CS M148 Final Project

Spotify Prediction Model

Rohan Kumar, Kriteen Jain, Ishan Garg, & Christian Solano

December 13th, 2024

Main Document

Data Set

The dataset used for this project is the Spotify Tracks Dataset, available on Hugging Face. It comprises features such as ‘artists,’ ‘popularity,’ ‘duration_ms,’ ‘danceability,’ ‘energy,’ ‘loudness,’ ‘speechiness,’ ‘acousticness,’ ‘instrumentalness,’ ‘liveness,’ ‘valence,’ ‘tempo,’ and ‘track_genre,’ among some others. To prevent the model from overfitting, we decided to only focus on key attributes such as genre, danceability, energy, and explicitness. These features are critical for building a good recommendation system and performing genre classification. The dataset contains over 114,000 rows and 21 columns, representing a wide variety of tracks across a total of 114 genres. Overall, the data set provided us with a lot of data and room to explore too.

Problem Statement

The goal of this project is to develop a recommendation system that suggests songs tailored to user preferences by analyzing key musical features such as danceability, energy, and genre. By leveraging data from the Hugging Face dataset, the system will predict songs that align with individual tastes, creating a personalized listening suggestion similar to the actual one by Spotify.

Key Methodology

Our methodology involved several key steps. First, during preprocessing, we filtered out niche genres with low representation, reducing the number of genres from 114 to 57 to ensure computational efficiency and better generalization to new data. All numerical features were standardized using StandardScaler to facilitate clustering and improve model performance. The dataset was then split into training, validation, and testing sets to ensure a proportional representation of genres across all of the relevant splits. This allowed for an organizational split.

The Random Forest classifier was the most effective methodology for this specific music recommendation system. This approach worked well because it combines multiple decision trees (100 in this case) to make predictions, which is particularly suited to musical data's complex nature. Random Forests excel at capturing non-linear relationships between features such as energy, valence, and danceability, which simpler models such as linear regression would have struggled with. Additionally, Random Forests are less prone to any overfitting compared to individual decision trees, which was crucial given the diverse nature of musical characteristics in this data set. We came to this conclusion after a lot of testing and research with various methods.

Results and Evaluation

The Random Forest model achieved the best results with a calculated training accuracy of 99.99% and a validation accuracy of 80.8%. It also had high precision (83%) and recall (90%)

scores. Cross-validation was also performed using 5-fold validation, with an AUC score of 0.748, indicating consistent and good performance across all of the different subsets of the data. The model's evaluation metrics showed strong performance in identifying danceable songs (90% recall) while maintaining good precision in its generalizable predictions. In addition, the significant gap between the training and validation accuracy (99.9% vs 80.8%) suggests some model overfitting, however, this was still the best model when performed on the validation data.

Code Implementation

To implement this project's code on the Spotify dataset, you would start by installing the required libraries (scikit-learn, pandas, numpy) and loading the dataset using the Hugging Face datasets library. You can then run the main Jupyter Notebook to execute all of the included preprocessing, training, and evaluation pipelines. This will provide you with the same methods that we utilized.

Appendix

Exploratory Data Analysis

Our initial exploration of the Spotify dataset involved comprehensive visualization techniques to understand the underlying data structure present. We created histograms and box plots for key musical features including danceability, energy, and tempo, which revealed their distribution patterns. A correlation heatmap highlighted significant relationships between features, notably a strong correlation (0.74) between energy and loudness. Additionally, for data partitioning, we implemented a genre-stratified split: 60% training (34,199 samples), 20% validation (11,400 samples), and 20% testing (11,400 samples) ensuring a balanced genre representation in the sets.

Data Preprocessing

The specific preprocessing phase focused on data quality and standardization. We identified and addressed missing values in the artists' column and applied StandardScaler to normalize numerical features. A key decision was made to reduce computational complexity by focusing on the 57 most popular genres. This method maintained the dataset integrity with room for analysis.

Linear Regression

Our initial regression analysis employed linear regression to predict danceability using energy and valence as predictors. The model achieved an R^2 value of 0.191 on both training and validation sets, indicating a limited linear relationship between these features. We implemented Ridge regression with alpha = 1.0 to address potential overfitting, but similar results suggested the relationships in our data were non-linear. This suggests that regularization was not necessarily needed as it would have improved the results significantly enough for us to use Linear Regression. That is to say, while we performed linear regression, it didn’t result in much.

