Introduction

In today's digital music landscape, established indicators such as radio airplay and album acquisitions do not individually ascertain a composition's triumph. How music achieves prevalence has been altered by platforms such as Spotify, TikTok, and YouTube, frequently introducing ever evolving tracks or rekindling prior melodies into common recognition. Our undertaking, impacted by this concrete alteration, aspired to investigate Spotify's Track Score and inspired us to investigate how external platforms bear upon total streaming triumph. We aspired to dig into which factors most considerably contribute to a song's modern-day popularity with the expansion of cross-platform interactions between social media and music streaming services.

Our project was shaped via many necessary data mining questions. Spotify's Track Score was examined in order to determine a measurable effect deriving from external platforms, such as TikTok and YouTube. We also endeavored to ascertain the platform that correlates most strongly with Spotify's track score, to see whether older songs going viral on TikTok or YouTube cause songs to regain popularity, and to see if the recency of a song relates to its influence and success.

We favored this attempt due to individual incentives. The convergence of data science and the swiftly changing music sector interests us. A TikTok trend helped Lady Gaga's "Bloody Mary" re-enter Spotify's Top 200 by over a decade after release, indicating that a track's age no longer curtails potential success, as well as giving prominent examples. Useful anticipatory understandings have been formulated for artists, labels, and industry professionals, and machine learning techniques were employed to reveal latent trends inside this energetic environment.

Several technical challenges from this environment were encountered during the project's duration. The data aggregation that occurs across platforms was indeed quite difficult. Distinctive platforms outline interaction via different measurements, incorporating listens, shares, endorsements, and comments. Wide-ranging cleansing was necessary for the dataset because of irregularities such as absent values, incorrectly arranged numerals, and erratic publication dates. Matching viral trend chronologies to Spotify stream alterations introduced difficulties because streaming augmentation and platform function are not always synchronous. This, along with how music sometimes circulates over time, can lead to discrepancies in virality across multiple platforms.

With this, we employed a quite detailed preprocessing workflow involving cleaning of data, standardization of features, and normalization of values for addressing these challenges. After applying a collection of machine learning models, we categorized songs as either viral or non-viral using logistic regression, forecast continuous streaming results via random forest regression, and discovered latent correlations inside the data through principal component analysis (PCA). Furthermore, to increase predictive performance as well as diminish error, we employed more advanced models such as feedforward neural networks (FNN) and XGBoost.

Each of the approaches demonstrated our very strong results. In identifying viral songs, logistic regression exhibited close to perfect classification performance, with a recall of 100% and a precision of 97%. The random forest regression model attained an R^2 score of 0.95. XGBoost additionally improved results, exhibiting an R^2 score of 0.9569. The number of playlists and the extent of playlist reach were key components impacting a song's triumph, per the PCA analysis, while the TikTok and YouTube external platform views exhibited a reduced, yet sufficient, consequence.

To summarize, our observations substantiated that cross-platform virality impacts a song's revival. Nevertheless, placement of a song across many playlists determines Spotify's success with the greatest criticality. A few important understandings of those digital avenues influencing contemporary musical prominence can be ascertained via our project through data mining.

Evaluation Methodology

Our group used a dataset of all of the most streamed songs of 2024 from Kaggle.com. This data included 29 columns of data with thousands of entries in each column. This massive dataset gave us a lot of information to sort through, however, we had to review and clean the data as a few entries were missing information i.e. views or releases, or duplicated tracks which impacted our goal for relations to . While this data took some time to clean, this was the only underlying issue with the data provided.

First, we used Root Mean Squared Error (RMSE) as one of our evaluation metrics for the neural network model. RMSE allowed us to measure the average squared difference between the actual and predicted Spotify stream counts. We chose RMSE because it heavily penalizes larger errors, which is important in real-world scenarios where mispredicting a song's popularity by millions of streams could have serious marketing and financial consequences. Accurate forecasting is critical for planning release strategies and allocating promotional resources effectively.

In addition to the RMSE, we used several other models like Feedforward Neural Network, Random Forest, and XGBoost. We used these to compare the Spotify popularity to the other major platforms like YouTube or TikTok, and we then calculated the average streams for each release month. This helped us uncover important patterns that one model alone might not fully reveal. These insights are vital and can be useful in real-world music industry decisions, such as determining what platform(s) to release on and whether it would affect a song's exposure, as well as identifying optimal times of the year to release new music for the best results.

We also implemented a predictive scenario by simulating how a hypothetical song would perform if released in different months. By adjusting the release month feature while keeping other characteristics constant, we were able to predict which month would likely yield the

highest number of streams hypothetically by doing this. We felt that this kind of what-if analysis mirrors real-world planning, where labels and artists often need to decide the best timing for a release based on potential audience engagement.

Overall, we chose these metrics and algorithms because they not only helped us measure model performance but also provided actionable insights that could be applied in real-world decision-making. If we had more time, we discussed that it would be useful to add additional metrics like R^2 (to show how well the model explains the variance in stream counts) and MAE (for a more interpretable measure of average prediction error).

Data Mining Task

The core objective of this project was to explore and quantify the influence of external platforms- specifically TikTok and YouTube—on the success of songs on Spotify, measured by Spotify's Track Score. To accomplish this, we framed several data mining questions and approached them through a combination of regression and classification models. The aim was to determine which features most significantly drive a song's performance and whether these metrics could be used to predict success.

Our input data was derived from a Kaggle dataset featuring the most-streamed songs of 2024. It included 29 columns spanning platform engagement metrics and metadata such as Spotify streams, YouTube views and likes, TikTok view estimates, playlist appearances across Apple Music, Deezer, and Amazon, Shazam counts, release dates, and explicit content indicators. The output of our data mining efforts was twofold: first, a regression task to predict a continuous Track Score for each song; and second, a classification task where songs were binned into popularity levels—Low (0–30), Medium (30–60), and High (60–750)—based on their Track Score. This binning strategy allowed us to simulate how songs might be categorized by industry analysts or marketing teams based on performance tiers.

