Song Popularity Predictors and Similar Songs

Introduction and Dataset

Throughout people's lives, many people gravitate toward specific songs that were popular during their most impressionable years [1]. For example, people in their 50s in the 2020s generally have favorite songs that were released in the 1980s as compared to those released more recently. If you are an adult, it may be difficult to find similar songs to your favorite songs that were released more recently. On the other hand, if you are a child, it may be difficult to find similar songs to your favorite songs that were released more than two decades ago. For this reason, we seek to find a method to suggest to a user a new song based on the current song they are listening to. To accommodate this, we also seek to create a playlist that varies the popularity of the song to improve the likelihood the user identifies a new song that he or she will enjoy. In order to do this, we seek to predict the popularity of a song based on its attributes, since the popularity of a song may change but its attributes will always remain the same. With this motivation in mind, our project will more succinctly investigate the following three questions in order:

1. Can we predict whether a song is popular or not based on its attributes?
2. Can we predict whether a song is relatively more popular than another based on their attributes?
3. Can we suggest to a user a new song based on the current song they are listening to?

To investigate our questions, we found a dataset that includes over 170,000 songs from Github (spags093). These songs were released as early as 1920 to as recently as 2021. The dataset includes a wide variety of songs, from the 1939 Spanish guitar concerto "Concierto de Aranjuez" by Joaquín Rodrigo to Olivia Rodrigo's 2021 pop hit "Driver's License". Within the original dataset, the Spotify API contributes the important features that we will leverage to describe each song. The descriptions of each of these attributes are provided in Table 1.

Table 1. Spotify Dataset Feature Descriptions [2] [3]	
Acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
Artists	The artists who performed the track.
Danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements, including

	tempo, rhythm stability, beat strength, and overall regularity. 1.0 is most danceable, 0.0 represents least danceable.
Duration	The duration of the track in milliseconds.
Energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
Explicit	Whether or not the track has explicit lyrics. 1 represents yes. 0 represents no.
ID	The Spotify ID for the track.
Instrumentalness	A confidence measure of whether the track contains no vocals. 1.0 represents greater likelihood the track contains no vocal content.
Key	The key the track is in. 0 represents C, 1 represents C#, 2 represents D and so on. -1 represents no key was detected. Keys range from -1 to 11.
Liveness	Detects the presence of an audience in the recording. A value above 0.8 provides strong likelihood that the track is live.
Loudness	The overall loudness of the track in decibels (dB). Loudness values are averaged across the entire track. Values typically range between -60 to 0 dB.
Mode	Mode indicates the modality of the track. Major is 1. Minor is 0.
Name	The name of the track.
Popularity	The popularity of the track. The value will be between 0 and 100 with 100 being the most popular.
Release Date	The date the track’s album was first released.
Speechiness	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording, the closer to 1.0 the attribute value.
Tempo	The overall estimated tempo of a track in beats per minute (BPM).

Year	The year the track was released.
Valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive, while tracks with low valence sound more negative.

Importantly, as described in Table 1, Spotify's popularity score is a value between 0 and 100, with 100 being the most popular. Per Spotify, the popularity is calculated from the total number of plays the track has had, with more recent plays being weighted more strongly. Therefore, we should note here that as a song ages, its popularity score will assumedly continue to decrease. It is not a constant value assigned by, for example, its peak popularity. This attribute will warrant additional investigation in EDA.

Data Cleaning & EDA

To begin, our group noticed that there are many songs with the same name in our dataset, so we decided to sort songs by name. We discovered that most of the songs with the same name have similar features but with different IDs. One possible explanation for this is the phenomenon that many artists decide to re-release songs. The songs may have been released as singles and subsequently re-introduced into the artists' albums. We decided to delete these duplicates, and our dataset size decreased from 170K to 137K. By deleting the duplicates, we expect to have a more unbiased dataset, which will be helpful when we build our models.

