Can Lyrics and Tweets Predict a Hit Song?

Author: Chloe Woodcock
Supervisor: Dr Riccardo Di Clemente, University of Exeter
External Supervisor: Giulio Prevedello, Song CSL Paris

Abstract

The ability to predict a hit song comes with great advantage to not only the music industry, but all businesses through means such as advertising. A song can be analysed through internal features like lyrics or acoustic variables, or external measures such as social media, chart rankings, or streaming count. My project looks at predicting the success of a song according to the Billboard chart rankings and streaming service Spotify. I gather information about song lyrics and analyse the songs external popularity through associated mentions via social media platform Twitter. Training three machine learning classifiers led to the conclusion that these features can be used to predict a hit song with promising scores. The best model was a Random Forest classifier which was able to predict whether a song could appear in the Top 10 of the Billboard rank with an accuracy score of 0.74. Remaining classifiers were able to perform better than random chance, concluding it is possible to predict song success using my feature set.

I certify that all the material which is not my own in this document has been identified.

1. Project Motivation

The music industry has always been powerful and significant throughout history. Advancements in technology over the past decade has led to digital streaming becoming the most popular method for music consumption, with many streaming platforms available worldwide such as Spotify[1] and Apple Music[2]. Spotify is the most widely used streaming platform, with millions of new and repeating subscribers every year [1]. Social media platforms have also grown significantly in recent years, where nearly every individual now has a social media account of some sort. Twitter is a social media platform which has 217 million active users in the US from 2022 [2], and many more worldwide. A community has been created, where users react and post opinions about music, films, and other entertainment media. Users will share what music they are interested in and listening to, which is collectable data and can be analysed. The #nowplaying hashtag[3] is always trending and is used by many users to share their new and old music tastes.

Data about songs can be scraped from streaming platforms such as Spotify, social media sites such as Twitter, along with acoustic information and used to predict the success of said song. There are many song charts available such as UK Top 40[4] or Billboard Hot 100[5], which list the most popular songs weekly. Streaming platforms also offer their own success variables, for example, Spotify give a popularity score for each song. Gaining an understanding into what makes a song successful would benefit the music industry and specifically, music producers. Producers would be able to maximise their profits, whilst lowering the risk of producing new music, due to having some insight into what features increase the probability of a song being popular. Streaming services would also benefit because they would be able to create more advanced recommendation systems and select songs which are likely to be popular among the majority of people. This will increase customer satisfaction and higher repeat subscriptions to the service. Non-music-based businesses can use this information about popular songs to support advertising strategies, meaning more memorable songs will be used in adverts and resulting in the product being memorable and desirable.

2. Project Definition and Methodology

My investigation will use lyrical, Spotify and Twitter data to predict song success according to both the Billboard Hot 100 and Spotify. More specifically, I will be analysing the sentiment and explicitness of lyrics along with the sentiment and volume of associated tweets, to predict if a song will climb to a Top 10 rank in the Billboard charts or reach a Spotify popularity score of over 75.

The Billboard Hot 100 is the standard record chart in the US since 1958. The chart rankings are based on sales, radio play, and online streaming in the US alone [3]. The rankings change on a weekly basis and a song can obtain a ranking between 1 and 100, where a rank of 1 indicates the most popular song of the week. Once a weekly ranking is created, it does not change, and the ranking history can be seen. Spotify is an audio streaming service which began in 2006 and has over 406 million subscribers every year [4]. The Spotify popularity score is a value between 0 and 100 given to both tracks and artists to determine their popularity. It is calculated based on the number of plays and how recent these plays are [5]. Only the most present score can be seen, and this can change in time. My investigation is therefore looking to predict a hit song according to a) all music consumption

[1] https://www.spotify.com/uk/
[2] https://music.apple.com/us/listen-now
[3] https://twitter.com/hashtag/nowplaying
[4] https://www.officialcharts.com/charts/uk-top-40-singles-chart/
[5] https://www.billboard.com/charts/hot-100/

methods but in the US only and b) just digital streaming from Spotify subscribers only but these subscribers are worldwide.

I will initially use a dataset containing the Billboard rankings from 2010 to 2020, and then gather:

- Song Lyrics: I will use the Genius API[6] and other online sources to gather the lyrics of each song. Once these are collected, I will use the Spotify API[7] to determine the explicitness of the lyrics and implement sentiment analysis to discover the proportion of positive and negative sentiment in the lyrics. This is an internal feature about the song, and I will explore if the lyrics chosen by an artist impacts the song's success.
- Tweets: I will also investigate the predictive power around microblogging. To do this, I will collect 4 weeks' worth of tweets associated with each song, using the Twitter API[8]. The first two weeks will be prior to the song's release date and the second two weeks will be the first two weeks after the song's release, which also contain the #nowplaying hashtag. The volume and sentiment analysis from these tweets will be recorded.

Once all data is collected, three machine learning classifiers (Logistic Regression, Support Vector Machine and Random Forest models) will be trained and tested using the data, to predict if the song is a hit according to two hit criteria:

1. Top 10: The classifier will have a binary output of whether the song is Top 10, or any other rank in the chart, i.e., rank 11-100.
2. Spotify > 75: The classifier will again have a binary output of whether the song popularity score is 75 or above, i.e., 75-100, or below, i.e., 0-74.

All models will then by evaluated using a variety of metrics such as precision, recall, and AUC score. Finally, I look at feature importance and then use all information to conclude the best model for my project.

3. Literature Review

My investigation comes under the topic of 'Hit Song Science' (HSS). Pachet describes HSS as 'an emerging field of investigation that aims at predicting the success of songs before they are released on the market' [6]. The main idea behind HSS is that there are common features in songs which will make them attractive and therefore popular to many individuals.

Dhanaraj and Logan [7] were the first to research into HSS, using acoustic and lyric features to build Support Vector Machine and Boosting classifiers to determine a Top 1 rank on the Oz Net Music Chart, from any other position. Timbrel aspects such as MFCC were gathered, along with the lyric sentiment, to train using 10-fold cross validation, and achieve a high AUC score of 0.86. Lyrical features appeared more useful than acoustic but, to their surprise, concatenating the two did not increase the classifier's accuracy. Their time frame was particularly large, from 1956 to 2004, but due to their feature collection method, they only resulted in 1700 songs with the full feature set. With a more consistent method of collecting data and therefore more data to train with, they may have increased their accuracy even further. Lee and Lee [8] also found promising results when also using MFCC features but they chose to include music complexity audio features. With 17k songs, they predict whether a song appeared on the Billboard charts at any rank and were successful.

Most of the literature addressing HSS use acoustic features to predict a hit song. Pachet and Roy [9] were accomplished researchers within this field and contradicted the claims on HSS by carrying out an investigation to no success. They used a 32,000-song dataset with 632

[6] https://docs.genius.com/
[7] https://developer.spotify.com/documentation/web-api/
[8] https://developer.twitter.com/en/docs/twitter-api

manually entered labels per title to predict low, medium, or high popularity using a Support Vector Machine model with RBF kernel. This was a considerably larger dataset than Dhanaraj [7] used, and used three different features sets (generic acoustic, specific acoustic and human produced metadata) to avoid bias. Unlike the results seen by Dhanaraj [7], their F-score was not high, and the experiment did not show any promising results. Pachet and Roy believe other successful claims were either based on spurious data or biased experiments, and they concluded that the features commonly used for feature analysis were not informative enough for hit predictions [9]. Borg and Hokkanen [10] reached the same conclusion when using audio features to predict the YouTube view counts of a song. They summarised that audio features do not have the embedded information relevant in making the song popular [10].

Ni, et. al, [11] carried out an investigation which directly opposed Pachet and Roy's [9] results and concluded that HSS is possible and promising. Using the Echo Nest API to extract acoustic features from 5947 songs, they attempted to distinguish between a Top 5 rank on the UK Top Charts versus a lower down position of 30-40. There is a large gap between these two groups, whereas Pachet and Roy used three different hit ranges, with no gap in-between. Ni, et. al, achieved better than random accuracy, and believed the difference in problem specification, as said above, were one of the reasons why they saw success. They also believe that they saw better results than Pachet and Roy because they used more novel acoustic features along with a time-shifting perceptron model which accounted for the time aspect [11]. They discovered differences between hit songs throughout time and suggested that songs have become faster and louder over the past decade, but simple song are more likely to be hits regardless of the year. Herremans, et. al [12] also used the Echo Nest API to obtain musical features and try to distinguish between different ranks in the Billboard charts: Top 10 versus 30-40, Top 20 versus 30-40 and Top 20 versus 20-40. This is a variety of different gaps in the hit thresholds, and they found distinguishing Top 10 from 30-40 produced better than random results using a SVM model. The difference in their method was that they focus on Dance music only, eliminating acoustic differences between genres.

Middlebrook and Sheik [13] continue to use acoustic features but gather 27 features from the Spotify API, instead of the Echo Nest API. The Billboard Hot 100 was their definition of a hit song, where any song appearing at any rank was considered a hit, like Lee and Lee [8]. They trained four classifiers: Logistic Regression, Neural Network, Support Vector Machine and Random Forest, and found a Random Forest model was the most robust and achieved 88% accuracy. This contrasts to other investigations which aim to distinguish between rankings on the chart, instead of being on the chart or not, such like Herremans, et. al [12]. Raza and Nanath [14] continue to look at rank distinguishing and attempt to predict if a song is Top 20 of the Billboard Hot 100 or in the bottom 10. Similar to Middlebrook and Sheik [13], they use the Spotify API to gather audio features but also collect lyrics and use sentiment analysis, as seen by Dhanaraj and Logan [7]. Their time frame was significantly smaller and only gathered songs between 2017 and 2019. In comparison to Dhanaraj and Logan's results, their classifier preformed the same as random chance and contradict the claim that lyrics and acoustic features can be used to predict a hit song in this context. They suggested their dataset was either too small or external features of the songs may lead to a better accuracy.