We developed the following key questions to guide our data mining process:

- Which external platform correlates most strongly with Spotify success?
- Can sudden increases in YouTube or TikTok engagement predict an uptick in Spotify streams?
- Do older songs gain renewed popularity through viral exposure on these platforms?
- Does the recency of a track's release date influence its performance on Spotify?

These questions helped us frame the relevance of our analysis from both an academic and an industry perspective.

To answer these questions, we implemented both regression and classification models. For regression, we used Random Forest and XGBoost to predict exact Track Scores based on platform and metadata features. For classification, we applied logistic regression and a Feedforward Neural Network (FNN) to categorize songs into discrete popularity levels. Each model was trained on a cleaned and normalized dataset, with standardization and log transformation applied to features where appropriate. Grid search and cross-validation were

used to tune hyperparameters and evaluate model robustness. We also used Principal Component Analysis (PCA) for exploratory analysis and dimensionality reduction, helping us identify the most influential features driving Spotify success.

Throughout the project, we encountered several technical challenges. One major issue was the inconsistency and incompleteness of TikTok data, which limited some analyses. Additionally, platform metrics were recorded in different units such as views, plays, and likes, requiring careful normalization. There was also no temporal metadata associated with trend events, making it difficult to align viral moments with changes in Spotify performance. Finally, because some platforms naturally dominate in volume (e.g., Spotify stream counts), we needed to normalize by total listens to better assess relative influence across platforms.

Addressing these challenges required detailed preprocessing and feature engineering. Ultimately, this setup allowed us to train high-performing models and uncover meaningful relationships between cross-platform engagement and Spotify success. Our analysis not only highlighted predictive signals but also revealed structural insights about how virality and platform reach intersect to shape modern music performance.

Technical Approach

In this project, we focused on evaluating the performance of three machine learning models: a Feedforward Neural Network (FNN), Random Forest, and XGBoost. The FNN, a type of deep learning model, was used primarily as an exploratory approach to assess how well a relatively simple neural network architecture could handle our regression task. Given the nature of our dataset, which includes tabular, non-sequential data with limited size, FNNs are not typically the most effective choice, as they often require large datasets and benefit more from spatial or temporal structure (as in image or time-series data). As a result, we did not expect the FNN to outperform more structured, ensemble-based models.

In contrast, Random Forest and XGBoost are ensemble learning algorithms that are well-suited for tabular data and have demonstrated strong performance in both classification and regression tasks. Random Forest builds multiple decision trees using bootstrap sampling and averages their outputs, reducing overfitting and variance. XGBoost, on the other hand, employs gradient boosting to build trees sequentially while optimizing for a chosen loss function with regularization, often leading to superior predictive accuracy. Because both models are designed to handle heterogeneous features and non-linear relationships effectively, we anticipated that they would outperform the FNN in both accuracy and robustness across our prediction task.

The FNN used in this project was designed with a straightforward, three-layer architecture optimized for regression on tabular data. The input layer received the scaled feature vector, consisting of 13 numerical features derived from streaming and engagement metrics across various music platforms. The first hidden layer contained 128 neurons, followed by batch normalization and a ReLU activation function to promote non-linearity and stabilize

learning. A dropout layer with a rate of 0.3 was applied to prevent overfitting by randomly disabling a fraction of neurons during training. This pattern was repeated in the second hidden layer, which consisted of 64 neurons, again followed by batch normalization, ReLU activation, and dropout. The final output layer was a single neuron with no activation function, appropriate for a regression task where the model outputs a continuous Track Score. The model was trained using the Adam optimizer with a learning rate of 0.001 and used L1 loss (mean absolute error) as the loss function, which is more robust to outliers and better reflects real-world prediction accuracy. The training was conducted for 1,000 epochs with periodic logging of the loss to monitor convergence. This architecture provided a flexible and interpretable baseline for assessing neural network performance on the dataset.

The Random Forest model was implemented as an ensemble learning algorithm optimized for regression on structured tabular data. It was trained using 300 decision trees with a maximum depth of 12 on a standardized dataset consisting of 13 engineered and cleaned numerical features. These features represented various indicators of a track's streaming performance, such as Spotify Playlist Reach, YouTube Views, and TikTok engagement. Unlike neural networks, Random Forest does not require batch normalization or activation functions, making it more straightforward to train on datasets of moderate size. It inherently handles nonlinear interactions between features and is robust to outliers, making it a strong candidate for this task. The model was evaluated using standard regression metrics and visual diagnostics to assess performance.

The XGBoost model was implemented as a gradient-boosted decision tree regressor, optimized through grid search cross-validation to select the best combination of learning rate, number of estimators, and tree depth. As an advanced ensemble method, XGBoost is particularly suited for structured, tabular data with complex, non-linear relationships. The model was trained on a scaled dataset of 13 numerical features derived from music streaming platforms and engagement metrics, with hyperparameter tuning performed using a 5-fold cross-validation setup to maximize the R^2 score. Once the optimal model was selected, it was evaluated using both regression and classification metrics to measure its performance.

