# The Spotify Effect

Fionna Chen (fyc3) and Megan Yin (my474)

**Digital platforms like Spotify have drastically changed the ability to distribute and listen to music. Founded in 2008, it has allowed all different types of artists and listeners to simply share and enjoy music. Because of the accessibility that this platform provides for its users, we are interested in exploring how popular music- "hits" - have changed over time and what are the features of the songs that drive those hits. We used OLS Regression, Hinge/Logistic Loss with Quadratic Regularization, Decision Tree, Random Forest, and InterpretML to explore our objective.**

## Introduction

Spotify is the leading music streaming service in the US (maybe even the world) and much of its success owes to the use of machine learning algorithms to categorize, rank, and even recommend songs. In the early days of Spotify, the service was lauded for its *Discover Weekly* playlist that suggests a whole playlist of 30 new songs to the user every week. This feature was great for listeners and artists alike by opening up the user to music they may like but otherwise would not have discovered and by helping new up and coming artists gain more exposure helping them grow. Today, amidst rising competition with big names like Apple Music and Youtube Music, Spotify has still remained the top music streaming platform due in large part to its early adoption of these recommendation algorithms. Currently, Spotify is also using similar algorithms to create Daily Mixes and other playlists for the user making the act of listening to music an adventure in and of itself. Because of the success of these algorithms, we wanted to explore how data analytics can be used to run a successful music platform.

These days, artists are competing for the attention of listeners as they hope to come out with the next big "hit". But what makes a song a hit and is it possible to train a model to predict such hit songs in a given year? Using models like OLS Regression, Quadratic Regularization, Decision Tree/Random Forest and Explainable Boosting Machines we aim to not only predict hit songs accurately, but also explain what makes a song a "hit".

Currently, models like decision tree/random forest and logistic classification are already proving useful in predicting hit songs (Middlebrook, 2019; Reimann, 2018). There is even research into creating a synthetic pop robot using Neural Networks (Hanson, 2020).

## Data

In this project, we are using the Spotify dataset found on Kaggle. This dataset has 160,000+ songs with features on artist, genre, etc derived from the Spotify Web API. We have a total of 19 features and 169,909 examples. We have numerical, boolean, and categorical features described below.

| Feature | Value Type |
|---|---|
| Acousticness | *float 0.0-1.0* |
| Danceability | *float 0.0-1.0* |
| Energy | *float 0.0-1.0* |
| Duration_ms | *int* |
| Instrumentalness | *float 0.0-1.0* |
| Valence | *float 0.0-1.0* |
| Popularity | *int 0-100* |
| Tempo | *float* |
| Liveness | *float 0.0-1.0* |
| Loudness | *float* |
| Speechiness | *float* |
| Year | *int* |
| Mode | *boolean* |
| Explicit | *boolean* |
| Key | *int* |
| Artists | *string* |
| Release Date | *date* |
| Name | *string* |

## Data Visualization

We did some initial data visualization to get a sense of the data we were working with. Something interesting that we found was that most songs fall on the extremely low end of speechiness and liveliness when we thought these features may actually be a good indicator of popularity. A correlation plot was made to see if there are any major determining features that dominate in determining popularity, but most features had really weak correlation and could have been just due to noise rather than an actual relationship. We observed that there may be potential bias in our dataset pertaining to the year feature because there are much more songs from recent years compared to older songs from 1920 for instance. Because of this, we were careful about handling data from different decades by partitioning it accordingly.

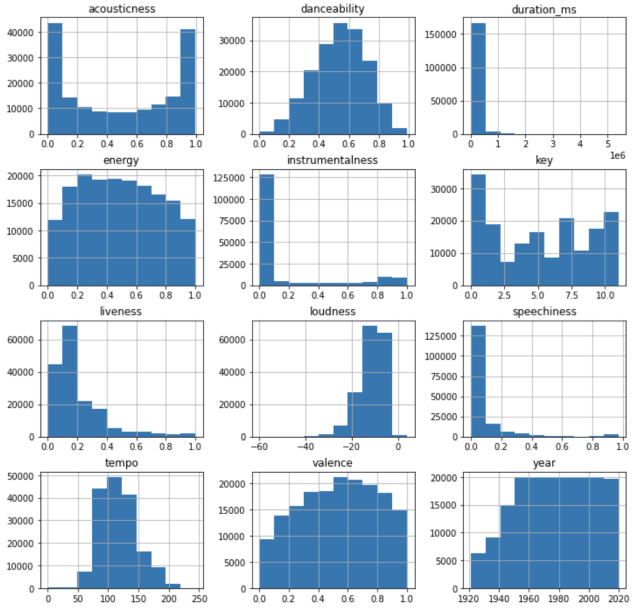


**Figure 1**

Part of our initial data exploration included fitting linear models for each decade of music to see if the model can predict the popularity of a song in its given year. We noticed that the training set error was always consistently higher than the test set error, telling us that there is an issue with

