Most Streamed Spotify Songs 2024

Names: Eric Villanueva, Jamie Ann Visconde, Julian Ting, Kristen Ng

Introduction

The dataset includes 4,600 of the most-streamed songs on Spotify in 2024, demonstrating the types of music that people were listening to. Each entry includes the track title, album name, artist, release date, "All Time Rank," and a calculated "Track Score" to help compare each song's popularity. Along with Spotify's stream counts, the dataset includes each song's playlist appearances and audience reach. The data also includes metrics from other platforms, such as radio spins on AirPlay, playlist counts and reach on Deezer and Amazon Music, Pandora streams and station plays, Soundcloud streams, Shazam tag counts, and whether a track is labeled as explicit. Since the dataset provides detailed insights into each song's characteristics, popularity, and presence on various music platforms, it can be used as a valuable resource for music analysts, enthusiasts, and industry professionals.

Question 1: Using the metrics, is the Spotify Playlist Count or Spotify Playlist Reach the better predictor of Spotify Streams?

The original question in the analysis plan used TikTok views and initial Spotify streams as predictors. However, it was changed to Spotify playlist count and Spotify playlist reach to better understand the impact each has directly on the same Spotify platform.

Methods

Before creating the model, the data needed to be cleaned up. Any columns or rows with missing or unimportant values were removed. Since some data, such as playlist counts, playlist reach, and total streams, are stored as text with commas (for instance, "4,789,123"), those commas were removed and the values were converted back into numeric form so calculations could be performed. Spotify playlist count and Spotify playlist reach were used as inputs to predict total Spotify streams. To evaluate this prediction method, the songs were randomly split into a training group (80% of the data) to teach the model and a test group (20%) to check its accuracy with a fixed random seed (in this case it was 50) so the split outputs the same result each time it runs. Before fitting the model, all the predictors were placed on the same scale so they could be measured easily. Finally, a linear regression model is built. This model learns from the training set to see how Spotify playlist count and reach relate to Spotify streams, then applies that relationship to the test set to see how accurately it predicts stream numbers.

Results

	Train	Test
Spotify Playlist Count R^2	0.7252	0.7386
Spotify Playlist Reach R^2	0.2405	0.3172

The R^2 represents the percentage of variation in total Spotify streams that the linear regression model can explain with a single predictor. When Spotify Playlist Count was used, the model accounted for approximately 72.5% of the variation on the songs it was trained on and 73.9% on songs it had never seen before. These high and similar numbers demonstrate that playlist count is a strong and reliable predictor of how many times a song will be streamed on Spotify, with no signs of overfitting. Spotify Playlist Reach has a lower R^2 of 24.1% for training and 31.7% for testing. This means it explains fewer differences in Spotify stream totals, even though it generalizes slightly better than the original data. When predicting total streams, how many playlists a song appears in is more useful than how many listeners those playlists reach.

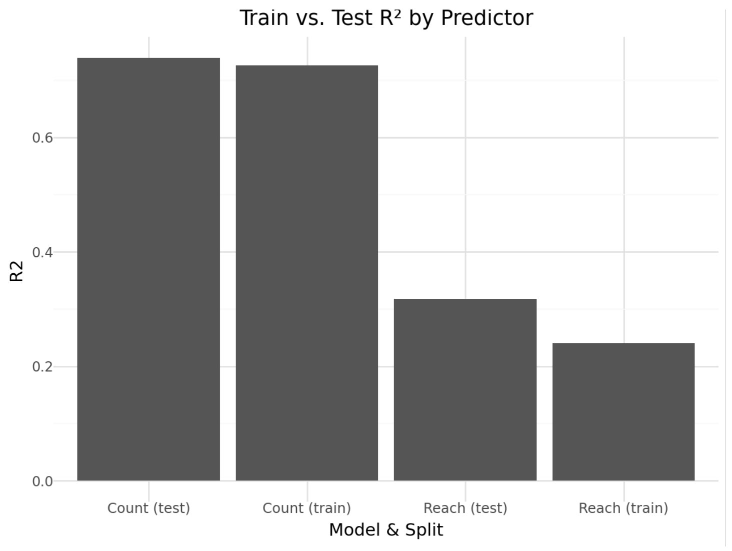

Figure 1: A bar chart displaying the train and test R^2 scores for Spotify Playlist Count and Spotify Playlist Reach

The bars show that using playlist count alone explains about 72-74% of the differences in stream numbers, so it's a strong predictor. Playlist reach only explains about 24-32%, so it's much weaker. Since the training and test bars are similar, both results are stable and generalize somewhat well to unseen data.

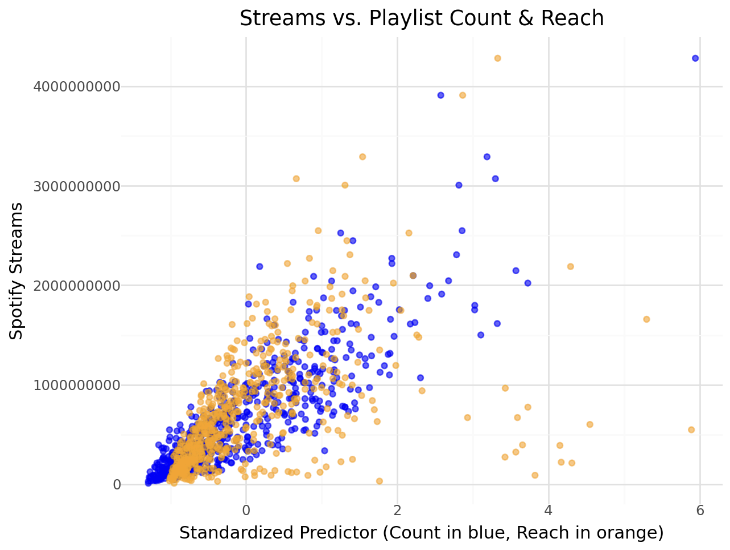

Figure 2: A scatterplot displaying Spotify Playlist Count and Spotify Playlist Reach against Spotify Streams to visualize the strength and direction of the relationship

In this plot, each point represents a song, with its playlist count score on the horizontal axis (blue points), its playlist reach score (orange points), and its total Spotify streams on the vertical axis. There is a clear upward pattern for the blue points, so as playlist count increases, stream totals tend to increase. The orange points also trend upward but are more scattered, indicating many songs reach large audiences without matching high stream counts, and vice versa. Together with the bar chart, this visual confirms that playlist count has a tighter and more consistent relationship with streams, whereas playlist reach shows that reaching a larger audience doesn't consistently lead to higher stream counts.

Discussion

Spotify Playlist Count clearly outperforms Playlist Reach in terms of predicting total Spotify streams. In the tests, a linear regression model based on the number of playlists containing each song explained approximately 73% of the variation in streaming numbers, whereas a model based on listener reach explained only about 30%. This means that knowing how many playlists a track appears in can provide an accurate estimate of its streaming success. Artists and record labels can use playlist count data to predict performance and decide where to focus their marketing.

However, this approach has limitations. It considers only one factor at a time and assumes a linear relationship, so it may overlook important influences such as fanbase size and social media influence. We removed any songs with missing values, which could bias our results, and allowed a few mega-popular songs to skew the model in ways that harmed predictions for other songs. To improve the model, missing data could be filled rather than discarded, and experimenting with more flexible methods like decision trees could capture non-linear effects and reduce the impact of outliers. Including features such as artist follower counts and social media metrics would result in a more reliable tool for predicting how a song will perform as well.

Question 2: With an 80/20 train-test split, how accurately do different degrees of polynomial regression predict Spotify Streams? What are the MSE, MAE, and R^2 for each model on training and testing sets, and do they indicate signs of overfitting?

The original question in the analysis plan used linear regression and polynomial regression with only a degree of 2. However, it was changed to only using polynomial regression and comparing the metrics from using degrees 2, 3, and 4. This change was made to determine the best complexity that is best suitable to model the data.

Methods

To analyze the factors that influence Spotify streaming numbers, we built a polynomial regression model using various song-related statistics like chart rankings, playlist placements, radio plays, and streaming data from platforms such as Spotify, Pandora, and SoundCloud. We used a method called train-test split, where 80% of the data was used to train the model and 20% was set aside to test how well the model performs. Before training, we cleaned the data by removing commas from large numbers and converting text values into numeric form that we could use to model with. We also standardized the data so all values were on a similar scale and converted the “Explicit Track” category into a format the model could understand. To test how complex relationships might improve predictions, we tested three versions of the model: one using polynomial degree 2, the other with degree 3, and the last one with degree 4. These polynomial models help capture non-linear patterns in the data, and we compared how well each performed at predicting Spotify streams on both the training and test data.

Results

	Degree 2	Degree 3	Degree 4
	Test		
MSE	2.4587	1.6731	3.8763
MAE	25071	49784	39925
R^2	0.4085	-401.4966	-931.5000

	Train		
MSE	2.0988	4.8612	2.7527
MAE	11141	3.2646	9.9398
R^2	0.9344	1.0	1.0

The results from the analysis show that while the more complex models fit the training data extremely well, they performed poorly on the test data, indicating overfitting. The degree-2 polynomial model achieved a moderate level of accuracy, with an R^2 of about 0.41 on the test set, meaning it was able to explain around 41% of the variation in Spotify streams using the input features. However, the degree-3 and degree-4 models, while nearly perfect on the training data ($R^2 = 1.0$), did very poorly on new data, with R^2 scores of -401 and -931, respectively. These negative values show that the predictions were worse than simply guessing the average. The large increase in error metrics with MSE and MAE for the higher-degree models on the test data confirms this overfit. This suggests that the simpler degree-2 model has a better balance between capturing patterns in the data and being able to generalize to new songs, while the more complex models memorize the training data too closely and fail to make reliable predictions on unseen data.

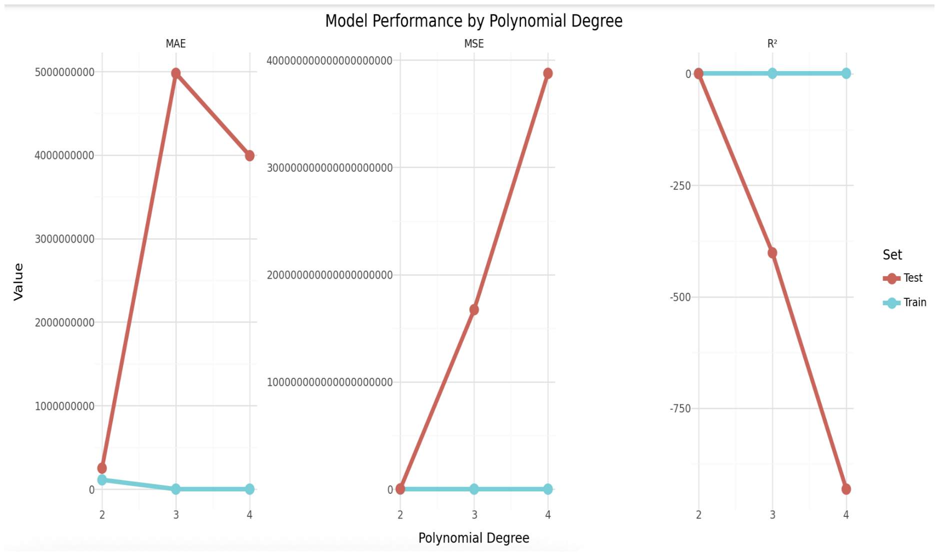

Figure 1: A line chart comparing MSE, MAE, and R^2 for each model on the training and testing sets

This graph shows how well each polynomial model performed using three evaluation measures: Mean Absolute Error (MAE), Mean Squared Error (MSE), and R^2 (which measures how well the model explains the data). For the training data (in blue), all models, especially the degree-3 and degree-4 ones, performed almost perfectly, with very low errors and R^2 values close to 1.