External features such as associated tweets could be used in attempts to predict a hit song, and many researchers have investigated this potential. Tsiara and Tjortjis [15] aim at using Twitter data, related to the songs in the Top 10 of the Billboard Hot 100 charts, to preform sentiment analysis and use this to predict chart ranges: Top 5, 10 and 20. The Twitter API search function was used to gather tweets and the VADER lexicon was used for sentiment analysis. The findings suggest moderate correlations between the volume of related tweets and the chart position. The best scoring algorithm for predicting Top 5 hits achieved 90%

accuracy. Zangerle, et. al [16], carry out similar research into whether tweets associated with the song but containing #nowplaying are correlated with the Billboard Hot 100 charts and trained a multivariate time series prediction model. Supporting Tsiara and Tjortjis, they found incorporating Twitter data into the prediction process lowers the RMSE score. Kim, et. al, [17] also used the concept of the #nowplaying trend on Twitter to predict the Billboard Hot 100 charts. They found a hit prediction classifier can predict a Top 10 hit with 90% accuracy when using tweets and weeks on chart. Weeks on chart will cause some bias however in the data, as it is expected for the length of time on the charts to directly impact how high of a rank it achieves, hence why I removed it from my feature set. Other researchers, such like Araujo [18], use rankings within Spotify as their success criteria instead of external charts like Billboard or UK Top 40. Arajuo used audio features from songs between 2018 and 2019, to train a Support Vector Machine and predict if a song will appear on the Spotify's Top 50 rankings. A high AUC score was achieved of above 80%. The promising results from Tsiara and Tjortjis, and Kim, et. al, has led to my research predicting a Top 10 hit from the Billboard Hot 100, versus any other ranking to attempt to replicate their high accuracy scores, using a different problem definition.

When lyrics are involved in the feature set, sentiment analysis of the lyrics is carried out. Probabilistic Latent Semantic Analysis was used by Dhanaraj and Logan as their method and VADER was used by Tsiara and Tjortjis. Logan, et. al [19], looked at predicting artist and genre similarity using the sentiment from 16k songs and found determining similarity based on lyrics is feasible but is not as effective as acoustic features. VADER lexicon analysis so will be used to analyse my tweets, which will use #nowplaying such like Kim, et. al. To the best of my knowledge, I am one of the first reports which uses lyrics and tweets with #nowplaying, without audio features, to determine if a song is a hit.

4. Initial Dataset

The initial dataset I will be using is the 'Billboard "The Hot 100" songs' dataset, created by Dhruvil Dave [20]. This dataset can be assessed and downloaded from Kaggle.com[9]. The dataset shows the Top 100 chart rankings from the Billboard Hot 100 each week from 1958 to 20201. This dataset contains 7 features:

1. *Date: date of the weekly ranking*
2. *Rank: the rank of the song in that weekly ranking*
3. *Song: the name of the song*
4. *Artist: the artist of the song*
5. *Last Week: the rank of the song on last week's ranking*
6. *Weeks on board: the total duration the song has appeared consecutively on the charts*

My project only uses a subset of this data from 2010 to 2020, so I imported this dataset into a panda's data frame using python library 'pandas'[10] and filtered it to the required years. This is due to the research by Ni, et, al [11], who conclude significant differences in hit songs between decades. As songs remain on the chart for multiple weeks, songs are on the dataset more than once, meaning these duplicates need to be removed. I removed all duplicates and kept the most recent entry of the song, as this will have the up-to-date peak rank. This left a total of 5256 songs, which is approximately 1/6 of the Kaggle dataset, showing the volume of songs remained on the charts for multiple weeks.

[9] https://www.kaggle.com/
[10] https://pandas.pydata.org/

Initial Data Analysis

I used python package 'matplotlib.pyplot'[11] and its functions to create bar charts to see which songs are remaining in the charts for the longest time period, and which artists have the highest count of unique songs in the dataset, throughout the 2010 decade.

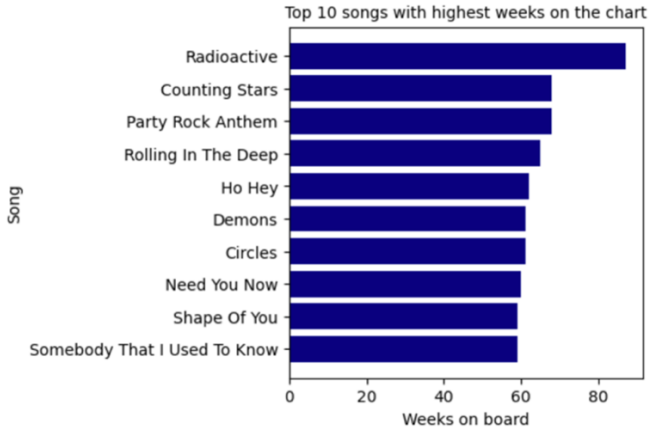


Figure 1: Songs on Billboard Hot 100 with highest weeks on the board

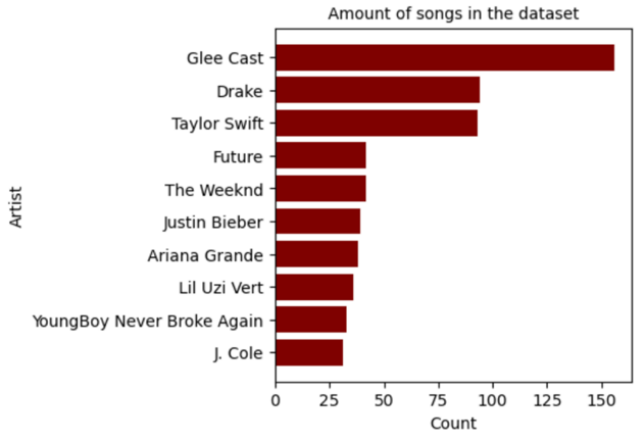


Figure 2: Count of artists songs on the Billboard Hot 100

Looking at Figure 1, we can see that 'Radioactive' by Imagine Dragons has the longest time on the board and has remained on the board for at least 10 weeks longer than any other song. Excluding this, the remainder of the top songs lasted around 60 weeks on the charts. Looking at Figure 2, we can see that the Glee Cast has a lot more songs on the board than any other artist. The Glee Cast is from the Tv show 'Glee' who cover pop songs and have a total of 6 seasons from 2009-2013[12], so there is no surprise there are so many popular songs, all which appear between 2010-2013 in my dataset. We can also see from Figure 2 that Drake and Taylor Swift still have significantly more songs on the chart than any other artist by a count over 40, who have no relation to a TV show, and are therefore the most popular artists throughout the decade.

I explored the correlation between the current rank, last rank, peak rank, and weeks on the board [See Appendix 1]. All variables have a high correlation with each other, with current rank and last rank having the highest correlation score of 0.8. This is expected as the rank of a song in the current week will depends on the rank in the previous week. The peak rank

[11] https://matplotlib.org/3.5.1/index.html
[12] https://www.imdb.com/title/tt1327801/episodes

and weeks on the board are also significantly correlated with score -0.7, meaning bias will be created in the classifier which are attempting to predict the hit rank. Kim, et. al [17] decides to keep it includes but I will be removing weeks on board along with rank and last rank because they were useful for initial analysis but are not useful to data gathering or model training.

5. Data Gathering and Preparation

APIs

An API (Application Programme Interface) is a set of programming code which allows computers or applications to communicate with eachother and exchange information [21]. I will be using three APIs in my project: Genius API, Spotify API, and Twitter API. A variety of uniquely generated API key are needed to access each API.

Genius API

I used the Genius API Python package 'lyricsgenius'[13] to query the song and artist name, and then took the first outputted result as the set of song lyrics. Multiple problems occurred with implementing this, such that the API would sometimes output spam like text instead of lyrics, or frequently give the correct lyrics but in a different language [See Figure 3].

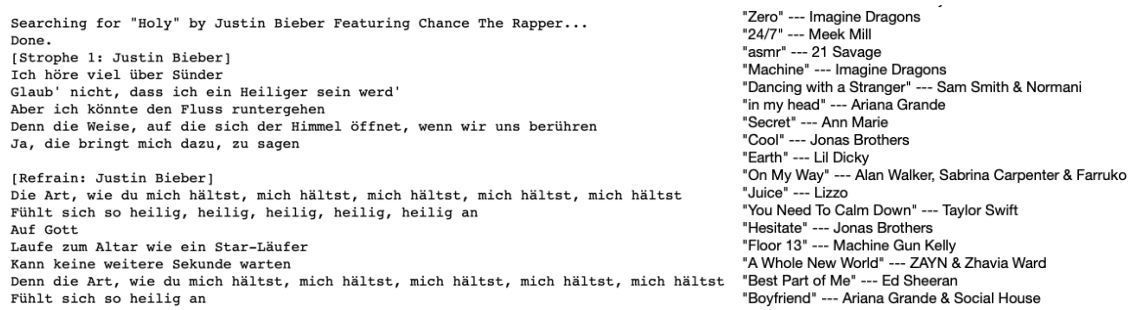


Figure 3: Examples of non-English lyrics or spam output from API

Genius.com[14], the corresponding website the API uses, offers lyric translations and is what the API is outputting at times. To ensure the language and format of the lyrics were correct, I manually checked through all lyrics as they were queried. I then searched for remaining lyrics that could not be found by the API, by hand, and initially used Genius.com to search for the songs, to find the English translation. Remaining songs were found on online sites lyrics.com[15] and songlyrics.com[16], to fill in the dataset as much as possible. There were some songs which did not have an English translation such as 'We Speak No Americano' by Yolanda be Cool, and therefore could not be used in the analysis. There were only 6 cases where lyrics could not be found from any source, leaving 5250 songs in my dataset. Excluding the mentioned song, all the remaining songs which were removed all reached ranks 90-100 in the charts and were perhaps lesser known, and therefore potentially had no available lyrics.