our model. We speculate the error occurred because there aren't enough data points when we partition the songs by year. However, while doing this we noticed an interesting trend in how the error changed throughout time. In particular, we noticed a general upward trend. We plotted the Mean Absolute Error throughout time to explore this further (Figure 2).

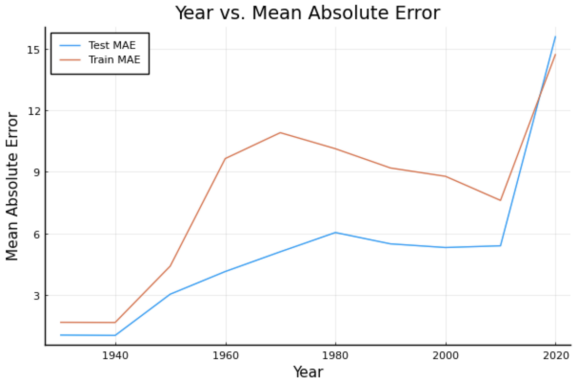


**Figure 2**

Something interesting that we observed is the sharp increase around 2010. Spotify launched in 2008. Our hypothesis was that because Spotify quickly became more popular, it became a platform that many artists became familiar with and thus had a platform to share their music. This allowed more diversity in the music industry since essentially anyone is able to upload to the music service. Music also became much easier to distribute and was more accessible to a wide audience. This combination of diversity in artist and audience is the perfect formula for more diverse hit tracks since different people prefer different styles of music. Perhaps, the diversity is why the error keeps increasing. As time progressed, hit songs are harder and harder to predict since they may be much more spread out in style, genre, and other features.

To explore the changes throughout time, we plotted each feature and its weight throughout time for the songs that were classified as hits.

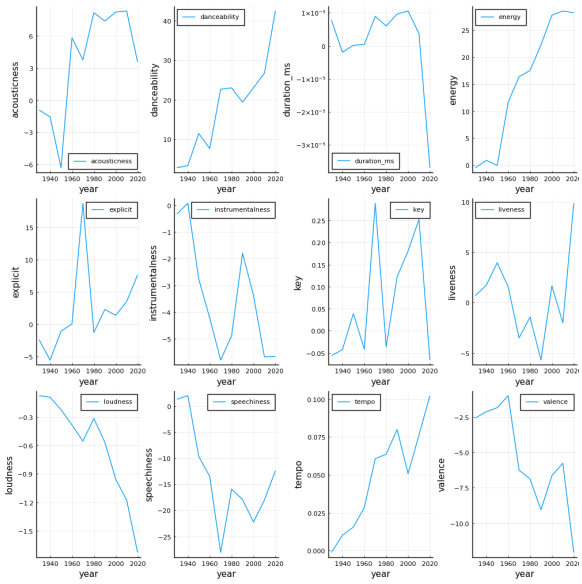


**Figure 3**

There are some very interesting observations we can make from Figure 3. Danceability, energy, and tempo have consistently been more and more important as time progressed. This makes sense as people tend to choose to listen to upbeat songs to attempt to improve mood (Ferguson, 2013). Since people tend to make an active choice in listening to music with these features, they tend to be more popular. Duration and loudness has taken a sharp decrease in its importance to a hit song. In fact, the results we get for duration indicate that the longer a song is, the less likely it is to be a hit. This is interesting because it could be related to the general trend of the generation where content is enjoyed in short bits- see TikTok, for

example, where content is no longer than 1 minute long.

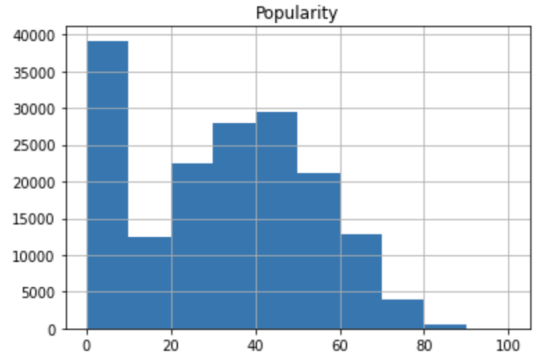


**Figure 4**

In figure 4, we see that songs are heavily skewed toward the lower end of the popularity scale that ranges from 1-100. Since we want to explore how the features can determine whether a song is a hit or not, we wanted to pick a threshold that would allow us to have enough hit songs in our training sets and test sets so that our model does not severely overfit. We decided to make our threshold to be 50. There will be enough songs in our dataset, but they will be popular enough that it is considered a "hit".