However, on the test data (in red), the errors increased dramatically for the higher-degree models, and the R^2 values became extremely negative. This indicates that while those models learned the training data very well, they failed to make accurate predictions on new data. In contrast, the degree-2 model had a good balance since it performed reasonably well on both the training and test sets. This pattern shows that the higher-degree models overfit the training data and are not suitable for real-world prediction.

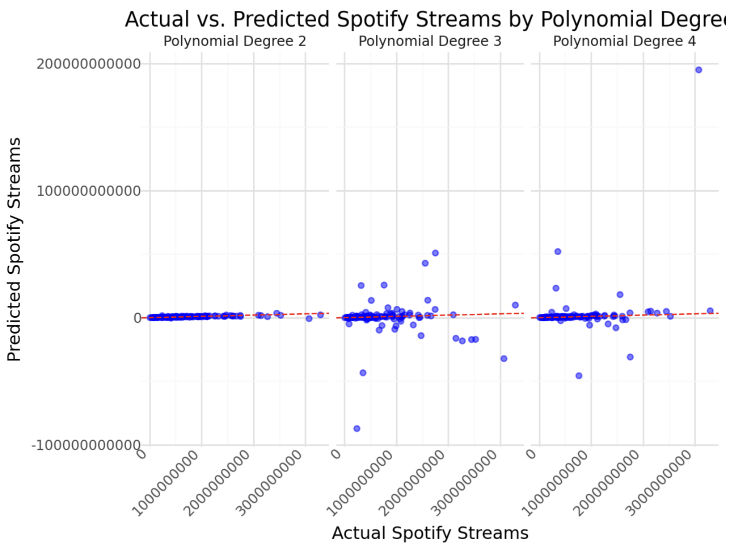

Figure 2: A scatterplot of actual vs. predicted Spotify Streams for each model to visualize prediction fit

The second graph compares the actual Spotify stream numbers to the predicted values from each model. In an ideal model, the points would fall along the red dashed line, meaning the predictions closely match the real values. For the degree-2 model, the predictions follow the line fairly well, showing that it generally captures the overall pattern. However, for the degree-3 and degree-4 models, the predictions are scattered and often very far from the actual values. These models produce extreme outliers and unrealistic stream counts, which confirms they are overfitting and not reliable for real-world use. This visual comparison supports the conclusion that the degree-2 polynomial model is the most appropriate choice for this dataset.

Discussion

With an 80/20 train-test split, the performance of polynomial regression models in predicting Spotify stream counts varied significantly depending on the degree of the polynomial. The contrast between training and test performance for higher-degree models indicated overfitting since they captured noise and specific characteristics of the training data instead of learning general patterns. Therefore, while polynomial regression can help explore non-linear relationships, the degree-2 model is the only one that generalizes well enough to be useful for prediction in this case.

That said, the model still has several limitations. The model assumes that relationships between predictors and the target (Spotify Streams) are polynomial in nature, which may not be the best functional form for this data. To improve the model in future analyses, it would be best to use more advanced models like decision trees or random forests that handle non-linearities better. Collecting more features and improving data quality would also likely improve prediction accuracy and model usefulness.

Question 3: For the five most popular artists in the dataset that appear the most, how does their average Track Score compare with the average Track Score of all remaining tracks in the dataset?

Methods

I used the function [‘artist’].value_counts().nlargest(5).index.tolist() and set it to a variable this gets the top 5 artists that show up most in the dataframe. I then made two different dataframes one with only the top 5 artists and one without any of the top 5 artists. Then I found the mean of the track scores of both dataframes. I made three separate ggplots one comparing the two averages of the average track scores, one comparing track score and amount of spotify streams with it being color coded of being made by a top 5 artist and not and lastly a histogram of the top 5 artists and their average track scores.

Results
What were the results of your model(s) and analysis?

Average Track Score for Top 5 Artists: 74.75384615384615
Average Track SCore for Top 5 Aritists: 60.20545808966861

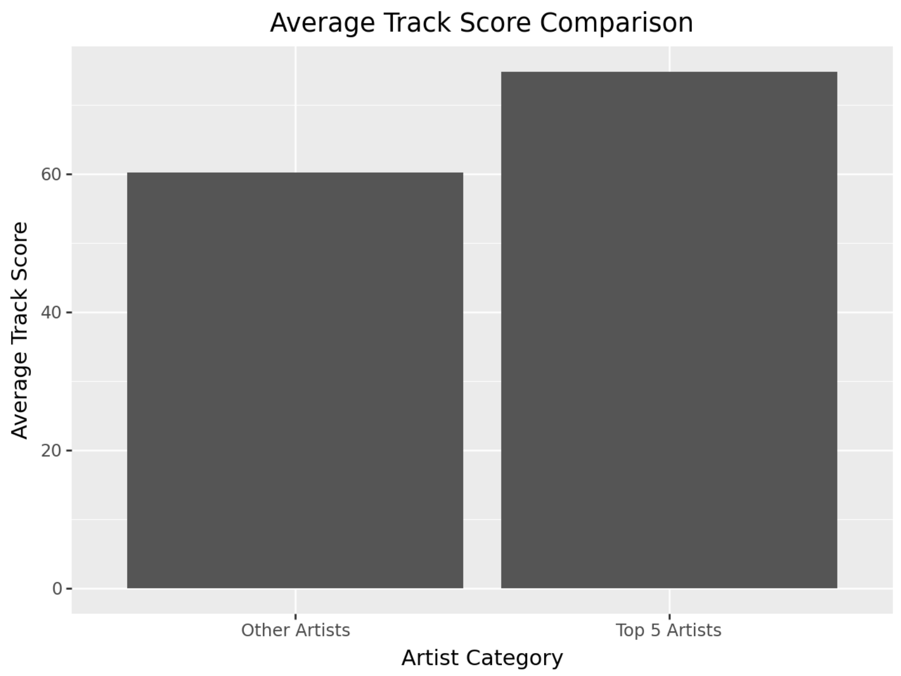

Figure 1: A histogram comparing the average track score of top 5 artists and other artists

In the first ggplot we can see a clear disparity in track score between the top five artists and the rest of the pack with average track score of the top five artists being 74 and 60 for the rest. However we need more analysis before coming to conclusions.

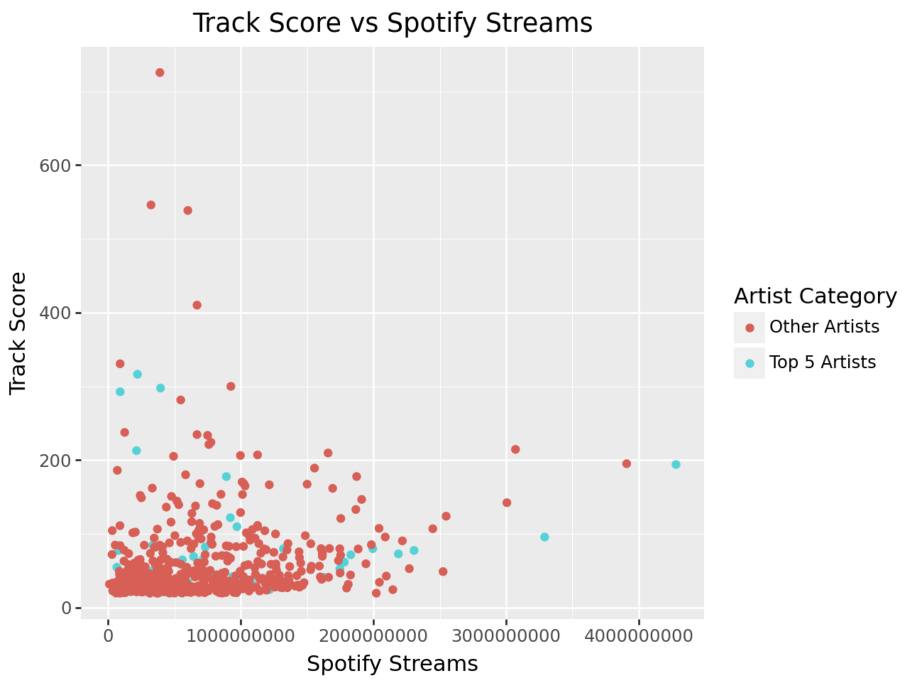

Figure 2: A scatterplot of every songs track score vs the amount of spotify streams they have color coded as being made by a top five artist or not

In our ggplot above we can see that there is not much of a visual difference between the amount of top five artists with high spotify streams and track score, and non-top five artists with high spotify strreams and track scores. This tells us that the top five artists may not necessarily hog all the top popular songs in terms of streams and track score.

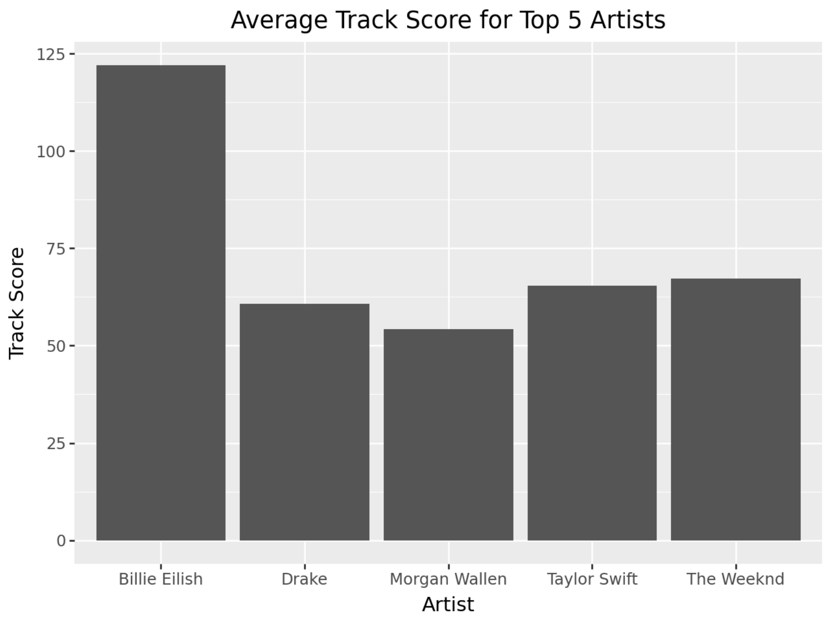

Figure 3: A histogram showing the average track score of each of the top 5 artists music

In this scatter plot, we can clearly see that Billie Eilish's music clearly had a much higher average track score than the rest of the top five artists. With this imformation we can make a better conclusion to our questions

Discussion

After visualizing everything, its clear that the top 5 artists do get higher track scores than the rest of the pack. But this may be misleading because we can see when looking at the top 5 artists that Billie Eilish is carrying the top 5 average higher while the rest of the artists have track scores around the same as the rest of the pack. Like in the scatterplot, we can visualize it and see that there is not much disparity in track score with the top 5 artists compared to the rest. In conclusion, Billie Eilish is an outlier and is skewing our results to make it look like the top 5 artists have a higher average track score when in reality without her, they would just be average. This model is useful in telling us that while many may assume that the top artists worldwide may create better quality music than the "underground" artists, the number of times an artist shows up in the 2024 most played songs doesn't correlate to quality or ammount of streams. There are still some limitations to my analysis as I didn't find certain outliers of songs that could be affecting Billie Eilish's average being so high as well. What I could have done that may improve my analysis in the future is maybe a boxplot or violin plot to point out possible outliers such as Billie Eilish's songs who seem to be highly rated compared to the other four top artists which I highly disagree with as her music is trash in my opinion.