Secondly, we identified inconsistencies in the "release date" feature. Many release dates were in the format of month-day-year [1/29/1920] but others were only listed by the year [1920]. The latter is redundant with the "year" feature. We initially decided to revise this column to simply be "release month" because we were interested in whether the release month contributed to a song's popularity, e.g. summer hits versus holiday tunes. However, due to the aforementioned release date formats being inconsistent, i.e. many did not include a month, and we did not have a reasonable method to fill the missing months, our dataset size shrunk to well below our self-selected 100K threshold. We believed any gain we get from leveraging release month would be offset by the loss of sampling in our dataset. Therefore, we decided to not leverage "release date" at all in our modeling.

Following the cleaning, our group began to concentrate on the features. We started using Tableau for histograms and python for scatter plots to see if there were any trends or relationships between different numerical features. Excluding "Artists", "Name", and "ID", all other features are numerical data that are primarily about song characteristics,

such as "danceability" and "loudness", and we went through each numerical column to see if there were any anomalous data we should be aware of. We concluded that there were no anomalous data points that we should be concerned about. We then confirmed that “popularity” is highly related to “year”. We see from Figure 1 that a disproportionately large number of older songs have low popularity scores, while more recent songs are more likely to have greater popularity scores. We should be aware of this relationship when developing our models. Finally, we discovered that there are many non-linear relationships between different variables versus popularity. We decided to utilize both Pearson and Spearman correlations to examine some probable associations derived from our observations. The results, shown in Figure 2 and 3, show there are both positive and negative correlations between different variables versus popularity, some stronger than others, e.g. “year”, “instrumentalness”, and “loudness”. With these strong relationships, we felt confident that we could build models leveraging these features to answer our primary questions.

