212 Final Project Report

By Group 3: Xueting Li, Jialu Li, Sammi Chang, Junguo He

How to Produce Popular Music

Abstract: According to previous research, the popularity of music can be explained by the inherent features of the music. Spotify Web API makes this idea testable by providing the information including audio features of songs published. Collecting data via Spotify Web API, we are allowed to collect information about songs and conduct analysis. In this analysis report, we first show how we connected to Spotify API and gathered the data. Then we described and gave some insights based on data visualization. Finally, we established classification models to enhance our conclusions. According to the analysis, we found that artists' followers have an overwhelming influence on the popularity of songs. Putting artists' followers aside, a song's loudness, speechiness and tempo have the greatest effect among all audio features.

1. Introduction

What features contribute to a popular song? How can a song, although newly released, become a hit? In this study, we explore the common features among popular songs, such as the loudness, tempo, and key. In addition, we analyze the extent to which the number of Spotify followers of the artist affects the popularity of a track.

Spotify provides us an opportunity to explore these questions by collecting data via their API. With the Spotify Developer Platform, we are able to read calculated audio features of tracks to learn about its danceability, energy, valence, and more. For more advanced use cases, it is possible to obtain an in-depth analysis data about tracks such as the audio features.

If we find evidence that there are indeed some common features that contribute to the popularity of the songs, it may inspire music producers to adjust some parameters of their music products. On the other hand, it may help operational managers of music companies to improve their music recommendation algorithms for users.

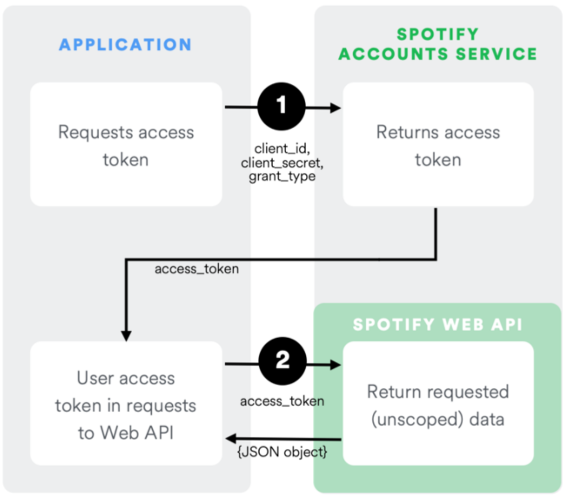
2. Data Collecting through Spotify Web API

2.1 Brief Introduction of Spotify Web API

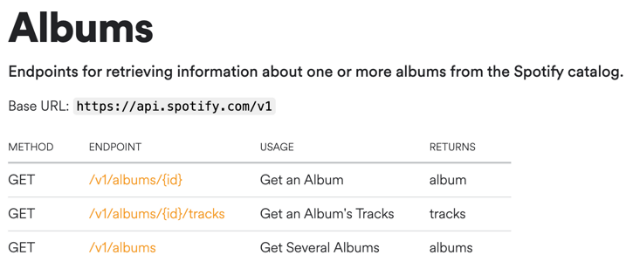
To start our analysis, we choose our sample and gather data. Spotify Web API offers several sources for developers and analysts to use the data from Spotify music platform from different endpoints. For a better understanding of the following steps, we make a simple introduction of Spotify Web API first.

Spotify Web API provides three ways to get authorized: authorization code flow, implicit grant flow and client credentials flow. We selected the last one for our study. Since we chose Client Credentials Flow, we demonstrate the following the steps below:

- Requesting the access **token** from Spotify to acquire authorization.



- Turning to the **endpoints** we need

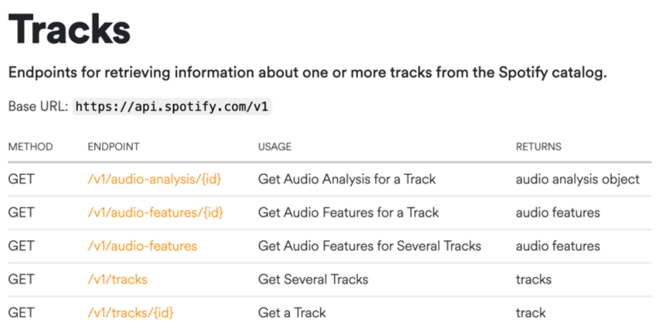


Albums

Endpoints for retrieving information about one or more albums from the Spotify catalog.

Base URL: `https://api.spotify.com/v1`

METHOD	ENDPOINT	USAGE	RETURNS
GET	/v1/albums/{id}	Get an Album	album
GET	/v1/albums/{id}/tracks	Get an Album's Tracks	tracks
GET	/v1/albums	Get Several Albums	albums



Tracks

Endpoints for retrieving information about one or more tracks from the Spotify catalog.