There were some inconsistencies between the format of song and artist names on the Genius API database, in comparison to my dataset. This meant that querying the songs became difficult and a wider knowledge of music was known to correct them. The first inconsistency was with the song name, where the name slightly differed between the API and dataset. An example of this is 'Move that Doh' and 'Move that Dope', both of which are

[13] https://pypi.org/project/lyricsgenius/
[14] https://genius.com/
[15] https://www.lyrics.com/
[16] http://www.songlyrics.com/

the same song by Future. The second inconsistency with the name of the artist where bands or groups had the separately listed names on the dataset, but the API used the group name. An example of this is 'Lemonade Mouth' who consists of members: Bridget Mendler, Adam Hicks, and others. The most common inconsistency was the use of * to avoid writing explicit words. My dataset used * in all cases but the Genius API used the raw versions of the explicit words.

When importing the lyrics from the dataset into a panda's data frame in Jupyter Notebook, Unicode language was used to translate blank lines or spaces in the lyrics. This means the lyrics contains strings such as '\n' or \u2028' which needed to be removed from the lyrics for efficient use of sentiment analysis.

Spotify API

Using Spotify API Python library 'spotipy'[17], I got the Spotify popularity score, explicitness, and release date for each song in the dataset. I did this by querying the Spotify API using the song name and artist. Like the Genius API, there were inconsistencies which led to queries returning incorrect songs and meant manually checking was necessary, to ensure every song found from the API was correct. I did this by querying a song and getting the API to print out the song and artist name from the results of the query, and then choosing whether it is correct or not.

Like Genius, the song and artist name on the billboard dataset differs from how they are stored in Spotify. An example of this includes 'Bigger > You' and 'Bigger Than You' by 2 Chainz. If any song name contained a bracket e.g. 'Don't Stop (Color on the Walls)', the Spotify API would not return any results, meaning they need to be removed before querying the song name, i.e., just use 'Don't Stop'. The main problem was when a song has multiple artists and, in the dataset, these names would be joined using connectors: *Featuring, Feat., With, Duet with, And, &, +, X, x,* and others. Some examples of this are: 'Dev & Enrique Iglesias', 'Drake Featuring Jay-Z' and 'Nicky Jam x J Balvin'. If the artist was queried as it appears in the dataset, then the API would fail. When the first artist was queries only, the API found the correct song, meaning the format of these artists needed to be changed also before being used. After implementing this, I was able to successfully find the explicitness and popularity score for most songs in my dataset. Some songs were not available on Spotify such as 'Iris' by Phoebe & Maggie, as I checked via my personal Spotify account. These were removed from the dataset.

There were many cases where the release date was either incorrect or not able to be found using the API. This is because the API finds the release date of the album, not the single, meaning if the song was not a part of an album, then a date can't be found. Equally, the album release dates are not always the same as the single dates. I attempted to find all release dates using the API and checked the release date was before its peak. Manual searching for the remaining release dates were carried out, and any song without a correct release date was removed. I then checked for any song that has peaked 2 years after its release and taken these songs out and stored them separately as '**Anomalies**' to be analysed later. A dataset with 5168 songs remained. Finally, I calculated the difference in days between the song releases and peaks and discovered 15 days was the median value. For this reason, I used 14 days (2 weeks) as my tweet timeframe.

Twitter API

Python library 'tweepy'[18] along with supporting functions was used to query tweets from the Twitter API. I used the Twitter API to gather 1) up to 100 tweets referring to the song name and artist 2 weeks prior to its release and 2) up to 100 tweets containing #nowplaying, song

[17] https://spotipy.readthedocs.io/en/2.19.0/
[18] https://www.tweepy.org/

name and artist 2 weeks after its release date. The API needed to find historical tweets, so Full-Archive Search[19] was needed. This is a search tool which lets you specify past dates to collect tweets from. Python library 'time'[20] was needed to be imported in order to do this. I queried the song and artist name, start date, end date, maximum results and #nowplaying for the post-release tweets. The default maximum is 100 and can't be any higher.

The API when iterated, would often give Error code 429, meaning the API seemed to time out and needed to be restarted. If certain characters were in the query, like the connectors seen in the Spotify API, then the API would fail with Error code 400. These were all removed along with Boolean operator words 'AND', 'OR', and 'NOT' where the API would fail. There were cases where the API could not find any tweets are all relating to the song within the 4-week time frame and gave Error code 200. This meant the query did not fail nit no tweets were outputted. I removed songs where this was the case, leaving a total of **5150** songs for training the classifiers.

Sentiment Analysis and VADER

Medhat [22] described Sentiment Analysis (SA) as 'the computational study of people's opinions, attitudes, and emotions towards an entity; in our case, the entity is a song. SA will discover the amount of positive, negative, or neutral sentiment portrayed in text. I will be using the VADER method which is a lexicon and rule-based sentiment analysis tool which is specifically altered to analyse sentiments which is expressed in social media [23]. This makes it a good tool for analysing the tweets. The benefit of using VADER of other lexicons is that it can adapt to language using emoticons and slang [23].

The song lyrics, title, and both sets of tweets will be analysed using VADER sentiment analysis to find the compound score. Python has library 'nltk'[21] which can be used to implement this. I took the compound score which is the overall sentiment where 1 is positive, -1 is negative and 0 is neutral.

When implementing sentiment analysis on the Tweets, pre-processing of the tweets was required. The tweets contained the song name and artist many times, which would impact the sentiment score and bias the overall compound score from VADER. For example, 'Bad Guy' by Billie Eilish has a negative name, and as this is repeated in the tweets, the overall tweet sentiment will be impacted negatively, when this was not the intention behind the user's tweet. To account for this, for each set of tweets, I filtered out the song name and artist. I attempted to remove any alternative version of the song and artist name as well, these being lower- and upper-case variants, as well as the inclusion and exclusion of brackets, backslashes, punctuation, and connectors such as 'Featuring', described in both Spotify and Twitter API sections.

6. Machine Learning Methods

Python libraries 'pandas', 'matplotlib', 'numpy'[22] and 'seaborn'[23] are again needed to import and manipulate datasets, and for plotting graphs and results. 'sklearn'[24] in a library which is used for all machine learning methods in my project. The package offers functions

[19] https://developer.twitter.com/en/docs/twitter-api/premium/search-api/quick-start/premium-full-archive
[20] https://docs.python.org/3/library/time.html
[21] https://www.nltk.org/
[22] https://numpy.org/
[23] https://seaborn.pydata.org/
[24] https://scikit-learn.org/stable/

'train_test_split'[25] to split my data. Along with this, it offers 'svm'[26] , 'LogisticRegression'[27] and 'DecisionTreeClassifier'[28] for training my models and 'RepeatedStratifiedKFold'[29] and 'GridSearchCV'[30] for determining the best model parameters. 'sklearn' also offers 'metrics'[31] to evaluate these models. 'SHAP'[32] is a Python package which will be used for feature importance and further analysis on the models.

Cross Validation and Grid Search

Grid Search is a technique which can be used to find the optimal hyperparameters of a machine learning model. These are parameters which are not directly learnt by the model, as is it fit to the training data, and can be specified and therefore be better suited to different problem types [24]. Cross Validation will repeat this process with different subsets of the data to ensure bias is reduced. K Fold Cross Validation allows us to specify a number of sections the data is split into and Repeated Stratified K Fold, repeats the Stratified K Fold n times with different randomization in each repetition [25]. Stratified means the class frequencies in each fold reflect the corresponding class frequencies in the data.

Logistic Regression

Logistic Regression (LR) is used to solve classification problems. LR differs from Linear Regression because instead of fitting a straight line or hyperplane to split the data, the LR model used a sigmoid function to reduce the output of the linear equation between 0 and 1 [26]. 0.5 is the most common used threshold and means an outcome greater than 0.5 will predict class 1 and below will predict class 0 [27]. I will use Binary Logistic Regression to determine if a song if a hit (1) or non-hit (0). An advantage of LR is that the algorithm does not just output the predicted label, but also the probability of the data point being in that class [26]. The probability of a song being a hit with 97% accuracy is undoubtably better than with 56% accuracy. I will alter three hyperparameters in my Grid Search: solver, penalty, and C value. The solver is the algorithm used in the optimisation training, and the penalty can be specified to this solver, and is a regularisation parameter [28]. The C value determines how strong the regularisation is.

Random Forest

A Random Forest (RF) model is an extension of a Decision Tree and can be used for both regression and classification problems. For this investigation, it will be used for classification. The RF model is a collection of a high number of decision trees, which operate as an ensemble [29]. This means that the algorithm aims for the combination of all decision trees to be more accurate than each tree individually. The uncorrelated predictions from all trees in the random forest is collects and the mean is taken to give the final class prediction from the model [29]. Typically, Decision Trees were limited and often suffered from the problem of overfitting. RF overcome this as using multiple trees and taking the average will reduce the variance. I will alter three hyperparameters in my Grid Search: criterion, n of estimators and max depth. Number of estimators is the number of trees in the forest, whilst max depth describes the maximum depth each tree will grow to [28]. Criterion is the measure of quality of a splitting.

---

[25] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
[26] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[27] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[28] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[29] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RepeatedStratifiedKFold.html
[30] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[31] https://scikit-learn.org/stable/modules/model_evaluation.html
[32] https://shap.readthedocs.io/en/latest/tabular_examples.html

Support Vector Machine
The aim of Support Vector Machine (SVM) is to find a hyperplane that distinctly classifiers the data points into two classes. There are many hyperplanes which can correctly do this but the key objective of SVM is to find the hyperplane that maximises the distance between the data points of both classes [30]. The larger the margin, the more generalisation in the classifier. SVM has a kernel function which can be specified and therefore suitable for a wider range of classification and regression problems. I will alter two hyperparameters in my Grid Search: the chosen kernel and C value, which like logistic regression, is the strength of regularisation applied [28].

Evaluation Metrics
I will first evaluate the classification models using the Accuracy, Precision, Recall, and F1-score. Accuracy is the amount of correctly classified data points, out of all data points. Accuracy is often inefficient to use when the dataset is imbalanced, but as I have ensured mine is, then it is a good metric to use. Precision looks at how many positive predictions were correct and is good when the cost of false positives is high [31]. Recall looks at the amount of true positives, out of all data which should be positive, and in contrast, is good when the cost of false negatives is high [31]. F1-score is an overall measure which combined both precision and recall. Following this, I will go on to plot the ROC curve and calculate the AUC score. An ROC curve plots True Positives against False Positives, and the area under this curve (AUC) can determine a 'good' model. An AUC score of 1 is a perfect model, and 0.5 is random choice.