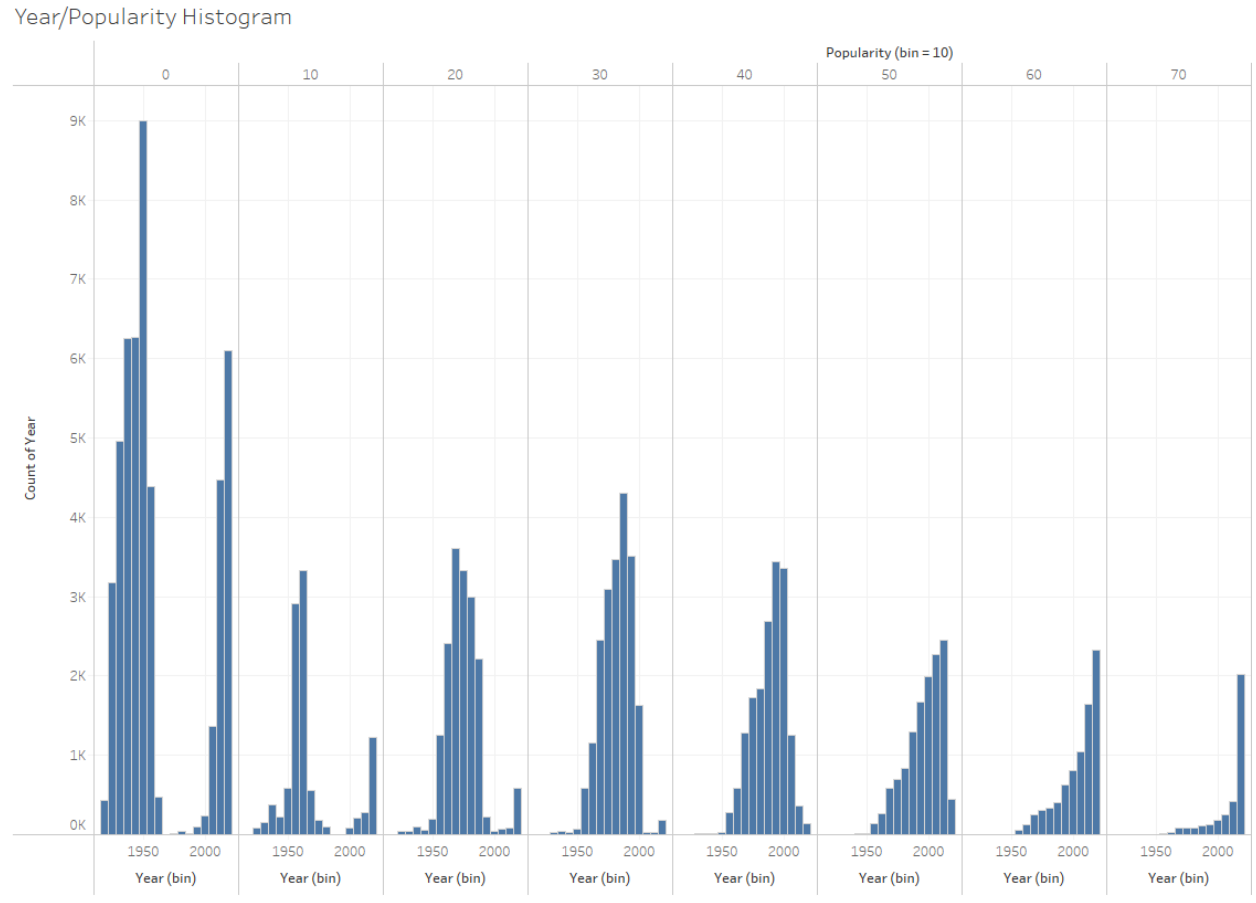

Figure 1. Song Count Histogram over Popularity and Year

Song Popularity Predictors and Similar Songs

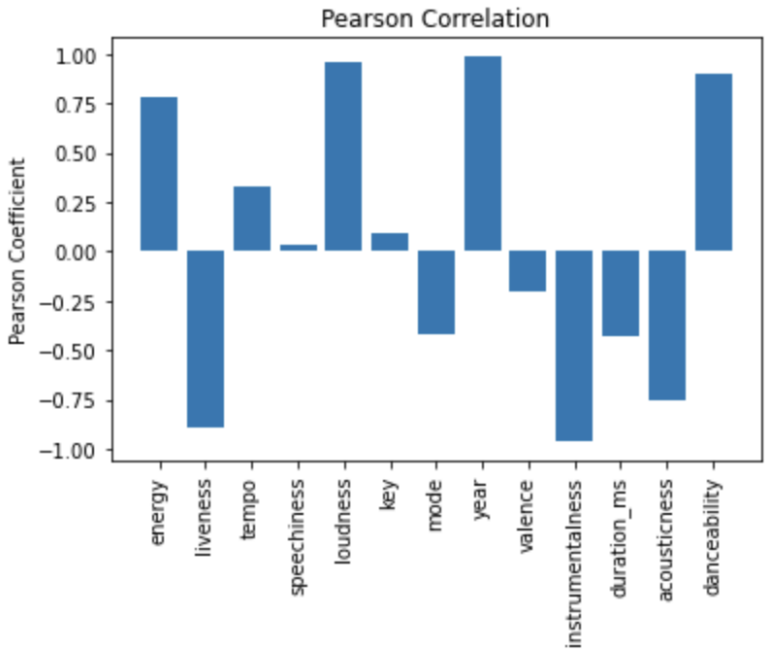

Figure 2. Pearson correlation coefficients vs dataset features

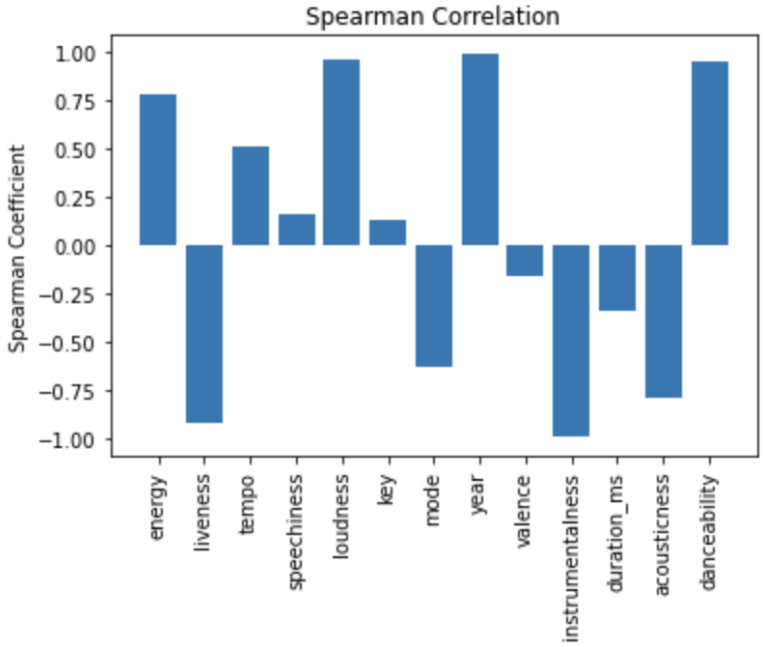

Figure 3. Spearman correlation coefficients vs dataset features

Data Modeling

Can we predict whether a song is popular or not based on its attributes?

The first question we wanted to look into was whether we can classify the songs into popular songs and not popular songs. To answer this question, we first separated our dataset of all 137K songs into popular and not popular songs based on the mean popularity score 26.196. We have about 69K songs that are not popular and 67K songs that are popular. We then used 80% of the data as training data and 20% of the dataset as testing data. The inputs for our model were all the numerical features in our dataset ('loudness', 'energy', 'instrumentalness', 'acousticness', 'liveness', 'duration_ms', 'speechiness', 'valence', 'danceability', 'tempo', 'explicit', 'key', 'mode', 'age'). The 'age' feature is the number of years since the song was released that replaced the 'year' feature in our dataset. We felt this was an appropriate method to expand the applicability of our models, since popularity depended on how recent a song was released and not necessarily the year it was released in. Finally, the target of our model is 0 for not popular data and 1 for popular data.

