Spotify Songs ML Project

Overview

This document summarizes the data preprocessing steps applied to the Spotify songs dataset from Kaggle (30000 Spotify Songs). The project's goal is to predict the popularity of Spotify songs based on their audio features, release information, and playlist context.

Research Question:
Can we predict the popularity of a Spotify song using its audio characteristics, release details, and associated playlist/genre information?

Project Goal:
Develop a predictive model to identify the key factors that influence track popularity and accurately estimate a song's popularity score.

1. Data Preprocessing step

1.1 Data Loading

- **Dataset Overview:**
  The raw dataset comprises approximately 32,833 rows and 23+ columns, including unique identifiers (e.g., `track_id`), descriptive fields (`track_name`, `track_artist`), audio features (e.g., `danceability`, `energy`, `loudness`), temporal data (`track_album_release_date`), and playlist information. The target variable (`track_popularity`) is numeric and ranges from 0 to 100.

1.3 Date Conversion and Feature Extraction

- **Date Conversion:**
  Converted the `track_album_release_date` column from an object to a datetime type using `pd.to_datetime(errors='coerce')`.
- **Feature Extraction:**
  - Extracted `release_year` and `release_month` from the date.
  - Created a derived feature, `song_age`.

1.3 Text Normalization and Type Conversion

- **Normalization:**
  Stripped extra whitespace and converted text in columns such as `track_name`, `track_artist`, and `track_album_name` to lowercase.
- **Type Consistency:**
  Applied a helper function to convert all object-type columns to the Pandas `"string"` dtype for uniformity.

1.4 Duplicate Handling

- **By Track ID:**
  Aggregated duplicate rows based on `track_id` using a custom function that returns the mode (most frequent value) for columns like `track_popularity` and `playlist_genre`, while retaining the first occurrence for other columns.
- **By Song Attributes:**
  Grouped by `track_name`, `track_artist`, and `duration_ms` to identify duplicate song entries, and kept only the earliest version based on the smallest `track_album_release_year`.
- **By Tempo Rounding:**
  Created a new column, `tempo_first3` (the integer part of `tempo`), then grouped by `track_name`, `track_artist`, and `tempo_first3` to select the earliest version based on both `track_album_release_year` and `track_album_release_month`.

1.5 Filtering Out Low-Quality Data

- **Tempo Filter:**
  Removed songs with a tempo below 20 BPM.
- **Duration Filter:**
  Excluded songs with durations shorter than 30,000 milliseconds (30 seconds).

1.6. Challenges in Preprocessing

- **Inconsistent Date Formats:**
  Required careful conversion and handling using error coercion.
- **Duplicate Records:**
  Multiple layers of duplicates necessitated a multi-step aggregation strategy.
- **Text Normalization:**
  Variations in casing and whitespace required standardized cleaning.
- **Outlier Filtering:**
  Extreme values in numerical features (e.g., tempo, duration) had to be addressed to ensure quality.

Summary

After preprocessing, the dataset has been:

- **Cleaned and Standardized:** Date fields converted, key features extracted, and text normalized.
- **Consolidated:** Duplicates were handled using a multi-step approach to retain the most representative version of each song.
- **Filtered:** Removed low-quality records (e.g., songs with very low tempo or short duration).

Chapter 2: Exploratory Data Analysis (EDA)

In this chapter, we explore the preprocessed Spotify songs dataset to understand its underlying patterns, relationships, and potential issues. The insights derived here will guide our feature selection and inform our modeling strategy.

2.1 Data Overview and Protocol

- **Data Loading and Overview:**
  The dataset (~32,833 rows, 23+ columns) was loaded from a pickle file and contains unique identifiers (e.g., `track_id`), descriptive fields (`track_name`, `track_artist`, `track_album_name`), audio features (e.g., `danceability`, `energy`, `loudness`, `tempo`), temporal data (`track_album_release_date`), and playlist information.
- **Data Protocol:**
  Data types, extreme values, missing counts, and unique counts were exported to Excel for reference. AutoViz was also used to generate initial visual reports.

2.2 Descriptive Statistics and Target Variable Analysis

- **Summary Statistics:**
  Using `df_spotify.describe()`, we reviewed the central tendencies, dispersion, and potential outliers in numeric features.
- **Target Variable – Track Popularity:**
  - A histogram with a density curve was plotted for **track_popularity** (range 0–100) to assess its distribution.
  - A box plot was used to detect outliers in track popularity.
- **Temporal Trends:**
  A `decade` column was derived from `track_album_release_year` and visualized using a count plot to examine the temporal distribution of songs.

