Project Title: Song/Artist Popularity Analysis

Team Members: *Neeraja Kirtane (kirtane3) , Sanil Chawla (schawla7), Omkar Gurjar (ogurjar2), Atharva Naik (annaik2)*

1. Abstract

We aim to study the various factors affecting the popularity of a song or artist. Specifically, we plan to develop various supervised and unsupervised ML and data mining models to predict song popularity and identify the properties of the song that determine it's popularity.

2. Introduction

The project aims to perform a comprehensive study of understanding the popularity of songs on online platforms using data driven methods. Specifically, we use a collection of $\approx 5.8M$ songs by $\approx 1.1M$ artists available on a popular streaming platform- Spotify. For every song, we are provided its popularity score along with a set of features determining its musical properties, its genre and the artists. As a part of this report, we perform an EDA of the data and visualize its various aspects to thoroughly understand its strengths, weaknesses and perform appropriate data processing steps. Next, we model a regression task of using song features to predict its popularity. We experiment with various models coupled with feature selection and parameter tuning. We also study the dependence of popularity of songs on the popularity of their respective artists using an unsupervised approach. Finally, we run a time-series model to predict popularity capturing temporal trends in the data. The code is available at our GitHub repository [1].

3. Motivation

Music has been the source of entertainment, expression and emotion for as long as we can remember. Even today, streaming platforms such as Spotify have over 551 million users [9] making it a goldmine of data related to users' music preferences. Machine learning algorithms coupled with the right data processing and feature engineering can help surface key patterns related to users' audio consumption. The project's potential applications can be extended to various music streaming platforms, where it can help make personalized recommendations and playlist curation. Additionally, it offers a remarkable opportunity to contribute to the ongoing research in both the data science and music communities.

4. Related work

Hit Song Science (HSS) [1] is the study of predicting whether a song would get 'hit' or popular on music platforms. Works such as [15][16][12] have tried to use song features derived using Spotify API to predict billboard song success and number of streams by training machine learning models. [17] aims to propose statistical tools for identifying features deterministic of a song's success. From a psychology point of view several works such as [11][19][10] have tried to explore the complexities of music preferences based on peoples' personality, mental state and other factors. On online ML comptetitions forums such as Kaggle , several users have independently compiled similar datasets for different time-frames and artists [2],[3] These datasets have been leveraged for other tasks including recommendation [4],[5] and classification [6],[7].

5. Methodology

i. Exploratory Data Analysis & Data Pre-processing:

[1] https://github.com/atharvanaik10/song-popularity-analysis

1. **EDA** - Our dataset contains two sub-datasets (1) Songs data containing some features such as acousticness, danceability and liveliness for 586672 songs along with their popularity which is the target variable. (2) Artist data containing information related to 1162095 artists included in the songs data. As a part of EDA, we compute the descriptive statistics(range, mean, median) for different features, investigate data-quality issues(missing values, any skews in data) and calculate inter-feature correlations. We also visualize the EDA results by plotting feature trends over the years and the Pearson's correlation matrix as a heatmap. Full details of the dataset(description of columns, quality issues etc) are included in the Appendix.

2. **Data Processing -** We model the task of predicting the song popularity as a regression task. **Before we went ahead with any pre-processing, we analysed a few data quality issues** -

   (a) Although null-values were relatively very less, the main issue we ran into was the presence of a highly skewed dataset i.e. a severely high number of songs with popularity as 0 (41,135 rows). This made scaling our target variable a necessity.

   (b) Another core concern we noticed was the column **genre**. On fine-grained observation, 8,56,500 songs(73.07% of our dataset) did not have any genre mapped. Every row that did have a value, consisted of an array containing several genres making it difficult to normalize, and hence being dropped out.

   (c) Lastly, we focus on modifying the release date column and extracting just the year from it as a feature, which gives a much better value to consider as multiple songs could be launched in the year, which could show some relevance to song popularity.

   For data pre-processing, we created a pipeline as follows -

   (a) First of all we merge the artist dataset with the track dataset on the column 'artists'. We notice that the track dataset has multiple artists for a single song, hence we explode the dataset i.e. create n rows for the same track where that song has n artists. Then we use this dataset to merge with the artist dataset. This is the only place we observe data loss as not all artists are present in the artist dataset as compared to the tracks dataset.

   (b) Min-Max[2] to normalize the features to the range (0, 1) to eliminate issues due to large ranges (e.g. number of followers). We also experimented with other normalization techniques such as z-score but received inferior results.