To reach a good model we tried different classifiers including logistic regression classifier, k-nearest neighbor classifier, decision tree classifier, random forest classifier, gradient boosting classifier, and voting and bagging ensemble classifiers based on different base classifiers. For all the models, we used 5-fold cross validation with the F1 scoring metric. We chose F1 to cautiously accommodate the difference in the size of the two classes. The highest score was the voting classifier with a random forest classifier base estimator with average 5-fold cross validation F1 score of 0.872 and an F1 score of 0.878 on the testing dataset.

To improve the performance of our model, we tried to tune the parameters: "criterion", "max_features", "warm_start", and "max_depth", but the best results typically resulted from the default parameters ("criterion" = 'gini', "max_features" = 'sqrt', "warm_start" = False, and "max_depth" = None). We set voting to 'soft' for our voting classifier in order to get the predicted probability for our testing dataset, allowing us to also ascertain the effectiveness of the classifiers from ROC curves. As seen in Figure 4, the voting classifier with random forest classifier base estimators still has the highest AUC score of all the better performing classification models.

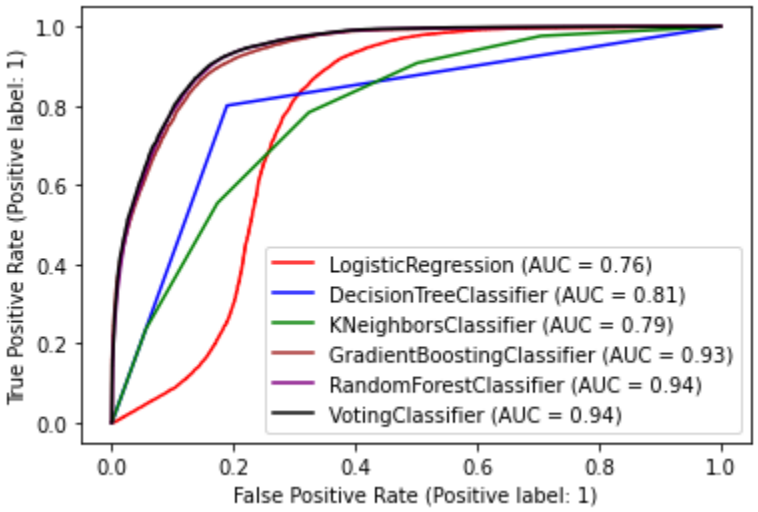

Figure 4. ROC curves for various classifiers

Can we predict the relative popularity of songs?