Question 4: Do songs with explicit tracks tend to have better ratings than those that are not explicit?

Methods

I used group_by to group the explicit vs non explicit songs. I then found the mean track scores of the explicit vs non-explicit songs. I made three seperate ggplots first being a histogram showing the mean average track score of the explicit vs non-explicit songs, a scatter plot showing track_score vs spotify streams again but this time color coded as explicit or not explicit, and finally a two box-plots which I should have done in the last problem to show the distribution of track scores between explicit and non-explicit songs.

Results

```
Explicit Track
0    59.024405
1    65.241921
```

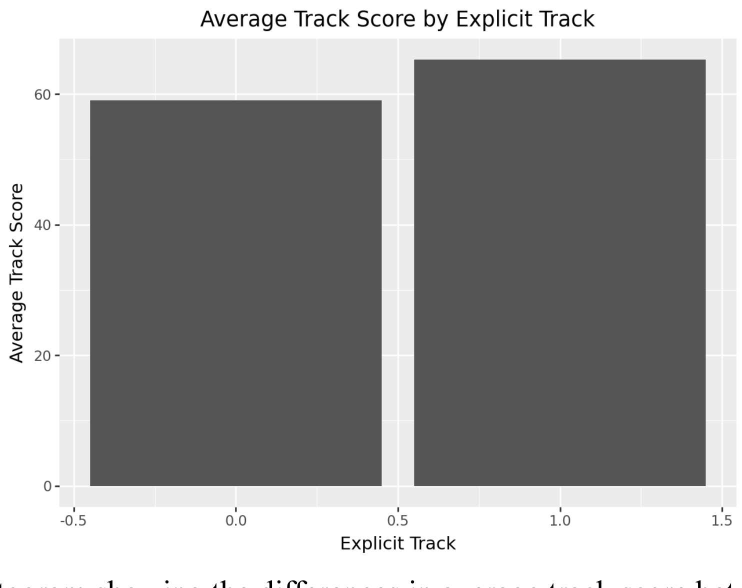

Figure 1: A histogram showing the differences in average track score between explicit vs non-explicit tracks

We can see in the histogram that explicit tracks have a slightly higher track score average but we need more visuals before coming to a conclusion

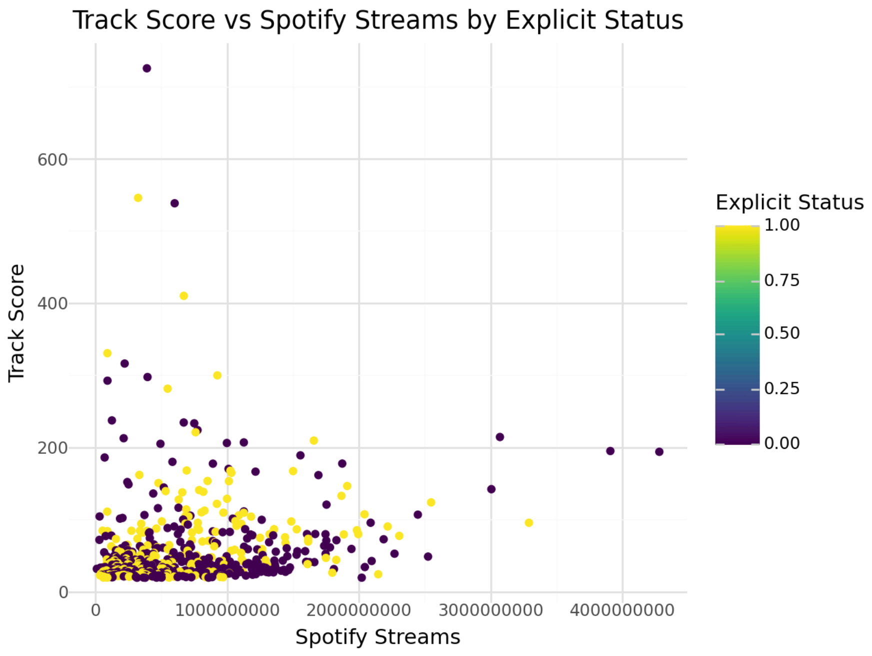

Figure 2: A scatterplot of every songs track score vs the amount of spotify streams they have color coded as explicit vs not explicit

In the scatter plot above we see a lot of explicit tracks slightly more spread out in the higher track score range and the amount of total spotify streams. With the non-explicit tracks we see a lot more outliers meaning there maybe a few more bangers that are non-explicit but more consistent higher rated explicit songs.

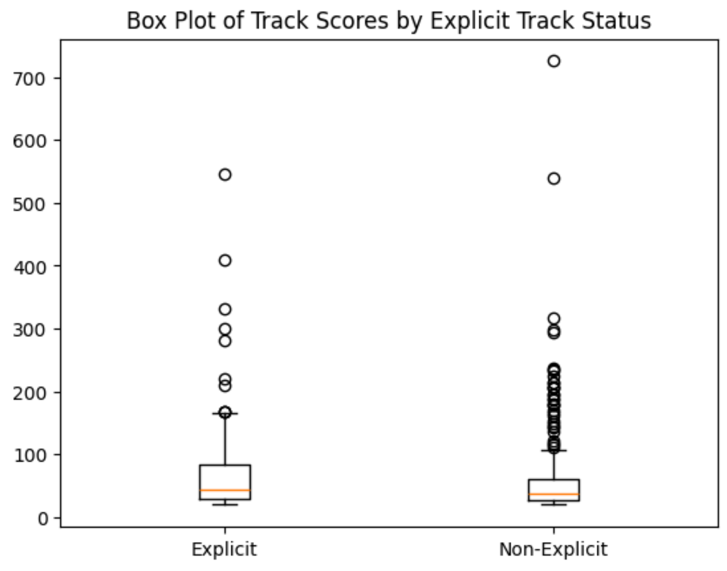

Figure 3: 2 box plots showing the distribution of track scores with explicit and non-explicit music

In the box-plot above, the non-explicit tracks show more outliers, especially extreme high values. There are several non-explicit tracks with very high scores (e.g., above 500 or even 700), while explicit tracks have fewer such outliers. This might indicate that while both types of tracks have similar average performance, some non-explicit tracks are exceptional hits. The interquartile range (IQR) — the height of the box — is slightly wider for explicit tracks, meaning their middle 50% of scores are more spread out with a lot more slightly higher track score songs in the middle 50% compared to the non-explicit tracks.

Discussion

Here is my very detailed discussion of the answer to this question (i.e., What is this model useful for?), as well as a discussion of the limitations of my model/question, and how I could make it better in the future?

In-conclusion, we can say that there are more extremely high-rated non-explicit tracks but more explicit tracks in the slightly better than average rated tracks. To put in a more clear context, the average explicit song was rated about in the range of 5-7 out of 10 while the average non explicit song was rated 3-6 out of 10 however there were more outliers that were rated 9-10 out of 10 for non-explicit songs. This model is useful in telling us the difference in how explicit and non-explicit songs are rated, and if adding profanity into music improves the ratings or not. This can help an artist decide wether they want to add profanity into their next song. There were no real limitations with this model as we were able to detect the outliers this time in our boxplot unlike the last question.

Introduction:

Recently, music labels want their artists to have a higher engagement with their social media so they can promote their music and have greater success. Platforms such as YouTube and TikTok are one of the many main hubs where artists self-promote. Additionally, consumers of their music can also create content to help their favorite artist gain more success. The objective is to predict the Spotify streams a song will receive based on a platform's likes. In our data set, the features present are: the TikTok likes a song gets, the YouTube likes from a song, and Spotify streams. If successful, the model could give a better understanding of who shops at their store. For example, record labels might prioritize their artists' social media engagement to help their careers.

Question 5: What clusters if any emerge when studying Spotify streams, Spotify popularity, siriusXM spins, and airplay spins? (Is there a relationship between the popularity of music on the Radio vs. on Spotify)

Methods

I started by selecting the predictors needed for this analysis. Since I am looking to characterize the dataset based on the clusters that have to do with Spotify and Radios, I selected the following variables: *Spotify Streams, Spotify Popularity, Airplay Spinds, SiriusXM Spins, Spotify Playlist Count,* and *Spotify Playlist Reach*. I cleaned the data by removing the commas and then z-scored it to standardize the model.From here, I decided to use a Gaussian Mixture Model to cluster the data since I know that there is a lot of overlapping data with different variances and the shapes are not normal. From finding out the AIC score of different values of K, I found the most optimal number of clusters to model. Creating, fitting, and predicting the model, I found the correlation between *Spotify Streams* and *SiriusXM Spins*, *Spotify Streams* and *Airplay Spins*, *Spotify Popularity* and *SiriusXM Spins*, as well as *Spotify Popularity* and *Airplay Spins*. To confirm the results seen in the graphs using GMM models, I also did two train-test split models with an 80/20 split with one of the Y values set to be *Airplay Spins* and the other one *SiriusXM Spins*, and the X values being all of the variables pertaining to Spotify. I once again cleaned the data, setting all values to be numeric and getting rid of any commas found. After z-scoring the *Spotify Streams* variable, a linear regression model was fit for both instances, and the R^2 was found.

Results

The most optimal number of clusters to model from ended up being 3, by looking at the AIC curve in Figure 5.1. When modelling clusters based on a gaussian mixture model, I found that across all of the different graphs(Figures 5.2-5.5), the model did not perform too well. To further prove that popularity on Spotify and on the radio do not correlate, I quickly did a HAC graph(Figure 5.6) and found that it also did not have a good relationship. When finding the R^2 for the linear regression models I had the R^2 for AirPlay Spins be 0.13, and the test set for it be 0.014. For SiriusXM, the train set's R^2 ended up being 0.0034 and the test set was -.00002698. Both of these model present overfit and indicate that there is no relationship between the two areas. The HAC model in figure 5.6 further shows that there is a lot of separation in the model even towards the bottom of the graph, indicating that there is not that much cohesion. Since all of nodes have long legs, we can conclude that the different clusters are not very cohesive indicating the variables aren't closely related.