Base URL: `https://api.spotify.com/v1`

METHOD	ENDPOINT	USAGE	RETURNS
GET	/v1/audio-analysis/{id}	Get Audio Analysis for a Track	audio analysis object
GET	/v1/audio-features/{id}	Get Audio Features for a Track	audio features
GET	/v1/audio-features	Get Audio Features for Several Tracks	audio features
GET	/v1/tracks	Get Several Tracks	tracks
GET	/v1/tracks/{id}	Get a Track	track

- Using indexes, such as album ids and track ids, we get access to the data points and download the data.

2.3 Sample Selection and Data Collecting

For our study, we aimed at figuring out the contribution of each of the features to popularity. Fortunately, Spotify published both the popularity and the features of each song. We were able to acquire information of the albums by searching the endpoints[1]. However, we encountered an obstacle: Spotify only returned 50 results per search, but we have far more than 50 instances as our sample. We figured out one way to resolve this in order to obtain a list for indexing.

Firstly, we obtained the album lists from the Billboard website by web scraping and determined the Billboard Top 200 Albums to be our searching index. Three points support this decision: 1. Using Billboard ranks can help filter out albums that do not contain pop music. 2. Two hundred albums a year covers most of newly released albums. 2. The Billboard website is easy for web scraping.

We set our sample period to be from 2005 to 2020, and then we used the album list we scraped from Billboard to collect album information from Spotify. Through the 'Search' endpoint, we finally attained 2433 unique albums and the following variables:

Album

- album id
- album name

[1] GET https://api.spotify.com/v1/search

- artist id
- artist name
- release date

In the next step, we used the Spotify album id as an index to acquire the data of artists and tracks of albums through the Artist and Album endpoints. At this step, we obtained the following variables:

Artist	Track
• artist id	• track id
• artist name	• track name
• genres	• duration_ms

Finally, we use the track ids as an index to collect data of track features through Track endpoints. After filtering out the duplicated observations, we acquired 23,487 unique tracks. According to the Spotify Web API, the track features and their definitions are showed below:

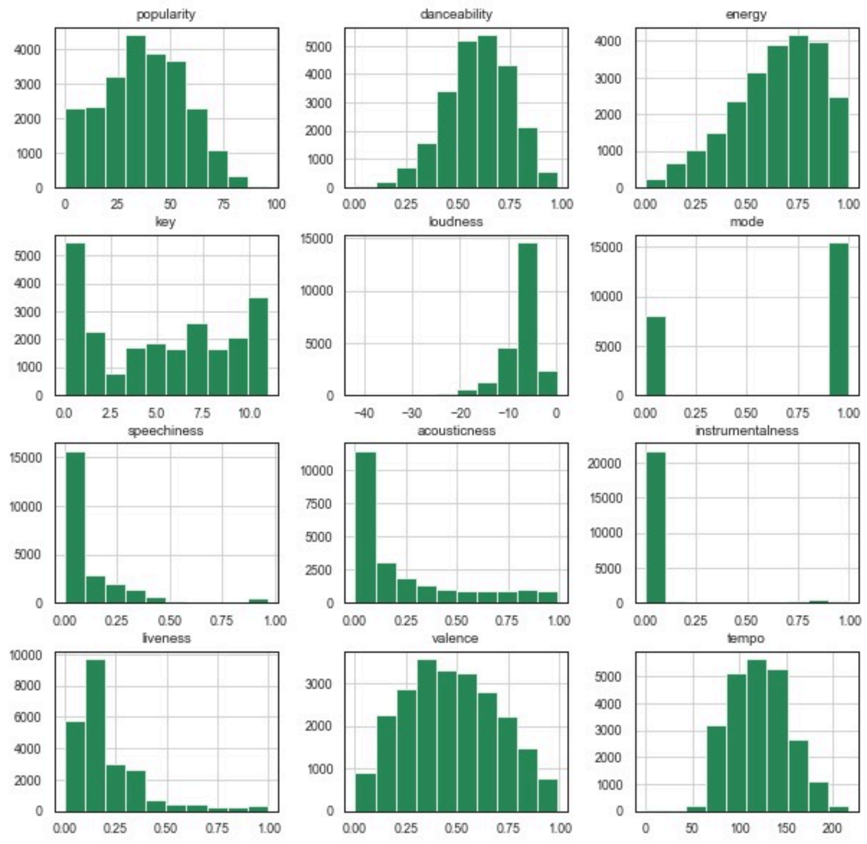
Key	Type	Value Description
Track_Id	string	The Spotify ID for the track.
Popularity	int	The popularity of the track. The value will be between 0 and 100, with 100 being the most popular. The popularity of a track is a value between 0 and 100, **with 100 being the most popular**. The popularity is calculated by algorithm and is based, in the most part, on the total number of plays the track has had and how recent those plays are. Generally speaking, songs that are being played a lot now will have a higher popularity than songs that were played a lot in the past. Duplicate tracks (e.g. the same track from a single and an album) are rated independently. Artist and album popularity are derived mathematically from track popularity. Note that the popularity value may lag actual popularity by a few days: the value is not updated in real time.
Key	int	The estimated overall key of the track. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, … 12 = B. If no key was detected, the value is -1.
Mode	int	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is represented by 0.
Time_Signature	int	An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
Acousticness	float	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
Danceability	float	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. A value of 0.0 is least danceable and 1.0 is most danceable.
Instrumentalness	float	Predicts whether a track contains no vocals. “Ooh” and “aah” sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly “vocal”. The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
Liveness	float	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
Loudness	float	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typical range between -60 and 0 db.
Speechiness	float	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
Valence	float	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).
Tempo	float	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