With moderate success in classifying songs as popular and not popular, we next attempted to predict the popularity score from a song's attributes. To accomplish this, we leveraged the same features mentioned earlier ('loudness', 'energy', 'instrumentalness', 'acousticness', 'liveness', 'duration_ms', 'speechiness', 'valence', 'danceability', 'tempo', 'explicit', 'key', 'mode', 'age') to predict a popularity score. We again used all 137,000+ songs for this analysis, using a similar 80-20 training-testing split. To identify the best model for this question, we compared R^2 scores after 5-fold cross validation. We began with an optimistic basic linear regression. Thereafter, in seeing the non-linearity of the data and the variability of the songs, we investigated support vector regression, nearest neighbors regression, and decision tree regression. With unsatisfactory results, we finally investigated ensemble methods AdaBoost regression, random forest regression, and gradient boosting regression. The model scores are shown below in Table 2.

Table 2. Regression Model Comparison

Model	**R^2 scores**
Linear Regression	0.41
Support Vector Regression	0.08
Nearest Neighbors Regression	0.28
Decision Tree Regression	0.35

AdaBoost Regression w/ Decision Tree Regressor base estimator	0.61
Random Forest Regression	0.68
Gradient Boosting Regression	0.64

With random forest regression and gradient boosting regression being the models with the best success after cross validation, we tuned the models to reach a final R^2 value of 0.68. As seen in Figure 5, the gradient boosting regressor peaked/plateaued at an R^2 value of 0.68 with a learning rate of 0.07 and a max depth of 10. Both of these models also have an R^2 value of 0.69 on the testing set. We were not necessarily satisfied with these values, but, most importantly, we wanted to investigate the relative popularity of songs, rather than how well we could reproduce Spotify's popularity score. Notably, the popularity score itself is less important since a song's popularity score will presumably decrease as the song ages. Therefore, we investigated the pairwise ranking accuracy of our models. Pairwise ranking accuracy assesses each song's ranking amongst its peers and calculates how many of its peers it is still ranked higher than over all possible pairs of songs. The gradient boosting regressor produced a pairwise ranking accuracy of 0.82, which we consider a moderate success due to variability of songs and user interest over time.

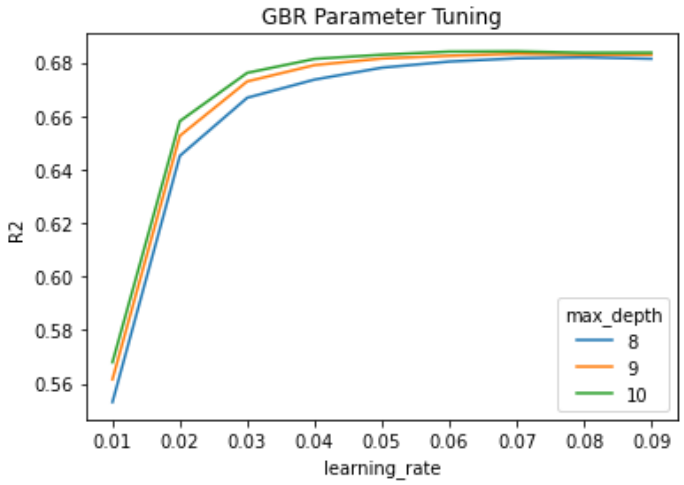

Figure 5. Gradient boosting regressor parameter tuning

Finally, we reviewed the most important features that contributed to our gradient boosting regressor. As shown in Figure 6, we confirmed that "year" is an important feature when predicting popularity, reaffirming our discussion of the Spotify popularity

algorithm earlier. The next most important features were “instrumentalness” and “loudness”, which are in line with our initial correlation assessment as well.