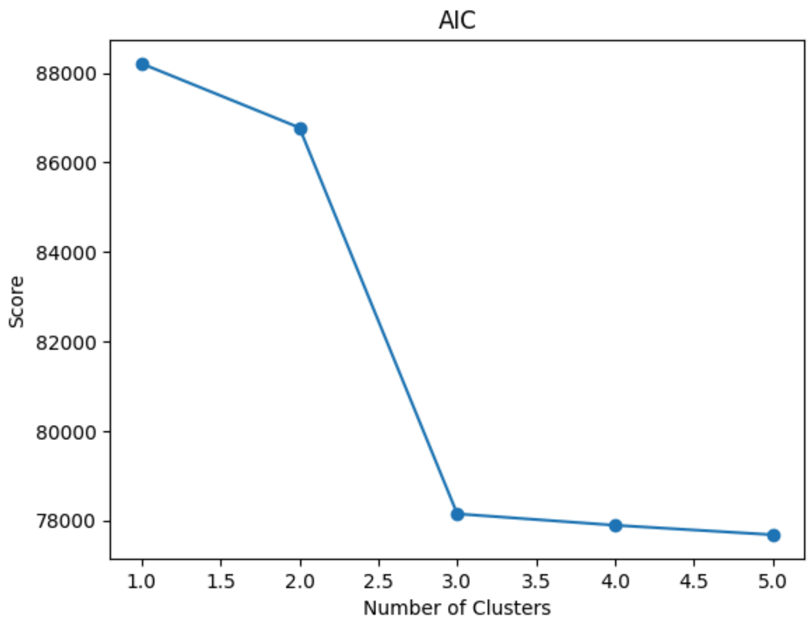

Figure 5.1: AIC Graph for Determining the # of Clusters

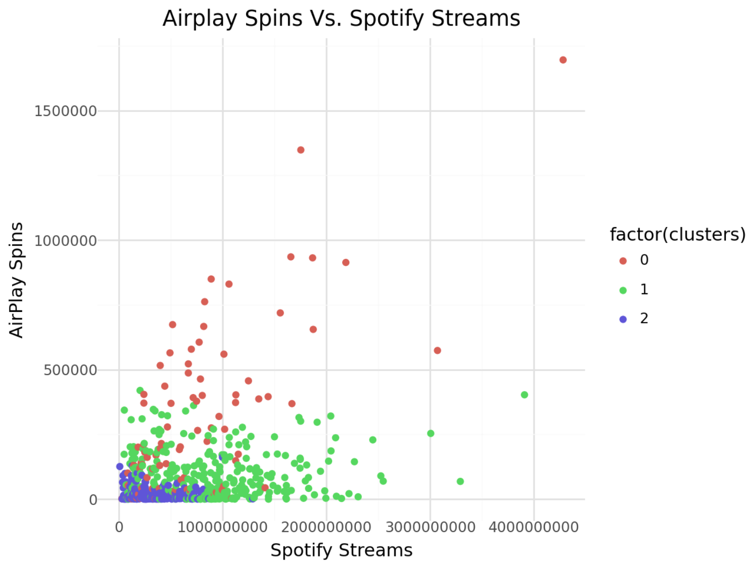

Figure 5.2: Relationship between Airplay Spins and Spotify Streams

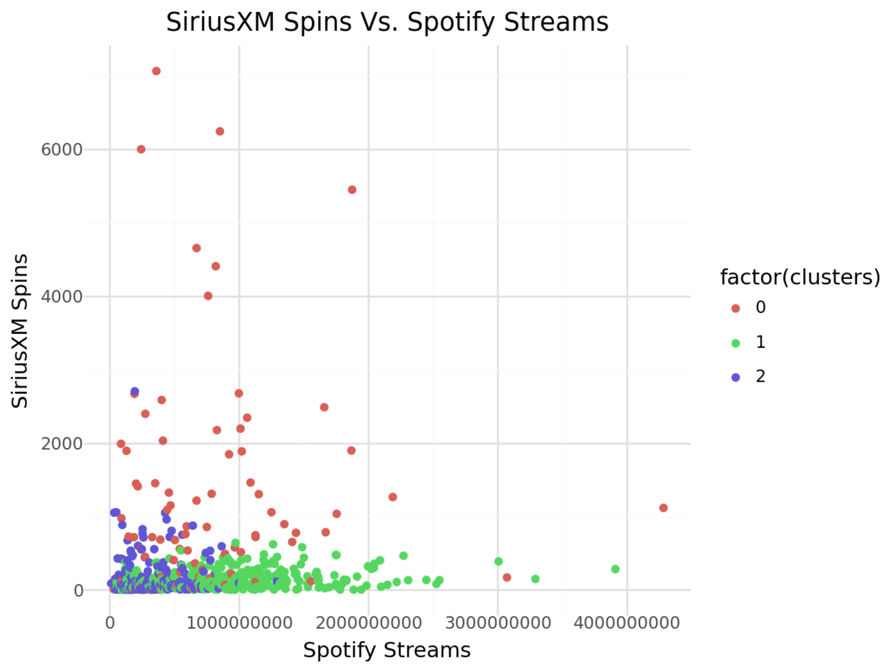

Figure 5.3: Relationship between SiriusXM Spins and Spotify Streams

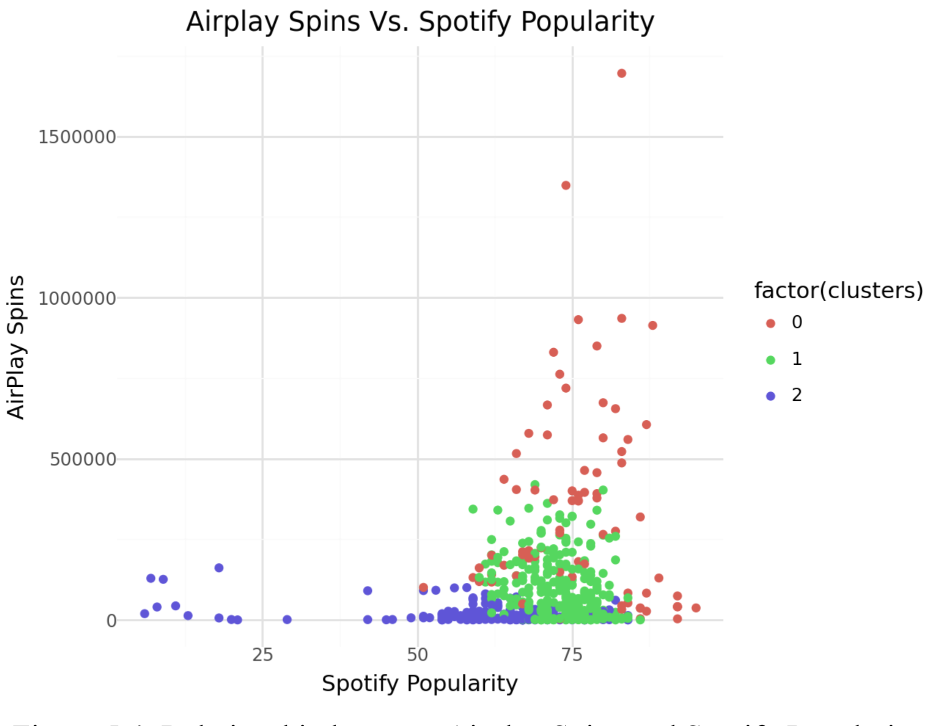

Figure 5.4: Relationship between Airplay Spins and Spotify Popularity

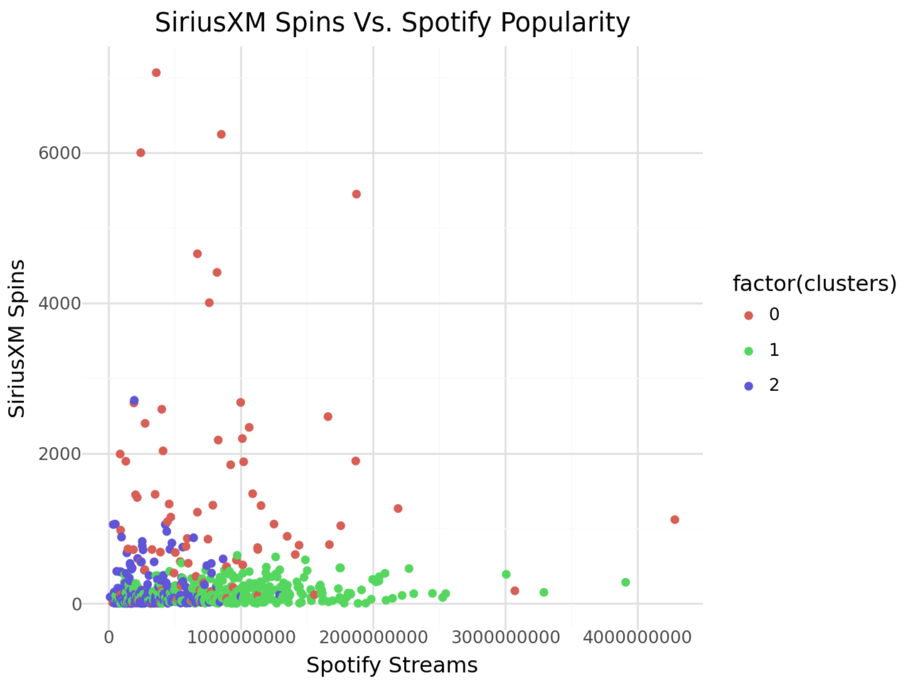

Figure 5.5: Relationship between SiriusXM Spins and Spotify Populartiy

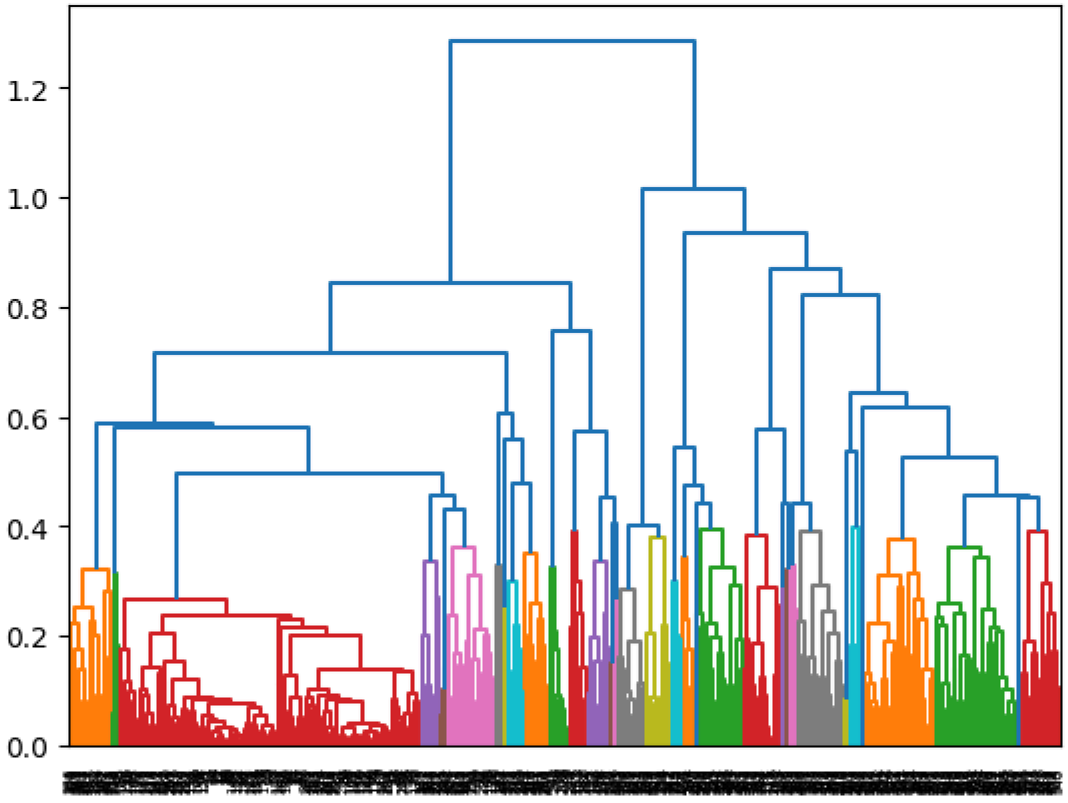

Figure 5.6: HAC based on the Spotify and Radio Variables

Discussion

Initially, I had thought that the popularity of a song on the radio and on Spotify would be correlated at least a little bit, but I was proven wrong. I know that the radio has severe bias in the songs that they play. Music on Spotify is dictated by each individual user and open to listen to any genre or music from a specific era. The modelling and clustering I was attempting, proved to not show a correlation in any of the three methods I used. The GMM clustering did not show any distinct clusters, the simple linear regression had very low R^2's and the dennogram shows that all of the clusters or splits are very separable, even towards the bottom.

The music industry is very complex and I was unable to characterize a positive relationship between Spotify and Radio popularity.

For the future, I could explore more complex modeling techniques to be able to explain the relationship between the two different things better.