7. Final Dataset

After collecting information from the three APIs and carrying out sentiment analysis, I now have all the data needed to train my classifiers. I will remove the song name, artist, lyrics, and tweets and summarise the features of my dataset:

- *Lyric sentiment: lyric sentiment compound score*
- *Lyric label: positive or negative label from the sentiment*
- *Lyric explicit: Boolean label from Spotify, whether lyrics are explicit or not*
- *#nowplaying volume: number of tweets 2 weeks after release containing song, artist, #nowplaying*
- *#nowplaying sentiment: compound sentiment of tweets 2 weeks after release containing song, artist, #nowplaying*
- *Pre-release volume: number of tweets two weeks prior to songs release containing song and artist*
- *Pre-release sentiment: compound sentiment of tweets 2 weeks prior to release containing song and artist*
- *Title sentiment: compound sentiment score of title*
- *Peak rank: the highest rank the song reached on the Billboard Hot 100*
- *Popularity: the Spotify popularity score*

Final Dataset Analysis
I created two correlation heatmaps with the variables and the hit criteria: peak rank and Spotify popularity [See Figure 4]. We can see that no variables have a significant correlation with peak rank or popularity and will all therefore be used to train my classifiers.

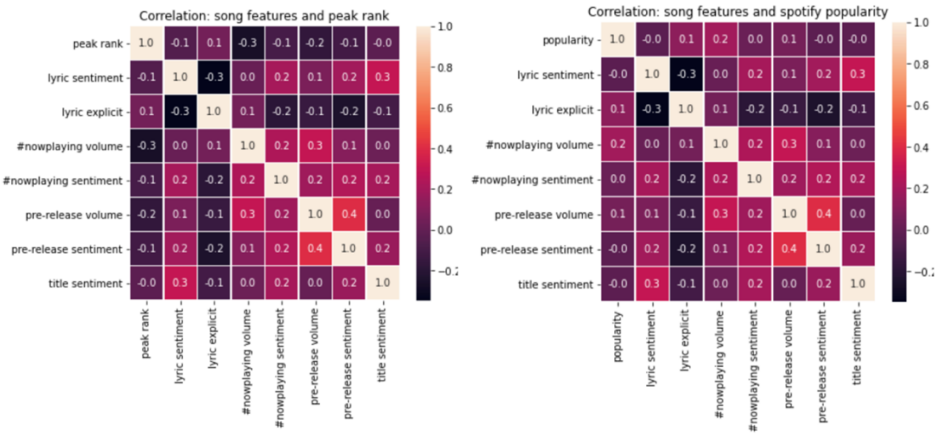


Figure 4: Correlation heatmap with dataset and peak rank (left) and popularity score (right)

I went on to compare hit songs from each year, according to both peak rank and popularity score. First, I plotted the volume of hit songs each year according to both criteria [See Figures 5 and 6]. As the most recent song entry was kept, when removing duplicates, it is expected for 2020 to have more hit songs in the Top 10 songs [See Figure 5], as there would be more song data overall within this year. Otherwise, the amount of Top 10 hit songs remains relatively constant throughout the rest of the years. Looking at the Spotify > 75 hit songs [See Figure 6], the volume of hits significantly increases each year. This could mean more songs are achieving a popularity score of above 75 easier, or there are more songs on Spotify in general as years go on meaning more have higher popularity scores. The popularity score is also the most current value, which could mean songs in older years have had their popularity score increase over recent years and is now considered a hit in 2020.

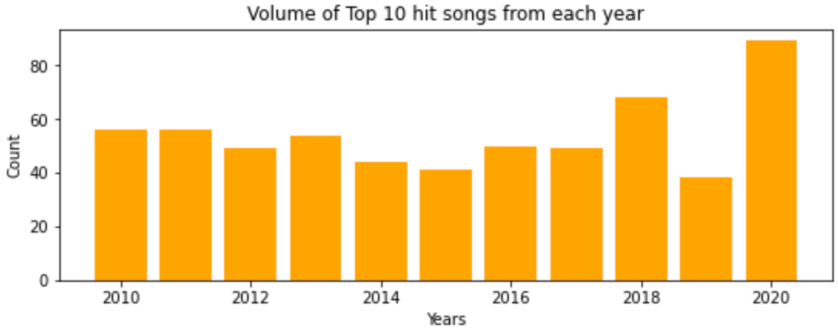


Figure 5: Quantity of Top 10 Hit songs yearly

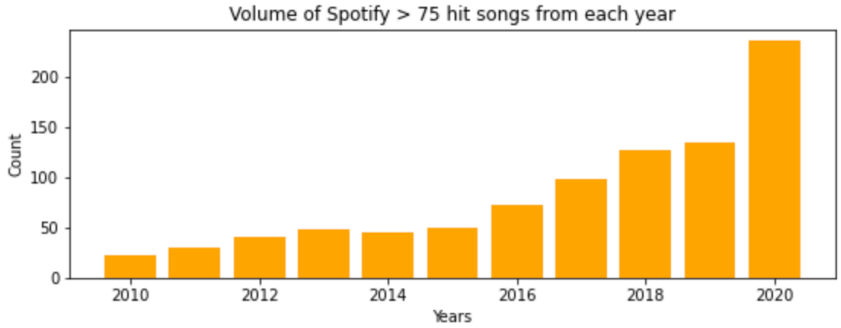


Figure 6: Quantity of Spotify > 75 Hit songs yearly

Looking at the amount of hit songs with explicit lyrics, we can see that in general Top 10 hit songs have a higher percentage of explicit lyrics compared to Spotify > 75 hits [See Figures 7 and 8]. Individually, we can see Top 10 hits saw more explicit songs in earlier years, which decreased by nearly 10% from 2017 onwards [See Figure 7]. Spotify > 75 saw the

opposite trend, where songs were getting more explicit in recent years, in comparison so 2014 which only has 20% of its hit songs being explicit [See Figure 8].

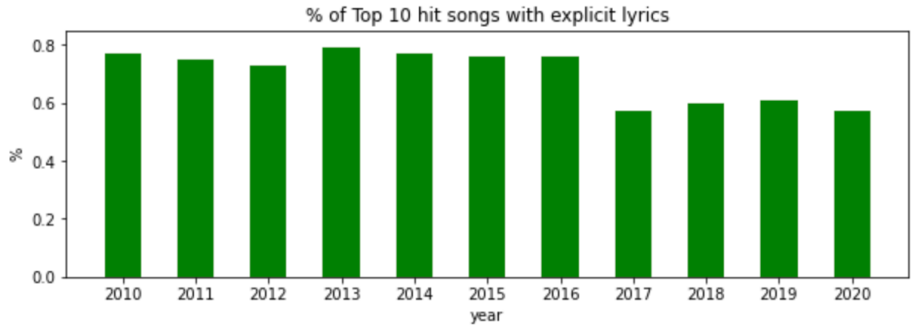


Figure 7: Percentage of explicit lyrics in Top 10 hit songs

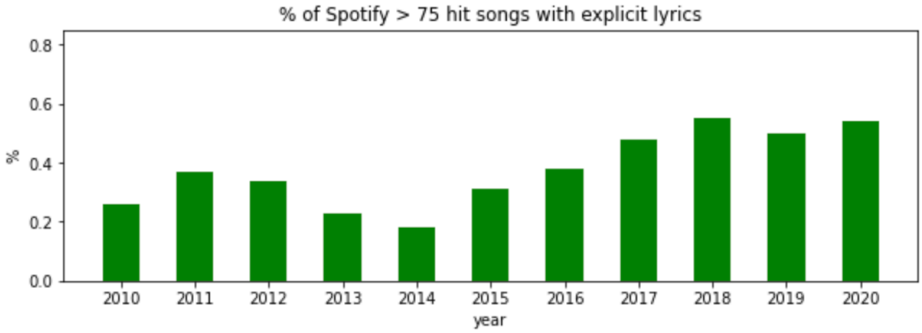


Figure 8: Percentage of explicit lyrics in Spotify > 75 hit songs

I carried on analysing lyrics and looked at the percentage of positive and negative lyrics in hit songs from each year [See Figures 9 and 10]. A similar trend is followed through both hit criteria, where the percentage of negative lyrics increases as years go on. 2010 saw very little negative lyrics, but this increased by 2020 where it was a 40:60 split. Both hit criteria also saw that there were more positive than negative lyrics in every year.

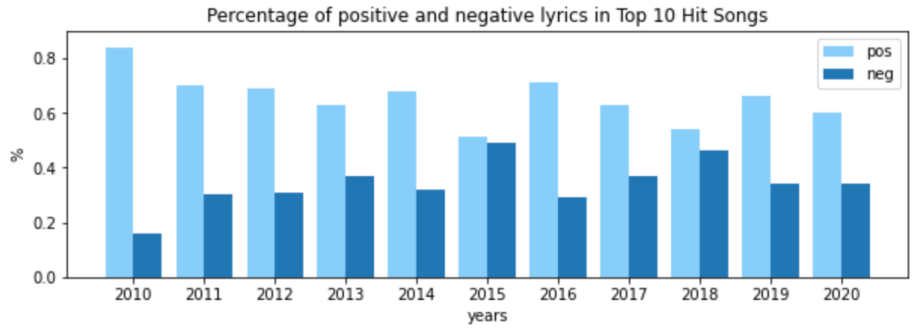


Figure 9: Percentage of lyric sentiments in Top 10 hit songs

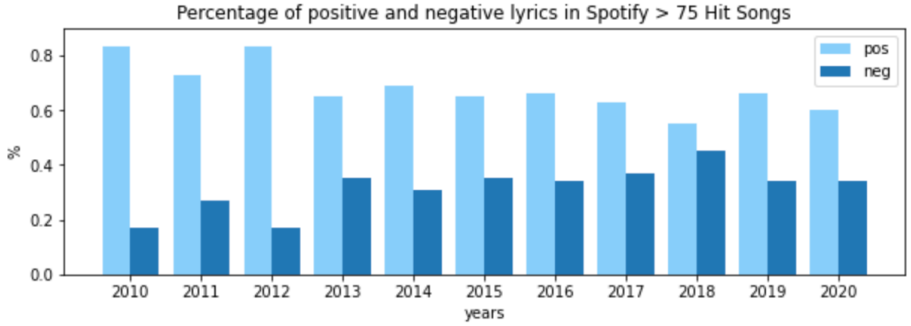


Figure 10: Percentage of lyric sentiments in Spotify > 75 hit songs

I finally looked at the volume of pre- and post-release tweets from hit songs in each year. Looking at Top 10 hits, the volume of pre-release tweets began to climb and remained consistently high until 2017, where a significant decrease was seen [See Figure 11]. This was also the same for Spotify > 75, meaning there were less tweets associated the song prior to its release in more recent years [See Figure 12]. Looking at the volume of tweets post-release, Spotify > 75 hits also seem the same downwards trend, whereas the Top 10 hits still see a high volume of tweets after a song release, even in recent years.