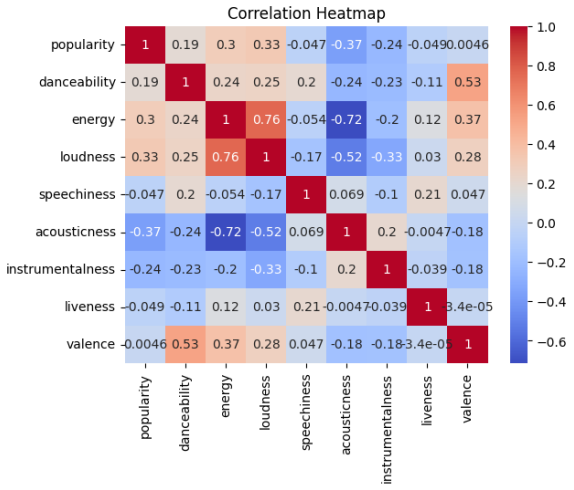
   (c) We also scale our target variable popularity to be between 0 and 1 using Min-Max scaling to minimize its range of values from 0 to 99 to 0 to 1.

ii. Regression

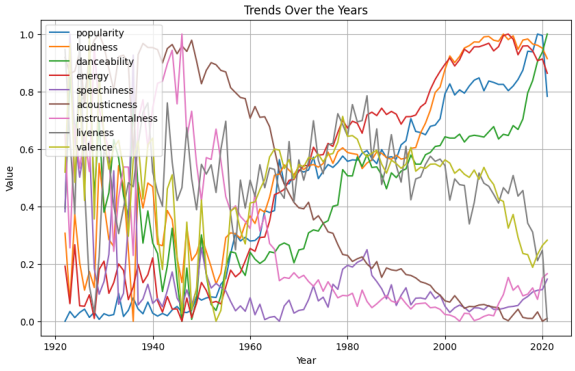
Next, we perform our modeling task to predict the song's popularity based on 12 song-related features. We train a suite of supervised regression algorithms including linear/lasso regression, elastic net, decision tree, various gradient boosting algorithms, and K Nearest Neighbor. We divide our dataset into train and test with a 70:30 ratio using a fixed seed value. Evaluation metrics- Mean Squared Error (MSE), Mean Absolute Error(MAE) and coefficient of determination (R^2) are used to compare the performance. We use Grid search and Randomized Search to select the best hyperparameter in all the models followed by feature selection to select the best set of features.

To add explainability to our model results, we also conduct a SHAP analysis [13]. SHAP uses a

[2] we utilize MinMaxScaler from Sklearn



(a) Pearson's Correlation Heatmap of Features



(b) Yearly Min-Max Normalized Means of Features

Figure 1: Results of EDA and data pre-processing. We observe interesting correlations and temporal trends: (a) Energy and loudness are highly correlated, while acousticness has a negative correlation with both. Loud and energetic songs tend to be more popular. (b) Yearly trends shows an upward trend in danceability and loadness over the years, while instrumentalness has gone down.

game-theory-based approach to quantify the impact of each feature on model predictions. Due to its model-agnostic nature, it can be applied to all our models.

iii. Quantifying the impact of artist popularity As the next analysis, we look to utilize the artist dataset and quantify the extent to which the popularity of songs is driven by the popularity of the artists. To have a fair comparison, we first group similar songs together. For this, we utilize KMeans clustering to cluster songs based on the 12 song-related features in our dataset. We initialize the cluster centers using the kmeans++ method which uses a sampling-based approach over random initialization. Further, we experiment with different numbers of clusters and use the elbow method based on the Silhouette score [3] to select the number of clusters. Next, we identify the top artists by taking the top 1% artists by their popularity. Finally, for each cluster, we compare the mean popularity of songs by top artists to the rest. We perform the one-tailed z-test to check the significance of our results.

iv. Time series analysis We also perform a time series analysis on the popularity. The data is first resampled every 6 months over the mean of the features during that period. The train test split is performed as a time-series split: the first 0.8 of the data is the training set, and the remaining 0.2 of the data is the test set. Then, we train SARIMAX (Seasonal Autoregressive Integrated Moving Average eXogenous model)[8] and Facebook Prophet [18] models on popularity. In order to train the ARIMAX model, we use hyperparameter tuning with auto-Arima to find the best parameters. These models are able to handle exogenous features (i.e. independent explanatory variables) that allow us to understand the impact of the various features on popularity. Finally, we calculate the MSE, RMSE, MAE, and r^2 scores for the predictions.

6. Empirical Results

Regression: We pick linear regression as our baseline. Ridge regression is picked to handle multi-collinearity observed during EDA. Different tree-based gradient boosting models are tried owning to their ability to handle non-linearity and large number of features.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score

The results of hyperparameter tuning can be observed in Table 1. To perform hyperparameter tuning, we ran GridSearchCV (GS) and RandomizedSearchCV (RS) on all the regression models. For both techniques, we passed an input parameters grid that contained a list of parameters out of which the parameters that gave the best score were selected. On a deeper look we observe very minimal changes in our baseline Logistic Regression model when we compare MSE, but we see significant fall in the MSE in the DecisionTree, ElasticNet and K-Neighbors regressors. After hyperparameter tuning the best model we can obtain is the XGBoost Regressor followed by the LightGBM Regressor as a close second on comparing all the 4 metrics of MSE, RMSE, MAE and R^2 scores. We still see that MLP Regressor has had a significant performance dip compared to our pre-tuning models, this can be attributed to the parameter grid not containing an optimal input giving a result that was better than the default values.