Logistic Regression

We transformed the danceability prediction into a binary classification problem using a 0.5 threshold. The logistic regression model achieved 70.3% accuracy with an AUC score of 0.72. These results still indicated that we needed more complex models to more accurately find a relationship between all the features. Regularization once again was not necessarily needed as it would have improved the results significantly enough for us to use logistic regression as before.

Classification Models

Our implementation compared K-Nearest Neighbors (k = 5) and Random Forest (100 trees), with Random Forest being the calculated better model. The KNN classifier achieved a training accuracy of 81.8% and a validation accuracy of 73.4%. The Random Forest model significantly outperformed KNN with a training accuracy of 99.9% and a validation accuracy of 80.8%. The model performed well due to its ensemble nature and ability to capture complex, non-linear relationships in the musical features through multiple decision trees. The significant gap between training and validation accuracy (99.9% vs 80.8%) suggests some overfitting however this was still the best model when performed on the validation data. Although, there is room to improve.

PCA and Clustering

Principal Component Analysis (PCA) and clustering were critical steps in uncovering latent structures within the dataset. PCA was employed to reduce the dimensionality of the dataset while preserving its most informative variance. By transforming the data into a two-dimensional space, PCA enabled us to visualize the clustering of genres based on acoustic features such as danceability and energy. Through PCA, we observed that the first two principal components explained approximately 63% of the variance in the dataset. This indicated that most of the variability in the data could be captured in a reduced-dimensional space, allowing for more efficient computation and analysis. In the specific context of the core dataset, this was sufficient.

For clustering, we experimented with KMeans and Agglomerative Clustering. KMeans (k = 5) was selected based on the elbow method and relevant silhouette analysis. The silhouette score of 0.62 suggested moderate separability among the clusters, with distinct groupings observed in the PCA-transformed space. Agglomerative Clustering yielded similar results but was more computationally intensive. That is to say, the clustering revealed genre groupings with similar acoustic profiles. For instance, genres with high danceability and energy formed distinct clusters, highlighting the utility of these features in defining musical styles. However, some overlap between clusters indicated potential noise or the need for additional features to improve separability. While PCA and clustering provided valuable insights into the dataset's structure, they were not directly used in the final prediction model due to the loss of feature interpretability and the modest clustering performance. These methods, however, contributed to feature selection and informed the choice of subsequent classification techniques, so they still yielded core results.

Neural Networks

A simple feedforward neural network, implemented in PyTorch, was specifically tested for genre classification. Despite extensive hyperparameter tuning (learning rates: 0.001 – 0.01, epochs: 10 – 50), the network plateaued at 72% validation accuracy, underperforming compared to the Random Forest. This underscores the dataset's limitations for deep learning due to its size and feature diversity. In this way, the neural network still provided relevant information for this data.

Hyperparameter Tuning

Hyperparameter tuning was an integral step in optimizing the performance of our models and ensuring their robustness. For the Random Forest classifier, we employed grid search to identify the optimal values for all of the key parameters, including max_depth, n_estimators, and min_samples_split. The grid search revealed that a max_depth of 15 and n_estimators set to 100 provided the best trade-off between the mode’s complexity and generalization performance. These values ensured the model captured non-linear relationships effectively without overfitting to the training data. Similarly, we experimented with varying `min_samples_split` to control tree growth and reduce overfitting, specifically finding a value of 4 to yield consistent results across cross-validation folds. Based on this, we were able to build on these results and build our model.

For the neural network, hyperparameter tuning was conducted iteratively, focusing on learning rates, the number of hidden layers, and the size of each layer. Learning rates in the range of 0.001 to 0.01 were tested, with a rate of 0.005 achieving the best convergence speed and stability during the training. The network architecture was also tuned, with three hidden layers and 128 neurons per layer emerging as the most effective configuration for capturing the complexities of the dataset. Additionally, the number of training epochs varied from 10 to 50, with 30 epochs providing a balance between computational efficiency and performance. These tuning efforts, though less impactful for the neural network compared to the Random Forest, demonstrated the importance of systematically exploring all the parameter spaces to enhance model performances.