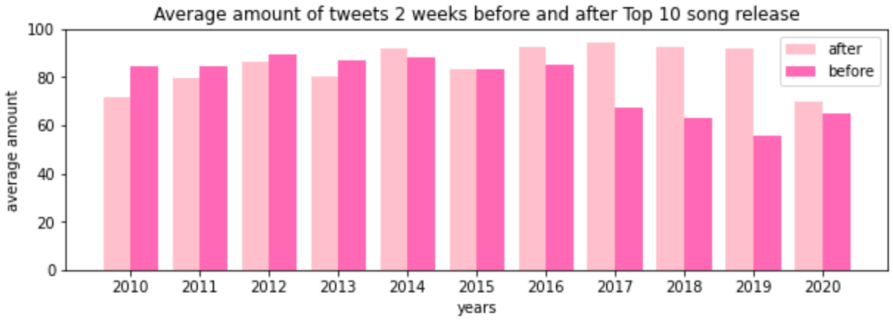


Figure 11: Pre- and post-release yearly tweet volume for Top 10 hits

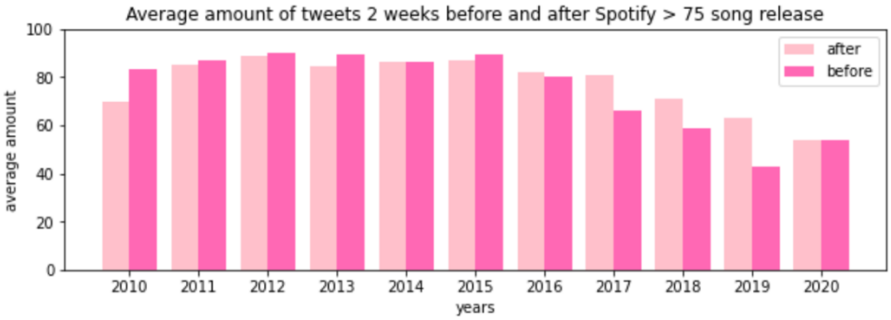


Figure 12: Pre- and post-release yearly tweet volume for Spotify > 75 hits

8. Training Classifiers

I have created six datasets with the features described above, each containing a Boolean variable indicating whether the song is a hit or not according to hit criteria: peak rank in Top: 10, 20, 5 and Spotify popularity score >: 75, 80, 70. Top 10 and > 75 are the two main hit thresholds we are interested in for the investigation, but the remaining ones will be looked at for comparison and further understanding.

Before training the three classifiers with the six datasets, I split the data each time into training and test data. The data was split using an 80:20 split, so there was 20% unseen data to evaluate the performance of the models. When I first trained the model using Top 10 and Spotify > 75 data, I achieved accuracies of above 90% but realised this was due to the datasets being heavily unbalanced. Within the Top 10 data, 594/5150 were hits and within the Spotify > 75, 900/5150 were hits. Training with unbiased data can lead to a naïve model, where the model will predict a non-hit every time and still achieve high accuracy scores there are a small quantity of hit songs in the data. This can be seen in the confusion matrix [See Figure 13], where I fit a Logistic Regression classifier to the unbalanced Top 10 data, and the model did not predict a hit once.

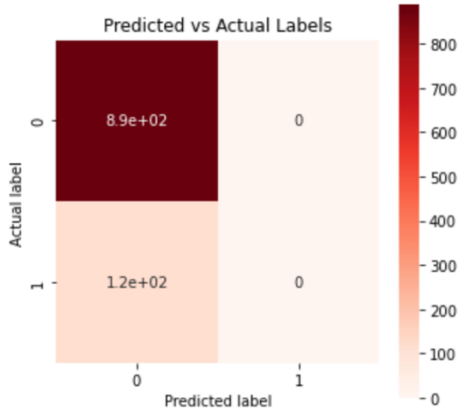


Figure 13: Unbalanced Top 10 data confusion matrix from LR model

I balanced each dataset, so there was an equal amount of hit and non-hit songs according to each of the six hit criteria. I will show these dataset sizes below [See Figure 14].

Size of datasets – Hit : Non-hit Ratio						
	Top 10	**Top 20**	**Top 5**	**> 75**	**> 80**	**> 70**
Ratio	594:594	1039:1039	355:355	900:900	370:370	1600:1600

Figure 14: Table to show balanced datasets

The final method included in training the classifiers was to carry out Cross Validation Grid Search, to see which parameters for the LR, RF, and SVM models were best for each the Billboard and Spotify Criteria. After implementing these methods, I will outline the best parameters for each classifier, for each hit criteria [See Figure 15].

	Top 10, 20, 5	**Spotify > 75, 80, 70**
Logistic Regression	solver = sag penalty = l2 C = 0.01	solver = newton-cg penalty = l2 C = 0.01
Support Vector Machine	kernel = poly C = 0.01	kernel = poly C = 0.01
Random Forest	n_estimators = 40 criterion = entropy max_depth = 5	n_estimators = 80 criterion = entropy max_depth = 5

Figure 15: Optimal parameters for each model

9. Results

The results obtained from training and testing the three machine learning classifiers, using their optimal parameters, can be seen in the tables below [See Figures 16 and 17]. The accuracy score has been entered into the table.

Peak Rank			
	Top 10	**Top 20**	**Top 5**
Logistic Regression	0.672	0.642	0.683
Support Vector Machine	0.672	0.647	0.641
Random Forest	**0.744**	0.620	0.662

Figure 16: Accuracy Scores from classifier: using peak rank as hit criteria

Spotify Popularity Score			
	> 75	**> 80**	**> 70**
Logistic Regression	0.603	0.561	0.611
Support Vector Machine	0.594	0.547	0.576
Random Forest	**0.641**	0.574	0.615

Figure 17: Accuracy Scores from classifier: using Spotify popularity score as hit criteria

Looking at Figures 16 and 17, all classifiers manage to predict a hit song according to Top 10 with an accuracy score over 67% and all Spotify > 75 with an accuracy score over 59%. A Random Forest was by far the best model for predicting both a Top 10 hit (0.744 accuracy) and a Spotify > 75 hit (0.641 accuracy). Both RF classifiers used parameters: max_depth = 5, criterion = entropy, but the Top 10 model found using n_estimators = 40 more optimal, whilst Spotify > 75 used n_estimators = 80. For both best classifiers, I will show the confusion matrix, ROC graph and feature importance.