# Models

The main objective we wanted to explore in this project was if we could use the features of a song to determine whether or not it'll be a hit. We used the popularity feature given to us and turned it into a boolean value. A song was considered a hit if it reached the threshold of 50/100 on the popularity scale. Because our dataset is so large and contains songs from 1921-2020, we wanted to fit a model focused on 1 year rather than all years since music trends have probably changed significantly. We decided we want to explore the most recent music trends and thus chose songs from just the year 2020. We split the data into a training set and test set with a 70/30 split.

# Hinge/Logistic Loss

For a first attempt at classifying these hit songs, we used the proxgrad function fit with Hinge Loss and quadratic regularizer. This resulted in a training misclassification rate of about 0.93 and a test misclassification rate of about 0.92. We also tried logistic loss which resulted in a training misclassification rate of 0.37 and a test misclassification rate of 0.71. Evidently, Hinge Loss seems to be underfitting the model especially when compared to the Logistic Loss model, but both of these don't seem to generalize well. Even for the Logistic Loss model, there is a large discrepancy between the training set accuracy and test set accuracy. This shows that simple misclassification models may not be enough to predict what makes a song a hit. On the one hand we need a more complex model, but we should also be careful about overfitting.

# Decision Tree/Random Forest

We thought Decision Trees would be a good model to use for our dataset because we have a lot of features this model can use to determine the best split for each one. It is easy to interpret, handles the mixed data, and captures the interactions between each feature. We tried different depths with the decision tree classifier. What we found was

that even for shallow decision trees without multiple layers tend to overfit the data. When we used a maximum depth of 2, we got an error of 0.37 for the training set and 0.96 for the test set. Clearly, this model overfits.

Random Forest provides a special type of bagging technique that combines multiple Decision Trees to find the best splits between features. We read that because Random Forest chooses the best branches of the trees, it would increase the accuracy and precision of the model. It also has been successfully used in some previous Spotify projects for these reasons (Middlebrook, 2019). Using the same training and test set as the other models, we got an error of 0.36 for the training set and an error of 0.74 for the test set which also indicates that the model overfits. This wasn't too surprising since we saw that our Decision Tree model also overfitted despite having a shallow depth. Since both Decision Tree and Random Forest overfit, we decided to turn to EBMs (Explainable Boosting Machines). Boosted Trees and Random Forests have high accuracy but low intelligibility, while EBMs have both high accuracy and high intelligibility.

## Interpret ML

Inspired by the guest lecture from Rich Caruna, we decided to look into how Explainable Boosting Machines (EBMs) in interpret.ml can be used in more accurately predicting Spotify hit songs. Interpret.ml is a machine learning library developed by Microsoft Research that implements many glass box methods that allow much more freedom in model training and interpretability. Historically, Linear models are very easy to interpret but result in low accuracy, but more complex models like the Tree based models we used previously are blackbox and hard to interpret. EBMs are able to reap the benefits of both sides by bringing Generative Additive Models to machine learning steroids. Whereas most linear models are trained as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

GAMs are trained as additive functions on individual features:

$$y = f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n)$$

It can even learn pairwise features allowing for more complex interactions. This makes the model much more versatile than traditional black box methods. Thus, we wanted to fit not only an accurate model, but also one that can be explained easily to record labels, up and coming artists, etc to know how they should optimize their music for popularity.

We trained an EBM Classifier on our data that performed remarkably well on training and test sets. Training accuracy was 0.91 and test set accuracy was about 0.9. Note that there is very little discrepancy between these values which shows that EBMs are also very good at correctly fitting the model.

One of the hallmarks of EBMs and glassbox models is the ability to interpret the model and derive meaningful insights. In fact, EBMs allow us the ability to look into

exactly which features were most influential in predicting if a song is a hit.

