Spotify Data Analysis and Recommender System

Submitted to:

Prof. Roselyne Tchoua

School of Computing and Digital Media

DePaul University

Report By:

Ian Mulchrone

Miguel Saucedo

Shyamkumar Moradiya

Ava Roelle

June 11, 2025

Executive Summary

This project explores how audio features from over 114,000 Spotify tracks can be used to classify music characteristics, predict song popularity, and create personalized song recommendations. Through machine learning, our goal was to discover which features are most predictive of a song's overall success and to generate a working recommender system. The project consisted of four tasks: Data Preprocessing and Feature Engineering, Regression Analysis, Classification, and Recommender System.

Data Preprocessing and Feature Engineering – Shyamkumar Moradiya and Ian Mulchrone:

This includes cleaning the data, encoding categories, normalizing values, and creating new features. Important features like the Energy-to-Danceability Ratio, Loudness per Genre, and Explicit Content Impact will be added to improve predictions and recommendations.

Regression – Miguel Saucedo:

Use a linear/polynomial regression to identify R squared, MSE, and MAE to make predictions on song popularity and other song features. KNN for regression can also be useful to be able to recommend similar songs based on several song features. A decision tree can be made to predict the popularity of a song.

Classification – Ava Roelle:

Predict song characteristics such as genre or popularity based on audio features. Implement various classification algorithms such as logistic regression, KNN, and random forest in order to find the best approach.

Recommender System – Ian Mulchrone:

Given a list of recently listened to songs, use the songs' metadata to recommend similar songs. Use feature selection from regression and classification models and compare results. Implement cosine and Euclidean distance metrics and compare results.

Data Schema and Size

Objects: 114,000

Features: 'track_id', 'artists', 'album_name', 'track_name', 'popularity', 'duration_ms', 'explicit', 'danceability', 'energy', 'key', 'loudness', 'mode', 'speechiness', 'acousticness', 'instrumentalness', 'liveness', 'valence', 'tempo', 'time_signature', 'track_genre'

Plan for Evaluation

Regression: Identify the target variable to make these predictions. Evaluate regression metrics on the data set, split data to be able to train the model for predictions, and create visuals to get a better interpretation of the data itself and how the model is working

Classification: Measure the overall accuracy of the different classification models to assess their predictive power. Use precision and recall to evaluate performance for each class. Generate confusion matrices to identify common errors. Use k-fold cross validation to ensure the model generalizes well to new data.

Recommender System: Compare song recommendations based on feature selection and distance metric. Observe unique results and common themes among all criteria. Report which results are deemed best to fit the user's interest.

Conclusions

For both regression and classification, a random forest model was found to be the most effective model, in terms of accuracy, at predicting song popularity and classifying songs by genre. The important features from the random forest model were then used for the recommender system. This allowed for both better efficiency as a result of feature reduction, while also maintaining quality recommendations for the user.

Task 1: Data Preprocessing & Feature Engineering

The initial stage of this project involved preparing the Spotify dataset for downstream modeling tasks. This process included data cleaning, transformation of categorical values, creation of new features, and normalization. Preprocessing is a critical component in any data science pipeline, as it directly impacts model performance, interpretability, and reliability.

Feature Engineering

To enhance the dataset's predictive power, multiple derived features were introduced using existing audio metrics:

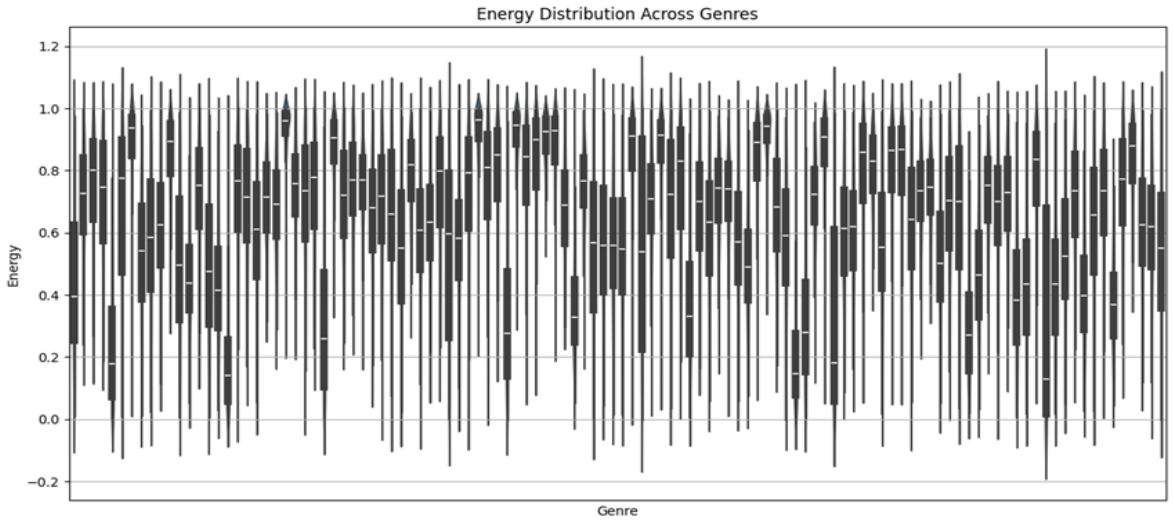
- Energy-to-Danceability Ratio: Captures the relationship between a track's energy level and its danceability, offering insight into musical dynamics.
- Loudness per Minute: Adjusts the loudness feature relative to song duration to normalize across varying track lengths.
- Tempo × Valence Product: Merges the song's tempo and mood, providing a proxy for emotional pacing.
- Acousticness-to-Energy Ratio: Differentiates between acoustic and high-energy tracks by comparing softness to intensity.
- Duration in Minutes: Converts duration from milliseconds to a more interpretable unit.

These engineered features offer more granular insight into the structure of each song and provide meaningful signals for regression, classification, and recommendation tasks.

