Predicting The Popularity of Kendrick Lamar's Tracks on Spotify

Problem Statement:

Spotify is one of the leading music streaming platforms globally, hosting millions of songs and attracting a vast user base. For my final project, I aim to develop a predictive model capable of estimating the popularity scores of Kendrick Lamar's songs on Spotify and analyzing the underlying trends and patterns influencing these scores. Popularity scores, ranging from 0 to 100, serve as a critical metric for understanding a song's success and listener engagement.

This model will be particularly beneficial for music producers, record labels, and artists seeking to optimize their strategies for song creation, promotion, and audience targeting. By identifying the key factors (e.g., feature, release year, energy, and tempo) that drive popularity, stakeholders can make data-driven decisions to produce music that resonates with listeners, maximize exposure, and allocate resources effectively.

The successful development of this model will not only improve predictive capabilities for song popularity but will also provide actionable insights into audience preferences and music trends. For instance, artists can tailor their sound and content to align with listener engagement, record labels can invest strategically in songs with high potential for success and streaming platforms can recommend songs more effectively, enhancing user satisfaction.

Furthermore, understanding how factors like song attributes, production quality, and trends evolve over time will empower industry stakeholders with a competitive edge in the rapidly evolving music industry.

Dataset Description:

Data for this project is sourced from Kaggle in csv format, providing comprehensive information about the various features of Kendrick Lamar's tracks. The dataset includes categorical variables: Explicit (whether the track is explicit or not), is_album_track (if the song is from an album) and is_feature (if the song has a Kendrick feature). I'll be using these three categorical variables, release year of the tracks and few numeric variables chosen from the dataset based on their correlation with popularity. Challenges may arise in constructing an accurate regression model due to the inherent volality of the data. Track popularity scores (ranging from 0–100) depend on listener opinions and behavior, making them inherently volatile and difficult to predict with high precision. Identifying the most impactful predictors (e.g., loudness, tempo) is crucial to building an accurate regression model. The dataset provides a foundation for analyzing track

characteristics and their relationship with popularity. Despite challenges related to subjectivity and data quality, I believe certain variables will serve as strong predictors for track popularity.

Exploratory Data Analysis

The spread of Popularity:

The distribution of track popularity is highly skewed to the left, with a significant number of tracks having very low popularity scores near 0. The mean popularity is 42.7, indicated by the red dashed line. A smaller portion of tracks achieves high popularity scores (above 60), indicating these are less frequent but more successful. The data suggests a large disparity in track performance, with most tracks struggling to gain significant traction. The graph alongside the standard deviation of 24.01 suggests that the data is widely spread and skewed.

Histogram of Numerical Data:

1. Tempo: Right-skewed distribution; most tracks have tempos between 90–120 BPM.
2. Loudness: Normal distribution centered around -10 dB, indicating most tracks are moderately loud.
3. Speechiness: Right-skewed; majority of tracks fall between 0.1–0.4, showing minimal speech content.
4. Acousticness: Heavily skewed; most tracks have low acousticness (< 0.2), indicating dominance of non-acoustic tracks.
5. Instrumentalness: Highly skewed; majority of tracks have near-zero instrumentalness.
6. Duration (min): Peaks around 4–5 minutes, with very few tracks extending significantly longer.
7. Liveness: Skewed distribution; most tracks have liveness values below 0.2, suggesting studio-recorded tracks dominate.

The data exhibits notable skewness across features like speechiness, acousticness, instrumentalness, and liveness, suggesting uneven spread, with most tracks falling into specific ranges.

Bivariate Analysis:

Scatterplots:

1. Popularity vs. Tempo:No clear relationship; popularity is spread across all tempo values.

2. Popularity vs. Loudness: Slight trend: Tracks with higher loudness (closer to -5 dB) tend to have higher popularity.
3. Popularity vs. Speechiness: No strong correlation; most popular tracks cluster between 0.1–0.3 speechiness.
4. Popularity vs. Acousticness: Negative trend: Tracks with low acousticness are more popular.
5. Popularity vs. Instrumentalness: Highly skewed; popular tracks generally have very low instrumentalness.
6. Popularity vs. Liveness: No clear relationship; popularity is spread across all liveness values, though clustering occurs below 0.3.

