track (major or minor)
- **Speechiness:** Presence of spoken words in the track
- **Acousticness:** Likelihood of the track being acoustic
- **Instrumentalness:** Probability of the track being instrumental
- **Liveness:** Presence of a live audience in the recording
- **Valence:** Musical positiveness conveyed by the track
- **Tempo:** Tempo of the track in beats per minute

Implementation Steps

1. **Import Libraries and Load Dataset**

   We will start by importing the necessary Python libraries and loading the dataset.

   ```python
   python
   CopyEdit
   import pandas as pd

   # Load the dataset
   spotify_data = pd.read_csv("Spotify_data.csv")

   # Display the first few rows of the dataset
   print(spotify_data.head())
   ```

2. **Data Preprocessing**

   We need to preprocess the data by handling missing values, encoding categorical variables, and selecting relevant features.

   ```python
   python
   CopyEdit
   # Drop unnecessary columns
   spotify_data.drop(columns=['Unnamed: 0'], inplace=True)

   # Check for missing values
   print(spotify_data.isnull().sum())

   # Drop rows with missing values
   spotify_data.dropna(inplace=True)

   # Convert 'Explicit' column to numerical (True/False to 1/0)
   spotify_data['Explicit'] = spotify_data['Explicit'].astype(int)

   # Select relevant features for prediction
   features = ['Duration (ms)', 'Explicit', 'Danceability', 'Energy',
   'Key',
               'Loudness', 'Mode', 'Speechiness', 'Acousticness',
               'Instrumentalness', 'Liveness', 'Valence', 'Tempo']
   X = spotify_data[features]
   y = spotify_data['Popularity']
   ```

3. **Split the Data**

   We will split the data into training and testing sets to evaluate our model's performance.

   ```python
   python
   CopyEdit
   from sklearn.model_selection import train_test_split

   # Split the data into training and testing sets
   X_train, X_test, y_train, y_test = train_test_split(X, y,
   test_size=0.2, random_state=42)
   ```

4. **Train the Model**

   We will use Linear Regression to train our model on the training data.

   ```python
   CopyEdit
   from sklearn.linear_model import LinearRegression

   # Create and train the model
   model = LinearRegression()
   model.fit(X_train, y_train)
   ```

5. **Make Predictions**

   We will use the trained model to make predictions on the test data.

   ```python
   CopyEdit
   # Make predictions on the test data
   y_pred = model.predict(X_test)
   ```

6. **Evaluate the Model**

   We will evaluate the model's performance using the Mean Squared Error (MSE) metric.

   ```python
   CopyEdit
   from sklearn.metrics import mean_squared_error

   # Calculate Mean Squared Error
   mse = mean_squared_error(y_test, y_pred)
   print(f"Mean Squared Error: {mse}")
   ```

7. **Visualize the Results**

   We will visualize the actual vs. predicted popularity scores to understand the model's performance.

   ```python
   CopyEdit
   import matplotlib.pyplot as plt

   # Scatter plot of actual vs. predicted popularity
   plt.scatter(y_test, y_pred)
   plt.xlabel('Actual Popularity')
   plt.ylabel('Predicted Popularity')
   plt.title('Actual vs. Predicted Popularity')
   plt.show()
   ```

Conclusion

In this project, we learned how to predict the popularity of music tracks using Python. We explored data preprocessing techniques, feature selection, and implemented a simple Linear Regression model to make predictions. This project provided insights into the practical applications of machine learning in the music industry, enhancing our data analysis and coding skills.

Disclaimer

The projects provided on this website are for educational purposes only. They are designed to help students understand fundamental concepts in AI, machine learning, and programming. While we strive for accuracy and quality, we do not guarantee the correctness or reliability of the projects for real-world applications.

Users are encouraged to modify and test the projects as per their requirements. We are not responsible for any issues, data loss, or unintended consequences resulting from the use of these projects.

For any queries or further assistance, please get in touch via email at contact@aimltechacademy.org.