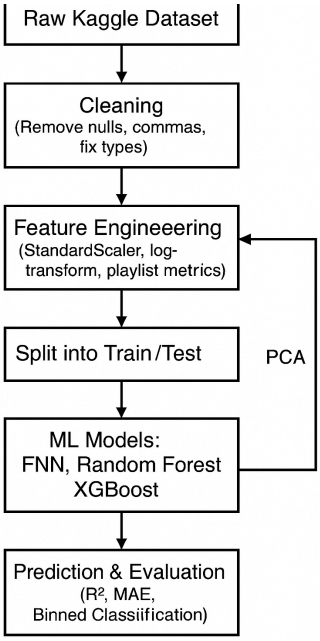

Figure 1: Workflow of how we used the data set

Results and Discussion

First, we wanted to perform PCA on our data to see how much of the variance was related to each of the different features in our dataset. With this information, we were going to change how we approach our data mining tasks. After preprocessing the data and getting all the features that meet the specific threshold of non-null values, we are left with these features:

```
# Step 1: Select relevant features (excluding Track Score)
feature_cols = [
    'Spotify Streams', 'YouTube Views', 'TikTok Views',
    'Apple Music Playlist Count', 'AirPlay Spins', 'Shazam Counts'
]
```

Figure 2: List of features that were used for PCA

These remaining features are being analyzed using PCA to see how each of them affects the variance of the track score. After standardizing and transforming the data for PCA, we are left with these results:

```
Correlation between PCA components and Track Score:
PC1    0.387599
PC4    0.225510
PC2    0.121805
PC6    0.104000
PC5   -0.070945
PC3   -0.104347
Name: Track Score, dtype: float64
```

Figure 3: Results of PCA

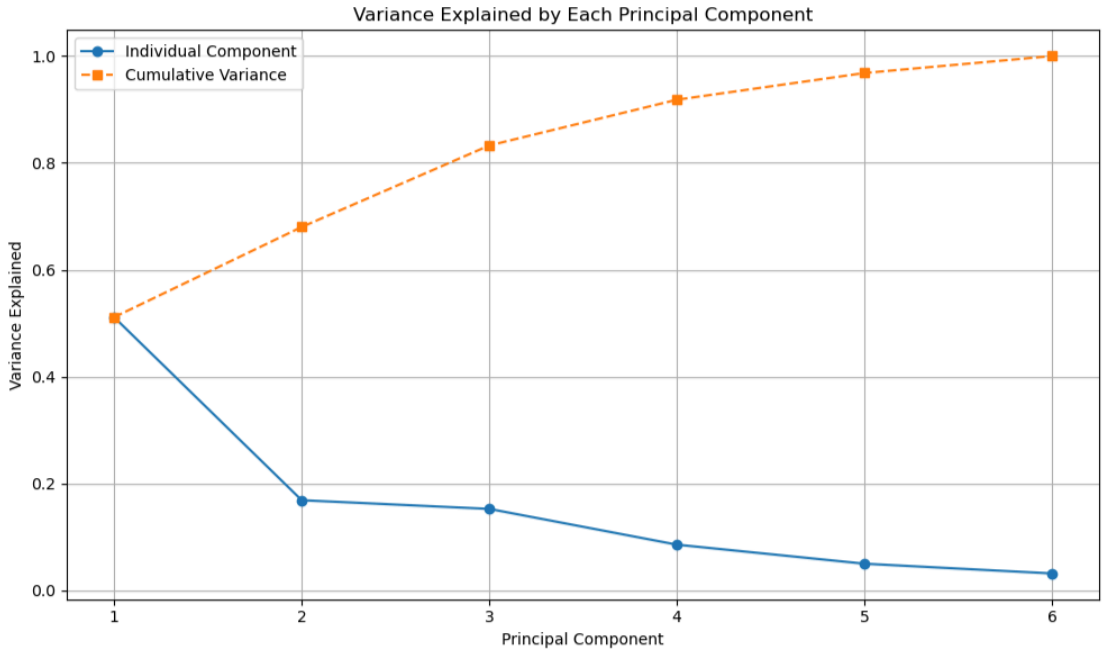

Figure 4: Graphical representation of the contribution of each principal component

While the information here is valuable, it doesn't mean anything to us since these components have no meaningful labels and aren't related to any of the features yet. Instead, we are more interested in PC1 and PC2, as they are responsible for the majority of the variance. We are determining this with a loading value, which represents the correlation or contribution of a feature to a principal component.

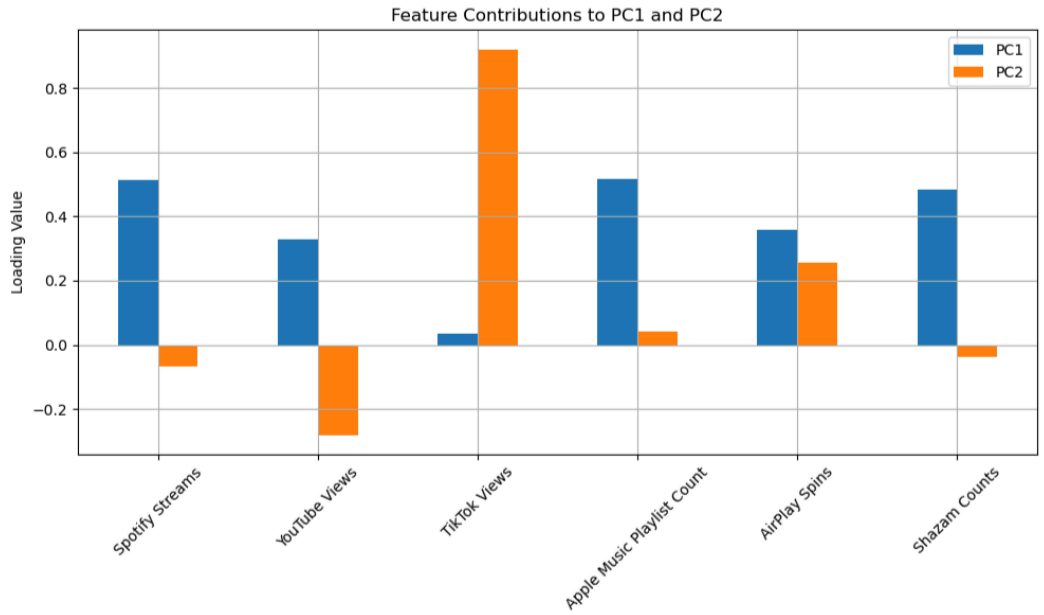

Figure 5: contribution of PC1 and PC2 to variance on track score

Looking at the graph, we can conclude that TikTok views are most likely related to PC2. As for PC1, it is not apparent which feature it relates to, as Spotify Streams, Apple Music Playlist Count, and Shazam count all have similar loading values for all of the features with similar loading values for PC1, which captures overall streams across all the different platforms in a conventional listening sense. While PC2 captures a different medium, being TikTok, which isn't really a platform to listen to songs, but is still a good metric for how much a song is trending.

