Predicting Whether Songs from Spotify Will Chart Billboard's Weekly Hot 100s

Jon Carlo Bibat
University of California, San Diego

Elton Ho
University of California, San Diego

Kelly Lo
University of California, San Diego

Abstract **Finding today's next hit is a complex endeavor taken by many producers in the music industry in order to ensure prolific success for their respective record label. The task of manually listening to every single track that enters their studio can be time-consuming and tedious as many of them may sound similar in an attempt to either replicate what currently lies at the top of the charts or provide a 'unique' twist on popular genres, which, in a sense, is not all that absurd to do. After all, only certain kinds of songs make it to the top of the Billboard. To verify this assumption and facilitate the process of finding the next Billboard hit, we developed a classification model that would categorize inputted songs as one that either makes Billboard's Weekly Hot 100 chart or one that does not. To accurately generate these predictions, we isolated and derived features that we determined to be significant from the literature that we consulted, and from experimentation on the contents of two datasets created by the Spotify API: one containing audio features and one containing artist features. Once we found these significant features, we compared the performance of several different classification models, eventually finding that our support vector machine (SVM) model had the best performance with a TPR of 0.8358, a TNR of 0.8095 and a F1 Score of 0.4557. This was a significant improvement on our baseline models which were naive classifiers dependent on the threshold of certain audio features. These models had highly-imbalanced TPRs and TNRs and F1 scores less than 0.3. We later discuss the success of the model that we created as well as the many iterative steps we took to arrive at our final model.**

1 INTRODUCTION

1.1 Data

We used a combination of three datasets for our predictive task. For the first dataset, we used a Spotify dataset that we have found from Kaggle. This dataset contains over 170,000 distinct songs from 1921 to 2020. For the second dataset, we used a dataset from the same Kaggle which contained data on 26,680 artists including a list of genres associated with each artist and their calculated popularity. However, these two datasets do not contain whether or not the song has charted the Billboard. Thus, for the third dataset, we curated a Billboard dataset by extracting all the Billboard songs from Billboard.com's Weekly Hot 100. This dataset contains over 325,000 songs from 1958 (the inception of Billboard's Hot 100) to 2020. With this dataset, we created a dictionary mapping each year to a set of songs that has charted Billboard's Hot 100 for that year. Then, we added a new column on the Spotify dataset to indicate whether that song has charted on Billboard. To do this, we looped through our dictionary of songs, searched the song name and artist with Spotipy [6] to find the song's ID and accessed the row in the dataframe that matched the ID to mark the boolean indicating whether it was ever on Billboard's charts or not.

1.2 Exploratory Data Analysis

The resulting dataset contains songs from 1940 to 2020. We didn't include songs from any year prior since our first instance of a song that has charted Billboard's Weekly Hot 100 is from the 1940s. The dataset consists of **155,978 songs**, where 14,511 songs have charted Billboard and 141,467 songs have not.

Each song has **20 properties,** where 14 of these properties describe the song's audio characteristics:

- `duration`: duration of track in milliseconds
- `valence`: musical positiveness by track
- `acousticness`: how acoustic a track is
- `danceability`: suitability of track for dancing
- `energy`: how energetic (fast/loud/noisy) track is
- `explicit`: whether track has explicit lyrics
- `instrumentalness`: whether track contains vocals
- `key`: overall key of track which maps to pitches
- `liveness`: presence of audience in recording
- `loudness`: overall loudness of track in decibels
- `mode`: modality (major or minor) of track

- `popularity`: popularity of track calculated by total number of plays and how recent plays are
- `speechiness`: presence of spoken words in track
- `tempo`: overall tempo of a track in beats/min

First, let us look at the distribution of songs in the dataset to get an idea of the songs we are working with.

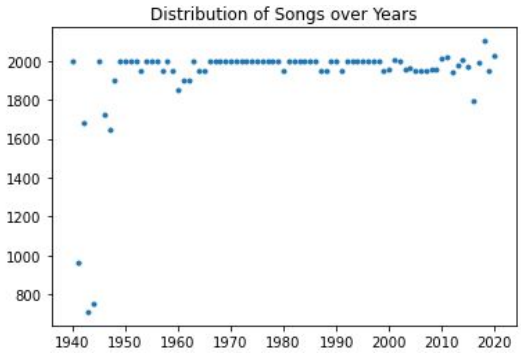

Fig. 1. Distribution of Songs over Years

It can be seen from the above figure that some years in the 1940's do not have many songs in the dataset. This could be due to the fact that digital streaming is a relatively recent phenomenon and that older songs aren't being accounted for because they are less listened to.