Search Method	**Model Name**	**MSE**	**RMSE**	**MAE**	**R^2**
RandomizedSearchCV	LinearRegression	0.0257	0.1603	0.1300	0.1765
RandomizedSearchCV	Ridge	0.0257	0.1603	0.1300	0.1765
GridSearchCV	Lasso	0.0311	0.1763	0.1451	0.0040
RandomizedSearchCV	ElasticNet	0.0294	0.1716	0.1411	0.0571
RandomizedSearchCV	DecisionTree	0.0241	0.1553	0.1232	0.2278
RandomizedSearchCV	AdaBoost	0.0258	0.1606	0.1308	0.1742
GridSearchCV	LGBM	0.0224	0.1498	0.1190	0.2814
GridSearchCV	**XGB**	**0.0224**	**0.1498**	**0.1190**	**0.2810**
GridSearchCV	MLP	30.5914	5.5309	5.0929	-978.9932
GridSearchCV	KNeighbors	0.0329	0.1813	0.1470	-0.0534

Table 1: Summary of Hyperparameter Tuning using GridSearch and RandomizedSearch for all models. The MSE,RMSE,MAE and R^2 scores depict the best set out of the both hyperparameter tuning results

Model	**Features Selected**	**MSE**	**RMSE**	**MAE**	**R^2**
LinearRegression	9	0.0257	0.1603	0.1300	0.1763
Ridge	9	0.0257	0.1603	0.1300	0.1763
Lasso	5	0.0311	0.1763	0.1451	0.0040
ElasticNet	6	0.0311	0.1763	0.1451	0.0040
DecisionTreeRegressor	12	0.0439	0.2096	0.1622	-0.4078
GradientBoostingRegressor	10	0.0234	0.1531	0.1227	0.2493
XGBRegressor	**11**	**0.0225**	**0.1500**	**0.1191**	**0.2796**
LGBMRegressor	12	0.0227	0.1505	0.1199	0.2741
MLPRegressor	10	0.0312	0.1767	0.1453	-0.0003

Table 2: Summary of Model Metrics using Recursive Feature Selection and Black Box Estimation. Features Selected shows final set chosen out of the 12 available features.

We also performed feature selection on all of our models, and the results of the same can be found in Table 2. For all our models, we used Recursive Feature Elimination, starting from 5 features as our baseline and going all the way to 12 maximum features. The table contains the result for the best score that was obtained for each model during this recursive feature search. Notice that for the MLP Regressor, which is our black box neural network, we could not use the above methods directly due to the absence of feature importance or coefficients in the fitted model. Hence, we used another method called LIME (Local Interpretable Model-Agnostic Explanations) [14] which gives us out-of-the-box functions that can be then used as a wrapper around our trained neural network to then test multiple iterations of the same. Post feature selection we do not obtain improvements in the baseline LinearRegression, but we see a significant improvement in our MLP Regressor. The performance scores for MLP Regressor have had a significant improvement based on the black box estimation that we implemented

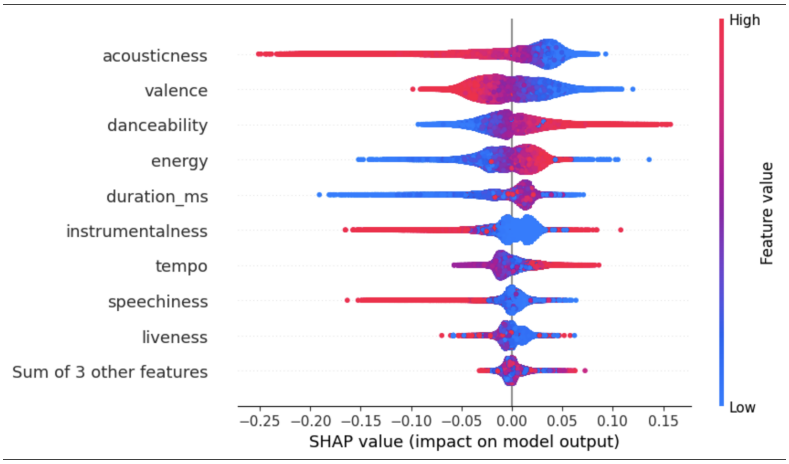
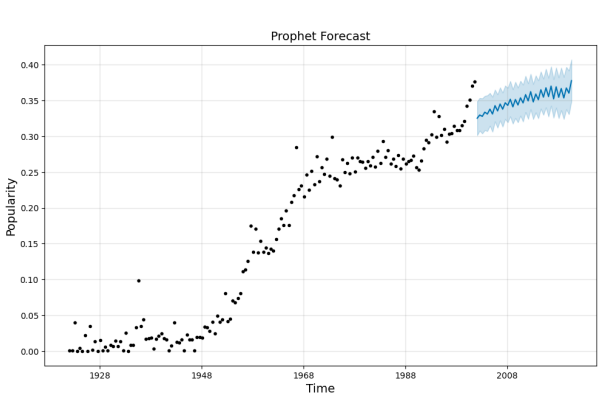


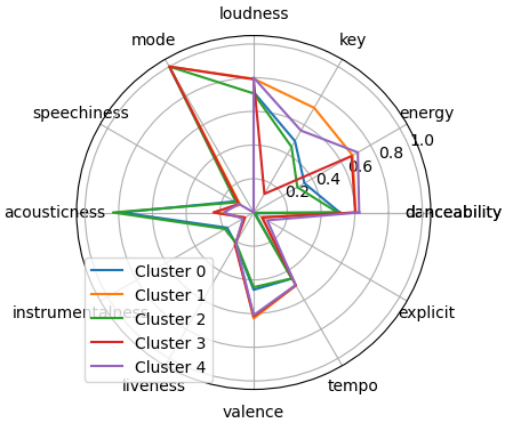
Figure 2: Feature importance using SHAP for all the predictions made by our best LightGBM model. Features are ordered by highest mean SHAP values from top to bottom. Positive SHAP values imply the feature positively affects the prediction. Red color refers to high feature value while blue means low value.

above. We still can see the trend of our XGBoost Regressor being our best model and LGMB Regressor being a close second on comparing the 4 scores of MSE, RMSE, MAE and R^2. A more detailed explanation of the final features that were chosen for each model and the best parameters derived from the tuning can be found in the Appendix.