Question 6: After finding the number of principal components needed to capture at least 90% of the variance, is there a clear distinction on what platform a song is popular? (ex. Spotify Streams Vs. TikTok Videos)

Methods

The data was subsetted for this question with the variables related to Spotify, TikTok, Youtube, and Apple Music being used. All variables were cleaned of any commas, converted to floats, and the NA values were dropped. After doing this, the data was z-scored and a PCA was fitted. The Y metric I used to quantify this model is *Spotify Popularity*, using the median popularity to split the data to be 0 and 1 to make it binary. After doing a train test split with an 80/20 split, a scree and cumulative variance plot was created to see how many PC's are needed to capture 90% of the variance. I then plotted PC2 Vs. PC1 to see what the relationship between the first two PC's are.The accuracy, precision, recall, F1, and ROC AUC were found based on a linear regression, and then found once again with the 90% variance using PCA model.

Results

After cleaning the data, and fitting the model to a PCA, I found that the optimal number of PC's is 5 as seen in figure 6.1. To see how separated the clusters were, I also graphed the first two PC's against each other. Seeing that there is not good separation or cohesion in the graph, there is no clear distinction in what platforms make a song popular. We know this to be true especially since accuracy, precision, recall f1, and the roc auc score is very high for the linear regression as well as PCA with 90% variance. The PCA did explain the data better as for every value in the train set being 0.15 higher than it is in the linear regression and a 0.2 increase for the test sets for every score. We also see a slight overfit across the board for both models. The true values can be seen in figure 6.3

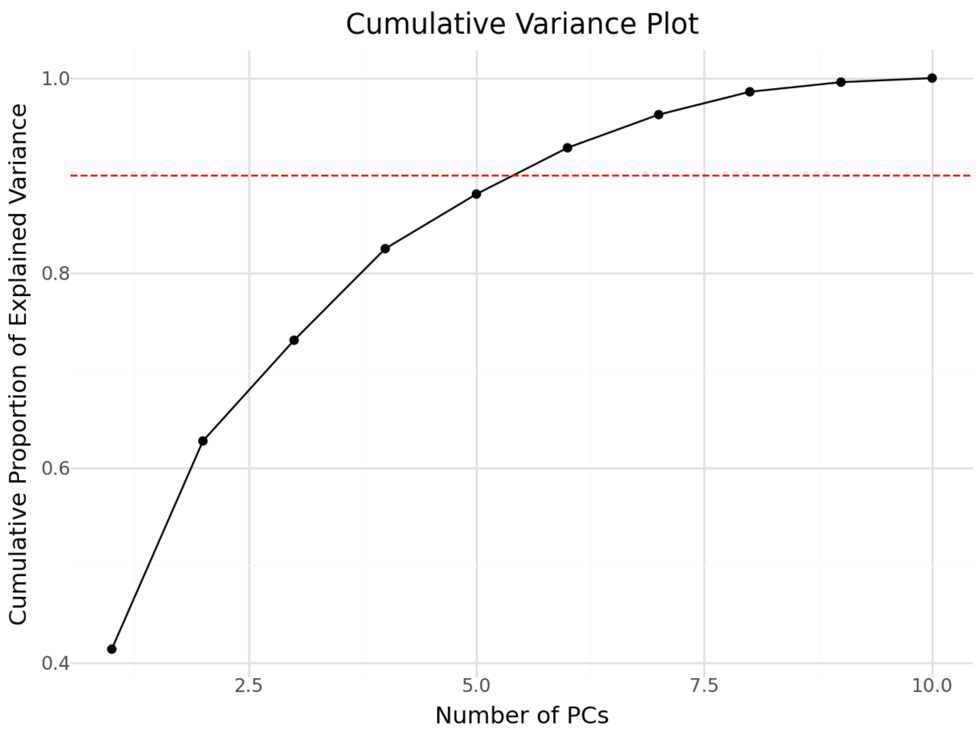

Figure 6.1: Variance Plot for the PCA's using Spotify, Tiktok, and Yotube variables

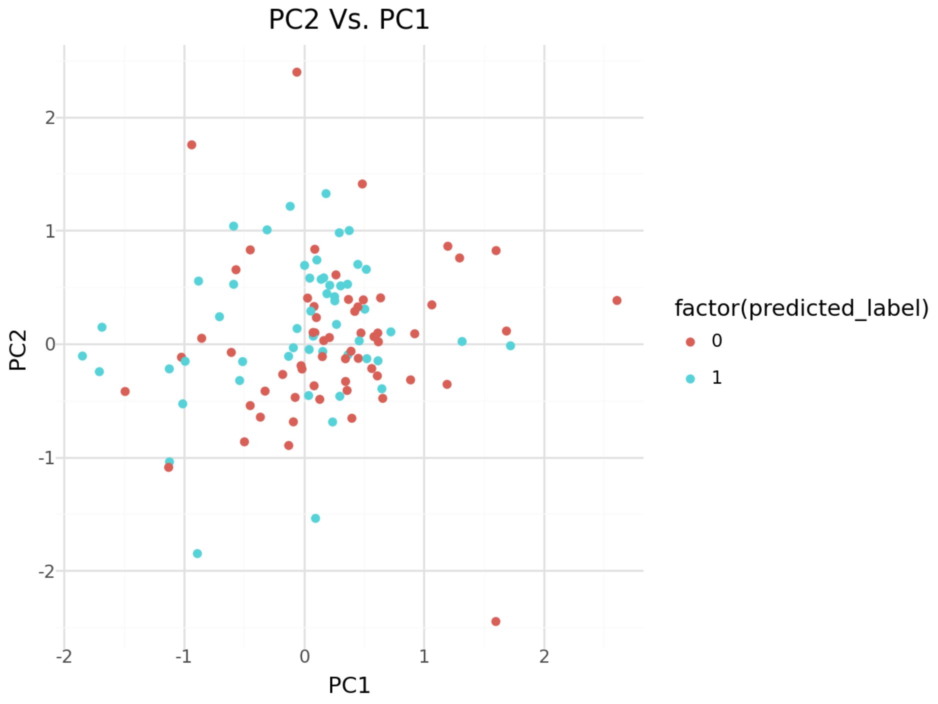

Figure 6.2: PC2 Vs. PC1

Linear Regression	PCA with 90% Variance
Train Acc : 0.8097345132743363	Train Acc : 0.9668141592920354
Train Prescision: 0.7990867579908676	Train Prescision: 0.963302752293578
Train Recall : 0.8064516129032258	Train Recall : 0.967741935483871
Train F1 : 0.8027522935779816	Train F1 : 0.9655172413793104
Train ROC AUC : 0.8640847141876655	Train ROC AUC : 0.9936464359250907
Test Acc : 0.7256637168141593	Test Acc : 0.911504424778761
Test Prescision : 0.6666666666666666	Test Prescision : 0.8571428571428571
Test Recall : 0.76	Test Recall : 0.96
Test F1 : 0.7102803738317757	Test F1 : 0.9056603773584906
Test ROC AUC : 0.7888888888888891	Test ROC AUC : 0.9714285714285715

Figure 6.3: Metrics for a Linear Regression and PCA with 90% Variance

Discussion
Different platforms have different impacts to how we listen to music. But, the differences aren't separable enough to signify what songs will be popular on what platform or if doing well on one platform generally means that it will perform well on another. If had been separable, we would see that there are distinct clusters in the PCA2 Vs. PCA1 graph, but we were unable to. Upon further exploration to confirm the results, doing simple ggplots also proved that there is a slight correlation between the *Spotify* variables and those relating to TikTok and YouTube.
For the future, I can test out different models and ask specifically about a single relationship in order to characterize the relationship better.

Question 7: Do more likes across social media platforms (YouTube and TikTok) predict the number of streams a song will receive on Spotify?

Methods:

Before creating my linear regression models, I prepared my dataset by selecting only the features I would be using in my models. I chose to do this because if I clean my data of null values using all the features, I will have an empty dataset, since the feature Tidal popularity is empty for all rows. After, I noticed that the values were in the type string, so I converted them into numeric values, which is necessary to perform calculations on the data. Once my data was in the correct format, I checked for null values and deleted the rows that had null values. After analyzing the full dataset, I realized there were data points that had extreme outliers, so I removed the top one percent of Spotify streams, YouTube likes, and TikTok likes to minimize their impact on the model. Once I ensured that extreme values weren't going to impact the accuracy of my model, I built my two linear regression models. I used an 80/20 train-test split for test validation. I then z-scored, which was not necessary, because it is a single-feature model, but I wanted to maintain consistency with best practices. Afterward, I built the two linear regression models separately, one for YouTube likes predicting Spotify streams and another for TikTok likes for Spotify streams. And fitted the training data, and made my predictions on the test set.

Results:

After making my linear regression models, both models showed to have poor metrics, but the YouTube likes model indicated predicting Spotify streams more confidently. For the YouTube likes model, the R^2 for the training set was 0.30 and the test set had an R^2 of 0.33, meaning that the model could explain 30-33% of the variation in Spotify streams. The mean squared error for the training set was valued at 1.47e+17, and the test set had a mean squared error of 1.53e+17. The mean absolute error of the training set was 274 million, while the test set had a mean absolute error of 277 million.

In contrast, the regression model for the TikTok likes had even worse metrics. For example, the R^2 of the training and test sets were 0.04 and 0.02, respectively, meaning that TikTok likes explain little to no of the variation in Spotify streams. Additionally, the mean squared errors of the training and test sets were 2.16e+17 and 2.41e+17, respectively. Furthermore, the mean absolute error for the training and test sets was 354 million and 378 million, respectively. The metrics of the TikTok models are a stark difference from those of the YouTube likes, and ultimately show that the YouTube model has a stronger predictive power in a linear regression mode, however, I should note that neither feature alone can provide a highly accurate model.