The popularity variable is our object of desire and our dependent variable since the study aims to predict the popularity of an audio track. The popularity variable is a value between 0 and 100, with 100 being the most popular. It is calculated by an algorithm based on the total number of plays the track has had and how recent those plays are. Acousticness is a measure, from 0 to 1, of whether the track is acoustic. With a value of 0, a song will have a more electric sound, and with 1, there will be less electric sound. The tempo attribute and loudness attribute, on the other hand, are self-explanatory. Danceability describes how suitable a track is for dancing based on a combination of several musical elements. If the danceability is small, the audio, for instance, will be more like the Irish-folk song “Danny Boy.” If it is closer to 1, then the audio will be more rhythmic, for instance, like the WALK THE MOON’s iconic dance track “Shut Up and Dance.” Energy is a measure that is similar to danceability. Instrumentalness represents whether a track contains no vocals, with 1 representing instrumental audio. Speechiness refers to the presence of spoken words in the track, like in the genre of rap. Valence represents the emotional attitude: positive or negative. Finally, the mode indicates if the track is using a major or minor key.

3. Insights from Data Visualization

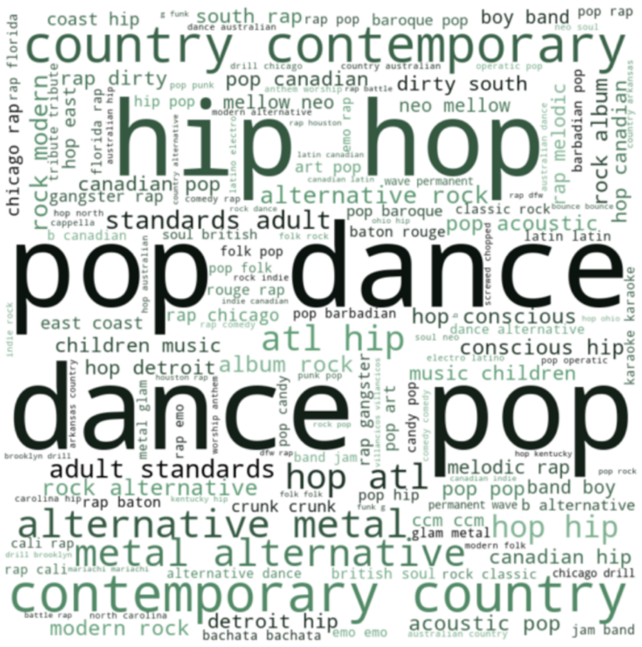
3.1 Overall Distribution of Variables



The overall distribution of variables shows that valence, tempo, and danceability look approximately normally distributed, although danceability has a slight skew. Popularity and energy are very skewed, which means that most values are gathered around the mode. Most songs have lower popularities, and most songs have higher energies. In the histogram of acousticness, the spike at 0 shows that most songs are minimally acoustic, which implies that electronic sounds are frequently used. The distribution for the binary variable, mode, shows that most songs are in major keys as 1 represents major and 0 represents minor. The distribution of instrumentalness shows most of the tracks are not instrumental audio.

3.2 Distribution by Genre

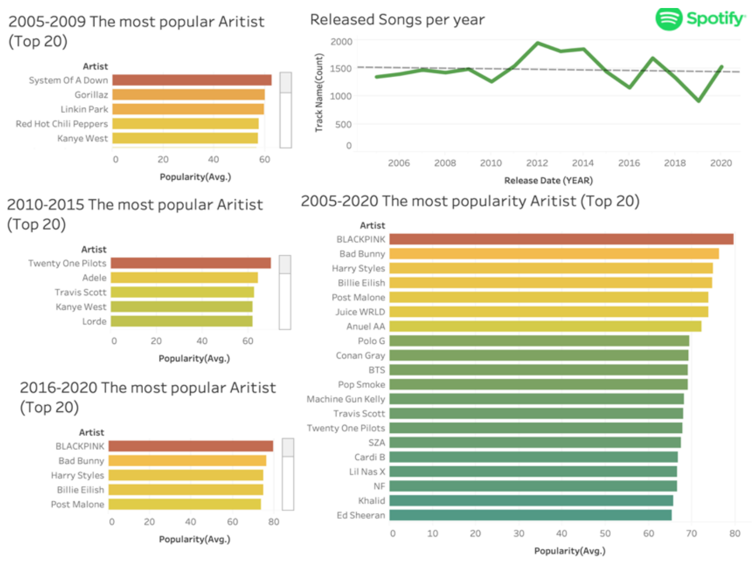
Using data from 2005-2020, the word cloud image implies how the genres are distributed amongst tracks. In the word cloud image below, the the bigger words are, the more frequently they appear among the tracks.