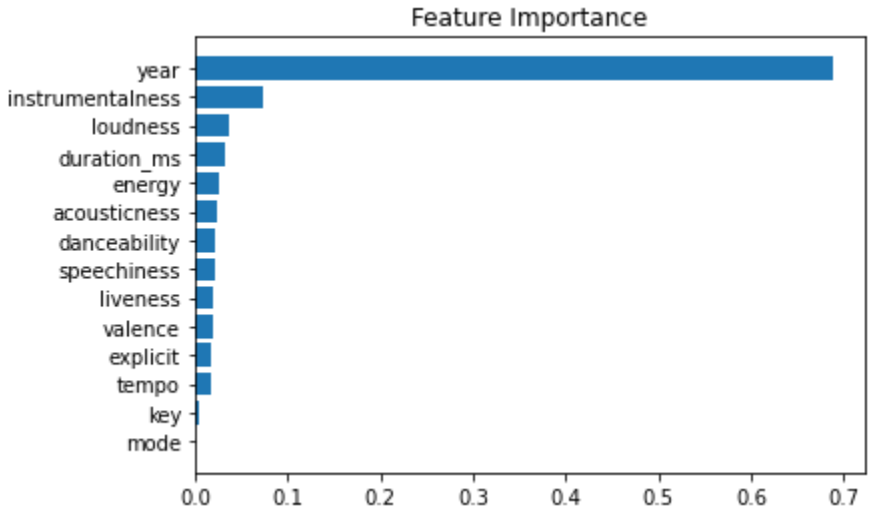

Figure 6. Feature importance for gradient boosting regressor predicting popularity

Can we suggest to a user a new song based on the current song they are listening to?

The third point we wanted to address is if we could locate comparable songs based on song qualities to recommend similar songs to users. We chose k-means clusters to assist to cluster the songs based on their attributes. When new songs are added, we can use the model to predict which cluster will be the best fit and suggest music from that cluster to users. To begin, we selected all of the numerical values that constitute song attributes as our variables (‘acousticness’, ‘danceability’, ‘duration_ms’, ‘energy’, ‘explicit’, ‘instrumentalness’, ‘key’ , ‘liveness’, ‘loudness’, ’mode’, ‘speechiness’, ‘tempo’, ‘valence’, ‘year’). We then used PCA to reduce the 14 dimensions to 2 dimensions, since PCA reduces noise in the data, allows for feature selection (to a certain extent), and produces independent, uncorrelated data features. After that, we used the distortion score elbow method to select the best number of clusters. The distortion score is the average of the squared distances from the cluster centers of the respective clusters, and the point with the most rapidly decreasing slope will be our optimized number of clusters, which will be 4, as seen in Figure 7. Figure 8 shows the resulting clusters over the PCA-reduced dimensions.

