Predicting Track Popularity

Shreya Musini
Vinuthna Hasthi

Abstract

Using a Kaggle dataset on Spotify tracks collected through Spotify's Web API and Python, we used numerical features such as danceability, paired with categorical features such as track genre and text features such as artist names to predict track popularity.

1 Introduction

We believe that figuring out what factors make a song popular is important to predict trends for artists, listeners in the music industry. This project uses machine learning to predict song's popularity by analyzing audio features and other artist information. By cleaning the data, creating new features with feature engineering, choosing the right models and fine tuning them we aim to build an accurate model that helps understand what listeners enjoy the most and improve music recommendations with overfitting.

2 Dataset

We chose the song popularity dataset from kaggle, [2] stored as 'dataset.csv'. We converted the csv file by importing the pandas package, it has 114000 songs from multiple artists, with some tracks repeated but classified under different genres. So each row is essentially a song and additional information about that specific track.

2.1 EDA statistics

The song popularity dataset basically looks at 89741 unique tracks and their specific artists, genre and other audio features like loudness, key, danceability etc. The popularity column ranges from 0 to 100, with most values being of 0 popularity.

3 Data Cleaning

The Spotify dataset had the following features in the dataset listed in Table 1.

The dataset was 114,000 instances with each row representing a track collected from the Spotify Web API. Looking at the columns, we dropped 'Unnamed:0' because it is the duplication of the index. As well, we dropped track_id from our data frame and kept the rest of the columns for feature engineering. All the columns except album_name, track_name, artists, and track_genre are categorical variables that measure factors of a song such as acousticness. The feature of 'explicit' was a binary variable that indicated if the track had explicit lyrics using True or False. We then investigated the number of missing values in the dataset and found three missing values, one in album_name, track_name, and track_genre. Because the missing values are consolidated into one entry in the data frame, we decided to drop the missing values instead of imputing them. Aside from those missing values, the dataset is relatively clean with no missing values.

No.	Feature
1	Unnamed:0
2	track_id
3	artists
4	album_name
5	track_name
6	popularity
7	duration_ms
8	explicit
9	danceability
10	energy
11	key
12	loudness
13	mode
14	speechiness
15	acousticness
16	instrumentalness
17	liveness
18	valence
19	tempo
20	time_signature
21	track_genre

Table 1: Features in Spotify Dataset

Before performing the machine learning predictive task, we investigated the format of the artists column in our data frame. We assumed that artist would play an influence on track popularity and wanted to include artists within our model through TF-IDF values. To do this, we wanted the vectorizer to treat an artist's whole name as a single word for numerical processing rather than treating the first and last name as separate words and indicate separate artists. Within our data frame, the artists were listed as artist_1;artist_2, which posed a problem in our processing. For example, taking the track "To Begin Again" by Ingrid Michaelson and Zayn, the artist is listed at "Ingrid Michaelson;Zayn", where the vectorizer would treat Michaelson;Zayn as a word instead of treating Michaelson to be part of the word "Ingrid" and "Zayn" as a separate word. We changed the artist column to have each artists' full name separated by underscores and replacing the semi-colon with spaces. Taking a look at "Ingrid Michaelson;Zayn", the new value for that track is "Ingrid_Michaelson Zayn."

4 EDA Findings

After looking at most columns of our dataset, we decided on looking closer into some of the columns that we believe will affect track popularity, first understanding how the distribution of the popularity column.

The popularity column rates the popularity of a track from 0 to 100. These are the numbers we are aiming to predict from the other

Conference'17, Washington, DC, USA
2025. ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

parametres of the dataset. Below is a histogram distribution of the popularity column song with a density curve.