In the PCA analysis, what worked well was the clear separation of feature contributions across the first two principal components, particularly the strong influence of Spotify Streams and TikTok Views. This was effective because PCA successfully captured the dominant patterns of variance in the dataset, and the loadings plot provided intuitive visual insight into which features drive overall popularity (PC1) versus viral or platform-specific trends (PC2). The transformation was especially meaningful given that the dataset was standardized and log-transformed, ensuring that skewed distributions and differing scales did not distort the component directions. However, one limitation was the interpretability of components beyond PC1 and PC2, as they captured less variance and offered less clear meaning. Additionally, PCA assumes linear relationships and orthogonality, which may not fully capture complex, nonlinear interactions between features (e.g., compounding effects across platforms). As a result, while PCA was helpful for initial exploration and dimensionality reduction, it may not reveal deeper, causal insights without complementary analysis.

```
Training Model...
Epoch 0, Loss: 43.1514
Epoch 50, Loss: 41.3385
Epoch 100, Loss: 39.7164
Epoch 150, Loss: 37.5127
Epoch 200, Loss: 34.5257
Epoch 250, Loss: 30.7437
Epoch 300, Loss: 25.7554
Epoch 350, Loss: 20.5714
Epoch 400, Loss: 15.8877
Epoch 450, Loss: 12.1128
Epoch 500, Loss: 10.4999
Epoch 550, Loss: 9.5265
Epoch 600, Loss: 9.1191
Epoch 650, Loss: 8.7568
Epoch 700, Loss: 8.4227
Epoch 750, Loss: 8.2459
Epoch 800, Loss: 8.3753
Epoch 850, Loss: 8.2529
Epoch 900, Loss: 8.2928
Epoch 950, Loss: 7.8300

Final Test MAE: 9.1044
Final Test MSE: 540.8133
R² Score: 0.7044
Root MSE: 23.2554
Mean Absolute Percentage Accuracy: 83.35%
Neural Network R²: 0.7044, MAE: 9.10, Accuracy: 83.35%
```

Figure 6: Results of FNN with loss values at every 50 epochs and final results

The Feedforward Neural Network (FNN) was trained over 1,000 epochs, progressively reducing the training loss from 43.15 to 7.83, indicating successful convergence. Upon evaluation, the model achieved a Mean Absolute Error (MAE) of 9.10, a Root Mean Squared Error (RMSE) of 23.26, and an R^2 score of 0.704, meaning the model explains about 70.4% of the variance in the track scores. The Mean Absolute Percentage Accuracy (MAPA) was 83.35%, suggesting reasonably strong performance on the regression task.

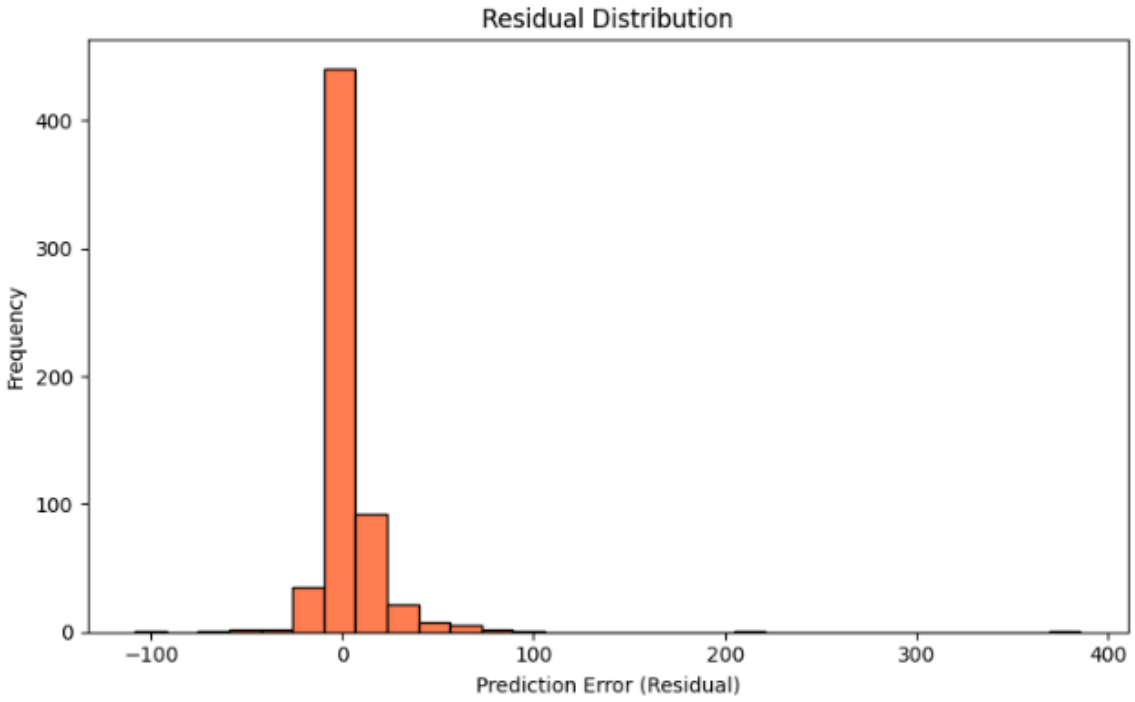

Figure 7: Plot of residual distribution

The residual distribution (Figure 7) is centered around zero with a right-skewed tail, indicating that most predictions are close to the actual values, but a few outliers resulted in significant overestimation. This suggests the model performs reliably for the majority of data points but struggles with edge cases, likely high-streaming outliers that skew error values.