The results of the SHAP analysis are shown in Figure 2. We observe that acousticness is the most-important feature and the high value of acousticness generally impacts the popularity negatively. Other important features include valence and danceability. High danceability positively impacts the popularity of the song. Finally, we infer that features like key and mode are the least important in driving the predictions.



(a)



(b)

Figure 3: showing clustering results. (a) Time series forecast of popularity using Prophet algorithm. (b) Plot showing the cluster centres for the KMeans clusters. Note that all features are min-max normalized.

Quantifying the impact of artist popularity We observe an "elbow" in our plot when the number of clusters is 5 and hence report the corresponding results. Table 3 shows the results of the analysis for top 1% most-popular artists. First, we observe that the percentage of songs by top-artists (and hence the percentage of songs by other artists) is fairly consistent across all the clusters. This means that the songs by top and other artists are actually similar when

Cluster	% top songs	$\mu(P_{TA})$	$\mu(P_{OA})$
Cluster 1	36.67	**27.65****	16.29
Cluster 2	46.34	**40.59****	25.07
Cluster 3	38.70	**25.98****	14.94
Cluster 4	46.36	**40.22****	24.51
Cluster 5	45.87	**42.69****	26.36

Table 3: Impact of artist popularity on the popularity of songs. % top songs is the percentage of songs in the cluster involving top 1% artists. $\mu(P_{TA})$, $\mu(P_{OA})$ is the mean popularity for songs by top and other artists respectively, for the given cluster. ** $p < 0.05$

it comes to their acoustic features and have a similar distribution of features represented by the five clusters. Next, we see that within each cluster, the mean song popularity for songs by top artists is always significantly higher as compared to their counterparts by other artists with a maximum of **74%** increase for Cluster 3. These results show that the popularity of the artists has a huge impact in determining the popularity of the song, in other words, a similar "sounding" song by a top artist would be more popular as compared to other artists.

Time-Series Analysis The time-series analysis is unique because it can provide insights into how various features have affected the average popularity of songs over time. The table below shows the metrics for both models. With these models, we can predict how a song will perform in a certain time period given its features. We notice that the ARIMA model performs better than the Prophet model. This is likely because SARIMAX explicitly includes exogenous variables in the autoregressive and moving average equations, while Prophet assumes that the impact of the external variables is linear and independent of the historical values of the target variable (popularity).

Model	MSE	RMSE	MAE	R^2 Score
SARIMAX	**0.0017**	**0.0419**	**0.0274**	**-0.5177**
Prophet	0.0027	0.0524	0.0456	-1.3718

Table 4: Time-series model metrics

8. Conclusions / discussions

In this study, we analyzed the Spotify popularity data and built models to uncover key results related to factors driving song popularity. We showed different insights through our visualizations, did the necessary pre-processing steps, and analyzed our regression models. We observe that for the regression task, boosting models perform better than linear models. XGBoost is the best-performing model for predicting song popularity. SHAP analysis identified acousticness and valence as the most impactful features. Using KMeans, we find that artist's popularity plays a crucial role in boosting the popularity of the song. We can also conclude that the SARIMAX model is optimal for forecasting popularity. For future work, we plan to study how the popularity measure was derived to understand the problem better and check its validity. We also want to see how our analysis will work out for non-English songs from regions across the globe.

References

[1] Hit song science. https://en.wikipedia.org/wiki/Hit_Song_Science.

[2] Kaggle dataset 1. https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset.

[3] Kaggle dataset 2. https://www.kaggle.com/datasets/jarredpriester/taylor-swift-spotify-dataset.

[4] Kaggle dataset 3. https://www.kaggle.com/code/vatsalmavani/music-recommendation-system-using-spotify-dataset.

[5] Kaggle dataset 4. https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system.

[6] Kaggle dataset 5. https://www.kaggle.com/competitions/machine-learning-pw4-part2-classification-evd1.

[7] Kaggle dataset 6. https://www.kaggle.com/competitions/machine-learning-pw3-evd2.

[8] Sarimax model. https://www.statsmodels.org/dev/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html.

[9] Spotify statistics. https://newsroom.spotify.com/company-info/.

[10] Bruce Ferwerda, Marko Tkalcic, and Markus Schedl. Personality traits and music genres: What do people prefer to listen to? In *Proceedings of the 25th conference on user modeling, adaptation and personalization*, pages 285–288, 2017.