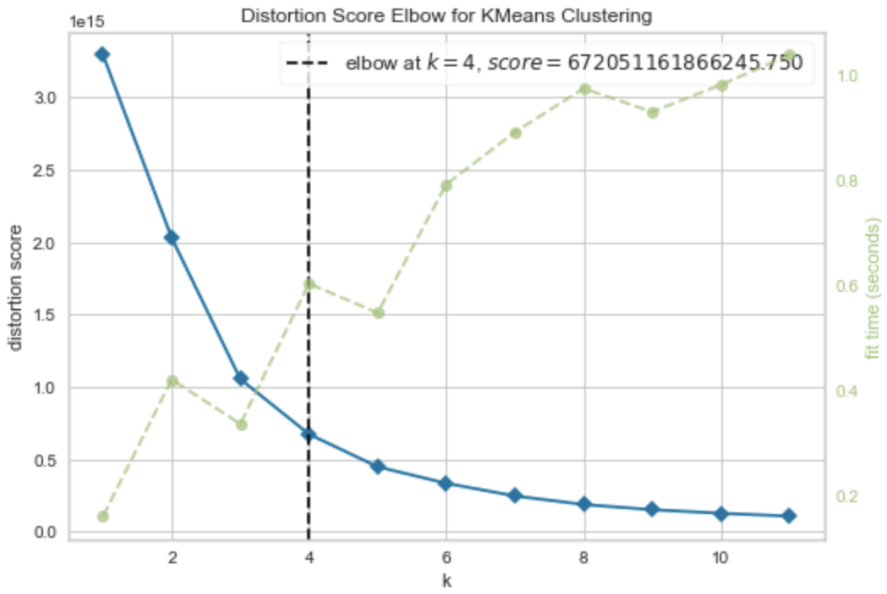

Figure 7. Elbow method for determining number of clusters for 2-dimension PCA-reduced data

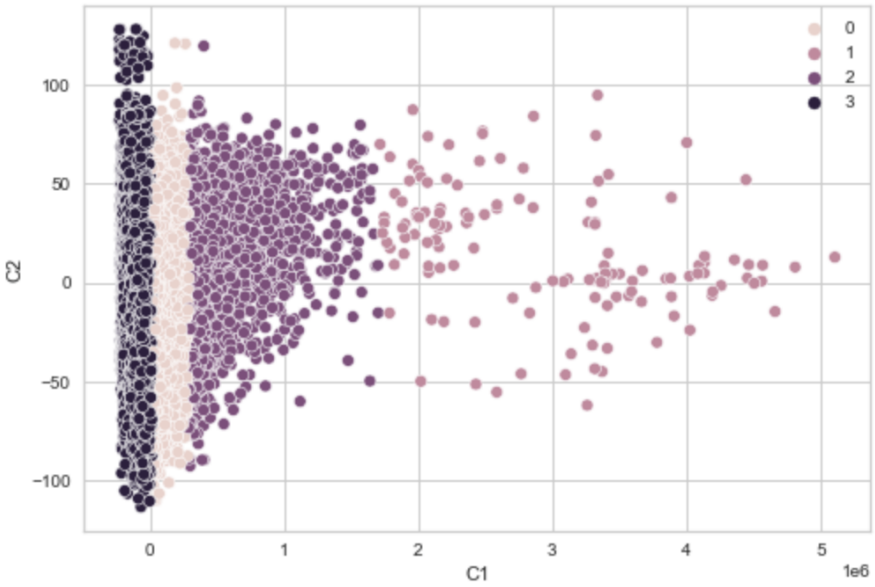

Figure 8. K-mean clustering over 2-dimension PCA-reduced data

Next, we wanted to see if we could suggest songs to a user based on attributes that are more favored by the user. For example, if a user has a specific taste, such as loud music, could we recommend a song that is in a similarly loud and popular cluster. For each feature (‘acousticness’, ‘danceability’, ‘duration_ms’, ‘energy’, ‘explicit’,

‘instrumentalness’, ‘key’, ‘liveness’, ‘loudness’, ‘mode’, ‘speechiness’, ‘tempo’, ‘valence’, ‘ year’) versus “popularity”, we used k-means clustering with the elbow method to choose the best number of clusters. An example is shown in Figure 9 and 10. From this k-means clustering methodology over both PCA-reduced dimensions and specifically chosen dimensions, we have a reasonable method to suggest new songs to a user based on specific similarities.