	Max	Min	Mean	Median
Counts	**2103**	**710**	**1925.654**	**2000.000**

Table 1. Distribution Statistics

Figure 1 and Table 1 show the distribution of the counts of songs over time in the dataset. We can see that we have a pretty even distribution of songs starting from the 1950's to 2020's making our data balanced in terms of years.

Next, we visualized how the audio features of each song compares. We represented the comparison in terms of the average trends between songs that have charted the Billboard and songs that have not. Here are figures of some of the audio features we found interesting.

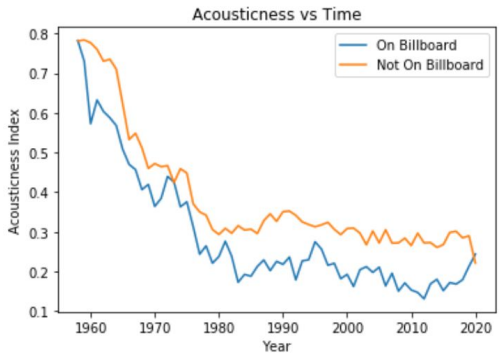

Fig. 2. Acousticness Metric over Time

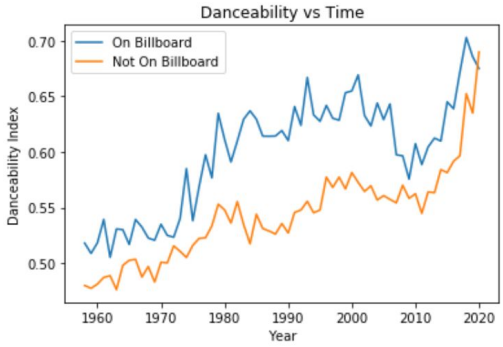

Fig. 3. Danceability Metric over Time

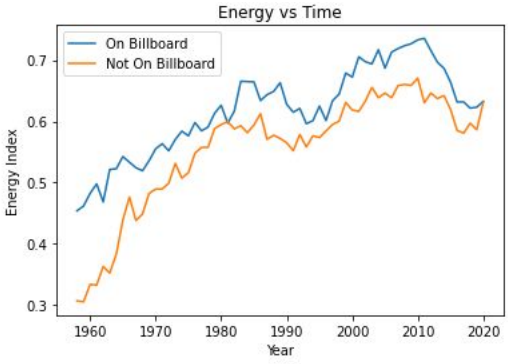

Fig. 4. Energy Metric over Time

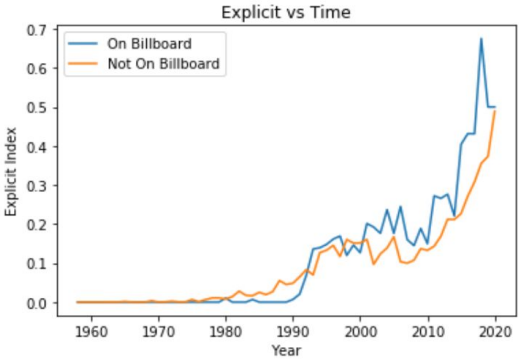

Fig. 5. Explicitness Metric over Time

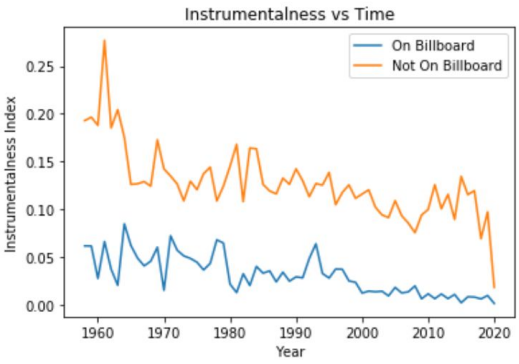

Fig. 6. Instrumentalness Metric over Time

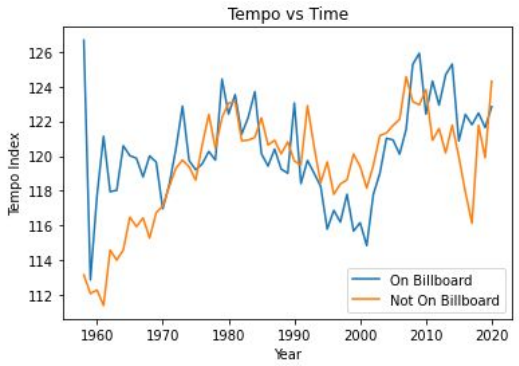

Fig. 7. Tempo Metric over Time

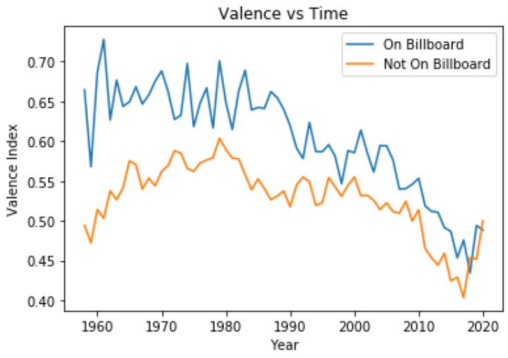

Fig. 8. Valence Metric over Time

Through this, we wanted to find audio features that have clear distinction between Billboard and non-Billboard songs. We found 5 mostly distinct audio features: acousticness, danceability, explicitness, instrumentalness, and valence.

Interestingly, we see that songs that were less instrumental and less acoustic tend to chart Billboard. Additionally, these features have a general downward trend. One feature that we predicted would play a big factor in our predictive model is danceability, with a general positive trend. We reason that these trends occurred due to the rise of electronic dance music and pop music. Recently, more pop songs tend to chart the Billboard, which has traits of higher danceability, less instrumentalness and less acousticness [1]. Surprisingly, we see that the valence feature had a general decrease which meant that songs generally evoked more negative emotions such as sadness and anger. We hypothesized that this could be due to the recent rise of anti-pop songs like Billie Eilish's "Ocean Eyes" which charted Billboard in 2019 and BROCKHAMPTON's "SUGAR" in 2020.

After some initial exploratory data analysis, we were able to identify a couple features we would like to explore in our predictive model. As we continue to build our predictive model, we will learn whether these features or other features would affect our model.

2 PREDICTIVE TASK