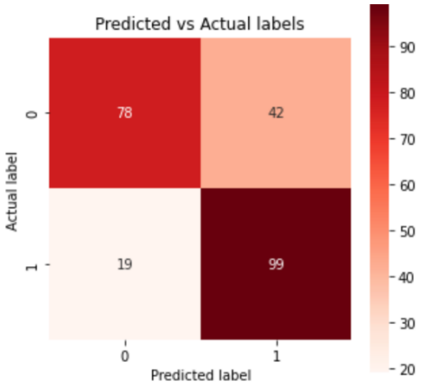


Figure 18: Top 10 RF Confusion Matrix

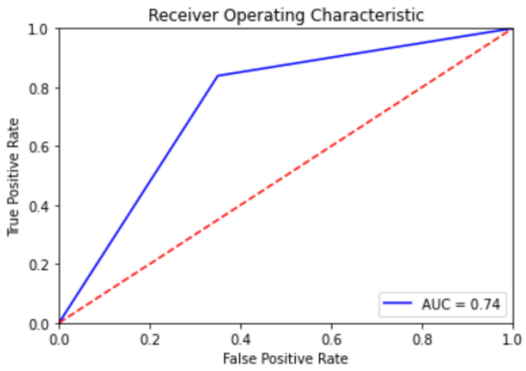


Figure 19: Top 10 RF ROC curve

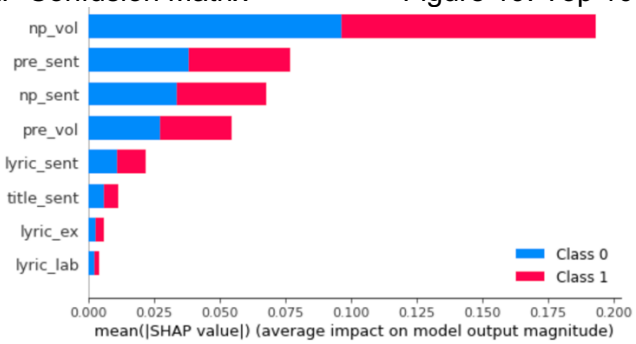


Figure 20: Top 10 Random Forest SHAP Feature Importance

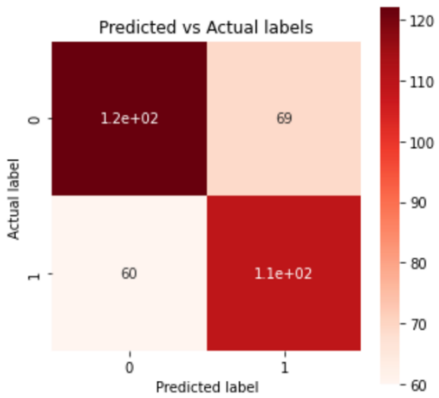


Figure 21: Spot > 75 RF Confusion Matrix

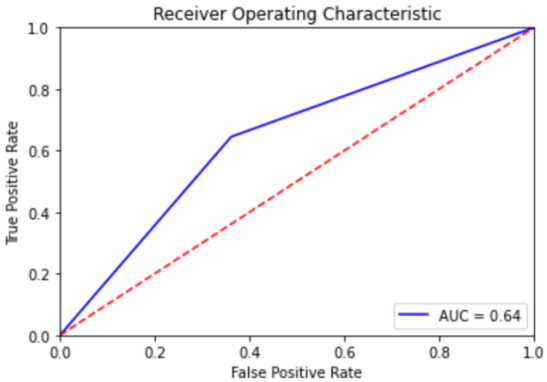


Figure 22: Spot > 75 RF ROC curve

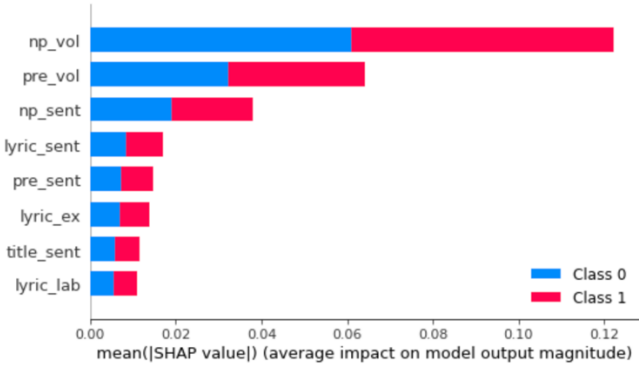


Figure 23: Spotify > 75 RF SHAP Feature Importance

Looking at Figures 18 and 21, we can see the models are no longer naïve and the classifier is predicting hit songs. The ROC curve for the Top 10 Random Forest classifier shows the AUC score is 0.74, which is over 0.5, meaning the model is predicting better than random choice [See Figure 19]. This is the same case for the Spotify > 75 RF classifier AUC score which is 0.64 [See Figure 22]. Looking at the feature importance for both best classifiers [See Figures 20 and 23], we can see that the volume of #nowplaying tweets has the highest predictive power in both models. The Top 10 Random Forest then finds the tweets sentiment has the next largest impact on the model, before the volume of pre-release tweets and the sentiment of the lyrics [See Figure 20]. This is opposed to the Spotify > 75 RF which finds the volume of pre-release tweets more helpful than the sentiments [See Figure 223. Both models find the title sentiment, lyric explicitness and positive/negative lyric label the least powerful in the predictions.

10. Discussion

The initial and most significant point to discuss is that the project was successful, and classifiers were successful in being able to predict a hit song according to both hit criteria with a promising accuracy score [See Figure 16 and 17]. Predicting a Top 10 proved a very positive accuracy with a score of 0.744 but the remaining accuracy scores achieved have a lot of room for improvement, but they give us a valuable indication that the combination of lyrics and tweets can be used in this way to predict if a song will become a hit. It is more important to be able to predict correctly when a song will become a hit, rather than thinking a song will become a hit and then it being unsuccessful. This is because producers do not want to spent time and resources on the creation, advertisement and release of a song thinking it will return profit, and then not doing well. The precision score from these classifiers is therefore more important, as the cost of false positives is high. When producing the classification reports [See Appendix 2 and 3], both the recall and precision scores matched the accuracy scores closely, so this is not a concern.

The feature importance [See Figures 20 and 23] gave insights into which features have high predictive power in this problem. As shown, the sentiment of tweets and lyrics, along with the volume of lyrics are key variables in training the classifiers for both hit criteria. This is information which will be useful to artists and music producers. Artists now know that the volume and sentiment of tweets relating to their music, pre- and post-release have the potential to significantly impact how well their song does in both the charts and on Spotify. Artists can then focus their advertising on social media, and make sure to frequently post and maintain an online presence, to ensure users are posting and tweeting about their excitement for the music release, and whether they are listening after the release.

Throughout all six hit criteria investigated in the previous section, Spotify > 80 consistency did not reach 60% accuracy across all classifiers [See Figure 17]. This may indicate that there is not much difference between songs with just below score 80 and just above,

meaning the classifiers sound it difficult to distinguish between a hit and non-hit, leading a low accuracy score. It is also interesting to note that all hit criteria achieved similar scores for the Billboard hit definitions, regardless of the dramatic differences in dataset size. The datasets varied from 700 to 3200 data points [See Figures 14], and the classifiers still managed to capture enough of the data relationship and trends to predict with above 0.6 accuracy even with the smallest data size.

Journey of One Song

I will look at the journey of one song within the dataset, to show in depth how it went through the data preparation and model training process. The song will be 'Radioactive' by Imagine Dragons, which was the longest remaining song on the dataset in terms of weeks on the board [See Figure 1]. 'Radioactive' was released 5 days before entering the charts on the 18th of October 2012 and left 87 weeks later on the 10th of May 2014. If we look at the songs ranking during these weeks [See Figure 24], we can see the song first reached its peak rank of 3. The song altered between ranks 3 and 4 for 9 weeks before declining down the ranks [See Figure 25]. The song is therefore a hit according to the Billboard criteria, as it is in the Top 10.

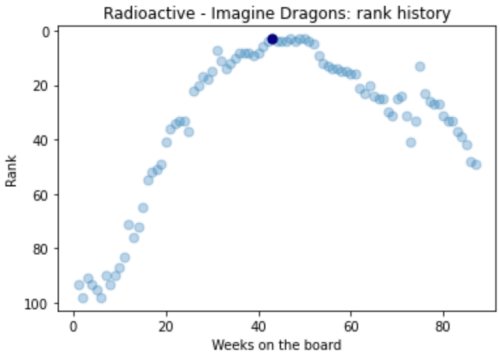


Figure 24: song rank versus weeks.

date	rank	song	artist	last-week	peak-rank	weeks-on-board
2013-07-06	3	Radioactive	Imagine Dragons	4.0	3	43
2013-07-13	4	Radioactive	Imagine Dragons	3.0	3	44
2013-07-20	4	Radioactive	Imagine Dragons	4.0	3	45
2013-07-27	4	Radioactive	Imagine Dragons	4.0	3	46
2013-08-03	3	Radioactive	Imagine Dragons	4.0	3	47
2013-08-10	4	Radioactive	Imagine Dragons	3.0	3	48
2013-08-17	3	Radioactive	Imagine Dragons	4.0	3	49
2013-08-24	3	Radioactive	Imagine Dragons	3.0	3	50
2013-08-31	4	Radioactive	Imagine Dragons	3.0	3	51
2013-09-07	5	Radioactive	Imagine Dragons	4.0	3	52

Figure 25: song peak ranking data entries

Using the Genius API query [See Figure 26], the lyrics were outputted [See Figure 27]. After removing '\n' strings from the lyrics, the VADER lexicon calculated a lyric sentiment compound score of 0.9579, meaning the lyrics have very high positive sentiment, as the score is close to 1.

```
genius.search_song("Radioactive", "Imagine Dragons").lyrics
```

Figure 26: Genius API song query

```
"Radioactive Lyrics[Intro]\nWhoah-oh\nWhoah-oh\nWhoah-oh\nWhoah\n\n[Verse 1]\nI'm waking up to ash and dust\nI wipe m
y brow and I sweat my rust\nI'm breathing in the chemicals\nYeah, ah\n\n[Refrain]\nI'm breaking in, shaping up\nThen
checking out on the prison bus\nThis is it, the apocalypse\nWhoa\n[Pre-Chorus]\nI'm waking up, I feel it in my bones
\nEnough to make my system blow\nWelcome to the new age, to the new age\nWelcome to the new age, to the new age\n\n[C
horus]\nWhoa-oh, whoa\nI'm radioactive, radioactive\nWhoa-oh, whoa\nI'm radioactive, radioactive\n\n[Verse 2]\nI rais
e my flags, dye my clothes\nIt's a revolution, I suppose\nWe're painted red to fit right in\nWhoa\n\n[Refrain]\nI'm b
```

Figure 27: Section of song lyrics from Genius API

Using the Spotify API query [See Figure 28], it outputted the Spotify popularity score as 73 and the lyric explicitness as False. The song is not a hit according to the Spotify criteria, as the popularity score is not above 75. This is unexpected as it has remained on the charts for so long. This shows there is not always a direct correlation with the Billboard charts and popularity score from Spotify.

```
sp.search(q='artist:' + "Imagine Dragons" + ' track:' + "Radioactive", limit=1)
```

Figure 28: Spotify API song query

Along with preliminary code, the Twitter API queries [See Figure 29] were used to find volume of pre- and post-release tweets. Both sets of tweets had a volume of 99, pre-tweets had sentiment 0.9952 and post-tweets had sentiment 0.9743. The title sentiment was also calculated and had score 0, meaning it is neutral.

```
create_url("radioactive, imagine dragons lang:en", "2012-07-31T00:00:00.000Z",
           "2012-08-13T00:00:00.000Z", 100)
create_url("#nowplaying, radioactive, imagine dragons lang:en", "2012-08-13T00:00:00.000Z",
           "2012-08-27T00:00:00.000Z", 100)
```

Figure 29: Twitter API song queries: pre- and post-release

The row of data [See Figure 30] was used to train the 6 classifiers, according to both hit criteria. We expect the Top 10 classifiers to output 1 (hit) and Spotify > 75 classifiers to output 0 (non-hit). Looking at the results [See Figure 31], we can see this is not the case and all three Spotify > 75 classifiers also predicted the song to be a hit. This is not unexpected as the classifiers are only 64-74% accurate, so 4/10 times a hit song prediction will be incorrect.

	lyric_sent	lyric_lab	lyric_ex	np_vol	np_sent	pre_vol	pre_sent	title_sent
0	0.9579	1	0	99	0.9743	99	0.9952	0.0

Figure 30: Song data entered in classifiers for prediction

	Top 10	Spotify > 75
LR	1	1
SVM	1	1
RF	1	1

Figure 31: Results table from song data predicting Top 10 and Spotify > 75

Anomalies

The ‘Anomalies’ are songs that were removed from the dataset in Section 5 because they had peaked over 2 years from its release date. The first trend to investigate is the concept of Christmas songs and over half of the Anomaly songs were Christmas songs. These 24 Christmas songs are ones which appear in the charts in December every year and are cyclic in nature, even though they were released early or pre-2000. Every year however, they seemed to reach further up the charts, as most of them peaked in recent years 2019 and 2020. Christmas is becoming more advertised and used by businesses every year to generate more profit. This means the songs will be played more and could lead to them rising in the charts each year. Another reason is because society is using technology more, especially digital music streaming. Families and older generations are also beginning to stream music, meaning they will stream Christmas songs, leading to them reaching higher ranks.