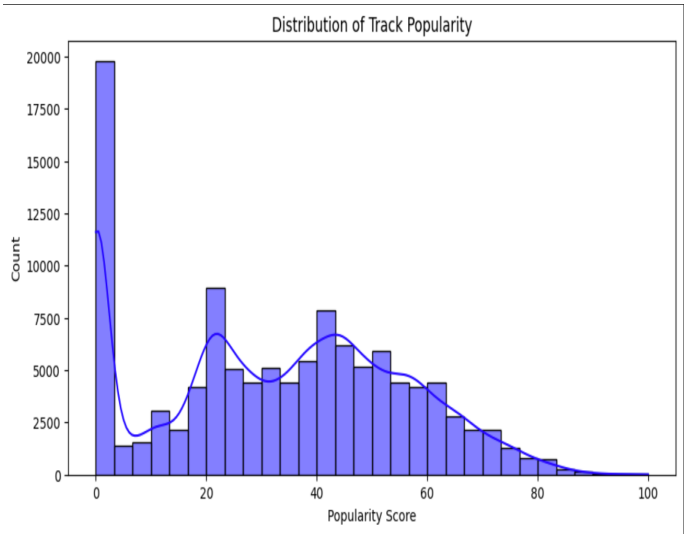

Figure 1: Popularity

As we can see above, there are a lot of tracks are concentrated at the lower end of the scale, peaking at 0 especially indicating a higher frequency of low popularity scores. There are smaller variations across the data but the general trend is that there is a decline in frequency as the score increases.

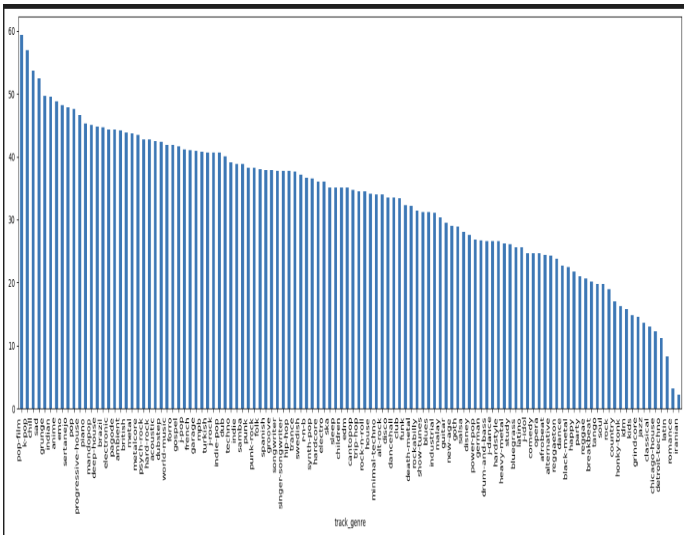

Figure 2: Popularity of music genres

The bar chart displays the average popularity of various music genres, sorted in descending order. The x-axis represents different genres, while the y-axis indicates the corresponding average popularity score. Genres like "pop film," "k-pop," and "pop" dominate the chart with the highest popularity, while genres such as "romance" and "iranian" have the lowest average popularity. The visualization effectively highlights the disparity in popularity across genres, with mainstream and globally recognized genres ranking higher.

Now, to identify if any other features have a strong correlation with track popularity, we make a correlation matrix below:

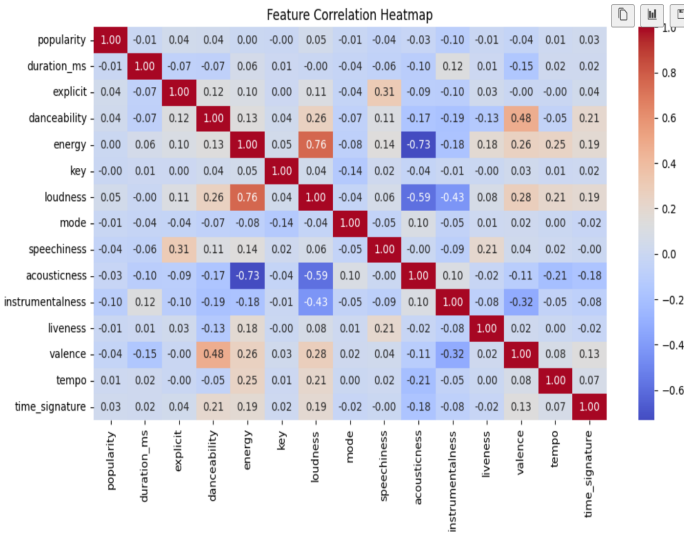

Figure 3: correlation matrix

The correlation heatmap visualizes relationships between music track features. For instance, loudness and energy show a strong positive correlation (0.76), while acousticness and energy have a strong negative correlation (-0.73). It offers insights into how musical elements interact, aiding in tasks like playlist curation or genre analysis. We came to a conclusion that there is strong correlation between audio features of tracks but not really with track popularity.