[11] Alinka E Greasley and Alexandra M Lamont. Music preference in adulthood: Why do we like the music we do. In *Proceedings of the 9th international conference on music perception and cognition*, pages 960–966. University of Bologna Bologna, Italy, 2006.

[12] Joshua S Gulmatico, Julie Ann B Susa, Mon Arjay F Malbog, Aimee Acoba, Marte D Nipas, and Jennalyn N Mindoro. Spotipred: A machine learning approach prediction of spotify music popularity by audio features. In *2022 Second International Conference on Power, Control and Computing Technologies (ICPC2T)*, pages 1–5. IEEE, 2022.

[13] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[14] Carlos Guestrin Marco Tulio Ribeiro, Sameer Singh. "why should i trust you?": Explaining the predictions of any classifier". *arXiv*, 2016.

[15] Kai Middlebrook and Kian Sheik. Song hit prediction: Predicting billboard hits using spotify data. *arXiv preprint arXiv:1908.08609*, 2019.

[16] Rutger Nijkamp. Prediction of product success: explaining song popularity by audio features from spotify data. B.S. thesis, University of Twente, 2018.

[17] Mariangela Sciandra and Irene Carola Spera. A model-based approach to spotify data analysis: a beta glmm. *Journal of Applied Statistics*, 49(1):214–229, 2022.

[18] Sean J Taylor and Benjamin Letham. Forecasting at scale. 2017.

[19] Sunkyung Yoon, Edelyn Verona, Robert Schlauch, Sandra Schneider, and Jonathan Rottenberg. Why do depressed people prefer sad music? *Emotion*, 20(4):613, 2020.

A Appendix

1. **Dataset Details and summary**
   The dataset that we have chosen contains the following attributes -

   Track ID Unique identifier for the track.

   Track Name Name of the track.

   Popularity Popularity of the track in the range 0 to 100.

   Duration (ms) Duration of the song in milliseconds.

   Explicit Indicates whether the track contains explicit content or not.

   Artists List of artists who created the track.

   Artist IDs Unique identifiers of the artists who created the track.

   Release Date Date of release of the track.

   Danceability A measure of how danceable a song is, ranging from 0 to 1.

   Energy A measure of how energized a song is, ranging from 0 to 1.

   Key The key the track is in. Integers map to pitches using standard Pitch Class notation ranging from 0 - 11

   Loudness How loud a song is in decibels (dB) normalized to the range of 0 to 1

   Mode The modality of the track, where 0 indicates minor and 1 indicates major.

   Speechiness The presence of spoken words in the track, ranging from 0 to 1.

   Acousticness How acoustic a track is, ranging from 0 to 1.

   Instrumentalness The absence of vocal sounds.The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.

   Liveness Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.

   Valence A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

   Tempo The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration. Values scaled to 0 - 1.

   Time Signature An estimated time signature. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures

   Followers The followers for an artist having an artist id. Values were scaled to between 0 to 1

   Artists Popularity The popularity for an artist having an artist id. Values were scaled to between 0 to 1

Metric	Mean	Std	Min	25%	50%	75%	Max
Popularity	28.39	17.49	0.00	15.00	28.00	41.00	99.00
Duration (ms)	226,738	114,759	3,344	175,647	214,787	261,516	5,621,218
Explicit	0.03499	0.18375	0.00000	0.00000	0.00000	0.00000	1.00000
Danceability	0.56591	0.15908	0.00000	0.46000	0.57700	0.68200	0.99100
Energy	0.55545	0.24411	0.00000	0.36700	0.56000	0.75500	1.00000
Key	5.22310	3.52034	0.00000	2.00000	5.00000	8.00000	11.00000
Loudness	0.76658	0.07065	0.00000	0.72674	0.77801	0.81879	1.00000
Mode	0.66443	0.47219	0.00000	0.00000	1.00000	1.00000	1.00000
Speechiness	0.10076	0.17809	0.00000	0.03320	0.04280	0.07130	0.96900
Acousticness	0.42915	0.33930	0.00000	0.08880	0.39400	0.74600	0.99600
Instrumentalness	0.09376	0.24188	0.00000	0.00000	0.00002	0.00512	1.00000
Liveness	0.21400	0.18524	0.00000	0.09800	0.13900	0.27800	1.00000
Valence	0.56433	0.25263	0.00000	0.36100	0.57600	0.77800	1.00000
Tempo	0.48391	0.12056	0.00000	0.39111	0.47910	0.55724	1.00000
Time Signature	3.87966	0.45668	0.00000	4.00000	4.00000	4.00000	5.00000
Release Year	1988.85	21.28	1900.00	1976.00	1992.00	2006.00	2021.00
Followers	0.01350	0.04881	0.00000	0.00016	0.00119	0.00755	1.00000
Artists Popularity	0.50311	0.19689	0.00000	0.38000	0.52000	0.65000	1.00000

Table 5: Summary Statistics of the Data

The dataset contains 586,672 rows of popular tracks, but after merging the tracks data with artists we minimze the dataset due to missing artist ids, thereby having a total of 470,038 rows. It contains Null values only in the following column -