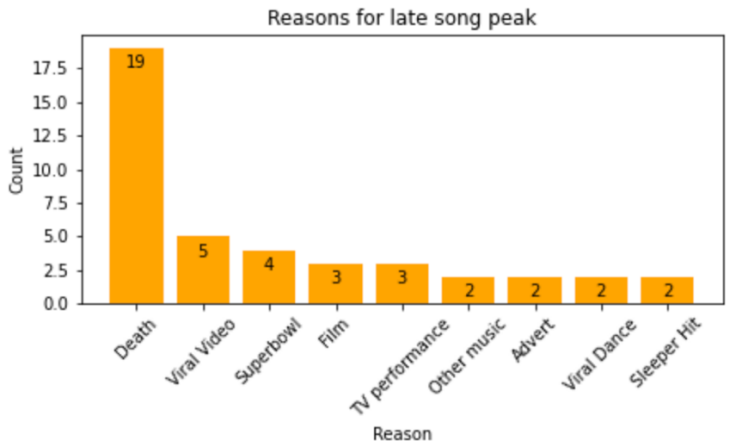


Figure 32: Potential reasons for anomaly songs re-peaking

The remaining songs potentially re-peaked for the following reasons:

- ‘Thriller’ – Michael Jackson peaked in November 2020 due to either Halloween the previous month, or a Netflix documentary about Michael Jackson being released in 2019
- ‘Alone’ – Marshmello peaked in February 2020 due to Marshmello preforming a virtual live concert in the video game Fortnite in 2020
- ‘Breakeven’ – The Script peaked in October 2010 because the video was voted number 2 in the 2020 VH’s Top 40 videos of the year
- ‘Bad Romance’ – Lady Gaga peaked in 2017 but I could not find any potential reasoning as to why this occurred.

We can see from Figure 32 that the death of the artist was the main reason why their music would re-enter and peak in the charts, years after the release of the song. There were cases where a lot of the artists old songs would enter in the same week of the chart. For example, Prince died in April 2016 and 8 of his songs peaked in May 2016, even though they were released in the 80s. It is interesting to see that songs have regained popularity after being used in viral videos. This shows that the internet and social media have a big impact on a song’s popularity. Examples of this are: ‘Wop’ – J. Dash which peaked in 2013 when Miley Cyrus posted a video of herself twerking to the video, or ‘Get Me Bodied’ – Beyonce who peaked in 2013 after a viral video of a flash mob dance was filmed and posted minutes before a patient had surgery. Artists and producers could use this information to advertise older songs in a variety of ways, in attempts to regain popularity. They want to aim to have the song appear in upcoming films and adverts, so they become re-noticed. The artist’s advertising team would also employ younger marketers who can focus on advertising via TikTok videos, using the song in hopes that it will become a viral video on the internet. TikTok is a social media where small videos are posted by users and become viral for other users to watch, react to, like and share.

A sleep hit is a song which takes longer than average to reach high ranks in the chart, but for no specific reason. It is sometimes the case that the public take a while to begin listening to it or the song naturally becomes popular in later years if it is played more frequently on the radio, in clubs, ect.

Literature

My results support those seen by Tsiara and Tjortjis [15], who also obtained promising results when predicting Top 10, 5 and 20 rankings on the Billboard charts using Twitter data. My project differed slightly by using #nowplaying in my Twitter queries and extending from Tsiara and Tjortjis by adding lyrical features. Tsiara and Tjortjis found moderate correlations between the volume of tweets and the chart position. The feature importance I carried out mirrors this and concluded the volume of tweets had high predictive power when predicting a Top 10 hit.

My methodology was similar to Kim, et. al [17], who used predicted Top 10 Billboard rank using #nowplaying tweets. Their investigation achieved higher accuracy levels of 0.9, but did include weeks on the board, which I chose to remove due to bias, and is a possible explanation for the decrease in accuracy. I used a larger volume of twitter data, including the pre-release tweets of the songs, which could allow the classifier to understand the relationship between a song tweets and Billboard rankings further. My results opposed those of Pachet and Roy [9], or Borg and Hokkanen [10] who concluded HSS was not yet a science. My research however, used a very different dataset which is the likely reason as to why better results were seen.

To my best knowledge, no literature used a combination of pre- and post-release tweets, along with lyrical sentiment and explicitness, which is therefore an extension of the work in HSS and could provide valuable insights into predicting a hit song. Significantly, I found few to no papers which address predicting the Spotify popularity score, only charts within Spotify like Araujo's [18] research. This means my research is one of the first within this field and could be useful to Spotify when choosing songs to recommend to users or which songs to make available to stream.

11. Ethical Issues, Considerations and Drawbacks

The primary acknowledgement to make is that half of my dataset is subject to the population of active Twitter users only. Out of the entire music listening population, only a niche group will be these active users. However, we do gain insight into how these users work and whether their influence online can determine a hit song according to both the music listening population of the US and the subscribers of Spotify. It is also important to note the Spotify popularity score is based on Spotify users only, which is a subset of the digital music streaming population. It is becoming more popular, and less people are buying records and CDs for music, however, so will begin to match chart rankings in recent years. These inconsistencies would be a potential reason why all classifiers, apart from 1, did not achieve an accuracy score above 0.7.

One drawback of my method, is that I balanced all the datasets, meaning some classifiers were trained with very little data. I could alter the time frame or hit criteria in order to gather more data, in attempts to achieve higher accuracies. Using Twitter and Spotify data limited my research as both were invented in 2006, so no data could be collected before this point. Christmas songs and other older songs which re-peaked needed to be removed as it was not possible to collect any information from them. This is not a concern for my project definition currently, as it is from 2010-2020 but would become a problem if I took the time frame any further into the past.

Even though the VADER lexicon is a good method for implementing sentiment analysis on social media text, it must still be recognised that it will not be perfect at recognising slang and sarcasm as the language used in younger generated changes to frequently, and a word can have multiple meanings. It would prove difficult for the VADER lexicon to know which meaning is being used. An example of this is the word 'sick' which means disgust but is used in younger generations to mean 'cool'. This is a limitation of my project, but I researched into different lexicons to select the one most suitable. Another limitation is my method of collecting the song release dates. The Spotify API finds the album date, and not the single, so a lot of manual corrections was required, meaning there will be missed and potentially incorrect release dates within the data. In future, a different method to collecting release dates could be used for more quality assured data collection. More consistent data collection methods, where the use of manually checking is not required, could lead to better results. I should also be noted that I collect the lyrics using the Genius API but analyse their explicitness through the Spotify API. Both APIs should have the same lyrics for each song, but if they do not, then again a more consistent method is needed and perhaps the lyrics could be obtained through a method using the Spotify API instead.

12. Conclusion

My investigation into whether lyrics and tweets can be used to predict a hit song showed promising, but not perfect results. There is evidence to suggest that lyrical sentiment and explicitness, along with the volume and sentiment of associated tweets can indicate if a song will be a hit according to both the Billboard charts and Spotify popularity score. My project consisted of using a premade Billboard dataset, using three APIs to collect a range of data related to the song, implementing sentiment analysis, and then using this information to train

Logistic Regression, Support Vector Machine and Random Forest binary classifiers. I found that a Top 10 hit is best predicted by a Random Forest classifier with 0.74 accuracy score and a Spotify > 75 hit is also best predicted by Random Forest model with 0.64 accuracy. Throughout the training and testing of the models, I implemented feature importance and determined that the volume and sentiment of the tweets were key indicators of the song's success. Analysing the Anomalies dataset concluded that Christmas songs re-enter the charts in cyclic nature and the older Christmas songs are the ones remain popular and do not get out-cast by newer versions. Along with this, I summarised that the primary reason a song will re-enter the charts years after its release, is if a popular artist dies, causing their old music to peak in the charts. My research falls in with other literature and replicated the results found, but also has the potential for new insights not yet researched into. To conclude, my investigation into HSS was somewhat successful and is valuable to outside businesses and music producers who can focus their marketing on social media such as Twitter and create a hype around the song's release. Society follows trends, so marketers also know the first two weeks after the song's release are critical and to get as many people listening and posting about the song as possible.

13. Further Investigation

In previous literature, investigations into HSS have taken place using audio features only, or a combination of audio with twitter or lyrical. However, to my knowledge, there are no cases which use all three feature sets together to predict a hit song. This is potential development for my project as the Spotify API can also be used to add audio features to my dataset, as seen by Middlebrook and Sheik. I could determine if a combination of all three features improve the classifier, instead of lyrical and twitter data only.

My project looks at predicting Billboard Hot 100 charts, which consist of popular songs, that are a mixture of genres but primarily pop. Following on from the work of Herremans, et. al [20], my classifier could be used on specific genre charts such as dance or rock. I could see if my classifier can predict hit songs in particular genres, or whether it is better at predicting for some genres over others.

I analysed song lyrics through sentiment analysis, but there are other methods which could scrape features from song lyrics. 'Bag of Words' could be used to find frequently used words in songs to see if there is a trend between certain words appearing and the likelihood of the song being a hit. The emotions portrayed in songs could also be analysed e.g., percentage of happy or anger sentiment, to see if this is useful for predicting a hit song.