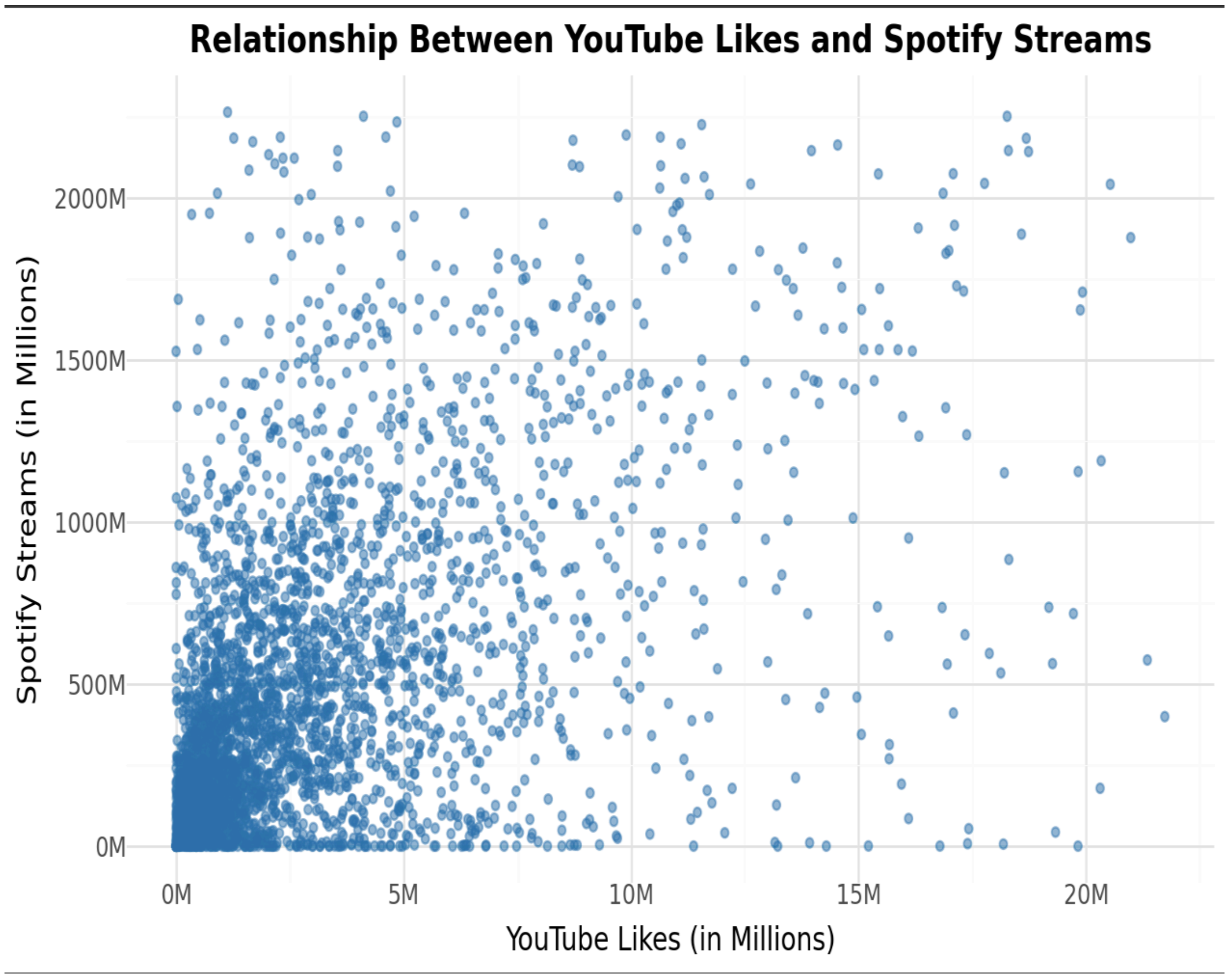

Figure 1: A scatter plot showing the relationship between YouTube likes and Spotify streams, units are in millions.

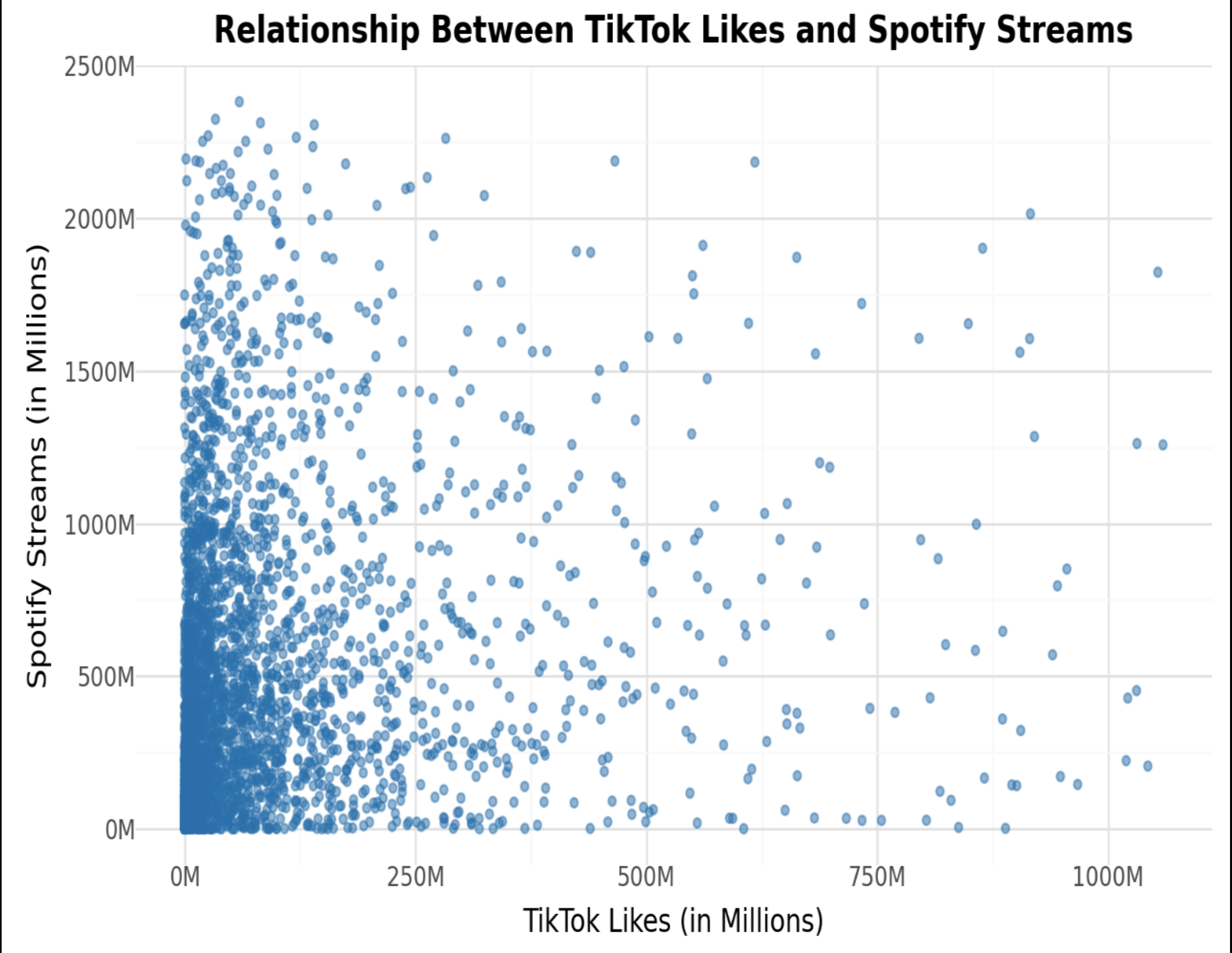

Figure 2: Figure 1: A scatter plot showing the relationship between TikTok likes and Spotify streams, units are in millions.

Discussion:

While social media engagement metrics can be of value when used in a linear regression model, the effectiveness differs by platform. The YouTube likes model has been shown to be more powerful in comparison to the TikTok likes model. After my assessment, I can conclude that YouTube metrics are more aligned with streaming behavior, perhaps because YouTube is known as a music platform in addition to a social media service where individuals listen to music and watch their favorite YouTubers. However, that is to say that both models should not be used at any record label, given that neither has highly accurate predictions. In the future, I would add more features to the linear regression model to capture the full picture. Additionally, I could use non-linear models or decision trees to model more complex relationships.

Introduction:

With the rise of social media (i.e., YouTube, TikTok), the modern recording industry is increasingly implementing on the realization of metrics such as views, likes, posts, shares, etc, to help understand a song's reach and the next impact of these behaviors. This is due in part to social media allowing for exposure with systematic engagement counts (user activities counted), this analysis will enable the researcher to determine if songs exist as meaningfully clustered on social media engagement features binned as clustering, and whether the average AirPlay Spins the radio predominant metric of radio popularity, differs for the clusters. The features I will be using in my clustering models will be Airplay spins, Youtube views, Youtube likes, TikTok posts, TikTok likes, TikTok views, Youtube playlist reach This analysis likewise, will provide a comparison of the two unsupervised learning approaches, DBSCAN and KMeans, to see which clustering algorithm resulted in fewer ambiguous distances to the clusters.

Question 8: When clustering songs based on social media metrics, which clustering method produces clearer group differences?

Methods:

The dataset contained thousands of Spotify's most-streamed songs, with columns including AirPlay Spins, YouTube Views, YouTube Likes, TikTok Posts, TikTok Likes, TikTok Views, and YouTube Playlist Reach. Upon data cleaning (commas removed, converted to numeric, null values dropped, and top 1% outliers removed), six social media metrics were available to use as predictors, which were standardized using Z-score normalization. Clustering was first attempted with the DBSCAN method, then K-Means immediately after.

Results:

An initial pass with epsilon as 1.5 and minimum samples of 10 achieved a high silhouette score of 0.5605 but only produced one cluster (plus some outliers)'s which is invalid for clustering in a meaningful way. Epsilon of 1.5 was chosen by using the elbow method, where one should select where the graph sharply increases as their epsilon. After testing different hyperparamters, I identified that the DBSCAN model with epsilon as 1.0 and minimum samples of 7 yielded 3 groups, and a silhouette score of 0.3936, however, there was a considerable imbalance in cluster sizes, with one cluster having 2,261 data points, one cluster having only 7, and 378 points classified as noise, making it dangerous to draw reliable inferences out of the DBSCAN clusters. K-Means clustering was also attempted with several values of k by graphing the different silhouette scores with different k-values, which are the number of clusters in the model. The best performing KMeans model was found at k equalling 2 with a silhouette score of 0.4877, though it is still lower than the original DBSCAN score, it produced two well-separated and well-balanced clusters. Analysis of K-Means clusters illustrates meaningful differences. In contrast, while DBSCAN offered some value in terms of assessing outliers and the potential for dense clusters, it simply could not provide us with balanced and interpretable clusters, even with the best version that produced more than one cluster, which still contains one dominant group and one nearly empty cluster.

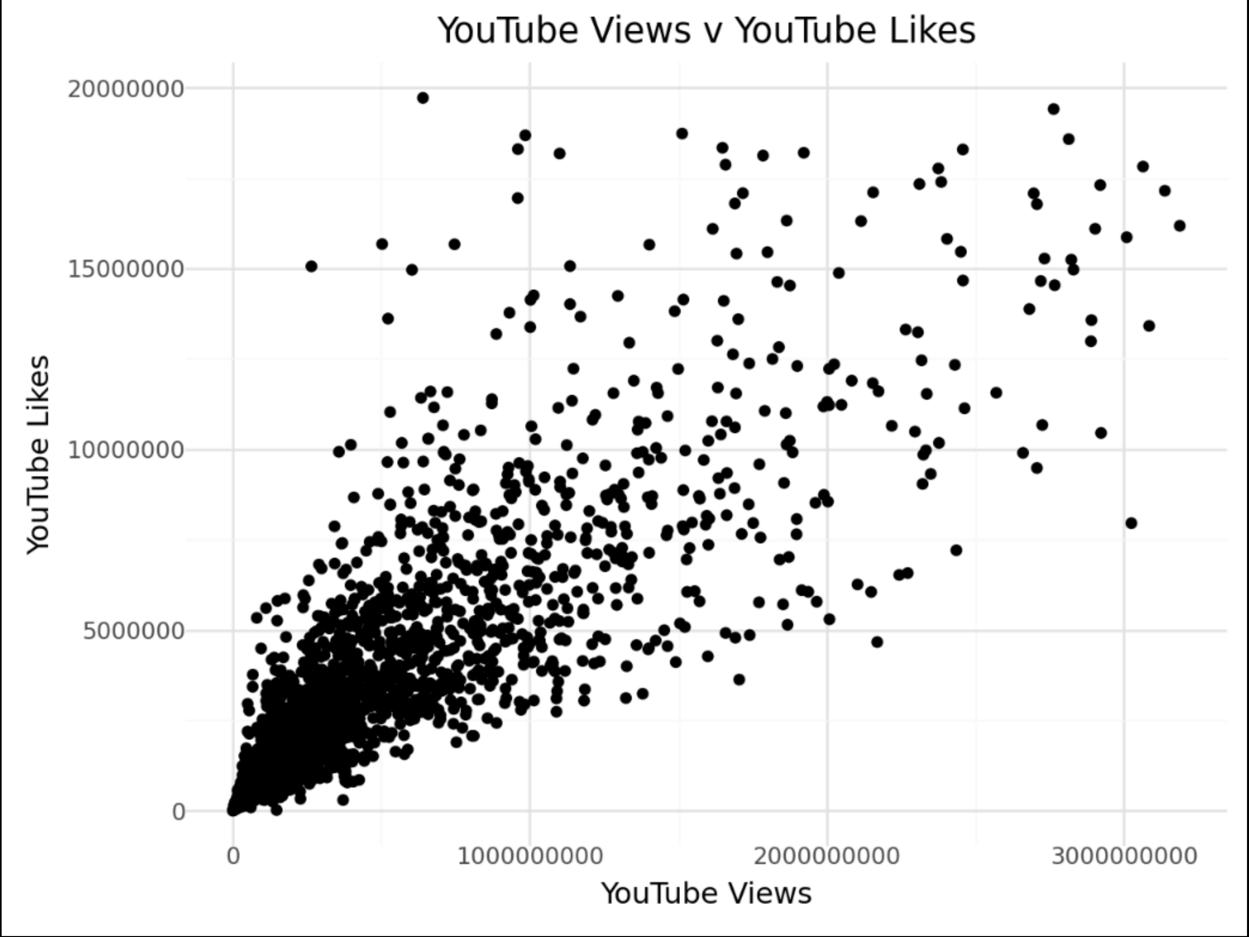

Figure 1: A scatter plot showing the relationship between YouTube likes and YouTube views.