For our predictive task, we attempted to predict whether a song will/has charted the Billboard's Weekly Hot 100 by using several prominent audio and artist features for a song. The combination of the features used in our model were determined when we evaluated which features provided the best performance. Because our data is imbalanced due to our dataset having a small number of Billboard songs compared to non-Billboard songs, we evaluated our models using the high true positive rate (TPR), true negative rate (TNR), and F1-score. These evaluation metrics attempt to balance precision and recall, taking into consideration accurately predicted results. . We also randomly shuffled and partitioned the dataset by 75:25. The first part is used for training and the second part is used for validating the model.

We used two baseline models for comparison. The first baseline model was a naive classifier that solely used the popularity feature. This model predicted if a song is on the Billboard chart by checking whether its popularity value surpassed a threshold, which we defined as the median of all popularity values in the dataset. Similarly, our second baseline model was built to classify songs based on thresholds defined for these 5 audio features that are characteristic of "pop" music: energy, tempo, danceablity, valence, and popularity. Both baselines were appropriate as they used features that were relatively good indicators of Billboard songs, and they did not require fitting a model to the data.

Model	TPR	TNR	F1
Baseline 1	**0.842635**	**0.543638**	**0.270436**
Baseline 2	**0.205282**	**0.889710**	**0.181185**

Table 2. Evaluation of Baseline Models

After the baseline models, we tried more sophisticated models using additional features and fine tuning of model parameters. We searched for an optimal model and a set of features that would allow us to substantially outperform our baseline models. For instance, in terms of finding the most optimal model, we compared the TPR, TNR, and F1 scores for other classification models like logistic regression, support vector machines, naive bayes, and random forests. We propose that SVM would be a high performing model as SVMs are inherently suited for binary classification and we are predicting whether or not a song will/has charted the Billboard. In addition, we experimented with the 14 audio characteristics described in Section 1.2 as features, or components of derived features, for our model. We wanted to explore audio characteristics that provided a high indication of Billboard success. We also wanted to derive several features from the audio characteristics such as exploring popularity of the artist(s) of a song, weighing the number of Billboard hits by the artists(s) of a song, and the genre of the song. We will dive deeper into how we derived these different features in Section 3.1, where we discussed how we selected our features for our final model.