(a) Track Name - 71 rows

(b) Followers - 11 rows

Model	**MSE**	**RMSE**	**MAE**	**R^2 Score**
Linear Regression	0.0257	0.1603	0.1300	0.1765
Ridge	0.0257	0.1603	0.1300	0.1765
Lasso	0.0311	0.1763	0.1451	0.0040
ElasticNet	0.0311	0.1763	0.1451	0.0040
Decision Tree Regressor	0.0440	0.2097	0.1622	-0.4090
Gradient Boosting Regressor	0.0235	0.1532	0.1229	0.2485
AdaBoost Regressor	0.0266	0.1630	0.1347	0.1485
K-Neighbors Regressor	0.0342	0.1849	0.1494	-0.0952
MLP Regressor	2.0150	1.4195	0.9874	-63.5507
XGBoost Regressor	0.0225	0.1501	0.1191	0.2787
LightGBM Regressor	**0.0227**	**0.1505**	**0.1199**	**0.2741**

Table 6: Regression Model Metrics- Preliminary Results

Table 7: Summary of Hyperparameter Tuning

Search Method	**Model Name**	**Best Parameters**	**MSE**	**RMSE**	**MAE**	**R^2**
RS	LinearRegression	{'copy_X': True, 'fit_intercept': True, 'n_jobs': -1}	0.0257	0.1603	0.1300	0.1765

Continued on next page

Table 7 – Continued from previous page

Search Method	Model Name	Best Parameters	MSE	RMSE	MAE	R^2
RS	Ridge	{'solver': 'cholesky', 'fit_intercept': True, 'alpha': 1.0}	0.0257	0.1603	0.1300	0.1765
GS	Lasso	{'alpha': 10.0, 'fit_intercept': True, 'selection': 'cyclic'}	0.0311	0.1763	0.1451	0.0040
RS	ElasticNet	{'selection': 'cyclic', 'l1_ratio': 0.1, 'fit_intercept': True, 'alpha': 0.1}	0.0294	0.1716	0.1411	0.0571
RS	DecisionTree	{'splitter': 'best', 'max_depth': 10, 'criterion': 'friedman_mse'}	0.0241	0.1553	0.1232	0.2278
RS	AdaBoost	{'learning_rate': 0.01, 'n_estimators': 200}	0.0258	0.1606	0.1308	0.1742
GS	LGBM	{'n_estimators': 200, 'max_depth': 5, 'learning_rate': 0.2}	0.0224	0.1498	0.1190	0.2814
GS	XGB	{'learning_rate': 0.2, 'max_depth': 5, 'n_estimators': 200}	0.0224	0.1498	0.1190	0.2810
GS	MLP	{'alpha': 0.0001, 'hidden_layer_sizes': (50, 50)}	30.5914	5.5309	5.0929	-978.9932
GS	KNeighbors	{'n_neighbors': 7, 'weights': 'uniform'}	0.0329	0.1813	0.1470	-0.0534

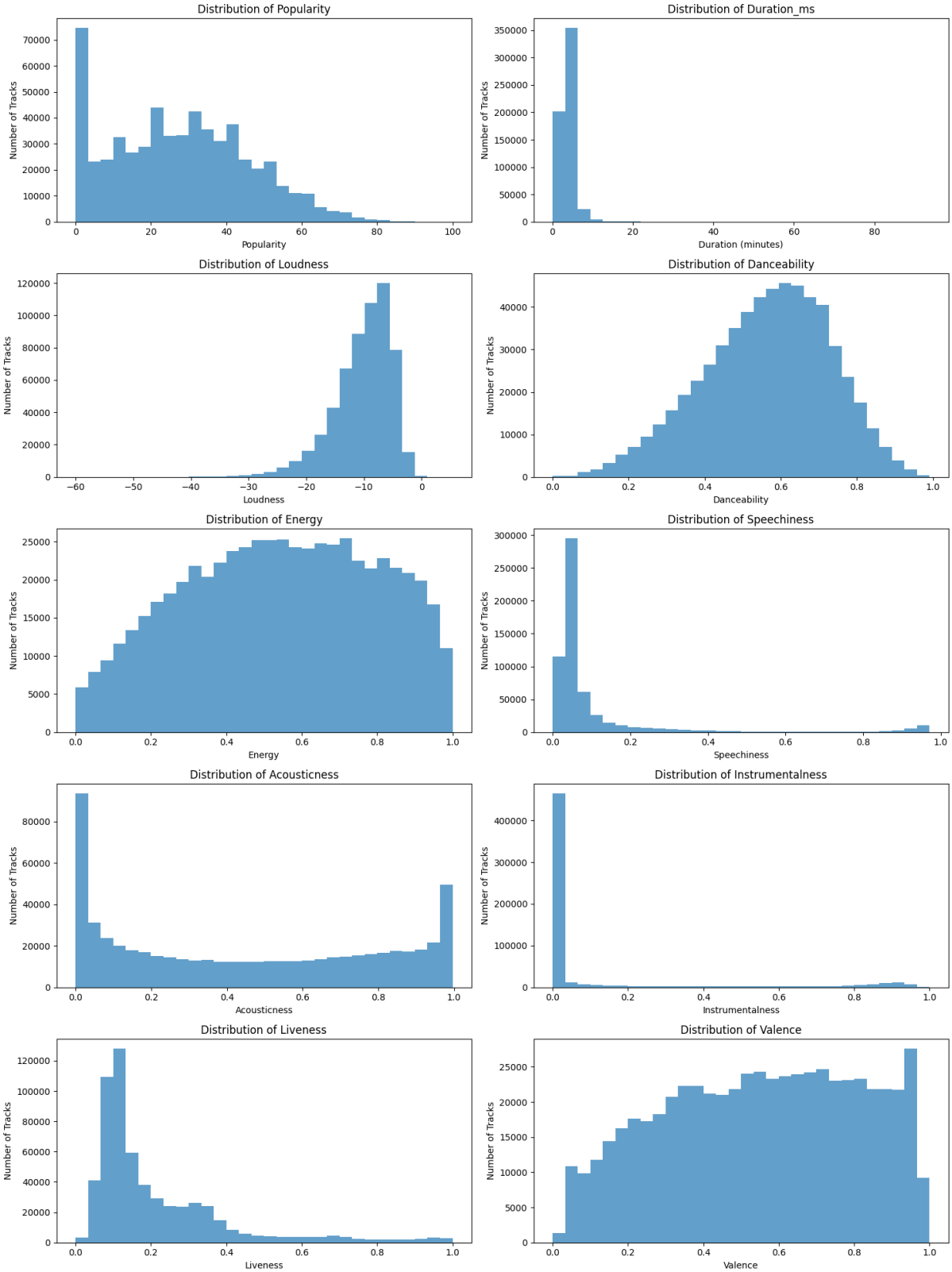
Table 8: Summary of Model Metrics and Feature Selection