Features like loudness and acousticness show slight trends, while other features exhibit weak or no correlation with popularity. Further analysis may be needed to identify significant predictors.

Correlation Matrix:

I made a correlation matrix for all the numeric variables. Based on the correlation between the predictor variables and the target variable: Popularity, I determined the variables that I'd be using for my Predictive Modeling. I decided to use energy, danceability, tempo and liveness given that they've the highest correlation values among all the variables. Alongside, I used feature, track and release year as they can impact the audience's reception.

Predictive Modeling

Baseline Model MSE:

I evaluated the success of each of my regression models by comparing it to the Baseline Model's Mean Squared Error. My Baseline Model has a MSE of approximately 574.73. This is calculated by predicting the mean popularity for all observations and comparing it to the actual popularity values.

Multiple Regression Model:

I chose to run a Multiple Regression to use the independent variables to predict the dependent variable as these predictors may have a collective influence on the popularity of the tracks. The multiple squared error for my training data is 329.78 while for my testing data it is 324.52. My testing data slightly outperformed my training data. The Multiple Regression Model also

significantly outperformed my Baseline Model as the MSE value for both my train and test data are lower than the Baseline MSE.

Album feature has the highest correlation of 0.47 becoming the most important predictor of popularity in this model. Release year and Danceability have a minimal positive effect with coefficients of 0.11 and 0.04. Tempo and Liveness have a minimal negative effect with coefficients -0.043339 and -0.017427 indicating a negative correlation.

The Multiple Regression Model has performed better than the Baseline Model because it leverages the relationships between the predictor variables (features) and the target variable (popularity), whereas the baseline model is a simplistic approach that uses the mean of the target variable as the prediction for all data points.

K-Nearest Neighbors (KNN) Regression:

I used K-Nearest Neighnors Regression next because unlike linear regression, which assumes linearity, normality, and homoscedasticity, KNN does not require these assumptions. KNN makes predictions based on the similarity of instances. For instance, if popularity is affected by the tempo, danceability or release year differently in clusters of observations then KNN can identify these local variations.

The Mutiple Square Error for my training data is 495.31 while the MSE for my testing data is 467.13. My testing data has performed slightly better than my training data. Both the training and testing MSE are lower than the Baseline Model which means the KNN model has performed better than the Baseline. However, the MSE for KNN model are higher than the MSE for multiple regression model hence my KNN model has performed worse than my multiple regression model. This might be because of KNN underfitting. KNN typically fits training data very closely with low bias. A high training error suggests underfitting, meaning the model fails to capture the patterns in the data. KNN also struggles in high-dimensional data where distances between points become less meaningful. Irrelevant or weak features (e.g., danceability, explicit) can dilute KNN's ability to find meaningful neighbors.

Release year is the most significant predictor for KNN as well, however its coefficient is 0.15 which suggests a weaker relationship as compared to the Mutltiple Regression Model. Energy, Liveness, and other features have very low or near-zero contributions.

Decision Tree Regression Model:

I use a Decision Tree because they can model complex and non-linear relationships between predictors and the target variables. They also have automatic feature selection where they assign importance to features and ignore irrelevant ones. The model is also easier to interpret as it

generates tree structure making it easy to understand and interpret decision rules. Decision trees also naturally handle handle feature interactions (e.g., combinations of feature and release year).

The plot of train and test Mean Squared Errors (MSEs) shows that the optimal max tree depth is 3, where the test MSE is the lowest (~297). Beyond this depth, the model overfits, as evidenced by the decreasing train error and increasing test error. The Train MSE for this model is 262.28 while the Test MSE is 297.93. The small gap between train and test errors at the optimal depth indicates a good balance between bias and variance, with the model generalizing reasonably well.

The decision tree reveals the hierarchy of splits: The first split is based on whether the track is a feature (is_feature), followed by conditions on release_year and energy. Terminal nodes show squared errors and predicted popularity values, providing interpretable thresholds for predictions. The most important features (based on permutation importance) are: is_feature_False (0.28), is_feature_True (0.27), release_year (0.25). These suggest that whether the track is a feature and the release year significantly influence popularity predictions. Other features like energy, tempo, danceability, and liveness have very low or zero importance, indicating they contribute minimally to the model.