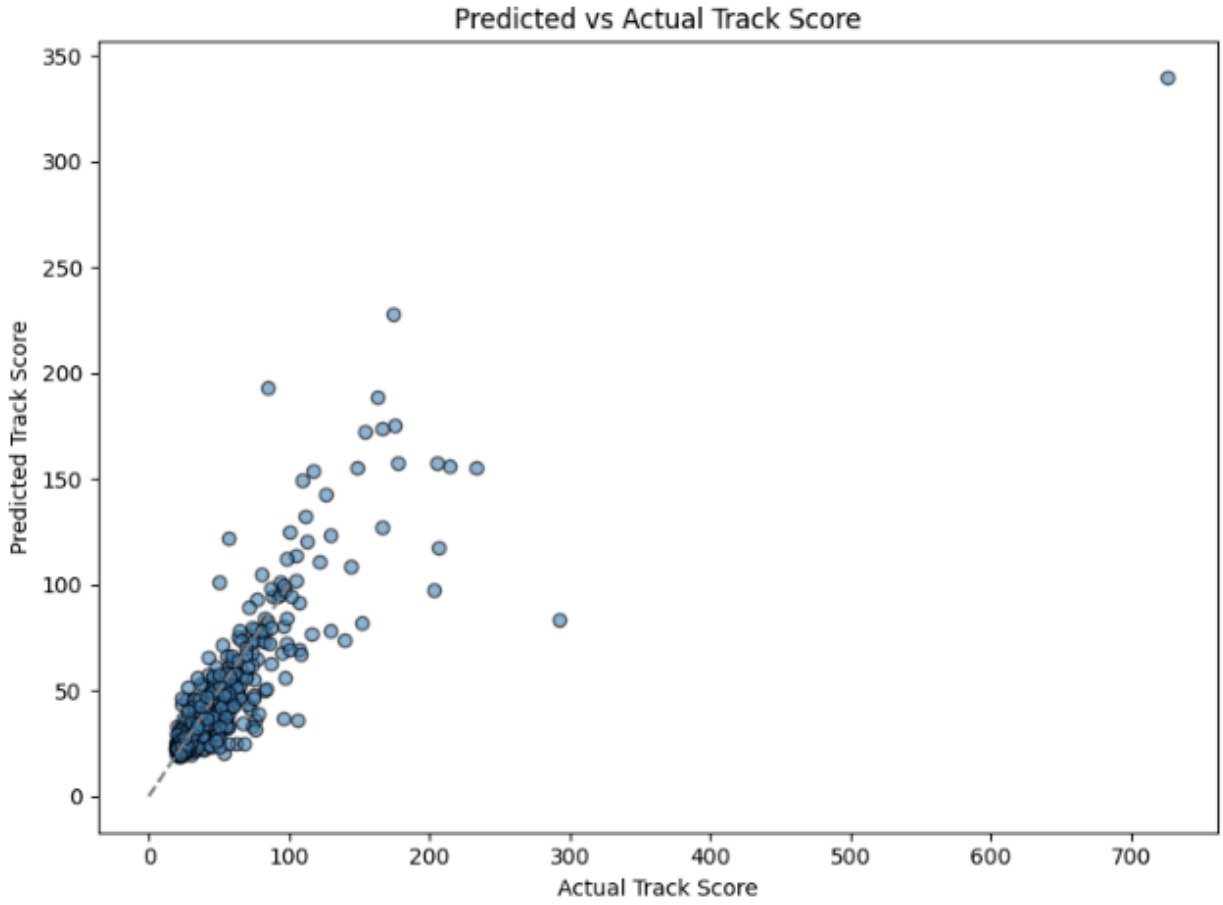

Figure 8: Plot of predicted score and actual score

The Predicted vs Actual plot (Figure 8) further confirms this: while most predictions lie near the ideal diagonal line (indicating strong correlation), a few extreme outliers deviate significantly, especially at higher actual scores. This underperformance on high-value predictions could be due to the model being trained on a data distribution where most values are concentrated in the lower to mid ranges, making it harder to generalize in the sparse, higher-value regions.

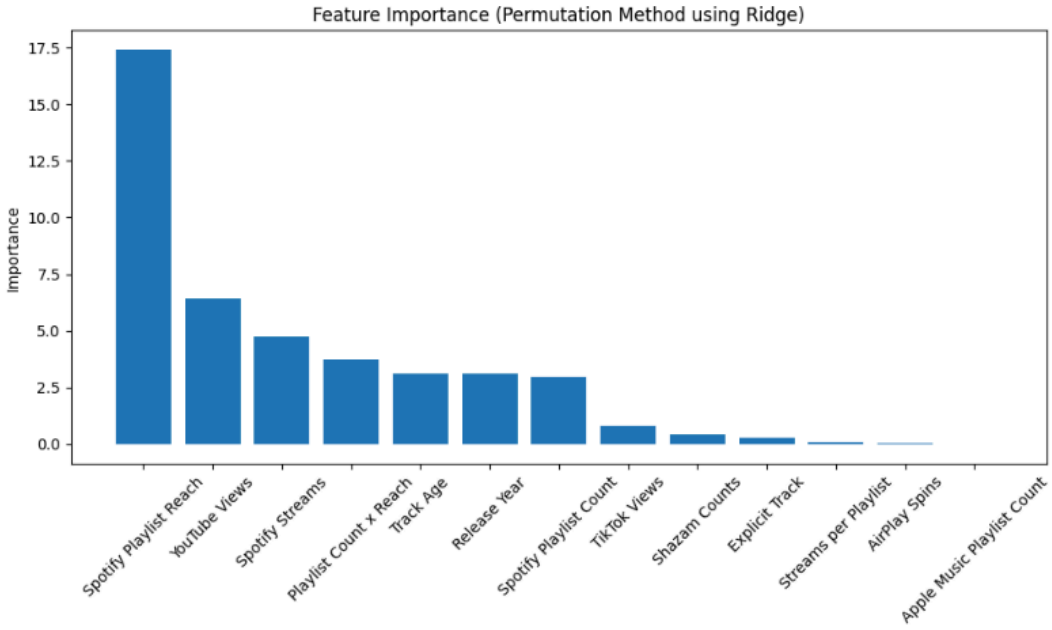

Figure 9: Feature Importance plot showing the degree of correlation of each feature to the track score