Model	Number of Features Selected	Selected Features	MSE	RMSE	MAE	R^2
LinearRegression	9	['acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'mode', 'speechiness', 'tempo', 'valence']	0.0257	0.1603	0.1300	0.1763
Ridge	9	['acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'mode', 'speechiness', 'tempo', 'valence']	0.0257	0.1603	0.1300	0.1763
Lasso	5	['duration_ms', 'speechiness', 'tempo', 'time_signature', 'valence']	0.0311	0.1763	0.1451	0.0040
ElasticNet	6	['duration_ms', 'mode', 'speechiness', 'tempo', 'time_signature', 'valence']	0.0311	0.1763	0.1451	0.0040

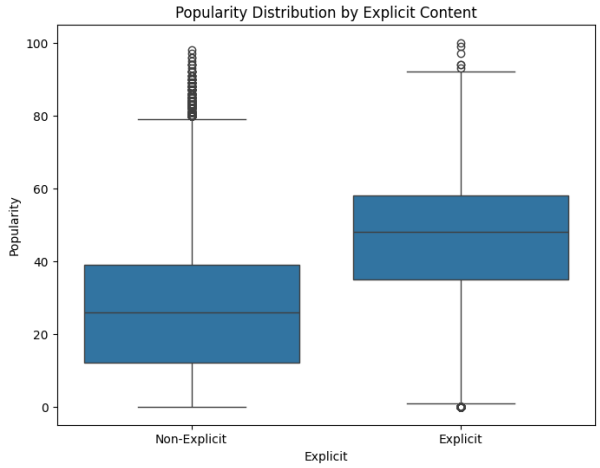
Continued on next page

Table 8 – Continued from previous page

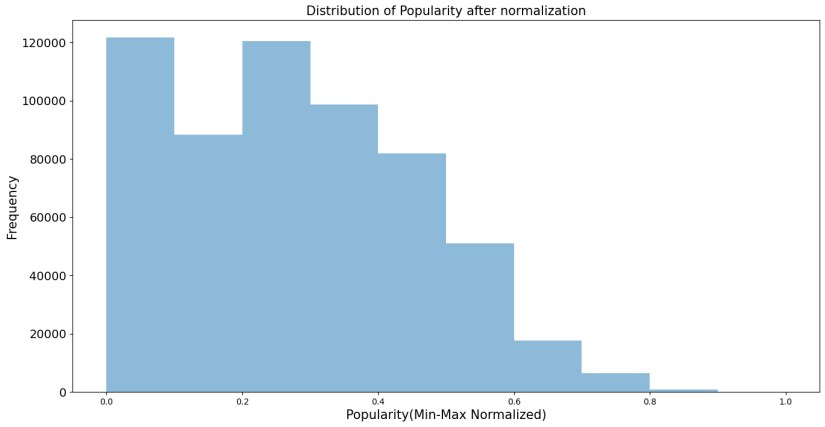
Model	**Number of Features Selected**	**Selected Features**	**MSE**	**RMSE**	**MAE**	**R^2**
DecisionTreeRegressor	12	['acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'key', 'liveness', 'mode', 'speechiness', 'tempo', 'time_signature', 'valence']	0.0439	0.2096	0.1622	-0.4078
GradientBoostingRegressor	10	['acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'liveness', 'speechiness', 'tempo', 'time_signature', 'valence']	0.0234	0.1531	0.1227	0.2493
XGBRegressor	11	['acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'key', 'liveness', 'mode', 'speechiness', 'tempo', 'valence']	0.0225	0.1500	0.1191	0.2796
LGBMRegressor	12	['acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'key', 'liveness', 'mode', 'speechiness', 'tempo', 'time_signature', 'valence']	0.0227	0.1505	0.1199	0.2741
PermutationImportance (MLPRegressor)	10	['acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'key', 'liveness', 'mode', 'time_signature', 'valence']	0.0312	0.1767	0.1453	-0.0003