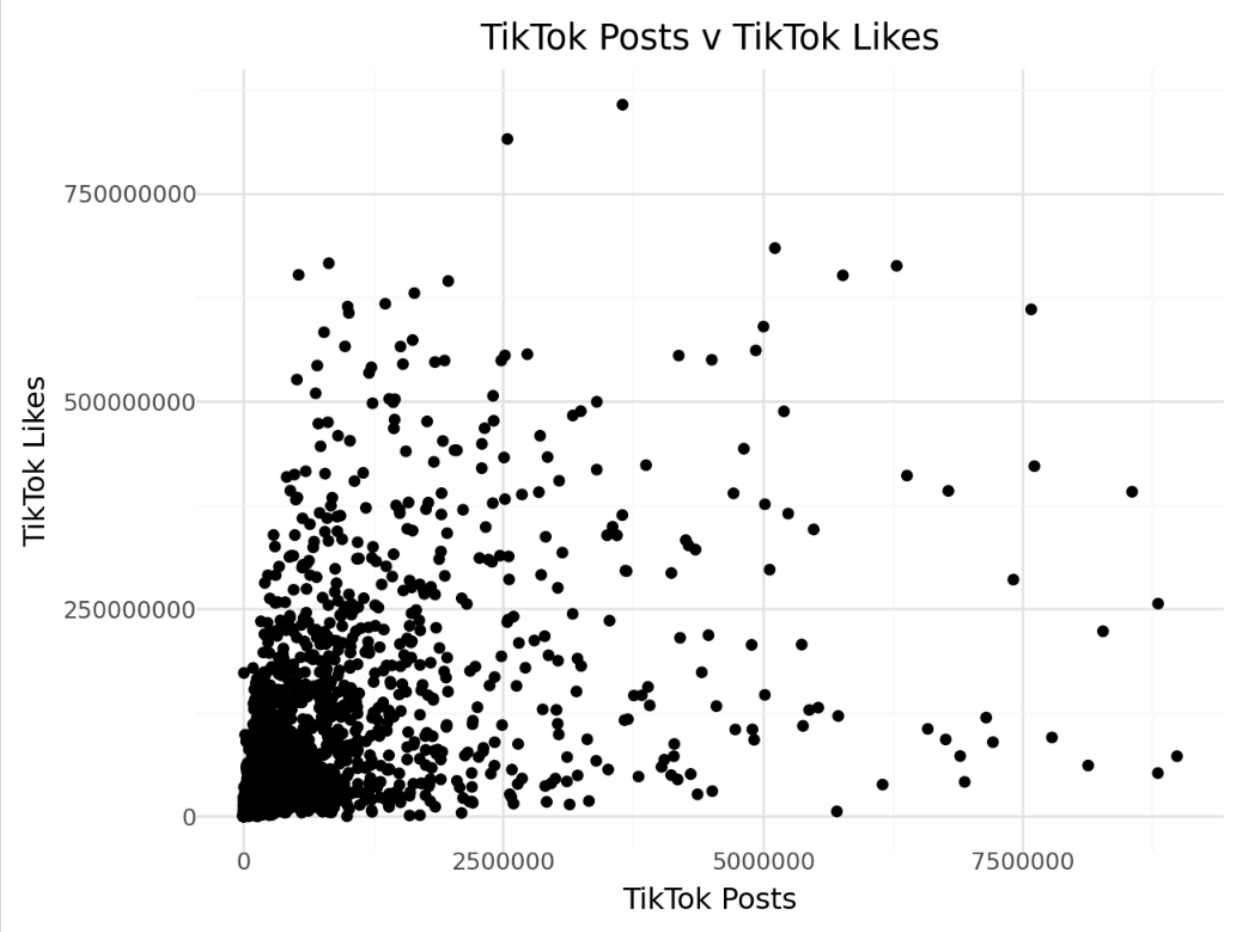

Figure 2: Figure 1: A scatter plot showing the relationship between TikTok posts and TikTok likes

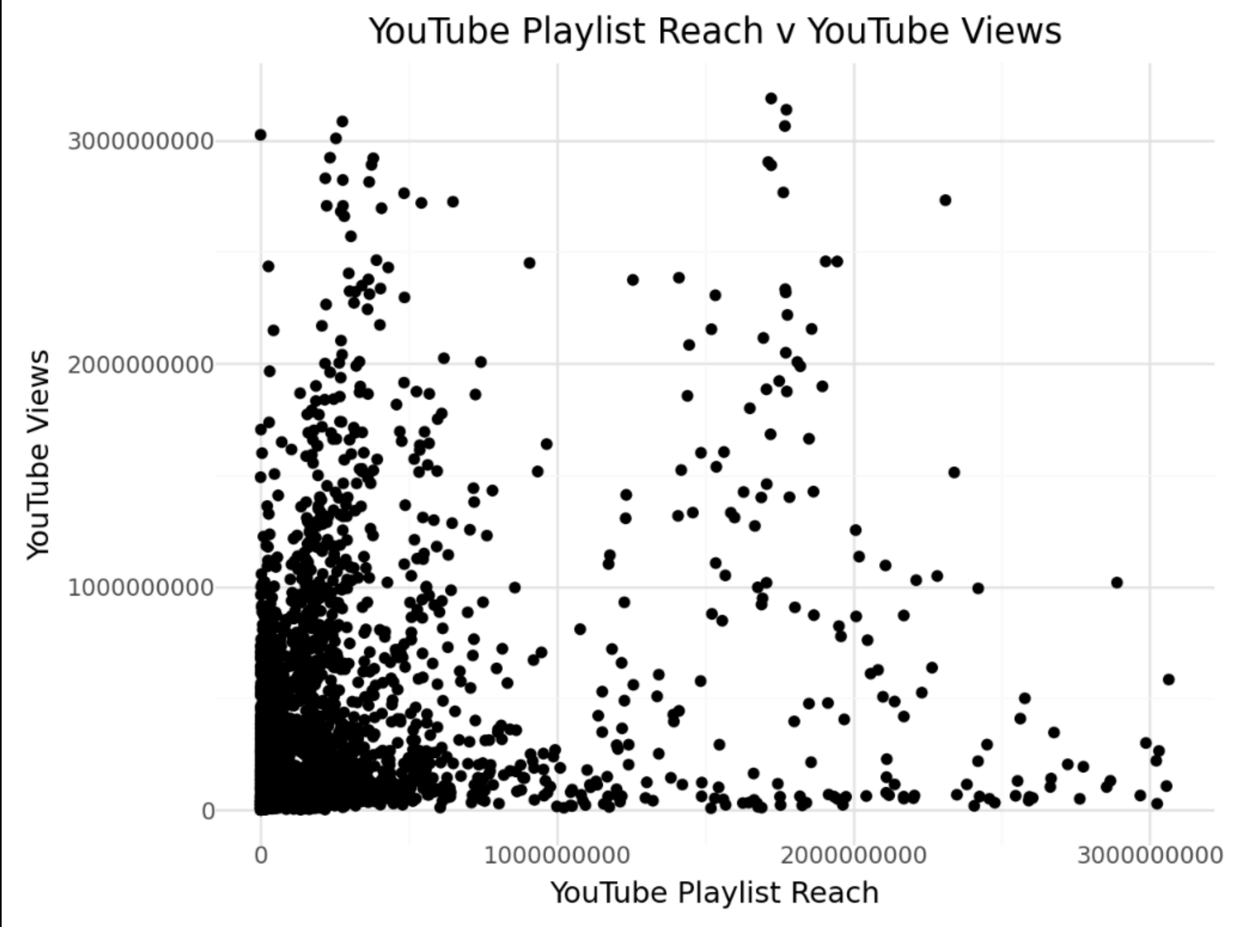

Figure 3: A scatter plot showing the relationship between YouTube Views and YouTube playlist reach

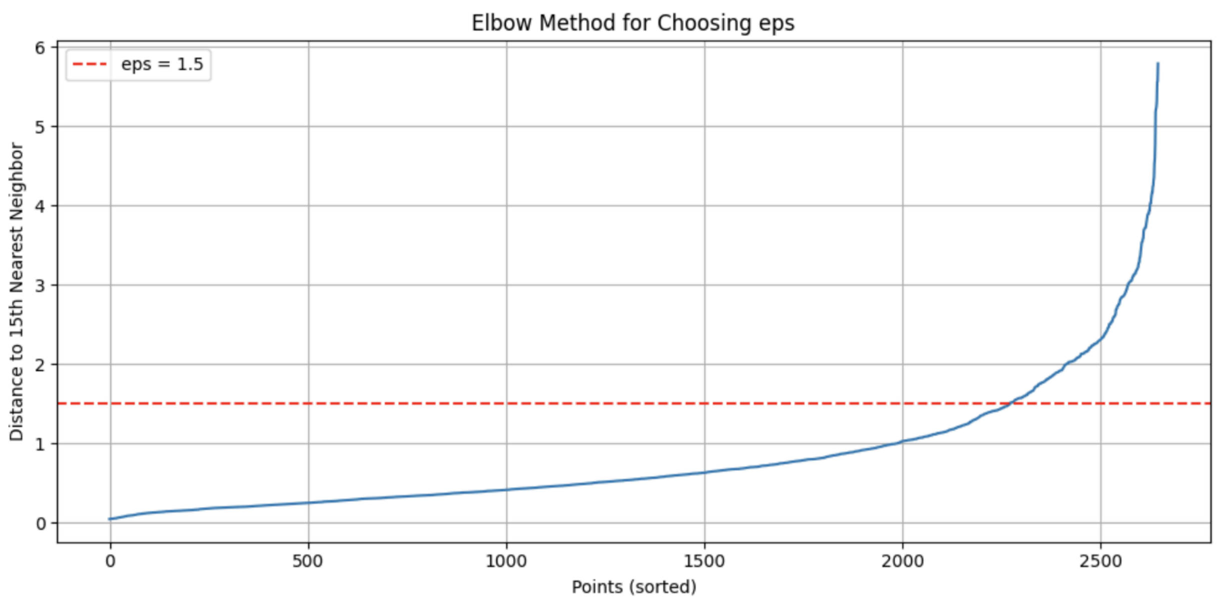

Figure 4: A graph showing the elbow curve I used to determine the epsilon. One would typically choose the eps where the graph is increasing rapidly.