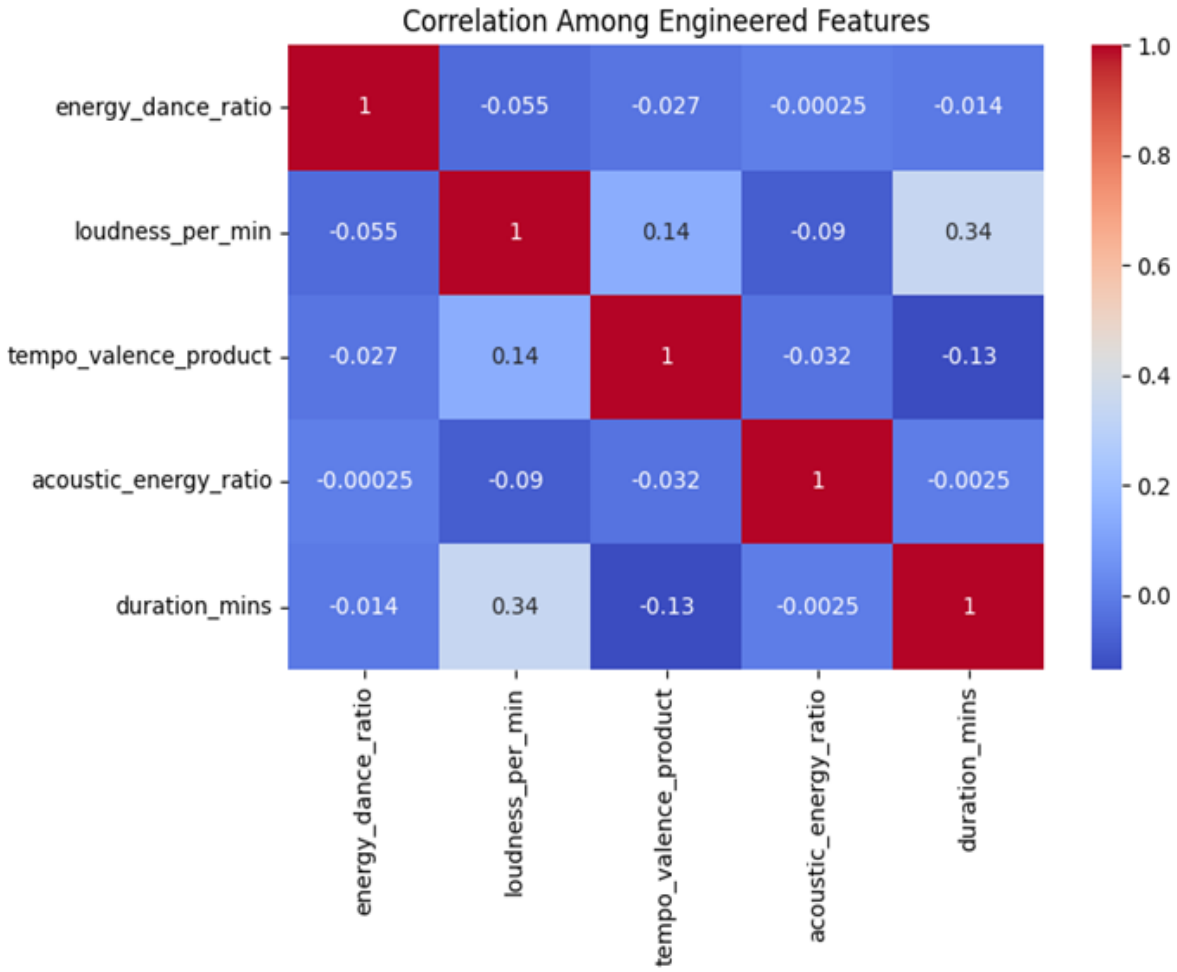
Exploratory Visualization

Two visualizations were generated to validate the feature engineering process:

The violin plot provides a comprehensive view of how the audio feature "energy" varies across different music genres in the dataset. Unlike a standard box plot, this visualization reveals both the distribution density and the spread of energy levels, highlighting nuances that are otherwise difficult to capture. For instance, genres like EDM and metal display consistently higher energy levels, reflecting their intense and upbeat nature, while acoustic and classical genres are characterized by lower energy values, aligning with their calmer profiles. These differences are particularly relevant when designing a content-based recommendation system, as understanding genre-level energy trends can help match users with music that aligns more closely with their listening preferences.



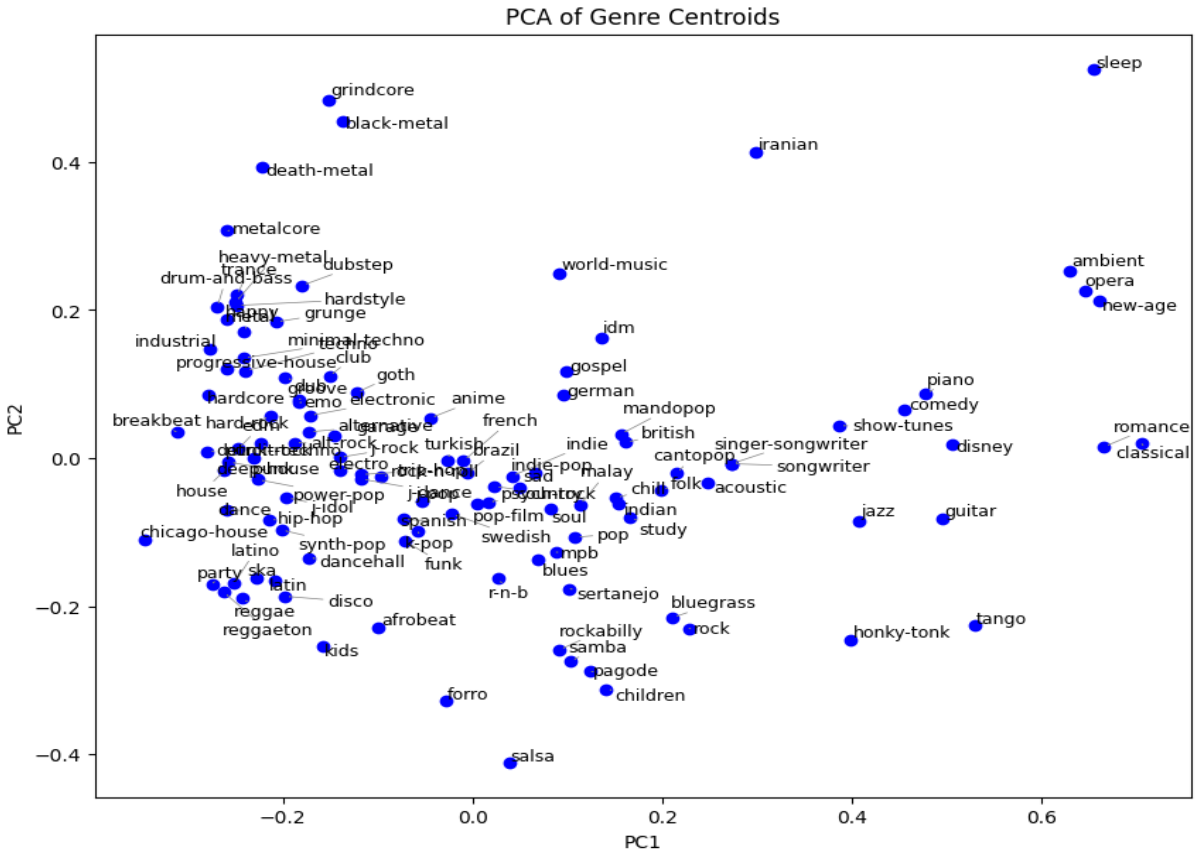
The heatmap of correlation coefficients among the engineered features allows us to analyze how these attributes relate to one another. This visualization played a key role in evaluating potential redundancies and dependencies between features. For example, the observed negative correlation between acoustic_energy_ratio and energy_dance_ratio indicates that songs with high acoustic characteristics tend to exhibit lower rhythmic intensity. Identifying such patterns is vital for refining feature selection in classification and regression models, ultimately improving predictive accuracy and model interpretability by minimizing the risk of multicollinearity.