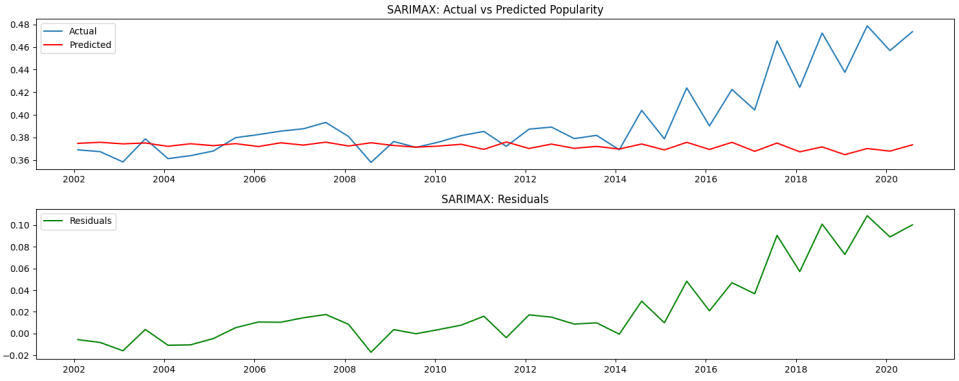
Distribution of datapoints for all features



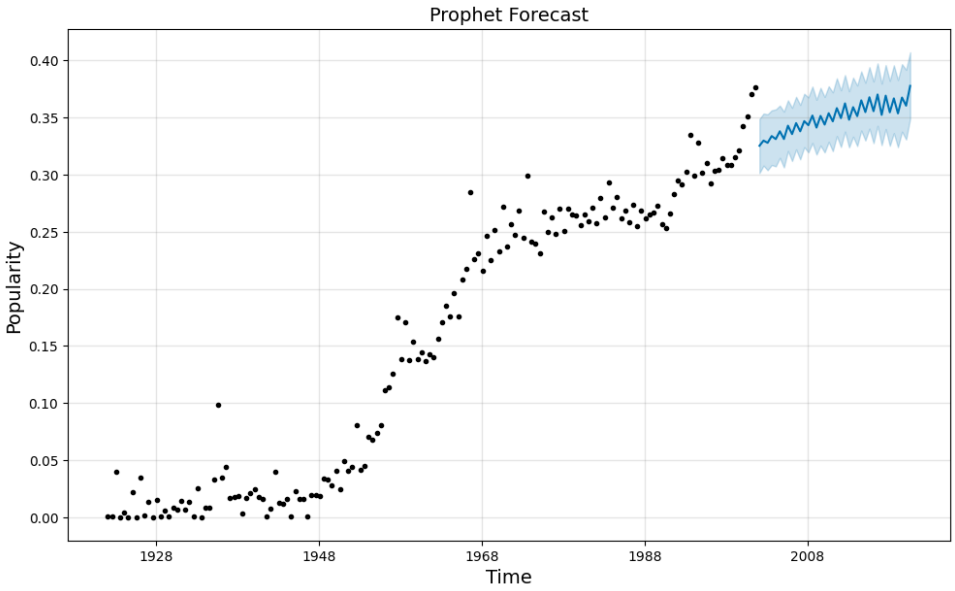
Boxplots of popularity for explicit and non-explicit content



Distribution of popularity after min-max normalization



Forecasts from SARIMAX



Forecasts from Facebook Prophet

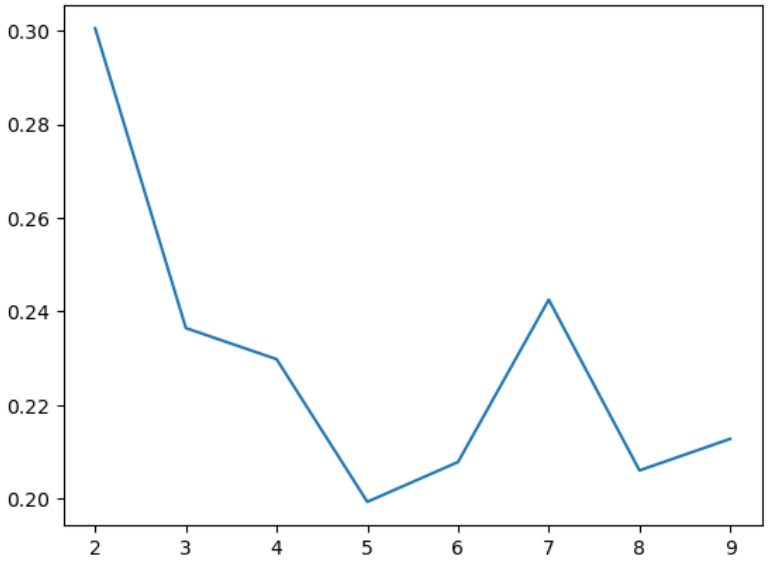


Figure showing the plot of Silhouette score vs the number of clusters. We observe the "elbow" for number of clusters = 5.