The feature importance plot (Figure 9), derived using permutation importance via Ridge regression, reveals that Spotify Playlist Reach, YouTube Views, and Spotify Streams were the most influential features for predicting the Track Score. Features like TikTok Views, Streams per Playlist, and Apple Music Playlist Count had minimal impact. This suggests that reach-based and traditional streaming metrics are more predictive of a song's overall performance than viral or auxiliary engagement metrics.

The Feedforward Neural Network (FNN) demonstrated several strengths in modeling the track score prediction task. Notably, it was able to effectively minimize training loss over 1,000 epochs and achieved a solid R^2 score of 0.704, indicating that it captured a substantial portion of the variance in the data. The use of feature engineering techniques, such as log transformation of skewed metrics and standardization, contributed significantly to the model's learning efficiency and stability. Additionally, the residual distribution showed that the majority of predictions were close to the actual values, affirming that the model generalized reasonably well on typical cases. The feature importance analysis using permutation methods also provided meaningful insights, highlighting that metrics like Spotify Playlist Reach, YouTube Views, and Spotify Streams were the most impactful features, which aligned well with domain expectations.

However, the FNN also had some limitations. It struggled with extreme outliers, particularly tracks with very high scores, leading to large prediction errors in those regions. This was evident in both the long right tail of the residual distribution and the scatter plot, where several high-value points deviated significantly from the ideal prediction line. This issue likely stems from the imbalanced nature of the dataset, where such extreme values were underrepresented, causing the model to generalize poorly in those areas. Furthermore, while

the FNN was adequate, neural networks are not always the most effective choice for tabular datasets, especially when the dataset size is limited. Ensemble methods like Random Forest and XGBoost, which are better suited for structured data, may offer improved performance and robustness with less sensitivity to outliers and better feature interpretability. There was also not as much testing data, as it doesn't use cross-validation like the Random Forest and the XGboost models.

```
Random Forest R²: 0.9501, MAE: 5.52, Accuracy: 85.66%
```

Figure 10: Final R^2, MAE, and Accuracy of Random Forest Model

The Random Forest model achieved a high R^2 score of 0.9501, indicating that it explained over 95% of the variance in track scores. The Mean Absolute Error (MAE) was 5.52, and the Mean Absolute Percentage Accuracy (MAPA) reached 85.66%, suggesting that predictions were both accurate and consistent across a wide range of scores.

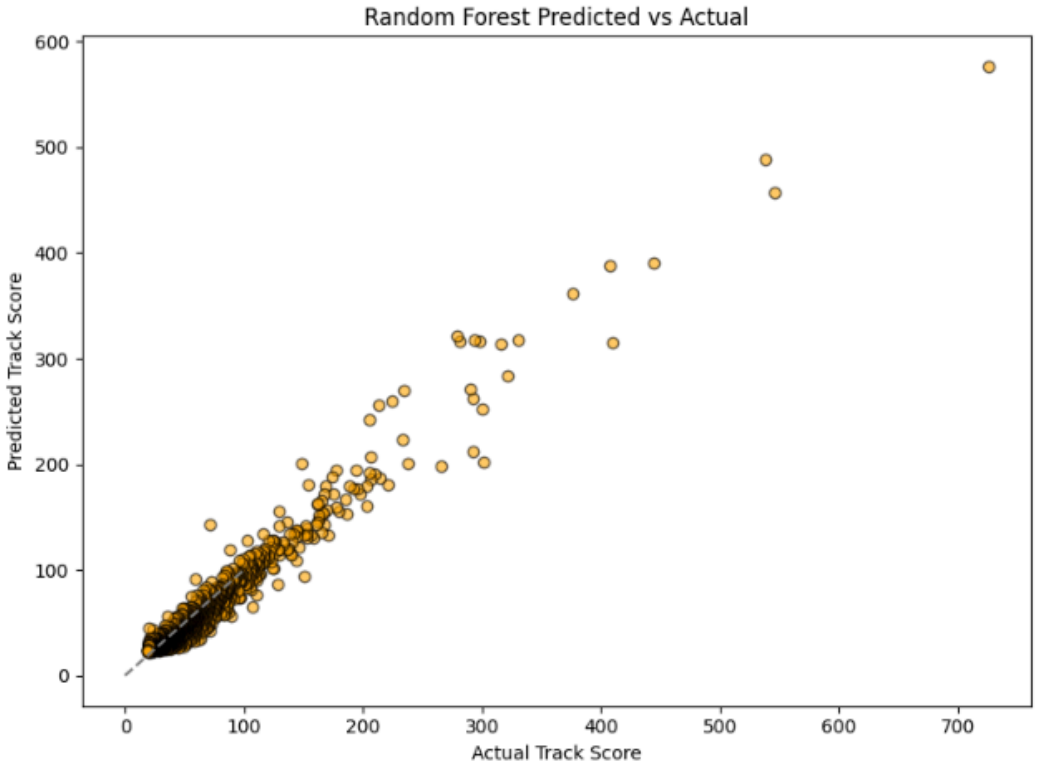

Figure 11: Random Forest Predicted vs Actual Plot

The predicted vs. actual plot (Figure 11) shows most points clustered along the ideal diagonal line, confirming strong alignment between predicted and actual track scores. Unlike the

FNN, the Random Forest model managed to maintain this alignment even for tracks with higher scores, reflecting its ability to generalize across the entire range of the target variable.