3 MODEL

3.1 Feature Selection

To build our final model, we used a feature vector that contained 6 features with 3 being derived:

1) **Instrumentalness and Danceability** – We used these 2 audio characteristics from Section 1.2 because these features had the biggest distinction in the trends between Billboard and non-Billboard songs.

2) **Popularity** – This feature was one of the clearer and more obvious choices to include in the model as popular songs tend to make the charts more often than not.

3) **Average Artist Popularity (derived)** – We used the popularity index of the artist data from our dataset. The artist popularity was averaged over the number of artists in a song to account for songs with more than one artist. By averaging, we mitigate the situation where songs by multiple artists are given extremely high weights in popularity.

   This feature, similar to song popularity, was a clear choice for the model as more well-known artists make it to the charts more often than lesser-known artists.

4) **Precedence (derived)** – Inspired by Interiano, et. al [2], we included a feature similar to their "superstar" variable which we coined as precedence. The precedence feature of a song takes into account the number of songs produced by an artist that has made it onto the Billboard charts.

   We decided to include this feature as it would allow for us to include an indirect temporal aspect into our feature vector. It was also another fairly obvious choice for a feature as artists who have made the charts many times in the past will most likely appear on the charts again.

5) **Genre (derived)** – We created a one-hot encoding of the song genres that appeared in our dataset by representing each music genre with an index in the encoding. However, due to the large number of genres, we only used the genres where the total number of artists that classify under that genre is greater than the mean number of artists over all genres.

   We included this feature as we determined that genre was fairly important in determining a song's success given that only certain genres ever make it to the top of the charts.

Features	TPR	TNR	F1
Instrumentalness, Danceability, Popularity	**0.761775**	**0.678953**	**0.314170**
Artist Popularity	**0.636265**	**0.617183**	**0.239287**
Precedence	**0.460931**	**0.843595**	**0.310899**
Genre	**0.816499**	**0.723680**	**0.364998**

Table 3. Evaluation of Features by Themselves on SVM

Finding this optimal vector of features required us to go through several iterations where we tested different combinations of features and feature representations:

- **Trial 1: Feature Thresholds** – This feature set and representation was largely based off of our baseline models where we simply predicted our labels based on whether a song's features exceeded a defined threshold. Each element in the feature vector for this particular set and representations is designated either 0 or 1 based on whether the calculated threshold was exceeded or not. However, this proved to be very inaccurate in our classification task, so we decided to move on to other sets and representations.

Model	TPR	TNR	F1
Naive Bayes	**0.0**	**1.0**	**0.0**
Logistic Regression	**0.0**	**1.0**	**0.0**
SVM	**1.0**	**0.0**	**0.172154**
Random Forest	**0.0**	**1.0**	**0.0**

Table 4. Evaluation of Feature Thresholds

- **Trial 2: Diverging Song Traits and Popularity** – This set of features was heavily influenced by our EDA in Part 1.2. In particular, the features we selected for this feature vector were ones that presented the largest disparity when comparing billboard and non-billboard songs. Such traits would allow us to more easily differentiate the two labels. We also included popularity as it was a feature that was highly indicative of Billboard success. This set, although a significant improvement from the baseline models, still didn't produce particularly satisfying results. As such, we decided to explore more features, but kept these features in mind to possibly incorporate in the final model.

Model	TPR	TNR	F1
Naive Bayes	**0.0**	**1.0**	**0.0**
Logistic Regression	**0.757148**	**0.681387**	**0.314078**
SVM	**0.761775**	**0.678953**	**0.314170**
Random Forest	**0.109447**	**0.967304**	**0.153728**

Table 5. Evaluation of Song Traits and Popularity

- **Trial 3: Average Artist Popularity Only** – The next feature vector we decided to try only contained a single feature as we wanted to see whether the isolated average artist popularity would improve the results of the prediction. After running this feature vector through our models, we discovered this feature actually decreased our overall F1 score; however, we also observed that it was an improvement on our baseline 2 model as it produced a higher F1 score and more balanced TPRs and TNRs. Because of this, we noted down that this was a possible feature to include in our final model.