In the word cloud of Genres, Pop, Dance, Hip Hop, and Contemporary Country are most conspicuous, with the three largest genres being Dance, Hip Hop, and Pop. Because high danceability and speechiness and low acousticness is closely related to these genres, this further affirms our hypothesis that a song is popular when the danceability and speechiness is high and the acousticness is low.

In addition, it is worth noting that Contemporary Country, Alternative Metal, Rock and Children music are also very conspicuous, which means that they also represent a part of the popular trend.

3.3 The most popular Artists ranks

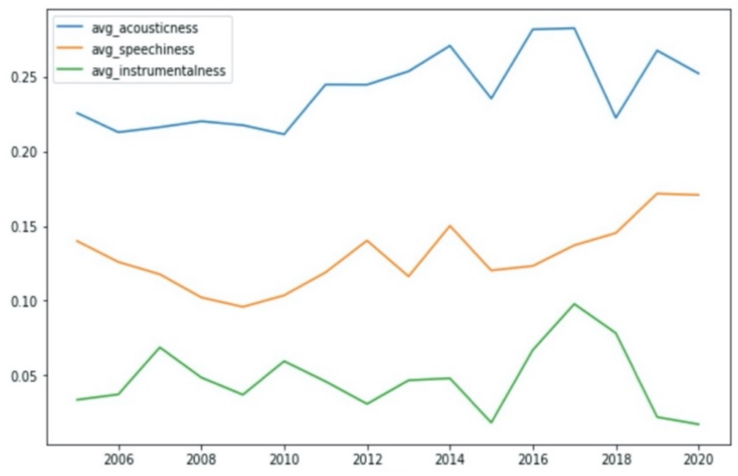


The most popular Artists rank for 3 different time intervals and also for the overall 16 years. The artists changed frequently in the ranks, but some keep their high popularity for more than 10 years. Recently, the most popular artist is Black Pink, Bad Bunny, Harry Styles, Billie Eilish and Post Malone. These top artists are also associated with our ranks for popular genres. For instance, the album that Blackpink released in 2020 included Pop and dance elements. So, we recommend that music producers, who want to make their songs popular, try styles like Dance, Pop, and Hip Hop. Interestingly, out of all genres that were popular in the 2000's, such as Alternative, Rock, R&B, Hip Hop, and Pop, only Hip Hop and Pop "survived," which insinuates that these two genres are the most robust, adaptable and versatile.

3.4 Tendency of popularity and audio features

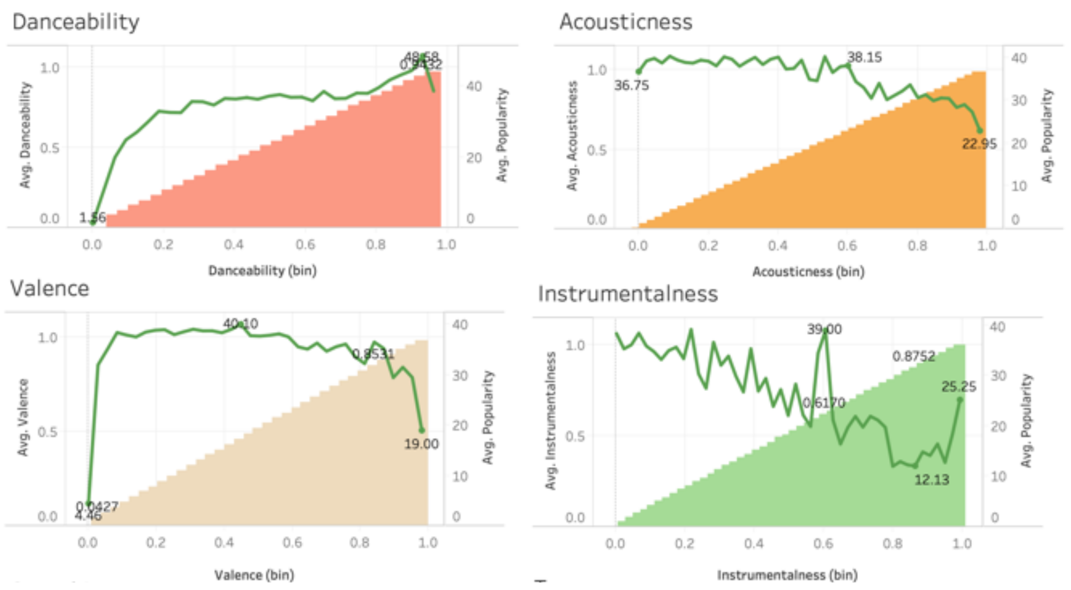
In the relationship between popularity and audio features plot, popularity is represented by the green line and the features are shown as different colors.