Dummy Encoding and Principal Component Analysis

Creating dummy variables for categorical data is necessary for them to be included in each model and perform calculations. For this, there are two variables that need to be dummy encoded. The first is 'key', where the song's pitch corresponds to a given integer (0 = C, 1 = C♯/D♭, 2 = D). The second is 'time_signature' which indicates how many beats are in each measure (3 for ¾, 7 for 7/4).

The 'genre' category, on the other hand, has 125 different genres. Therefore, encoding genre using dummies would be computationally expensive and could create issues with model accuracy. Instead, a genre value can be calculated using the centroids of each genre and performing PCA on the centroids to get a representation of which genres are most similar to each other. This will be of use later when recommending similar songs which may not necessarily be within the same genre.



Plotting the PCA values for the genre centroids gives a better understanding of which genres are most similar to each other. The large group on the left consists of more loud, high-tempo genres, while the right side is more relaxing and less noisy. This will help organize each song so it is less likely the user will be recommended genres that are too dissimilar. For example, it is possible that a user who likes grindcore will also be interested in black-metal or death-metal music. This is reflected in those genre's PCA values being similar and may result in a song being recommended from any of those genres. The purpose of this is to create a sort of normalization for each song to stay closer to its genre. It is likely not going to be obvious from the data itself which genre a given song belongs to. This way, the genre PCA value can serve as a type of anchor so that a recommendation does not stray too far from its genre.

Task 2: Regression

Within the regression task, my team and I wanted to determine whether song features alone can help predict song popularity. However, we wanted to learn if genre and the artist themselves can play a role in predicting song popularity as well. The data was split into two

data sets; one with those two features (artists and track genre) and one with only song features. To change these two categorical features to numerical, two new features were added, which included the average popularity of each artist and average popularity of each genre.

This task was a bit tricky because we wanted to ensure we were not giving the model the answer when predicting popularity. To avoid this issue, we first split the data sets into training and testing sets before adding in the new features. This allowed for the testing set to be unseen during training. We then calculated averages for both artists and track genre using only the training set. After those averages were calculated, they were mapped onto the testing set. Rows with 'Unknown' artists were dropped from the data set.

We decided to train and tune four different regression task models: linear regression, decision tree, KNN, and random forest. I used GridSeachCV with 5-cross validation to find the best parameters for a given model. Although RandomForest did better in both cases, we wanted to look further into the KNN model. Each model was evaluated based on RMSE, r2, and CV score.

In both cases, RandomForest was the better model. However, RandomForest outperformed every other model in the data set that did not contain those two features.

```
RMSE (Linear Regression): 22.03413899451539
R² Score (Linear Regression): 0.02133928016594755
CV Score (Linear Regression): 22.026012876873672


RMSE (KNN): 19.690266591403628
R² Score (KNN): 0.21847445036158342
CV Score (KNN): 20.353820705094925


RMSE (Decision Tree): 20.489049220672577
R² Score (Decision Tree): 0.15377938779415445
CV Score (Decision Tree): 21.868779124397093


RMSE (Random Forest): 14.8654476678133
R² Score (Random Forest): 0.5545530671935754
CV Score (Random Forest): 15.689183246290474
```