Model	TPR	TNR	F1
Naive Bayes	**0.0**	**1.0**	**0.0**
Logistic Regression	**0.643071**	**0.613475**	**0.239931**
SVM	**0.636265**	**0.617183**	**0.239287**
Random Forest	**0.111898**	**0.983241**	**0.175791**

Table 6. Evaluation of Average Artist Popularity Only

- **Trial 4: Precedence Value Only** – We decided to isolate another variable, which was inspired by one of the literature that we consulted: our coined "precedence" value. Running this feature vector through our chosen classification models yielded results that were similar to those of Trial 2, so we noted this feature as another possible feature to be included in our final model.

Model	TPR	TNR	F1
Naive Bayes	**0.0**	**1.0**	**0.0**
Logistic Regression	**0.517288**	**0.822052**	**0.320417**
SVM	**0.460931**	**0.843585**	**0.310899**
Random Forest	**0.026681**	**0.997452**	**0.050764**

Table 7. Evaluation of Precedence Value Only

- **Trial 5: Genres Only** – This was the last of the trials that had only an isolated variable. The feature vector, however, had significantly many more elements than the previous trials as we decided to represent the most popular genres as a one-hot encoding. We observed a significant improvement upon the F1 scores, the TPRs, and TNRs of our models when using this feature and its vector representation. As such, this was an

obvious candidate to be included in our final model.

Model	TPR	TNR	F1
Naive Bayes	**0.567111**	**0.857636**	**0.38**
Logistic Regression	**0.762592**	**0.759122**	**0.373890**
SVM	**0.816499**	**0.723680**	**0.364997**
Random Forest	**0.172339**	**0.981316**	**0.364997**

Table 8. Evaluation of Genre Only

- **Trial 6: Average Artist Popularity and Diverging Song Traits** – This trial signified the start of us combining features to formulate a final model. The features that we decided to combine first were our lowest performing features. We did this in order to see if together, these features would perform significantly better than when they were isolated. Running these features through our models, we discovered that this combination did, in fact, improve upon the baseline and even improved upon the isolated runs. Because of this, we decided to build off of this model to see if we can improve the overall performance and derive a decent final model.

Model	TPR	TNR	F1
Naive Bayes	**0.059079**	**0.992894**	**0.104805**
Logistic Regression	**0.747890**	**0.703722**	**0.325358**
SVM	**0.646882**	**0.786808**	**0.349926**
Random Forest	**0.126599**	**0.984571**	**0.198591**

Table 9. Evaluation of Average Artist Popularity and Song Traits

- **Trial 7: Average Artist Popularity, Precedence, and Diverging Song Traits** – For this trial, we added our precedence variable into our feature vector. The resulting F1 score was an improvement on the previous trial. As such, we decided that we would continue building upon the features in this model as it seemed likely that an improvement would occur given another feature, especially if the added feature was genre. As such, in our final trial we decided to add genre, the last feature we experimented with during our trials, as a feature in our final model.

Model	TPR	TNR	F1
Naive Bayes	**0.380615**	**0.866412**	**0.285598**
Logistic Regression	**0.735366**	**0.750998**	**0.356098**
SVM	**0.686359**	**0.797056**	**0.377310**
Random Forest	**0.263000**	**0.980552**	**0.362748**

Table 10. Evaluation of Average Artist Popularity, Precedence, Song Traits

- **Final Trial: Average Artist Popularity, Precedence, Genres, and Diverging Song Traits** – In the final iteration of our feature search and experimentation, we incorporated all of the features we encountered during our trials: diverging song traits, song popularity, average artist popularity, precedence, and track-artist genre. The resulting model was an even more significant improvement upon the baseline as it nearly doubled both of the F1 scores calculated for those naive models. The TPR and TNR also saw a significant improvement in terms of magnitude and balance between the values, meaning that classifications aren't being skewed toward one label or the other. It also doesn't appear that these classifications are random as both rates are fairly high, exceeding 0.8.

Model	TPR	TNR	F1
Naive Bayes	**0.503947**	**0.842547**	**0.333934**
Logistic Regression	**0.823849**	**0.806822**	**0.447534**
SVM	**0.835829**	**0.809455**	**0.455692**
Random Forest	**0.316090**	**0.976390**	**0.409668**

Table 11. Evaluation of Average Artist Popularity, Precedence, Genre, Song Traits

Before and during our trials, we considered a number of different other features to derive or incorporate into our model, however, we decided to forego those options in the interest of both time and simplicity. An example was some sort of temporal feature to account for the popularity of a song, artist, or genre during a certain decade as different musical trends exist during different periods in time. As mentioned above, we decided to forego this feature as it seemed too complex to implement, and we discovered that our derived "precedence" variable somewhat accounted for any temporal side effects that our data might have. It essentially did this by basing the success of a track on the historical success of the artist

rather than treating all tracks as if they were on the same level.