Features Through Time

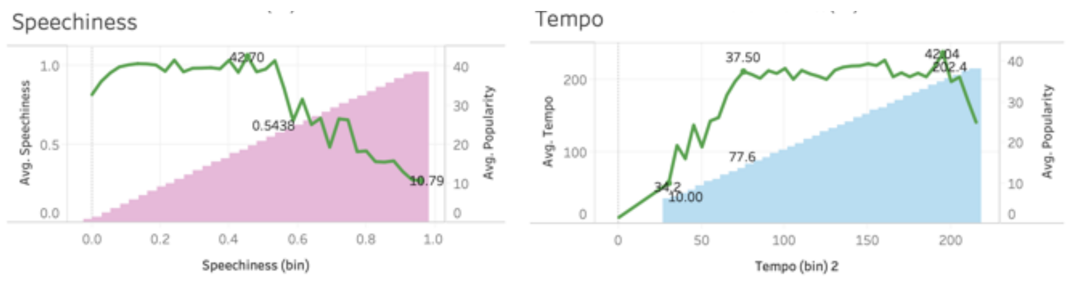


Plot of features' values against time. Features change by release years

Average Popularity vs. Features



Line graph describing the relationship between average popularity and audio features. The x-axis shows the value of the audio features (binned) and the y-axis measures the average popularity. -1



Line graph describing the relationship between average popularity and audio features. The x-axis shows the value of the audio features (binned) and the y-axis measures the average popularity. -2

According to the respective graph, popularity drops significantly when danceability is lower than 0.2, and popularity peaks when danceability is equal to 0.95. Overall, when it is greater than 0.2, the average popularity will remain in an acceptable range above 35, which indicates that danceability is an important variable for prediction.

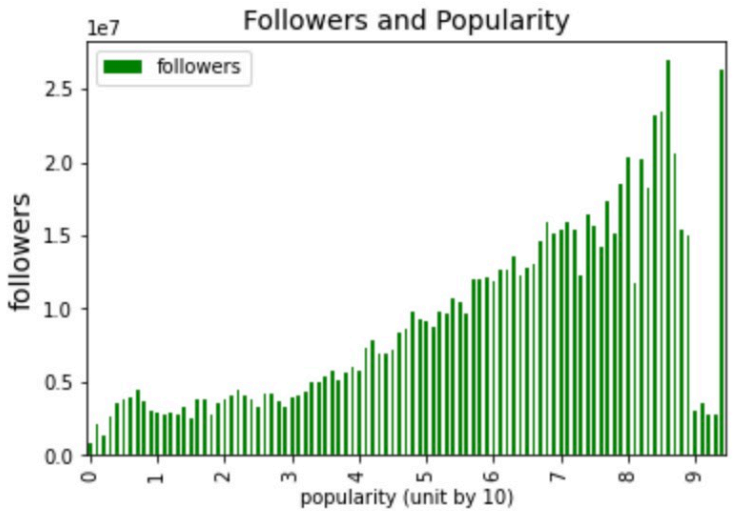
Acousticness also shows better average popularity when the value is higher rather than lower. Combined with the “Features Through Time” plot, we see that acousticness increases up until 2017 and then trends downward. This may indicate that the general public has, up until that point, been developing a preference for more acoustic music. Conversely, after 2017, electric sounds become more popular. Artists more or less add electronic elements to their music.

Valence plot shows that when the value is less than 0.1 and larger than 0.84, the popularity goes straight downward, which indicates that audiences don’t like the overwhelmingly emotional music.

Overall, the instrumentalness graph shows that popularity decreases with the increment of instrumentalness. However, popularity reaches a peak when instrumentalness is 0.62; and when instrumentalness approaches 0, which refers to instrumental music, the average popularity is also desirable.

Speechiness increases through time, especially in the recent 2 years, as shown in the “Features Through Time” graph, which gives us an insight that there is increasingly more rap in music. Comparatively, the average popularity is relatively high when speechiness is between 0 - 0.5. Lastly, for the tempo plot, when tempo is in the interval between 75 - 195, popularity remains in a sensible interval around 40.

3.5 The follower distribution per popularity intervals



Graph of popularity and the average number of followers of the musician. Popularity is in units of 10 and followers is in tens of millions.

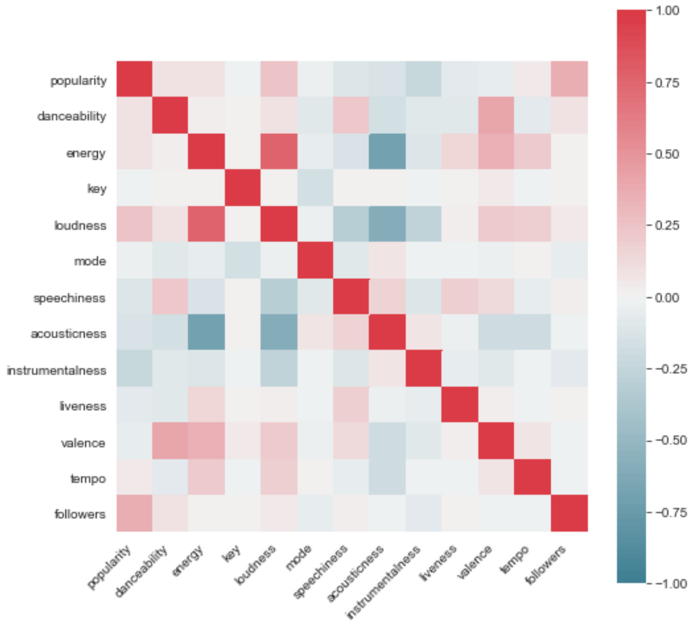
The last graph visualizes popularity and average Spotify followers. The above plot shows the followers' distribution per popularity intervals by 10. Overall, they have a positive correlation, and the highest amounts of followers crowd at popularity over 60. However, when popularity is between 90 and 96, the number of followers decreases. One possible explanation could be Tik Tok. Tik Tok is able to create hit songs in one night by virality, but it doesn't make the artists themselves famous. Therefore, these songs have a high popularity but fewer followers. However, when the popularity is 96, the number of followers jumps back to a high level of 25 million.