Random Forest Classification Report:

	precision	recall	f1-score	support
High	0.93	0.86	0.89	470
Low	0.88	0.83	0.86	1478
Medium	0.75	0.82	0.78	1118
accuracy			0.83	3066
macro avg	0.85	0.84	0.84	3066
weighted avg	0.84	0.83	0.83	3066

Figure 12: Classification Report for Binned Score Categories

To further evaluate categorical prediction quality, track scores were binned into three classes: Low, Medium, and High. The classification report (Figure 12) shows strong results, with the “High” class achieving a precision score of 0.93 and an F1-score of 0.89. The “Low” class also performed well, while the “Medium” class had slightly lower metrics (F1-score of 0.78), likely due to it being a transitional category between clearly defined low and high extremes. Overall accuracy across the bins was 83%, with balanced performance across categories.

The Random Forest model demonstrated substantial advantages for this task. It achieved higher predictive accuracy than the neural network and handled outliers more effectively. Its tree-based architecture naturally models non-linear feature interactions and does not require intensive parameter tuning. It also performed well in both regression and classification evaluations, making it highly suitable for the nature of the dataset.

However, one area where it underperformed slightly was in predicting Medium-scoring tracks, which likely resulted from less distinct boundaries between feature values in that range. Additionally, while the model performed well, its internal decision-making is less interpretable without tools like SHAP or feature importance scores, making it more of a black-box compared to linear models. Despite this, Random Forest proved to be a reliable, high-performing model for predicting music track success across multiple metrics.

```
XGBoost R²: 0.9522, MAE: 5.88, Accuracy: 84.82%
```

Figure 13: Final R^2, MAE, and Accuracy of XGBoost Model

The XGBoost model achieved an R^2 score of 0.9522, a Mean Absolute Error (MAE) of 5.88, and a Mean Absolute Percentage Accuracy of 84.82%. These results are highly competitive and on par with the Random Forest model, confirming that XGBoost can capture a large portion of the variance in track score outcomes while maintaining low prediction error.

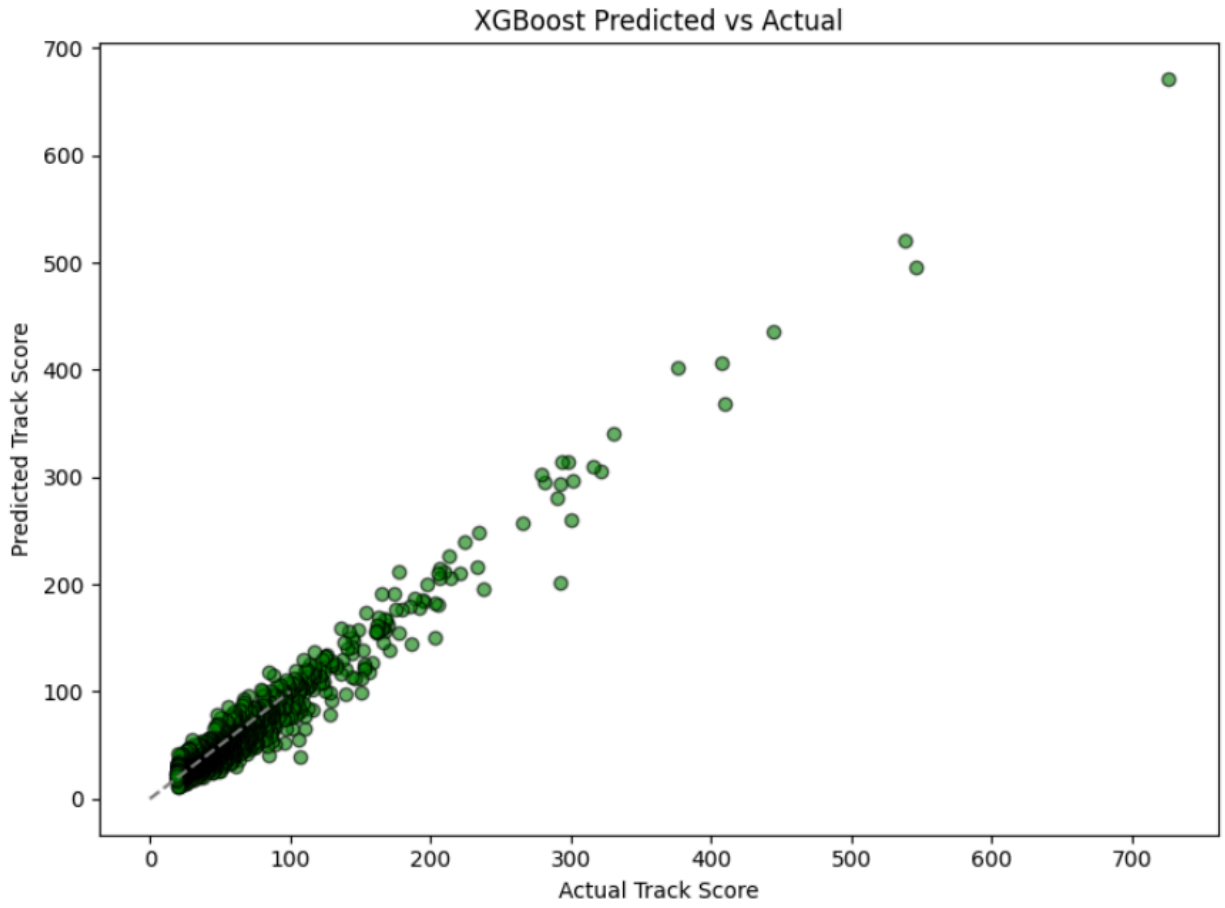

Figure 14: XGBoost Predicted vs Actual Plot

The predicted vs. actual plot (Figure 14) shows that most predictions are closely aligned with the ideal diagonal line, with a strong linear trend even in higher score ranges. A few minor deviations at the upper extremes suggest slight underperformance on the most popular tracks, but overall, the model displays strong predictive capability across the full range of track scores.