2.3 Univariate Analysis of Key Audio Features

- **Distribution Plots:**
  Histograms were generated for key audio features (e.g., **danceability**, **energy**, **loudness**, **tempo**, **duration_ms**, **song_age**) to understand their distributions, central tendencies, and potential outliers.
- **Outlier Detection:**
  Box plots were produced for numeric features (limited to numeric columns) to further inspect the presence of outliers.

2.4 Bivariate Analysis: Relationships with Track Popularity

- **Scatter Plots:**
  Scatter plots were used to examine relationships between **track_popularity** and key audio features. These plots revealed trends such as:
  - Higher danceability tends to be associated with higher popularity.
  - Extremely high energy levels might correlate with lower popularity.
- **Correlation Analysis:**
  A correlation matrix was computed and visualized via a heatmap, highlighting relationships such as a strong positive correlation (0.68) between **loudness** and **energy**, and a moderate negative correlation (-0.54) between **energy** and **acousticness**.

2.5 Categorical Analysis

- **Genre and Subgenre Distributions:**
  Count plots were created for `playlist_genre` and `playlist_subgenre` to assess the distribution of songs across genres.
- **Contribution Analysis:**
  A pie chart was plotted to illustrate the contribution of each genre to total track popularity.
- **Top 20 Analysis:**
  The top 20 songs by popularity were identified, along with the top 20 artists (filtered by minimum song count) and top 20 albums (by average popularity), to pinpoint the most influential entities in the dataset.

2.6 Textual Analysis: Word Cloud

- **Word Cloud of Artist Names:**
  All artist names were combined into a single string and visualized using a word cloud.
  Rationale:
  The word cloud provides a visual summary of the most frequently occurring artist names in the dataset. This analysis helps highlight which artists appear most often, offering insights into the overall composition of the data and potential influences on track popularity.

2.7 Statistical Testing

To statistically validate our observations, several tests were conducted:

- **T-Test for Genre (EDM vs. Non-EDM):**
  Dummy encoding of `playlist_genre` (e.g., creating a binary `genre_edm` variable) allowed comparison of mean popularity between EDM and non-EDM songs.
  Result: A significantly lower average popularity for EDM songs (t-statistic: -25.742, p-value: 0.000).
- **T-Test for Energy:**
  Splitting songs at the median energy value, we compared high-energy versus low-energy songs.
  Result: High-energy songs had significantly lower average popularity (t-statistic: -15.154, p-value: 0.000).

- **T-Test for Danceability:**
  Splitting by median danceability, we found that high danceability is associated with significantly higher popularity (t-statistic: 6.841, p-value: 0.000).

2.8 Skewness Analysis

- **Skewness Metrics:**
  The skewness of numeric features was computed and visually highlighted (values >1 or < -1) using a custom function.
  Note: Although some features exhibit skew, many modern models (especially tree-based methods) are robust to these distributional issues. The skewness analysis informs potential feature transformation if needed later.

2.9 Summary and Implications for Modeling

- **Key Insights:**
  The EDA revealed significant relationships between track popularity and both audio features (e.g., energy, danceability, loudness) and categorical/temporal variables (e.g., genre, decade). Statistical tests confirm these associations, providing a strong foundation for predictive modeling.
- **Modeling Guidance:**
  Given that **track_popularity** is a continuous variable (0–100), our subsequent modeling phase will employ regression techniques (e.g., Random Forest, XGBoost). The insights from EDA suggest that features like energy, danceability, and genre are valuable predictors.

This comprehensive analysis of the dataset through various visualizations and statistical tests ensures that we have a robust understanding of the data, which is crucial for building an accurate predictive model of track popularity.

Chapter 3: Feature Selection

In this chapter, we detail the feature selection process applied to our preprocessed Spotify songs dataset. Our goal is to identify the most predictive features for estimating track popularity. We use a combination of regression-based models to assess the importance of each feature and then select those that are consistently identified as relevant.

3.1 Overview and Rationale

Our target variable, **track_popularity**, is continuous (ranging from 0 to 100), so we approach feature selection through regression models rather than classifiers. We aim to leverage the strengths of various methods to determine which features contribute significantly to predicting popularity. This multi-model approach helps mitigate biases that might be introduced by any single method.

3.2 Methodology

We use the following regression-based methods for feature selection:

- **Lasso Regression:** Employs L1 regularization, encouraging sparse coefficients by shrinking less important feature weights to zero.
- **Ridge Regression:** Uses L2 regularization to penalize large coefficients; while not inherently sparse, it helps indicate feature relevance.
- **LinearSVR:** A support vector regression model with a linear kernel, which provides coefficient estimates for each feature.
- **Gradient Boosting Regressor:** A tree-based ensemble that computes feature importances based on how frequently features are used for splitting.
- **Random Forest Regressor:** An ensemble of decision trees that measures feature importance by how much each feature reduces impurity.

Each model is fitted using the same predictor set **X** (which contains only numeric and boolean features) and target **y**. For each method, a feature is considered "selected" if its coefficient (or feature importance) is non-zero.