3.2 Model Derivation and Refinement

- **Naive Bayes Model** – A simple classifier to use, but requires the assumption that the features are conditionally independent. However, such conditional independence does not often occur, and this was very much the case regarding our features, as artist popularity and song popularity are conditionally dependent. When we built our model with our final feature vector, Naive Bayes performed worse on TPR and F1 score.

- **Logistic Regression** – One of the easier models to implement. It also fixes a potential double counting issue that arises from Naive Bayes. However, this model does not optimize for the number of mislabels, and pays no special attention to the "difficult instances"

  We used Logistic Regression because it is more suited for a classification task. It performs better on data that cannot be split by a line as it uses the Sigmoid function to classify. From the results of our trials, we noticed this model generally performed well relative to the other models we employed, but did not outperform all other models due to it neglecting the songs that were more difficult to classify.

- **Support Vector Machine** – A classifier that attempts to divide up a plane of data points into two regions for classification using a linear line. Unlike Logistic Regression, SVM attempts to optimize for the number of mislabeled points by penalizing points that fall on the wrong side of the line. It also attempts to divide the plane such that the margin between the points and the line are as big as possible. However, SVMs are immensely inefficient, such that as features grow in complexity, the runtime grows exponentially.

  From the results of our final trial, we noticed that SVM surpassed the Logistic Regression model in all three evaluation metrics, whereas in the other trials, SVM mostly tied with Logistic Regression, outperforming in some measurements, and underperforming in others. The stark improvement in performance from the Logistic Regression model can be highly attributed to the SVM's penalization of mislabeled points as well as the fact that we were able to run the model with a maximum of 50000 iterations as opposed to the default 1000.

- **Random Forest** – Several of the documents that we consulted for our literature concluded that random forests were their best performing predictor and thus, we wanted to see if this would be the same for us. Random Forest models tend to be very efficient with large datasets and have less potential to overfit. Additionally, these models can handle both numerical and categorical features, which is the case for our feature vector.

  However, when we used this model to classify on our feature vector, the F1 score was relatively low and there was a substantial difference between TPR and TNR. A reason for why Random Forest models struggled to properly fit our data could be that these models tend to perform better on non-linear data and less sparse matrices. Our data is more linear, e.g. there is a positive correlation in artist popularity and danceability mapping to Billboard success.

3.3 Final Model and Features

Out of all the models and combinations of features that we tested, our SVM model performed the best as it gave us the best combination of TPR and TNR and one of the highest F1 scores. The most optimal feature vector used the average artist popularity, precedence, genres, and diverging song traits.

Initially, our SVM model did not converge, leading to poor performance. After increasing the number of iterations to allow our SVM model to converge, we noticed a significant increase in classification performance.

From Table 11, we can see that our SVM performed just slightly better than the Logistic Regression model in all evaluation metrics. We hypothesized that our SVM model outperformed our Logistic Regression model because the SVM model fits the data more "loosely" than the Logistic Regression model. In other words, the SVM model tried to divide the data into two parts using a straight line, while the logistic regressor attempts to use the Sigmoid function ($\sigma(t) = \frac{1}{1+e^{-t}}$) curve to fit the data. We believe that using such a curve hurt the accuracy

because it was a "tighter" fit, leading to potential overfitting. Additionally, the representation of our genre feature is a one-hot encoding over all genres, which means that our feature matrix is very sparse. Generally, SVMs tended to perform better on sparse data compared to Logistic Regressors.

4 LITERATURE

For our dataset, we used the Spotify Dataset from Kaggle. People previously used this dataset to create a danceability predictor and artist recommender, in addition to many other different projects. From Koenigstein et. al [3], researchers used Billboard chart data along with P2P search query data in an attempt to discover correlations between file-sharing popularity and chart performance. They made use of chart position data and analyzed temporal changes of songs with these rankings.