Now we're going to look at the artists table, there are 31437 unique artists included in our dataset, so instead of looking at a distribution of all artists we are going to look at the top 24 artists based on their average popularity below:

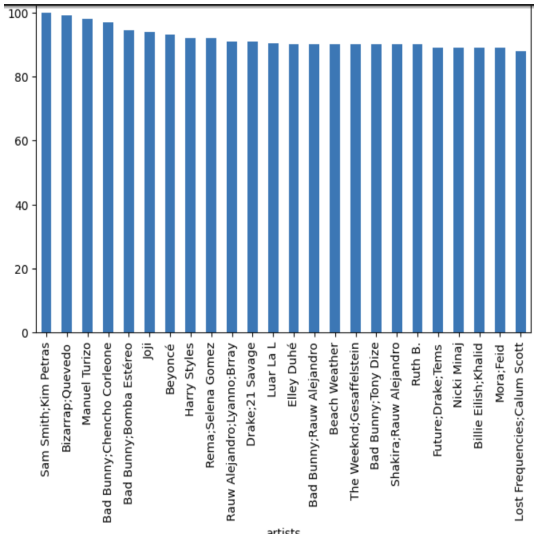

Figure 4: top artists based on popularity

The artists column is important in predicting track popularity because it identifies the creators of the track, and how many times they appear in our dataset. Popular artists often have a higher likelihood of chart-topping tracks. Additionally, unique artists have unique audio features qualities like energy, acousticness etc. making this column essential for predicting track popularity trends.

5 Prediction

5.1 Task

After cleaning and engineering, given artists, audio features, and track genres of tracks, our goal is to predict the popularity of a given track. This is a regression prediction problem.

5.2 Evaluation

Root Mean Square Error (RMSE) is our metric of choice for determining which model is most effective for our predictive task. We calculated R^2 score and RMSE but we believe RMSE is easier to interpret and tells us how wrong our predictions are in accuracy points. So now we have a better idea as to how many popularity points our prediction has i-on average from our actual popularity values. We are evaluating our performance based on average rmse score of the model when applied to a test data.

5.3 Baseline Models

We chose 4 models for our baseline: Linear regression model, Random Forest Regressor, XGB Regressor and LightGBM Regressor. **Linear Regression** model established a linear relationship between features and the target variable (popularity) making it easier to interpret.

Random Forest Regressor is a supervised machine learning algorithm and a type of ensemble learning method that combines multiple decision trees to improve performance. Unlike regression, it can also learn non-linear complex between features and how they impact the target variable. It is also less likely to overfit as it aggregates the prediction of multiple decision trees.

XGB Regressor can offer better predictive performance due to it's gradient boosting technique. It creates decision trees sequentially and it learns from the previous to make the current tree better.

LightGBM Regressor is trained to run faster compared to other boosting algorithms. It uses histogram based algorithms reducing computation which works the best on larger datasets.

The above models are trained on the same training data and the rmse score of each of these models is evaluated on the same test data.

5.4 Baseline Results

The rmse score for Linear Regression and Random Forest Regressor are kind of close with values 14.92 and 14.11 respectively. The XGBoostRegressor performed worse than we expected it would with an RMSE value of 17.19. Similarly Light GBM had an rmse score of 17.2. All models were run with their default parameters. To examine which features were important respective to each model, the feature importance was extracted and placed in a data frame to understand which features worked the best for each model.

For the feature importance of the Random Forest Regressor model, the top ten most important features were loudness, duration_ms, valence, acousticness, speechiness, danceability, tempo, energy, liveliness, and instrumentalness. The Random Forest model had the most important features be the numerical values associated with each track, with how loud the song is as most important for predicting track popularity. The top 10 features data frame with their importance is shown below.

	feature	importance
4	loudness	0.036187
0	duration_ms	0.036121
10	valence	0.035784
7	acousticness	0.035534
6	speechiness	0.035483
1	danceability	0.034571
11	tempo	0.034348
2	energy	0.033668
9	liveness	0.029287
8	instrumentalness	0.026970

Figure 5: Top 10 Features from Random Forest Regressor Baseline

The top 10 most important features of the XGBoost Regressor model were different track genres, with the 10 being Iranian, Romance, Pop-Film, Chill, Chicago-House, Detroit-Techno, Sad, Grindcore, Jazz, and Honky-Tonk. The most important features from this model became the one-hot encoded features of the tracks, showing that the model viewed the track genres as the most important features for predicting track popularity. Below are the top 10 features data frame and their importance level.

	feature	importance
72	track_genre_iranian	0.023808
106	track_genre_romance	0.015308
94	track_genre_pop-film	0.014705
28	track_genre_chill	0.013858
26	track_genre_chicago-house	0.013018
37	track_genre_detroit-techno	0.012048
107	track_genre_sad	0.011209
55	track_genre_grindcore	0.010935
77	track_genre_jazz	0.010444
65	track_genre_honky-tonk	0.009173