In the figure above, RMSE, r2, and CV score are shown for the data set that did not contain the features of artists and track genre. RandomForest had the lowest RMSE value, which indicated the lowest average prediction error. It had the highest r2 value, explaining 55% of variance in popularity and had the lowest CV score.

```
RMSE: (Linear Regression) 14.142367571691313
R² Score: (Linear Regression) 0.6207920139544598
CV Score (Linear Regression): 11.36400641518382


RMSE (KNN): 14.33458652697102
R² Score (KNN): 0.6104137913626342
CV Score (KNN): 13.024930215013459


RMSE (Decision Tree): 15.177169028984327
R² Score (Decision Tree): 0.5632682405222852
CV Score (Decision Tree): 14.112052097021316


RMSE (Random Forest): 12.302627346318712
R² Score (Random Forest): 0.7130349728681288
CV Score (Random Forest): 10.607649374324591
```

In the figure above, RandomForest was the best model for data containing both artists and track genres. This model had the lowest RMSE value, the highest r2 value, and the lowest CV score. The results show the importance of artists and track genre when predicting song popularity.

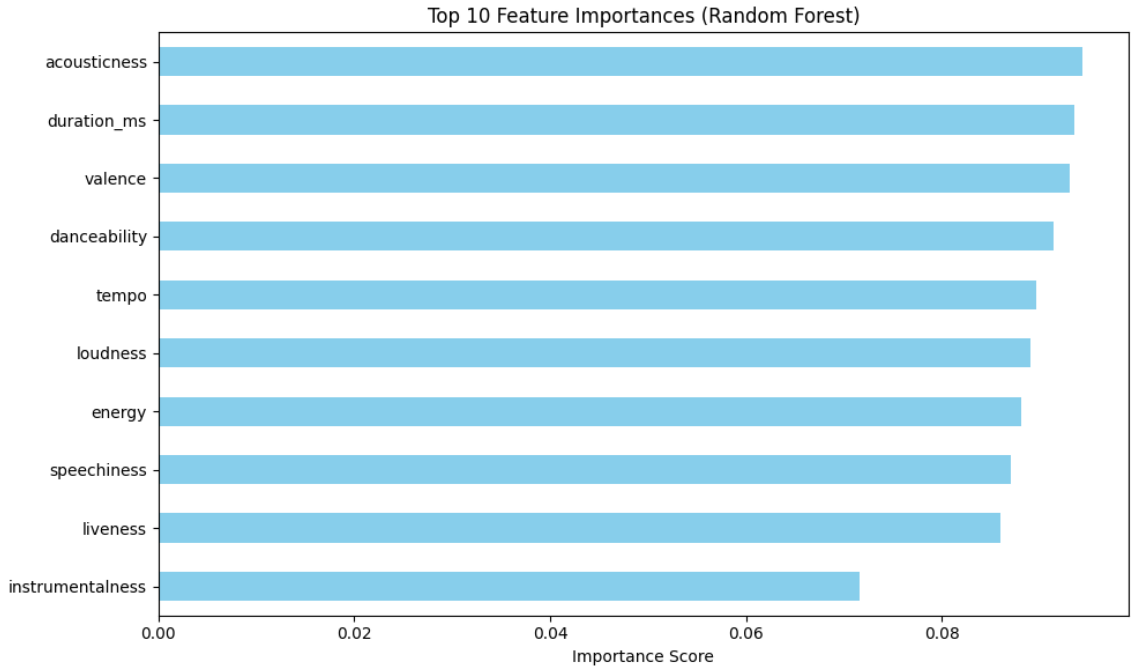
Task 3: Classification

To explore the classification task, we wanted to predict a song's popularity category (Low, Medium, or High) using Spotify's audio features. After the original popularity scores were scaled, we binned them into three categories using quantile cutoffs (Low: 0 to 0.022, Medium: 0.022 to 0.044, High: 0.044 and above). We included 12 numeric features (acousticness, loudness, energy, etc.) and one-hot encoded versions of key and time signature for the input features. The data was split 80/20 using stratified sampling to ensure that each of the three classes were proportionally represented in both the train and test sets.

We trained and tuned three different classification models (KNN, Random Forest, and Logistic Regression) using GridSearchCV with 5-fold cross validation. Each model was then evaluated on accuracy, precision, recall, and F1 score on the test set. Although we evaluated multiple metrics, we used accuracy as our main comparison metric since the popularity categories were fairly balanced. The macro and weighted F1 scores almost perfectly aligned with accuracy, which confirmed that the model performed the same across the different classes.