The model has performed better than the baseline, multiple regression and KNN. This can be due to the fact that that Decision Tree Modeling automatically identifies and prioritizes the most important features. In this case, it prioritized feature, and release year and ignored weaker features like danceability and explicit. Unlike KNN, which is sensitive to irrelevant features and distances, the Decision Tree ignores unimportant features during splits. This reduces noise and improves performance.

Random Forest Regression Model:

For my last model, I extended the decision tree model into a random forest regression model. By aggregating predictions from multiple trees, Random Forest reduces overfitting compared to a single Decision Tree. The grid search determined the best hyperparameters as: max_depth = 10 which imits the depth of the trees, reducing overfitting. n_estimators = 200: The number of trees in the forest, providing robustness. The training mean squared error for the model is 51.97 and the Testing MSE is 260.36. The large difference between Training MSE (51.97) and Testing MSE (260.36) occurs due to overfitting. This can be due to Model Complexity - 200 trees with max depth 10 might overfit the training data or small or noisy data and high variance in target variable (popularity).

The top features for this model are track feature, release year and energy with track feature having a coefficient of 0.57. The model confirms the importance of is_feature and release_year, aligning with earlier models. Other features like danceability and tempo have low but non-zero contributions.

The Random Forest Model has performed better than all the other models. This can be due to Random Forest aggregating predictions from multiple decision trees, reducing the variance that single Decision Trees suffer from. By limiting max_depth to 6 and using 200 trees, Random Forest achieves a balance between underfitting and overfitting. It ignores unimportant features during tree splits, which improves accuracy. It effectively captures non-linear interactions between is_feature, release_year, and other features.

Summary and Further Discussion:

In my analysis of popularity scores, all models demonstrated improved performance over the baseline predictor, emphasizing their significance in capturing patterns within the data. The models ranked in terms of performance are as follows: Random Forest Regression, Decision Tree Regression, Multiple Linear Regression, and K-Nearest Neighbors Regression.

Key Findings

The **Random Forest Regression** model emerged as the most effective, showcasing the lowest testing MSE and the best predictive capabilities. Its ensemble nature, which aggregates multiple decision trees, allowed it to capture **non-linear relationships** and **feature interactions** effectively, leading to superior performance. Across all models, the following features demonstrated the highest influence:

- is_feature: Strongest predictor of popularity, indicating its substantial role in audience perception.
- release_year: Highlighted the trend of newer releases receiving more attention.
- energy: Showed moderate importance in shaping predictions.

Other features like danceability, tempo, and explicit exhibited minimal importance and could be excluded in future refinements.

Model Comparisons:

Model	Training MSE	Testing MSE	Overfitting
Linear Regression	239.78	324.52	Slight overfitting
KNN Regression	495.31	467.13	Underfitting
Decision Tree	262.20	297.93	Minimal overfitting

Random Forest	**51.97**	**260.36**	Best generalization

The Random Forest Model's robust performance highlights its suitability for predicting popularity scores by leveraging ensemble methods to capture intricate patterns. The identification of is_feature and release_year as pivotal predictors aligns with the importance of new releases and specific attributes in influencing audience perception.

Next Steps/Improvements

To further enhance the predictive capabilities of the models and gain deeper insights into popularity scores, the following improvements can be incorporated:

1. **Incorporate Additional Features**:
    - **Data on Listener Demographics**:
      Adding demographic data (e.g., age, gender, region) can provide insights into how audience segments impact popularity.
    - **Streaming Metrics**:
      Metrics such as total streams, skips, and playlist inclusions could highlight actual listener engagement and song popularity.
2. **Text Feature Extraction**:
    - **Song Titles and Lyrics Analysis**:
      Applying **Natural Language Processing (NLP)** techniques to analyze song titles or lyrics can help identify keywords and themes that contribute to popularity.
    - For example, sentiment analysis on lyrics could reveal correlations between emotions and audience reception.
3. **Temporal Trends**:
    - Incorporate time-series data to analyze how popularity evolves over time, capturing trends influenced by seasonal events or viral marketing.
4. **Producer and Artist Insights**:
    - Include data on **producers**, **artists**, and **labels** to examine the impact of key individuals or entities on song success.
5. **Financial Metrics**:
    - Integrating financial data, such as production costs and marketing budgets, could reveal relationships between resource investment and song popularity.