Figure 6: Top 10 Features from XGBoost Regressor Baseline

The top 10 most important features of the LightGBM Regressor model were also generally numerical values being duration_ms, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness and loudness. The most important features from this model became numerical values about the track, showing that the model viewed the different aspects of the song as important for predicting track popularity. It is also interesting to note that the top 10 features for the LightGBM Regressor model are the same as the Random Forest Regressor model, but in different order and metric for importance. Intuitively, it makes sense for these numerical features to be most important rather than the track genre in the XGBoost Regressor's features of importance. Below are the top 10 features data frame and their importance level.

5.5 Hyperparameter Tuning Using RandomizedSearchCV

To determine the most optimal model for predicting song popularity, we performed hyperparameter tuning for Random Forest, XGBoost, and LightGBM using RandomizedSearchCV with 5 cross folds and 30 random iterations of hyperparameters. The other parameters we chose were the n_jobs = -1 to use all the CPUs available and verbose

	feature	importance
0	duration_ms	36
1	danceability	29
2	energy	31
3	key	2
4	loudness	43
5	mode	3
6	speechiness	19
7	acousticness	41
8	instrumentalness	35
9	liveness	20

Figure 7: Top 10 Features from LightGBM Regressor Baseline

= 2. We chose to use RandomizedSearchCV over GridSearchCV to be computationally faster in fitting models and less computationally taxing to apply our 91k data training set for each model. As well, we included a random seed to keep the randomness consistent.

The parameter grid for random forest included tuning for the parameters of number of estimators, max depth, max features, min samples split and min samples leaf. The parameters for each feature were: [50, 100, 200], [None, 10, 20], [sqrt, log2, None], [2, 5] and [1, 2]. The best model from the randomized search was $\{n\_estimators : 50, min\_samples\_split : 2, min\_samples\_leaf : 1, max\_features : log2, max\_depth : None, bootstrap : True\}$.. The resulting RMSE score was 13.93 and an R^2 value of 0.6088. The hyperparameter tuning resulted in a marginally better RMSE value in comparison to the baseline for random forest 14.11. As well, the feature importance table was also made to show the top 10 features for the best model. The table showed the top 10 features to be the numerical features of each song, which is the same as the features that were produced in the baseline. However, the order of the features and the importance is on different scale. Below is a table of the top 10 features and their importance.

The parameter grid for xgboost included tuning for the parameters of number of estimators, max depth, learning rate, subsample and colsample_bytree, gamma, reg_alpha and reg_lambda. The parameters for each feature were: [100, 300, 500], [6, 8, 10], [0.01, 0.1, 0.3], [0.8, 1], [0.8, 1], [0, 0.1, 0.2], [0, 0.01, 0.1], and [1, 1.5, 2]. The best model from the randomized search was $\{subsample : 1, reg\_lambda : 1, reg\_alpha : 0.1, n\_estimators : 500, max\_depth : 10, learning\_rate : 0.3, gamma : 0.2, colsample\_tree : 1\}$.. The resulting RMSE score was 14.36 and an R^2 value of 0.58. The hyperparameter tuning resulted in a significantly better RMSE value in

	feature	importance
7	acousticness	0.035335
1	danceability	0.034628
10	valence	0.034555
11	tempo	0.034126
6	speechiness	0.033979
2	energy	0.033638
4	loudness	0.033171
0	duration_ms	0.032579
9	liveness	0.030979
8	instrumentalness	0.029705

Figure 8: Top 10 Features from Random Forest Regressor Hypertuned

	feature	importance
72	track_genre_iranian	0.016394
106	track_genre_romance	0.009279
37	track_genre_detroit-techno	0.008211
107	track_genre_sad	0.007329
94	track_genre_pop-film	0.006827
55	track_genre_grindcore	0.006642
65	track_genre_honky-tonk	0.006381
26	track_genre_chicago-house	0.005988
79	track_genre_kids	0.005928
67	track_genre_idm	0.005530

Figure 9: Top 10 Features from XGBoost Regressor Hypertuned

comparison to the baseline for xgboost of 17.20. As well, the feature importance table was also made to show the top 10 features for the best model. The table showed the top 10 features to be the one-hot encoded track genres, which had some overlapping features while other track genres were different. The new track genres were kids and idm. The scale of importance was the same with values smaller than the baseline. Below is a table of the top 10 features and their importance.