4. Analysis

We find the best features contributing most to the popularity of a newly released song. Therefore, we build classification models to predict our target variable, popularity.

4.1 Basic EDA

In the first step, we make a simple correlation matrix for all the attributes we use in our models. The correlation heatmap is showed below:



As we can see from the correlation heatmap, the attribute ‘Followers’ is far more correlated with popularity than others. It is quite reasonable, because a popular singer always draws more attention to his/her albums, and their songs can become a hit one no matter the album’s quality. Moreover, popular artists also have more resources to produce or obtain a hit melody. Therefore, we attempted prediction in two different ways: one that leaves out the attribute, ‘Followers’, and another that includes it. We hope to find the most influential audio features besides ‘Followers’.

4.2 Classification Models

To build our models, we must define a ‘popular’ song and an ‘unpopular’ song by setting a cut off point for our class variable, popularity. Referring to the distribution of the attribute ‘popularity’, we found that there are few songs that report a popularity of over 60, which is only 3 thousand out of over 20 thousand tracks. Therefore, we cut the sample into two bins: tracks with popularity over 60 are classified as popular songs and others as unpopular songs. We labeled the popular songs as 1 and unpopular ones as 0 under the new binary popularity attribute. As with many other classification models, we run into a class imbalance problem. Since there are so many more instances of class 0 than of class 1, the models we train will tend to classify most songs as class 1. Even though this results in an extremely high accuracy, the results are ultimately impractical because most, if not all, of class 1’s are categorized incorrectly. Therefore, we use two methods: we oversample the minority class and also try SMOTE (Synthetic Minority Over-sampling Technique) on the training set. In this way, our classification models are able to capture more variation in the data. The non-preprocessed testing set is then used to validate the models.

Since we have more than 20 thousand unique tracks as our sample, we can build a both a large training set and test set. We split our sample with 2/3 training data and 1/3 testing data, which means we have 15,658 tracks in training set and 7,829 tracks in test set.

Since this is a binary classification problem, we used the following the models:

1. K-Nearest Neighbors

2. Gaussian Naive Bayes

3. Decision Trees

4. Random Forests

4.2.1 K-Nearest Neighbors

The K-nearest neighbors algorithm (KNN) is a non-parametric method proposed by Thomas Cover used for classification and regression. In KNN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small).[2] As an entry-level model, KNN provides good support for nonlinear problems due to its simplicity and reliability. Although all samples need to be saved, KNN is still active in various fields and provides relatively robust identification results. Since KNN algorithm calculates the relative distance between pairs of instances, we need to scale the data first.

After scaling the data, we used un-preprocessed data to conduct the model training and testing and obtained the table below. The second column in the weighted avg row is the overall accuracy of the model.

Table 4-1 Result with Non-preprocessed Sample by KNN

	WITH 'FOLLOWERS'				**WITHOUT 'FOLLOWERS'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.87	0.99	0.93	6772	0.86	0.99	0.92	6772
1	0.52	0.07	0.12	1057	0.1	0.01	0.01	1057
Accuracy			0.87	7829			0.86	7829
Macro Avg	0.7	0.53	0.53	7829	0.48	0.5	0.47	7829
Weighted Avg	0.82	0.87	0.82	7829	0.76	0.86	0.8	7829

To deal with the imbalanced data issue, we used oversampling and SMOTE methods. We conducted training with the oversampled data.

Table 4-2 Result with Oversampling by KNN

	WITH 'FOLLOWERS'	**WITHOUT 'FOLLOWERS'**

[2] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.9	0.73	0.81	6772	0.91	0.5	0.65	6772
1	0.23	0.51	0.31	1057	0.18	0.68	0.28	1057
Accuracy			0.7	7829			0.53	7829
Macro Avg	0.57	0.62	0.56	7829	0.54	0.59	0.46	7829
Weighted Avg	0.81	0.7	0.74	7829	0.81	0.53	0.6	7829

In the above table, we see that the overall accuracy is not very high. At a 50% recall rate for class 0, the model sacrifices a gain in recall for class 1 for more precision and less recall in class 0. When SMOTE was conducted, the opposite occurred: recall for class 0 rose by .2, but recall for class 1 dropped to under .5. Overall, KNN does poorly in the prediction of popularity.