Model	Accuracy	Best Parameters
KNN	0.55	n_neighbors=3
Random Forest	0.68	n_estimators=200, max_depth=None min_samples_split=2
Logistic Regression	0.42	C=10 penalty='l2'

Random Forest outperformed the other two models on every metric. This model also provided some insight into feature importance. The most important features in predicting a song's popularity are acousticness, duration, valence, and danceability.



Task 4: Recommender System

The recommender system is a function that takes a list of songs the user has listened to and returns a playlist of the most similar songs. The number of songs in the new playlist can be specified, as well as the similarity measure used to calculate and whether or not the genres should match the genres of songs the user has listened to.

Artist	Song	Genre
Diana Ross	I'm Coming Out	disco
Metallica	Nothing Else Matters	hard-rock
Ludwig van Beethoven;Paul Lewi	Sonata No. 14 "Moonlight" in C-Sharp Minor",	classical
Dua Lipa	Levitating	pop
Nature Sounds	Rainy Relaxation	sleep

With these five songs recently listened to by the user, what would be recommended in a new playlist? With a wide range of genres and styles of music in the recently listened list, it is important that each song be represented in the new playlist. This can be done by finding the most similar song to each one the user has listened to and creating a new playlist of songs with the highest similarity score. The result, using cosine similarity is:

Artist	Song	Genre
Barry White	Can't Get Enough Of Your Love, Babe	disco
Jimmy Eat World	Hear You Me	emo
Ludwig van Beethoven;Murray Pe	Piano Sonata No. 14 in C-Sharp Minor, Op. 27	classical
Dua Lipa;DaBaby	Levitating (feat. DaBaby)	pop
Binaural Beats Brainwave Entra	Rain Drops	sleep

Playlist Score: 0.9974418950399999

Notice that an emo song by Jimmy Eat World was recommended as most similar to the hard-rock anthem by Metallica. This is aided by the PCA values of emo and hard-rock being similar to each other and may be an acceptable result for the user, even if it means going outside of the hard-rock genre. For every other song, the recommendation is within the same genre and the average cosine similarity for each song is seen in the playlist score. The high score is likely aided by the recommendation for 'Levitating' being the same song re-released with a new featuring artist, as well as the rain songs for sleep not likely to have much variation between the two aside from the frequency and intensity of droplets.

However, we can choose to limit the songs used for comparison by only selecting songs in the genres that the user has listened to. The result, using cosine similarity is:

Artist	Song	Genre
Barry White	Can't Get Enough Of Your Love, Babe	disco
Planet Hemp	Mary Jane	hard-rock
Ludwig van Beethoven;Murray Pe	Piano Sonata No. 14 in C-Sharp Minor, Op. 27	classical
Dua Lipa;DaBaby	Levitating (feat. DaBaby)	pop
Binaural Beats Brainwave Entra	Rain Drops	sleep

Playlist Score: 0.99554917688

By limiting recommendations to only similar genres, the recommendation for the Metallica song is changed to another hard-rock song. This change is also reflected in the playlist score, which decreased slightly from 0.9974 to 0.9955. This also helps with efficiency, as the number of songs needed to calculate similarity is reduced.

For any playlist size greater than the user's listening playlist, the first songs in the playlist will be the same, up until the number of songs in the recommended playlist exceeds the length of the user's playlist. After that, the songs added to the playlist will be the ones with the highest similarity score to any song the user has listened to. For recommending 10 songs, using cosine similarity without matching genres is:

Artist	Song	Genre
Barry White	Can't Get Enough Of Your Love, Babe	disco
Jimmy Eat World	Hear You Me	emo
Ludwig van Beethoven;Murray Pe	Piano Sonata No. 14 in C-Sharp Minor, Op. 27	classical
Dua Lipa;DaBaby	Levitating (feat. DaBaby)	pop
Binaural Beats Brainwave Entra	Rain Drops	sleep
Vishal-Shekhar;Arijit Singh;Sh	Ghungroo (From "War")	pop
Pritam;Arijit Singh;Sunidhi Ch	Dilliwaali Girlfriend	indian
Pitty	Te Conecta	mpb
Don Omar	Pobre Diabla	latin
Salim-Sulaiman;Sunidhi Chauhan	Halkat Jawani	indian

Playlist Score: 0.9980140266399999

The first 5 songs remain unchanged, and the rest of the playlist is filled with the songs with the highest similarity scores. This is also reflected in the playlist score as more songs with

extremely high similarity are included in the playlist. Problems start to arise with straying too far from the user's genre preferences, however, where there are two Indian songs and a Latin song in the recommendations. Different similarity metrics can be tested to see if this problem improves. Using Euclidean similarity, the result is:

Artist	Song	Genre
Barry White	Can't Get Enough Of Your Love, Babe	disco
Jimmy Eat World	Hear You Me	emo
Ludwig van Beethoven;Murray Pe	Piano Sonata No. 14 in C-Sharp Minor, Op. 27	classical
Dua Lipa;DaBaby	Levitating (feat. DaBaby)	pop
Binaural Beats Brainwave Entra	Rain Drops	sleep
Pritam;Arijit Singh;Sunidhi Ch	Dilliwaali Girlfriend	indian
Vishal-Shekhar;Arijit Singh;Sh	Ghungroo (From "War")	pop
Don Omar	Pobre Diabla	latin
Pitty	Te Conecta	mpb
Salim-Sulaiman;Sunidhi Chauhan	Halkat Jawani	indian

Playlist Score: 0.9047590125300001

The order is different, but the songs in each playlist are identical. This suggests that the PCA fix for encoding genres has its limitations as more songs are added to the playlist. Going forward, therefore, the recommendations will be limited to only songs within the same genre. This is to ensure that the user is not recommended songs too dissimilar to their own interests and will also improve computation speed. Regarding the playlist score, comparing the scores for Euclidean and cosine similarity is not useful. Only comparing scores of playlists that use the same similarity measure gives insight.

To further improve efficiency, the features used in similarity calculations can be limited to the ones found to be most effective in classifying genres and predicting popularity. The random forest classification model and regression model both had the same list of top 10 most important features. The features are acousticness, duration_ms, valence, danceability, tempo, loudness, energy, speechiness, liveness, and instrumentalness. While continuing to filter for the same genres, the new playlist with feature selection using cosine similarity is:

Artist	Song	Genre
Barry White	Can't Get Enough Of Your Love, Babe	disco
White Lion	Til Death Do Us Part	hard-rock
Pascal Rogé;Cristina Ortiz;Lon	Le Carnaval des Animaux, R. 125: 7. Aquarium	classical
Vivat	Życie to film	disco
Waterfall Sounds	Waterfall Sounds	sleep
Charlie Brown Jr.	Peso Da Batida Do Errado Que Deu Certo	hard-rock
Jürgen Drews;Ben Zucker	Ein Bett im Kornfeld	disco
Lorenz Büffel	Schifoan	disco
Maxel	Już Nie Jesteś Sam	disco
Dejw	Pozamiatane	disco

Playlist Score: 0.99803211058

Interestingly, the most similar songs for the Metallica, Dua Lipa, and rain songs are different with these criteria. The recommendation for Dua Lipa is interesting because the former recommendation was the same song just with a new featured artist on the track. With all features available, the model was able to recognize that as the most similar song, but the reduced feature model recommends a disco song instead. This may not be a poor choice, depending on the user's preferences, but this playlist is lacking genre variability with disco representing 6 of the 10 spots. Changing the similarity measure may be more helpful now with the reduced features. The resulting playlist with Euclidean similarity is:

Artist	Song	Genre
Barry White	Can't Get Enough Of Your Love, Babe	disco
White Lion	Til Death Do Us Part	hard-rock
Pascal Rogé;Cristina Ortiz;Lon	Le Carnaval des Animaux, R. 125: 7. Aquarium	classical
Willi Herren	Da sprach der alte Häuptling der Indianer	disco
Deep Sleep Rain Sounds	Loopable Rain	sleep
Dua Lipa;DaBaby	Levitating (feat. DaBaby)	pop
Maxel	Już Nie Jesteś Sam	disco
Akcent	King Of Disco	disco
KISS	I Was Made For Lovin' You - Single Mix	hard-rock
Masters	Czy te oczy mogą kłamać	disco

Playlist Score: 0.94636850844

Euclidean similarity correctly identifies Levitating (feat. DaBaby) as worthy of being included in the playlist, even if it's still not ranked as the most similar to the original song. Overall, this playlist appears to be the one that captures the user's preferences the best. The largest problem that still remains is songs in different languages being ranked as the most similar. This could be addressed by comparing the names of each song to see if they are in the same language. However, many non-English songs have an English title, which would make this comparison difficult. A Spotify tag marking each song's language would have been useful to further improve the recommended playlist. If this user is open to experiencing music in many languages, this playlist using Euclidean similarity, genre filtering, and features selected by random forest models accomplishes that goal.