3.3 Feature Selection Process

1. **Data Preparation:**
   We define our predictors **X** by dropping non-numeric or non-boolean columns from the preprocessed DataFrame (**df_spotify_2**). The target variable **y** is **track_popularity**. Any missing values in **X** are filled with the column means.
2. **Model Fitting and Selection:**
   We fit the following models:
   - **Lasso Regression** (with α=0.01)
   - **Ridge Regression** (with α=0.01)
   - **LinearSVR** (with C=0.01, using dual=True to support the specified loss)
   - **Gradient Boosting Regressor**
   - **Random Forest Regressor**

   For each model, we create a binary indicator (1 if the feature is selected—that is, non-zero coefficient or importance—and 0 otherwise).

3. **Aggregation of Selection Results:**
   We compile the selection results from all five models into a summary DataFrame. We then sum the selections for each feature to determine how many models selected it.
4. **Final Feature Set:**
   We set a threshold of at least 4 out of 5 models. Only features meeting or exceeding this threshold are retained for the final modeling dataset. The final dataset is then created by combining these selected features with the target variable.

Below is the list of final features that were kept for modeling:

1. **danceability**
2. **energy**
3. **key**
4. **loudness**
5. **mode**
6. **speechiness**
7. **acousticness**
8. **instrumentalness**
9. **liveness**
10. **valence**
11. **tempo**
12. **duration_ms**
13. **track_album_release_year**
14. **track_album_release_month**
15. **song_age**
16. **decade**
17. **genre_edm**
18. **genre_latin**
19. **genre_pop**
20. **genre_r&b**
21. **genre_rap**
22. **genre_rock**
23. **track_popularity** *(target variable)*

3.4 Results and Discussion

The feature selection summary reveals that most features were selected by all five models—indicating that the audio features (such as **danceability**, **energy**, **loudness**, **tempo**, etc.), temporal features (e.g., **track_album_release_year**, **song_age**, **decade**), and encoded genre variables are all relevant predictors. One notable exception is **song_age**, which was selected by 4 out of 5 models. Setting our threshold to 4 allows us to retain **song_age**; however, if a stricter threshold is desired, this feature might be dropped due to potential redundancy with other temporal variables.

3.5 Implications for Modeling

Based on this multi-method feature selection process:

- **Robustness:** The consensus among multiple regression-based methods suggests that the selected features have strong predictive power for track popularity.
- **Modeling Strategy:** The final modeling dataset, **df_model_final**, now includes only the most relevant features alongside the target variable. This dataset will be used in the next phase for training regression models (e.g., Random Forest, XGBoost) to predict track popularity.

Chapter 4: Model Selection and Evaluation

After finalizing our feature set, we evaluated several regression models to predict track popularity. Since our target variable is continuous, our task is a regression problem. We split our dataset into training and test sets (80% training, 20% testing) to train and assess model performance.

- **Models Evaluated:**
  We experimented with a range of models, including:
  - **Linear Regression:** A baseline model for linear relationships.
  - **Decision Tree Regressor:** A non-linear model that partitions the feature space based on decision rules.
  - **Random Forest Regressor:** An ensemble of decision trees that reduces overfitting by averaging multiple trees.
  - **AdaBoost Regressor:** An ensemble method that combines weak learners to improve predictions.
  - **Gradient Boosting Regressor (GBM):** A powerful ensemble technique that sequentially builds trees to correct previous errors.
  - **Support Vector Regressor (SVR):** A model that finds the best hyperplane for regression in high-dimensional space.
  - **XGBoost Regressor:** An optimized gradient boosting model known for its efficiency and performance.
- **Evaluation Metrics:**
  For each model, we computed several regression metrics to quantify performance:
  - **Mean Squared Error (MSE)**
  - **Root Mean Squared Error (RMSE)**
  - **Mean Absolute Error (MAE)**
  - **R^2 Score**
  - **Root Mean Squared Logarithmic Error (RMSLE)**

  These metrics provide insights into the error magnitude, variability, and overall goodness of fit for each model.

- **Visual and Tabular Comparison:**
  Scatter plots of actual versus predicted values were generated for each model to visually assess performance. Additionally, all model metrics were aggregated into a summary table, which allows for straightforward comparison.
- **Insights and Next Steps:**
  The model evaluation phase identifies which regression algorithm performs best based on the error metrics and visual diagnostics. For example, if the Random Forest Regressor yields the lowest MAE and RMSE, it might be chosen for further tuning and deployment. The evaluation also informs whether additional feature engineering or hyperparameter optimization is required.

This comprehensive evaluation ensures that our final model is built on a robust set of predictive features, and that the chosen algorithm is well-suited to the task of predicting track popularity.