SMOTE (Synthetic Minority Over-sampling Technique) is a technique that generates new observations by interpolating between observations in the original dataset. Implementing SMOTE on our imbalanced dataset helped us with the imbalance of our labels (more not popular than popular tracks). Then, we conducted training by SMOTE method.

Table 4-3 Result with SMOTE by KNN

	WITH 'FOLLOWERS'				**WITHOUT 'FOLLOWERS'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.91	0.7	0.79	6772	0.89	0.7	0.79	6772
1	0.22	0.54	0.31	1057	0.2	0.47	0.28	1057
Accuracy			0.68	7829			0.67	7829
Macro Avg	0.56	0.62	0.55	7829	0.55	0.59	0.53	7829
Weighted Avg	0.81	0.68	0.73	7829	0.8	0.67	0.72	7829

From the resulting tables, we can clearly find that the overall accuracy decreases after resampling. However, we focus more on the precision and recall rate as is called for by the context of our goal.

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answers is: of all tracks that are labeled as popular by our model, how many are actually popular? High precision relates to the low false positive rate. We care more about the precision rate of popular songs instead of unpopular ones. Interestingly, after resampling, the precision of class 1 increases in the model without 'Followers,' but decreases in the model in which 'Followers' is included.

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the tracks that are truly popular, how many did we correctly label? A recall greater than 0.5 is desired. The result with resampling data significantly performs better than that with non-preprocessed data.

4.2.2 Naïve Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable.

After scaling the data, we used the same methods to conduct the model training and testing as in the previous section. We get the resulting table below:

Table 4-4 Result by Naïve Bayes

				NON-PREPROCESSED				
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.88	0.95	0.91	6772	0.86	0.99	0.92	6772
1	0.35	0.18	0.24	1057	0.12	0.01	0.01	1057
Accuracy			0.84	7829			0.86	7829
Macro Avg	0.61	0.56	0.57	7829	0.49	0.5	0.47	7829
Weighted Avg	0.81	0.84	0.82	7829	0.76	0.86	0.8	7829
				OVERSAMPLING				
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.89	0.9	0.9	6772	0.91	0.52	0.66	6772
1	0.32	0.31	0.32	1057	0.18	0.68	0.29	1057
Accuracy			0.82	7829			0.54	7829
Macro Avg	0.61	0.61	0.61	7829	0.55	0.6	0.47	7829
Weighted Avg	0.82	0.82	0.82	7829	0.81	0.54	0.61	7829
				SMOTE				
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.89	0.9	0.9	6772	0.89	0.75	0.82	6772
1	0.32	0.31	0.32	1057	0.2	0.41	0.27	1057
Accuracy			0.82	7829			0.71	7829
Macro Avg	0.61	0.61	0.61	7829	0.55	0.58	0.54	7829
Weighted Avg	0.82	0.82	0.82	7829	0.8	0.71	0.74	7829

In the 'Without Followers' group, we can clearly see that the overall accuracy decreases after resampling across all pre-processing methods. However, this does not mean that resampling is useless because the resampling increases the class 1 recall, which is precisely what we want for our prediction. In the test without 'Followers', the class 1 recall makes a bounce, compared to that with 'Followers'. Naïve Bayes model did a better job when handling the data without a leading attribute. However, the recall rates for class 1 are low, especially with the SMOTE method: class 1 had a recall rate less than .5. For Naïve Bayes, oversampling resulted in the best result.

4.2.3 Decision Tree

A Decision Tree is a decision analysis method to calculate the probability that the expected value of net present value is greater than or equal to zero, evaluate project risks, and judge its feasibility based on the known probability of occurrence of various situations. It is a graphical method to intuitively apply probability analysis.

Following the process under the last two models, we also built the Decision Tree model with the non-preprocessed, oversampling and SMOTE resampling data.

Table 4-5 Result by Decision Tree

UN-PREPROCESSED								
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.88	0.98	0.93	6772	0.87	0.99	0.92	6772
1	0.58	0.17	0.26	1057	0.24	0.02	0.04	1057
Accuracy			0.87	7829			0.86	7829
Macro Avg	0.73	0.58	0.6	7829	0.56	0.51	0.48	7829
Weighted Avg	0.84	0.87	0.84	7829	0.78	0.86	0.8	7829
OVERSAMPLING								
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.94	0.7	0.8	6772	0.91	0.55	0.69	6772
1	0.26	0.69	0.38	1057	0.19	0.66	0.29	1057
Accuracy			0.7	7829			0.57	7829
Macro Avg	0.6	0.69	0.59	7829	0.55	0.61	0.49	7829
Weighted Avg	0.84	0.7	0.74	7829	0.81	0.57	0.64	7829
SMOTE								
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.93	0.75	0.83	6772	0.9	0.62	0.74	6772

1	0.27	0.62	0.38	1057	0.19	0.56	0.28	1057
Accuracy			0.73	7829			0.61	7829
Macro Avg	0.6	0.68	0.6	7829	0.55	0.59	0.51	7829
Weighted Avg	0.84	0.73	0.77	7829	0.81	0.61	0.67	7829

Using Decision Trees, we get very similar results to the KNN model, with precision, recall, and f1-score all averaging around the same values.