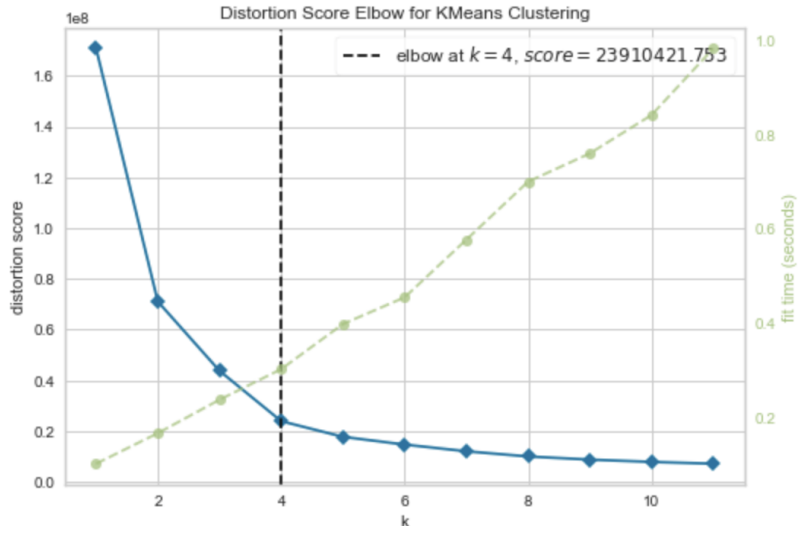

Figure 9. Elbow method for determining number of clusters for “year” vs “popularity”

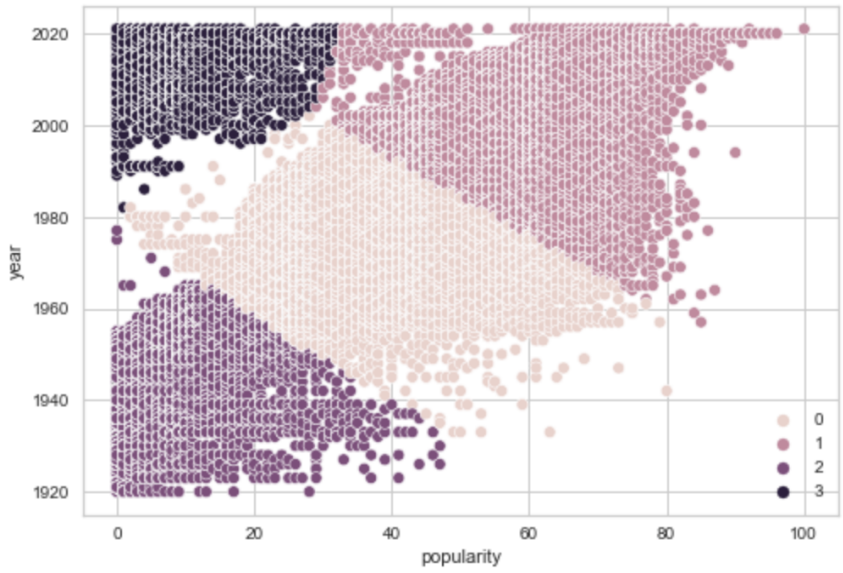

Figure 10. K-mean clustering over “year” vs “popularity”

Group I - Wanqin Chen, Yuwei Wang, Eric Li 6/7/2022

Song Popularity Predictors and Similar Songs

Summary

For this project, we investigated three primary questions:

1. Can we predict whether a song is popular or not based on its attributes?
2. Can we predict whether a song is relatively more popular than another based on their attributes?
3. Can we suggest to a user a new song based on the current song they are listening to?

To answer question 1, we found that we could use a voting ensemble classifier with a random forest classifier base estimator with default parameters to predict whether a song is popular or not popular. This model has an F1 score of 0.878 over our testing dataset, which we believe is satisfactory for our current goals. Next, to answer question 2, we arrived at a gradient boosting regressor with a learning rate of 0.07 and a max depth of 10 to produce a 0.82 pairwise ranking accuracy score. We believe that 82% is a reasonable accuracy score based on the variability of songs and user interest over time. Finally, we investigated PCA dimensionality reduction and k-means clustering to cluster similar songs to effectively suggest new songs to a user based on their current song of interest. From the elbow method, we determined that 4 clusters were most effective at describing the dataset. We believe this provides us with a reasonable sample of songs to suggest to a user while also allowing us to leverage our popularity predictors from question 1 and 2 to effectively produce a song playlist of enough variability. In summary, from these models, we can produce a song playlist of varying popularity given a song of interest to introduce new songs to users and better connect the musical world.