Interiano, et. al [2] analyzed the correlation between acoustic features and Billboard chart performance, and "explored the predictability of success of songs." Similar to our EDA results, this study also found that more successful songs tend to be ones that are "more party-like." Interiano, et. al. used random forests to predict song success by using the acousticness of songs and a "superstar" variable, where it took into account the popularity of the artist who released the song (how many other songs by the artist that have previously made it on the top charts). The paper noted that the inclusion of the superstar variable improved the accuracy of the predictor by 15%. Middlebrook, et. al [4] also attempted to predict which songs will top the charts using random forests. This paper also mentioned that the random forest model is their best performing predictor with an accuracy of 88%.

From the results of our findings, our random forest model was not our highest performing model unlike the random forest model used in Middlebrook, et. al [4]. However, this could be due to the fact that our data is imbalanced and thus leading to a subpar performance. The introduction of the precedence feature into our model also significantly improved our classifier performance similar to how the "superstar" variable improved the model's performance mentioned from Interiano, et. al [2].

5 RESULTS AND CONCLUSION

Through this project, we got a better idea of how certain features influence a song's probability of charting the Billboard's Weekly Hot 100. We observed that our SVM model outperformed the rest of the models in all our evaluation metrics.

Model	TPR	TNR	F1
SVM	**0.835829**	**0.809455**	**0.455692**

Table 12. Evaluation of SVM Using Final Feature Vector

As mentioned in Section 3.2, we attempted numerous models. However, some models were not as successful in one way or another and there are a number of factors that contributed to this. For instance, our Naive Bayes model did not perform well since it requires variables to be conditionally independent and this is not the case for our features, as they were highly correlated. For our Random Forest model, we learned that it is inherently suited for multi-class problems and this is not the case in our classification as we predict if a song is either a Billboard success or failure. Thus, because of these reasons, this could be why our support vector classifier was able to outperform the rest of our models.

We discovered that using a select few audio features allowed us to build a reasonably accurate classifier. We also discovered a few audio features that did not work as well. For instance, we forgone the use of our initial set of audio features (acousticness, danceability, explicitness, instrumentalness, and valence) as a lot of these features were highly correlated. Additionally, Table 4 clearly displayed how our model performed very inaccurately using these features. Thus, we only used a select few features that proved to be more independent. By using the instrumentalness, danceability, and popularity features of a song in our model, our model was able to outperform both baseline models. Additionally, we saw that deriving three new features from already defined properties in the dataset—(1) genres done by an artist, (2) average popularity of the artists in a song and (3) number of Billboard hits by an artist—provided additional valuable information and improved the model performance. As for why using the number of Billboard hits by an artist helps our model, we hypothesized that if an artist was able to chart the Billboard with numerous previous songs, then they would have a higher probability of charting the Billboard with a new song. This reasoning could be applied to our second derived feature as if there are more popular artists on a song, then this could increase the probability of the song charting the Billboard. In terms of the genre feature, as part of Section 1.2 and 4, we saw that specific genres gained/lost a lot in popularity during specific time periods. Thus, because of these rises and

drops of genre prominence, the genre performs as a good indicator of a Billboard success.

Through all this testing and validation, we were able to eliminate models that were unfit or did not perform well for our tasks. In the end, we were able to develop a model that would successfully predict whether the song is or will be on the Billboard's Weekly Hot 100 chart which matched our proposed model of SVM.

6 REFERENCES

[1] Beckwith, J. (2016). Jack Beckwith. The Evolution of Music Genre Popularity.

[2] Interiano, M., Kazemi, K., Wang, L., Yang, J., Yu, Z., & Komarova, N. L. (2018). Musical trends and predictability of success in contemporary songs in and out of the top charts. *Royal Society open science*, 5(5), 171274.

[3] Koenigstein, N., Shavitt, Y., & Zilberman, N. (2009, December). Predicting billboard success using data-mining in p2p networks. In *2009 11th IEEE International Symposium on Multimedia* (pp. 465-470). IEEE..

[4] Middlebrook, K., & Sheik, K. (2019). Song Hit Prediction: Predicting Billboard Hits Using Spotify Data. *arXiv preprint arXiv:1908.08609*

[5] SPOTIFY (2020). Spotify Dataset 1921-2020, 160k+ Tracks. https://www.kaggle.com/yamaerenay/spotify -dataset-19212020-160k-tracks.

[6] SPOTIPY API (2020). https://spotipy.readthedocs.io/en/2.16.1/.