And lastly, the parameter grid for lightgbm included tuning for the parameters of n_estimators, max_depth, learning_rate, num_leaves, min_child_samples, subsample, colsample_bytree, reg_alpha, and reg_lambda. The parameters for each feature were: [100, 200, 300], [-1, 10, 20, 30, 50], [0.01, 0.05, 0.1], [31, 50, 100], [5, 10, 20, 30], [0.8, 1.0], [0.8, 1.0], [0, 0.1, 0.5] and [0, 0.5, 1.0]. The best model from the randomized search was $\{subsample : 0.8, reg\_lambda : 0.0, reg\_alpha : 0.1, num\_leaves : 100, n\_estimators : 300, min\_child\_sam$ $5, max\_depth : 50, learning\_rate : 0.1, colsample\_tree : 1\}$.. The resulting RMSE score was 14.38 with an R^2 score of 0.58. The hypertuned model also did significantly better than the baseline model of 17.2. The feature importance table was also made to show the top 10 features for the best model. The table showed the top 10 features be the numerical features for each track, which is the same top 10 features as the baseline model. However, the order and scale of importance are much different from the baseline. For example, the importance of the duration_ms feature was 36 in the baseline but came to be 1186 in the hypertuned model. Below is a table of the top 10 features and their importance.

	feature	importance
0	duration_ms	1186
10	valence	1120
1	danceability	1077
7	acousticness	1023
6	speechiness	992
2	energy	952
4	loudness	928
9	liveness	878
8	instrumentalness	838
11	tempo	813

Figure 10: Top 10 Features from LightGBM Regressor Hypertuned

6 Literature

According to the project at the site, [1] Victor's project also aims to predict track popularity based on all the features in the dataframe. They proceeded on to do very detailed EDA on every feature of the data (histogram distribution and boxplot). They also dropped values of time signature column that are not 3 and 4 and also turned mode and time signature into categorical features. Some of the EDA graphs like popularity distribution and correlation matrix. The models they chose were Linear Regression, Random Forest Regressor, Ridge and Gradient Boosting. Comparing the RMSEs, ours were a little less than their models' accuracy even though we assumed it would be better with the additional data analysis they did. For hyperparameter tuning, they also used RandomizedSearchCV because GridSearchCV takes too much time to run especially for a dataset of this size. Their parameters were also different for randomized search, but they looked at different evaluation metrics like MAE while we focused on RMSE which makes it harder to compare our models. They also only chose Random Forest Regressor to hyper parameter tune on while we additionally also hyper parameter tuned on XGB Regressor and LightGBM.

7 Results

Based on our randomized grid search, the best model was the Random Forest Regressor with an RMSE value of 13.9. However, it is important to note that all the models produced RMSE values that were not too far off from each other, with the second best model being XGBoost Regressor with an RMSE of 14.36 and LightGBM being the last model with an RMSE score of 14.38. In addition to RMSE values, when we look at R^2 values of 0.60 for Random Forest Regressor and 0.58 for XGB Regressor while LightGBM had the same R^2 as XGBoost with a value of 0.58. As well, the top 10 features for Random Forest and LightGBM are the same with different levels of importance and order. Notably, LightGBM Regressor has a much higher importance value. The outcome of our randomized grid search did not match our assumption that LightGBM or XGBoost would be the best performing model but this can be attributed to the fact that we performed a randomized search of 150 different models rather than performing a brute force search for the best model. Based on our findings, it seems that the numerical features of a track such as their instrumentalness play a big impact on the resulting popularity rating on a scale from 0 to 100, with Random Forest Regressor being the best model to predict the track popularity. Results from the randomized grid search suggest that a less complex model like random forest, with the right hyperparameters can outperform more complex models such as XGBoost in terms of RMSE.

We understand that there could be some bias in some of our features, for example, time signature mostly has values of 3 and 4 which might not have had a relevant significance to predicting track popularity. As well, the randomized testing of hyperparameter combinations could play an impact in the outcome of our findings to result in Random Forest being better than the other two models we tested. If a brute force method such as GridSearchCV were performed, the result of the best model for predicting track popularity might be different from our findings.

References

[1] Eden Mapia. 2024. ML-FIL-A3-Victor-Mathias. https://www.kaggle.com/code/edenmapia/ml-fil-a3-victor-mathias Accessed: 16 March 2025.
[2] Maharshi Pandya. 2021. Spotify Tracks Dataset. https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset/data Accessed: 8 March 2025.