Even though the pre-processing did not improve the overall accuracy, they drastically improved the recall performance under the class of popular songs. Using oversampling and SMOTE, respectively, the recall rates increased from 0.17 to 0.69 and 0.62 with 'Followers' and from 0.02 to 0.66 and 0.56 without 'Followers'. We pay more attention to the recall rate of class 1 because we care more about whether the model can help us predict a popular song instead of unpopular one. Besides, the weighted average precision was not sacrificed for a higher recall. Therefore, the pre-processed decision trees performed better.

4.2.4 Random Forest

A random forest is a tree-based machine learning algorithm that randomly selects specific features to build multiple decision trees. The random forest then combines the output of individual decision trees to generate the final output. Advantages of Random Forests Random forests present estimates for variable importance. Random forest is a good method for working with missing data. Missing values are substituted by the variable appearing the most in a particular node.

Like how we conducted the test under the last models, we also built random forest model with un-preprocessed, oversampling and SMOTE resampling data.

Table 4-6 Result by Random Forest

UN-PREPROCESSED								
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.88	0.99	0.93	6772	0.86	1	0.93	6772
1	0.62	0.11	0.18	1057	0	0	0	1057
Accuracy			0.87	7829			0.86	7829
Macro Avg	0.75	0.55	0.55	7829	0.43	0.5	0.46	7829
Weighted Avg	0.84	0.87	0.83	7829	0.75	0.86	0.8	7829
OVERSAMPLING								
	With 'Followers'				**Without 'Followers'**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.94	0.75	0.83	6772	0.91	0.67	0.77	6772
1	0.3	0.68	0.41	1057	0.21	0.57	0.31	1057

	precision	recall	f1-score	support	precision	recall	f1-score	support
Accuracy			0.74	7829			0.66	7829
Macro Avg	0.62	0.71	0.62	7829	0.56	0.62	0.54	7829
Weighted Avg	0.85	0.74	0.78	7829	0.81	0.66	0.71	7829

SMOTE								
	With ‘Followers’				**Without ‘Followers’**			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Popularity:								
0	0.92	0.78	0.84	6772	0.9	0.71	0.79	6772
1	0.29	0.58	0.39	1057	0.21	0.49	0.29	1057
Accuracy			0.75	7829			0.68	7829
Macro Avg	0.61	0.68	0.62	7829	0.55	0.6	0.54	7829
Weighted Avg	0.84	0.75	0.78	7829	0.81	0.68	0.73	7829

According to the results, pre-processing decreases the overall accuracy under random forest models, just as that under other models, as expected. Similarly, pre-processing increases the recall value of class 1 from 0.11 to 0.68 and 0.58 under the group with the attribute of ‘Followers’ and from 0 to 0.57 and 0.49 without the same attribute. Moreover, the overall accuracy is acceptable, at over 66% for the ‘without followers’ group.

Combining the results of the above four models, random forest performs best among these four classification models with oversampling pre-processing. Random forest ran with SMOTE did indeed result in a slightly higher overall accuracy of ~.01, but the recall rates greatly decreased in return. The only insufficiency of random forest is that the model harms the precision of class 1. However, it maintains a relatively high overall accuracy and good class 1 recall.

4.3 Best Features

For a clearer understanding, we made a list of most influential features for the popularity of songs. The table features the correlation scores of all attributes.

Table 4-7 Best of All Audio features

With Followers		**Without Followers**	
Feature	**Score**	**Feature**	**Score**
followers	0.116745	loudness	0.013228
loudness	0.012699	speechiness	0.010417
danceability	0.01117	tempo	0.006935
speechiness	0.00955	acousticness	0.006615
energy	0.006946	instrumentalness	0.006611
tempo	0.006903	liveness	0.006436
instrumentalness	0.005855	danceability	0.006273
liveness	0.004353	energy	0.005461
acousticness	0.00423	valence	0.004337

valence	0.003143	mode	0.001798

The result indicates that ‘Followers’ is the most influential attribute and has an overwhelming effect over all other audio features. Except for that, loudness contributes most among audio features, no matter how we structured the attributes. Speechiness and tempo also affect the popularity more than other features. It is consistent with the results of data visualization.

Conclusion

Our project was intended to further investigate whether or not it is possible to predict hit songs using machine learning simply via their audio features and artist popularity. Even though the classification models did not offer very solid and convincing evidence, we can still draw some useful conclusions.

From the data visualization and classification results, speechiness and loudness are most related to the popularity of the songs among all audio features. However, the most influential attribute is always the popularity of the artists themselves. In our random forest model, for example, the overall accuracy rose almost 10% with the inclusion of follower data. This may be based on the fact that most songs that have high popularity also have a large follower value, so the models that do not include follower data have unexplained variation that the independent variables do not cover. However, based on the visualization graphs, it is clear to see that it is possible for a non-famous artist to produce a popular song even when he/she has a low number of followers, albeit chances are more slim. Based on these conclusions, music producers should pay more attention to the speechiness and loudness of their music products. And if they can afford the cost, it is always a short cut to invite pop stars as the singers.