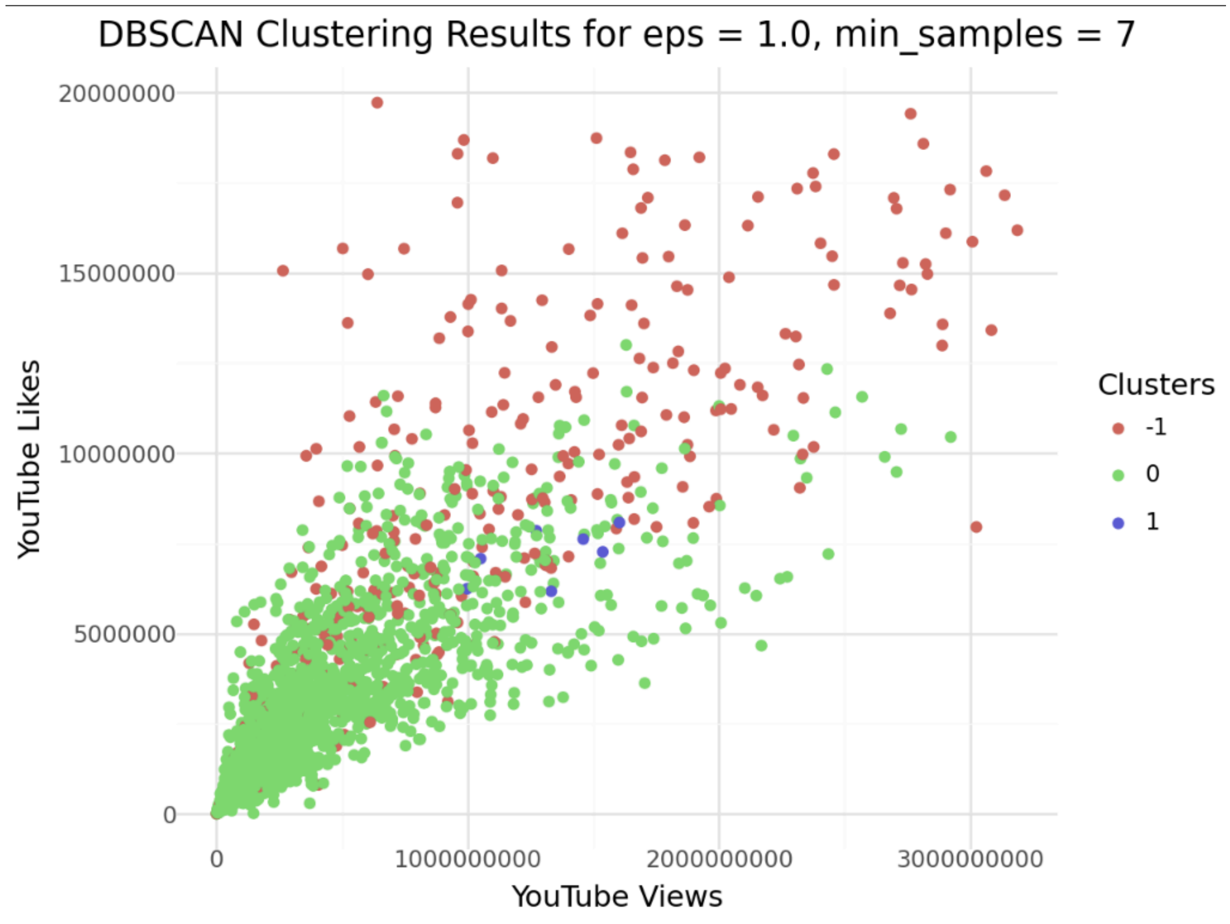

Figure 5: A graph showing a 2-cluster solution for YouTube likes and YouTube views using DBSCAN. The -1 cluster represents noise (points that are not assigned to a cluster)

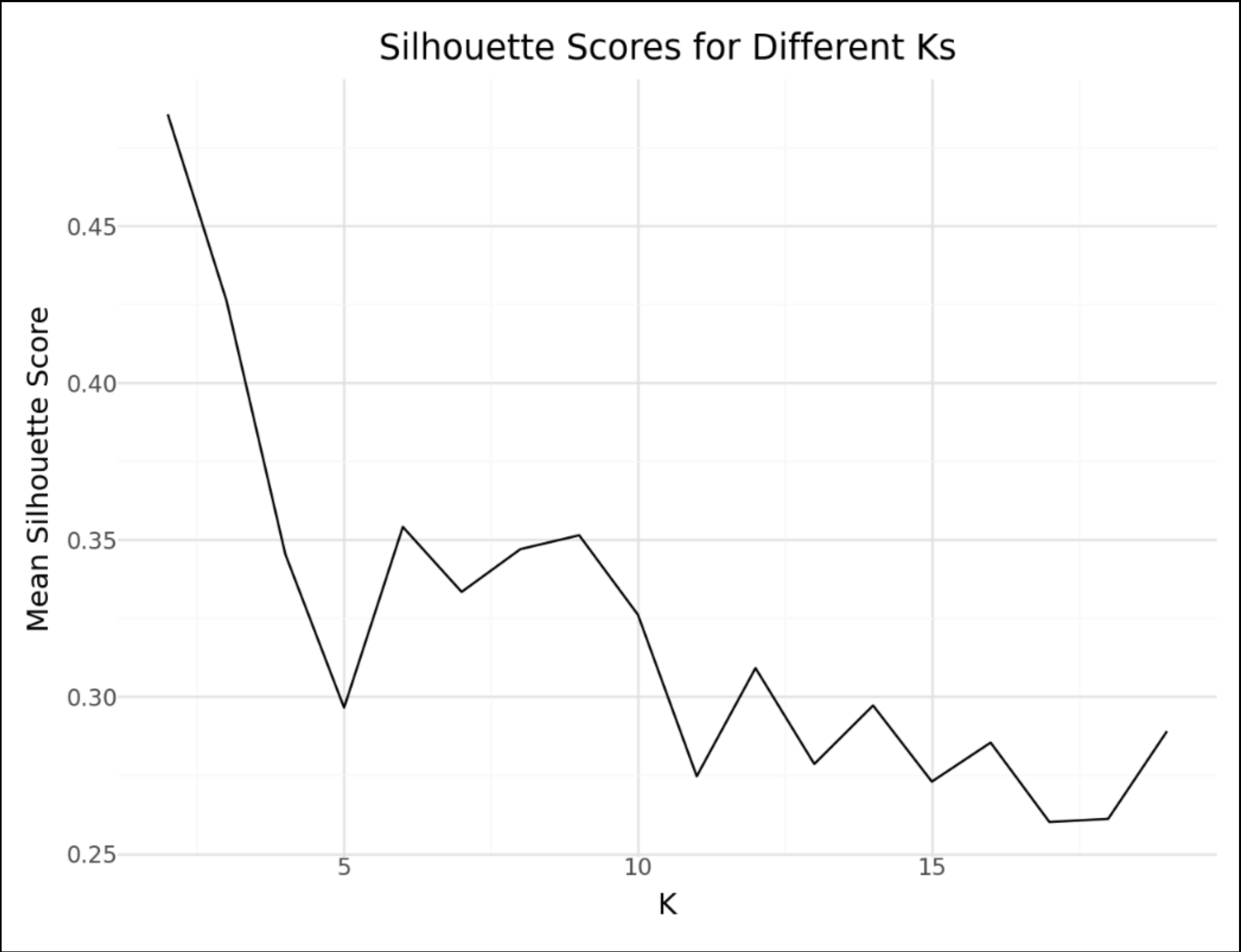

Figure 6: A graph used to show the silhouette scores for different numbers of clusters. Used to choose the number of clusters in a K-Means model

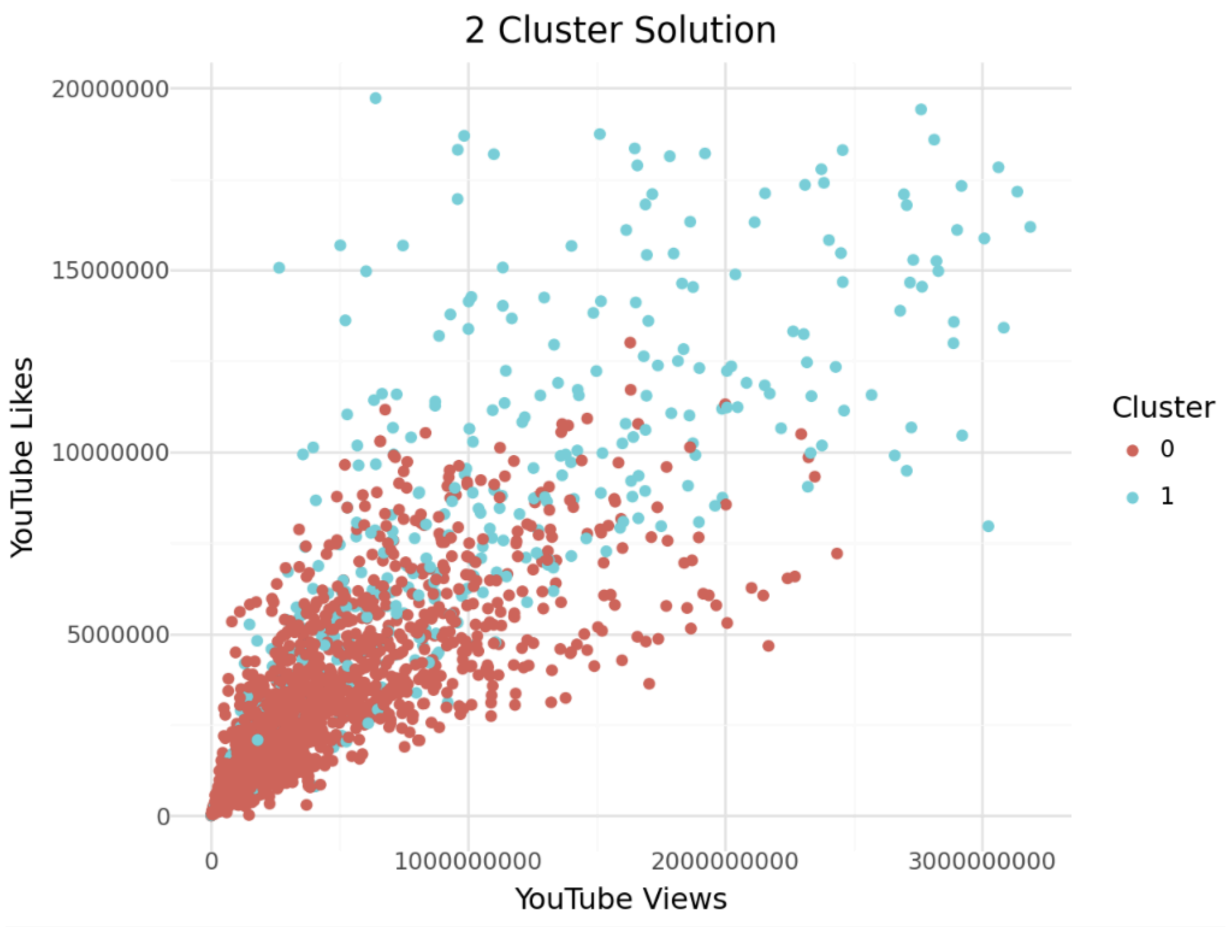

Figure 7: A graph showing the clustering of a K-means model using two clusters.

Discussion

In summary, while DBSCAN showed early promise by silhouette score alone, its inability to produce more than one meaningful cluster rendered it much less useful to this dataset. K-Means not only produced better-balanced clusters, which makes it a far better approach for segmenting songs based on social media behavior. This leads to the conclusion that K-Means may be better in terms of structured and interpretable clusters in this case, whereas DBSCAN may have a future place for identifying outliers or noise in potentially more irregularly grouped data distributions.