XGBoost Classification Report:

	precision	recall	f1-score	support
High	0.88	0.83	0.85	470
Low	0.87	0.79	0.83	1478
Medium	0.69	0.79	0.74	1118
accuracy			0.80	3066
macro avg	0.81	0.81	0.81	3066
weighted avg	0.81	0.80	0.80	3066

Figure 15: Classification Report for Binned Score Categories

Classification evaluation was performed by grouping track scores into Low, Medium, and High bins. As shown in Figure 15, the model achieved strong performance for the High class (precision = 0.88, F1-score = 0.85) and Low class (precision = 0.87, F1-score = 0.83). However, the Medium category again proved more challenging, with a precision of 0.69 and an F1-score of 0.74. This drop is consistent with trends seen in other models and likely reflects the fuzzy boundaries and less distinctive features among Medium-scoring tracks. The overall classification accuracy was 80%, with macro and weighted averages for precision and recall at 0.81.

The XGBoost model demonstrated exceptional performance in this project, balancing predictive accuracy with robust generalization. Its ability to optimize error sequentially through gradient boosting gave it a slight edge in handling subtle patterns in the data. What worked particularly well was the model's tuning flexibility and its capability to fit complex relationships with minimal overfitting. However, its slightly lower classification performance on the Medium category and its marginally higher MAE compared to Random Forest suggest that while powerful, the model may still struggle with ambiguous or borderline cases. Additionally, like Random Forest, XGBoost's internal mechanics are difficult to interpret without tools like SHAP or feature visualization. Nonetheless, XGBoost proved to be one of the top-performing models in this analysis and a strong candidate for music popularity prediction tasks.

In conclusion, the evaluation of the three models—Feedforward Neural Network (FNN), Random Forest, and XGBoost—revealed that ensemble tree-based models were far better suited to the task of predicting music track scores from streaming and platform engagement data. The FNN served as a baseline and demonstrated the ability to learn general patterns, but its performance was limited, achieving an R^2 of 0.704 and an MAE of 9.10. It particularly struggled with high-performing tracks and outliers, as it lacked the depth and flexibility needed

to model the non-linear and sparse nature of the dataset. Despite using dropout, batch normalization, and standardization, the neural network could not match the accuracy or consistency of the ensemble methods, which are inherently better equipped for tabular data.

The Random Forest and XGBoost models both achieved excellent predictive accuracy, with R^2 scores of 0.9501 and 0.9522, respectively, and similarly low MAEs (5.52 for Random Forest and 5.88 for XGBoost). Their scatter plots of predicted vs. actual track scores demonstrated a tight fit around the ideal diagonal, indicating that both models were able to capture the underlying relationships across a wide range of track scores, including at the higher extremes where the FNN underperformed. Classification analysis, where continuous scores were binned into Low, Medium, and High categories, also showed that both models performed well in detecting High and Low tracks, with some difficulty in the Medium range due to overlapping feature characteristics. Feature importance analysis further reinforced the dominant role of Spotify Playlist Reach, Spotify Streams, and YouTube Views, suggesting that sustained platform engagement is a stronger predictor of success than viral spikes from platforms like TikTok.

Despite their similar overall performance, the XGBoost model performed slightly worse than the Random Forest in terms of MAE and classification accuracy. This may be attributed to XGBoost's sequential learning process, which, while powerful, can be more sensitive to noise and overfitting, especially when not all patterns in the dataset are strong or well-separated. Additionally, XGBoost requires careful tuning of more hyperparameters such as learning rate and tree depth, and although grid search was used, it's possible that further optimization was needed to reach its full potential. In contrast, Random Forest's parallel tree construction and reliance on bootstrap aggregation (bagging) made it more stable and less prone to overfitting, resulting in more consistent performance across all score ranges. This stability gave Random Forest a slight edge in both predictive precision and robustness, solidifying it as the most effective model for this particular dataset and task.

Based on the results and feature importance analysis, we can infer that other digital platforms—particularly YouTube and to a lesser extent TikTok and Shazam—do have an impact on Spotify Streams, but their influence varies in strength and consistency. The most predictive features across all models were consistently Spotify Playlist Reach, Spotify Streams itself, and YouTube Views. This suggests that YouTube engagement correlates strongly with Spotify performance, likely because YouTube serves as both a discovery tool and an alternative streaming platform. High YouTube viewership may indicate broad listener interest or viral exposure, which can translate into more Spotify searches, playlist additions, and streams.

In contrast, platforms like TikTok and Shazam appeared less influential in the models' predictions, showing relatively low feature importance scores. This implies that while TikTok virality can spark initial interest or cause short-term spikes in attention, it may not consistently convert into sustained streaming volume on Spotify. Similarly, Shazam Counts—which reflect moments of discovery—may indicate awareness but don't necessarily guarantee user follow-through on Spotify.

Overall, these results suggest that platforms that support full-track listening and long-form content (like YouTube) tend to have a stronger predictive relationship with Spotify Streams than momentary engagement platforms (like TikTok or Shazam). While all platforms contribute to the broader digital ecosystem that influences a song's visibility, the most lasting and measurable effects on Spotify performance come from platforms that mirror its listening experience and allow for deeper user interaction.

Contribution

Abdur Islam: 23%

Aden Athar: 11%

Henok Gelan: 11%

Gabriel Compton: 11%

Justin Moos: 11%

Osaze Ogieriakhi: 11%

Kaleb Kebede: 11%

Jaeger Nelson: 11%