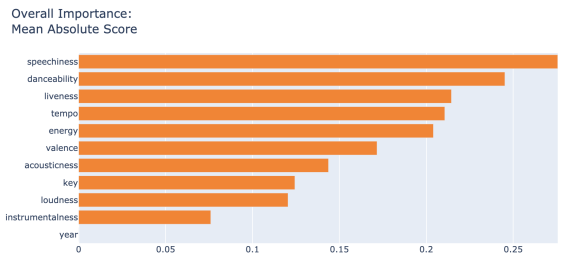


**Figure 5**

In the above graph, we trained an EBM classifier on songs in 2020 and found that speechiness is the most important feature for a hit song. This may be due to the rise in popularity of rap music or podcasts in recent years.

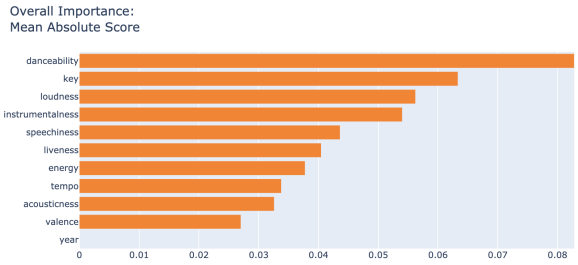


**Figure 6**

However, on a model trained on songs from 2010 (figure 6) we see that danceability and key were more important.

Overall, EBMs were very helpful in not only accurately predicting hit songs, but also in interpreting our results. Classical blackbox classifiers like Random Forest or Decision Tree may have helped us get good results, but they are hard to explain. EBMs got us much more accurate results than Random Forest and Decision Tree and allowed us to easily interpret the trends for each year of songs.

# Conclusion

## Results Overview

We started by using a simple OLS Regression to predict raw popularity scores to see which features would be upweighted and more important for high popularity. We found that predicting raw scores doesn't make sense because it is both a very hard problem to get right and we don't necessarily need that much precision since most artists aim to make a "hit song" rather than a song with a 75 popularity score, for example.

Thus, we set a hit song threshold of popularity 50/100 and used several classification methods including proxgrad with Hinge/Logistic Loss, Decision Trees and Random Forest and finally EBMs to predict which songs are hits (popularity > 50/100).

| Model | Test Set Accuracy |
|---|---|
| Hinge Loss | 0.08 |
| Logistic Loss | 0.29 |
| Decision Tree | 0.04 |
| Random Forest | 0.26 |
| EBM Classifier | 0.90 |

**Figure 8**

As we can see from the table above, EBM Classifier was the most successful at accurately predicting hit songs. In addition, we were able to interpret the model weights to learn how music tastes have changed over time.

## Weapons of Math Destruction

A Weapon of Math Destruction is a predictive model whose outcome is not easily measurable, whose prediction can have negative consequences, or creates feedback loops. Our outcome is simply a boolean value measured by the number of streams, which is a clear measure. In our case, the data does not contain sensitive information, which makes it extremely unlikely that there will be any harm caused. It could be possible that artists will target specific features to increase the possibility that their song will be a hit based on the model, which could create a feedback loop if the song does indeed become a hit, leading to a continuation of emphasizing the same features over and over again. However, since music is always changing and there are so many different styles, the features available to us are not the perfect indicator of what a song should or shouldn't have. Artists are also constantly reinventing themselves. Overall, the chance of a feedback loop is low.

## References


[1] Middlebrook, Kai, Kian Sheik. (Sept. 2019). "SONG HIT PREDICTION: PREDICTING BILLBOARD HITS USING SPOTIFY DATA". https://arxiv.org/pdf/1908.08609.pdf

[2] Spotify for Developers. Song popularity, 2020.

[3] Caruna, Rich, et al. (Dec 2020). "Friends Don't let Friends Deploy Black Box Models: Intelligibility, Transparency and Explanation in Machine Learning" https://people.orie.cornell.edu/mru8/orie4741/lectures/Tutorial4MadeleineUdellClass_2020Dec08_RichCaruana_IntelligibleML InterpretML_EBMs_75mins.pdf

[4] Ferguson, Yuna L., Kennon M. Sheldon. (May 2013). "Trying to be happier really can work: Two experimental studies." The Journal of Positive Psychology. https://www.tandfonline.com/doi/abs/10.1080/17439760.2012.747000

[5] Reiman, Minna, Philippa Örnell (May 2018). "Predicting Hit Songs with Machine Learning". https://kth.diva-portal.org/smash/get/diva2:1214146/FULLTEXT01.pdf

[6] David Hanson, et al. (Nov 2020). "Experiments in Human-AI Collaboration on Popular Music". https://arxiv.org/pdf/2011.10363.pdf