14. Code

All code notebooks and csv files are included. Inside the folder is a document called "Code Summary", which explains the purpose of each notebook and which datasets are imported and exported. For any notebooks where I used a website for code reference, I linked it at the top of the notebook and in the code summary. Some notebooks were used for testing and understanding, and others for the actual implementation. Any code used in this project can be accessed via link:
https://drive.google.com/drive/folders/1JBM4vt9No9RJfsw7mxvD6z04Mf69XYx1?usp=sharing

15. References

[1] M. Iqbal, "Spotify Revenue and Usage Statistics," *BusinessOfApps*, Jan. 19, 2020. https://www.businessofapps.com/data/spotify-statistics/ (accessed Apr. 14, 2022).

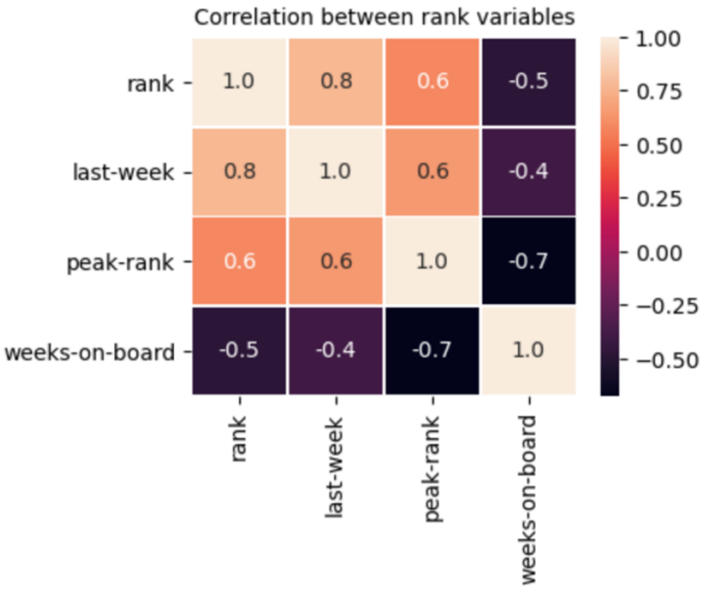
[2] Statista Research Department, "Twitter: number of monetizable daily active users worldwide 2017-2021," *statista.com*, Mar. 24, 2022.

https://www.statista.com/statistics/970920/monetizable-daily-active-twitter-users-worldwide/ (accessed Apr. 14, 2022).
[3] Billboard Staff, “Billboard Finalizes Changes to How Streams are Weighted for Billboard Hot 100 & Billboard 200,” *Billboard*, Jan. 05, 2018. https://www.billboard.com/pro/billboard-changes-streaming-weighting-hot-100-billboard-200/ (accessed Apr. 14, 2022).
[4] M. C. Götting, “Spotify’s monthly active users 2015-2021,” *statista.com*, Feb. 10, 2022. https://www.statista.com/statistics/367739/spotify-global-mau/ (accessed Apr. 14, 2022).
[5] O. Eichler, “Spotify Popularity – A unique insight into the Spotify algorithm and how to influence it,” *The Songstats Lab*, Oct. 07, 2020. https://lab.songstats.com/spotify-popularity-a-unique-insight-into-the-spotify-algorithm-and-how-to-influence-it-93bb63863ff0 (accessed Apr. 14, 2022).
[6] F. Pachet, “Hit Song Science,” 2012.
[7] B. Logan and R. Dhanaraj, “Automatic Prediction of Hit Songs. Automatic Prediction of Hit Songs,” 2005. [Online]. Available: https://www.researchgate.net/publication/220723049
[8] J. Lee and J. S. Lee, “Music popularity: Metrics, characteristics, and audio-based prediction,” *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 3173–3182, Nov. 2018, doi: 10.1109/TMM.2018.2820903.
[9] P. Roy, F. Pachet, and S. Csl, “Spotify HIT SONG SCIENCE IS NOT YET A SCIENCE,” 2008. [Online]. Available: https://www.researchgate.net/publication/220723429
[10] N. Borg and G. Hokkanen, “WHAT MAKES FOR A HIT POP SONG? WHAT MAKES FOR A POP SONG?”
[11] Y. Ni, R. ul Santos-Rodríguez, M. Mcvicar, and T. de Bie, “Hit Song Science Once Again a Science?” [Online]. Available: http://www.theofficialcharts.com/
[12] D. Herremans, D. Martens, and K. Sörensen, “Dance Hit Song Prediction,” *Journal of New Music Research*, vol. 43, no. 3, pp. 291–302, Jul. 2014, doi: 10.1080/09298215.2014.881888.
[13] K. Middlebrook and K. Sheik, “Song Hit Prediction: Predicting Billboard Hits Using Spotify Data,” Aug. 2019, [Online]. Available: http://arxiv.org/abs/1908.08609
[14] A. H. Raza and K. Nanath, “Predicting a Hit Song with Machine Learning: Is there an apriori secret formula?,” in *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics, DATABIA 2020 - Proceedings*, Jul. 2020, pp. 111–116. doi: 10.1109/DATABIA50434.2020.9190613.
[15] E. Tsiara and C. Tjortjis, “Using twitter to predict chart position for songs,” in *IFIP Advances in Information and Communication Technology*, 2020, vol. 583 IFIP, pp. 62–72. doi: 10.1007/978-3-030-49161-1_6.
[16] E. Zangerle, M. Pichl, B. Hupfauf, and G. ¨ Unther Specht, “CAN MICROBLOGS PREDICT MUSIC CHARTS? AN ANALYSIS OF THE RELATIONSHIP BETWEEN #NOWPLAYING TWEETS AND MUSIC CHARTS,” New York City, Aug. 2016. [Online]. Available: http://spoti.fi/J1Gqhs
[17] Y. Kim, B. Suh, and K. Lee, “Nowplaying the future billboard: Mining music listening behaviors of twitter users for hit song prediction,” in *SoMeRA 2014 - Proceedings of the 1st ACM International Workshop on Social Media Retrieval and Analysis, Co-located with SIGIR 2014*, 2014, pp. 51–55. doi: 10.1145/2632188.2632206.
[18] C. V. S. Araujo, “A Model for Predicting Music Popularity on Spotify,” *Extended Abstracts for the Late-Breaking Demo Session of the 21st Int. Society for Music Information Retrieval Conf.*, 2020.
[19] B. Logan, A. Kositsky, P. J. Moreno, and P. Moreno, “Semantic Analysis of Song Lyrics Semantic Analysis of Song Lyrics*,” 2004. [Online]. Available: http://www.cee.org/rsi/index.shtml
[20] D. Dhruvil, “Billboard ‘The Hot 100’ Songs.” Accessed: Apr. 14, 2022. [Online]. Available: https://www.kaggle.com/datasets/dhruvildave/billboard-the-hot-100-songs/metadata
[21] C. Hoffman, “What Is An API, and How Do Developers Use Them?,” *How-To Geek*, Aug. 12, 2021. https://www.howtogeek.com/343877/what-is-an-api/ (accessed Apr. 14, 2022).
[22] W. Medhat, A. Hassan, and H. Korashy, “Sentiment analysis algorithms and applications: A survey,” *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, Dec. 2014, doi: 10.1016/j.asej.2014.04.011.

[23] F. Singh, “Sentiment Analysis Made Easy Using VADER,” *Developers Corner*, Dec. 08, 2020. https://analyticsindiamag.com/sentiment-analysis-made-easy-using-vader/#:~:text=VADER%20(Valence%20Aware%20Dictionary%20and,the%20polarities%20i.e.%20positive%2Fnegative (accessed Apr. 14, 2022).
[24] R. Joseph, “Grid Search for Model Tuning,” *towardsdatascience.com*, 2018. https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e (accessed Apr. 22, 2022).
[25] GeeksForGeeks, “Stratified K Fold Cross Validation,” *GeeksForGeeks.com*, May 29, 2021. https://www.geeksforgeeks.org/stratified-k-fold-cross-validation/ (accessed Apr. 22, 2022).
[26] C. Molnar, “5.2 Logistic Regression,” *Interpretable Machine Learning*, Mar. 2022, Accessed: Apr. 14, 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book/index.html#summary
[27] H. Belyadi and A. Haghighat, “Chapter 5 - Supervised learning,” in *Machine Learning Guide for Oil and Gas Using Python*, Gulf Professional Publishing, 2021, pp. 169–295. Accessed: Apr. 14, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128219294000044
[28] J. Brownlee, “Tune Hyperparameters for Classification Machine Learning Algorithms,” *Machine Learning Mastery*, 2019. https://machinelearningmastery.com/hyperparameters-for-classification-machine-learning-algorithms/ (accessed Apr. 22, 2022).
[29] T. Yiu, “Understanding Random Forest,” *Towards Data Science*, 2019. https://towardsdatascience.com/understanding-random-forest-58381e0602d2#:~:text=The%20random%20forest%20is%20a,that%20of%20any%20individual%20tree (accessed Apr. 14, 2022).
[30] R. Gandhi, “Support Vector Machine — Introduction to Machine Learning Algorithms,” *Towards Data Science*, 2018. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (accessed Apr. 14, 2022).
[31] “Evaluation Metrics for Machine Learning - Accuracy, Precision, Recall, and F1 Defined,” *pathmind*. https://wiki.pathmind.com/accuracy-precision-recall-f1#one (accessed Apr. 14, 2022).

16. Appendix

Appendix 1 – Correlation Heatmap for Initial Dataset variables



Appendix 2 – Classification Report for Top 10 RF model

	precision	recall	f1-score	support
0	0.80	0.65	0.72	120
1	0.70	0.84	0.76	118
accuracy			0.74	238
macro avg	0.75	0.74	0.74	238
weighted avg	0.75	0.74	0.74	238

Appendix 3 – Classification Report for Spotify > 75 SVM model

	precision	recall	f1-score	support
0	0.67	0.64	0.65	191
1	0.61	0.64	0.63	169
accuracy			0.64	360
macro avg	0.64	0.64	0.64	360
weighted avg	0.64	0.64	0.64	360