Song Hit Prediction

Statistical Learning
Experimental Project Report

University of Milan
Department of Economics, Management and Quantitative Methods
Master's Degree in Data Science and Economics
Academic year 2021/2022

Abstract

The global recorded music market grew by 7.4% in 2020, the sixth consecutive year of growth, according to IFPI, the organization that represents the recorded music industry worldwide. The value of the market has been estimated to be $21.6 billion. It is hard to get a hint of how many songs an artist releases in a specific time frame but, given the fact that generally a viral song is very much less frequent than a not viral one, knowing if a specific song will be popular would give a hedge to producers and boost even more the revenues. In the current study, the problem of predicting whether a song would become a hit or non-hit was addressed using the main objective features of a song such as acousticness, tempo and danceability. A dataset of approximately 12.500 songs was created retrieving for each one the features from the Spotify Web API. After learning about the dataset through exploratory data analysis and clustering, the success of a song was able to be predicted with approximately 91% accuracy on the test set for the best model. The most successful models were the logistic regression and the tree random forest. The model that performed very poorly was the quadratic linear discriminant.

Table of Contents

1. Introduction

This section presents a general and brief overview of the sources and features of a song. In the final chapter, the structure of the paper is defined.

1.1 Sources of data

1.1.1 The Billboard Hot 100

The Billboard Hot 100 is the music industry standard record chart in the United States for songs since 1955, published weekly by *Billboard* magazine. Chart rankings are based on sales (physical and digital), radio play, and online streaming in the United States. In other words, the Billboard Hot 100 Chart remains one of the definitive ways to measure the success of a popular song. As you might have noticed, we constructed the definition of the *hit* variable around this context. If a song made it into the chart, it signifies that it is popular and therefore a hit. Thus, all the songs present in the charts of Billboard from each year since 1955 to 2021 were retrieved using the Billboard API. The library provided the *track_name* and the *artist_name.*

1.1.2 The Million Song Dataset

A dataset of 10,000 random songs was collected from the Million Songs Dataset (MSD), a free dataset maintained by labROSA at Columbia University and EchoNest. This was narrowed down to songs released between 2007 and 2021 in order to counterbalance the skewness of the dataset. The dataset provided the artist name and song title, as well as other miscellaneous features. Finally, we removed overlapping songs. At this point, tracks were labeled 1 or 0: 1 indicating that the song was featured in the Billboard Hot 100 (between 2007- 2021) and 0 indicating otherwise.

1.1.3 Spotify

Spotify API, Spotipy! to extract audio features for these songs. The Spotify API provides users with 11 audio features, and other information such as total followers and artist popularity on Spotify. This dataset was then merged with previous one to form the final one. Furthermore, the artist's name is associated with a binary variable *featuring* to indicate whether an artist has collaborated with one or more other artist for a song.

1.2 Features of a song

A song has several features: some are subjective, and some are objective. For this study, it was opted to use as inputs only objective features that are outlined briefly below.

1.2.1 Acousticness

A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.

1.2.2 Danceability

Describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.

1.2.3 Energy

Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy.

1.2.4 Instrumentalness

Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal." The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.

1.2.5 Liveness

Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.

1.2.6 Speechiness

Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.

1.2.7 Tempo

The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

1.2.8 Valence

A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

1.2.9 Time Signature

Time signatures consist of two elements: a top number and a bottom number. The top number tells us the number of beats in each measure. The bottom number in time signature tells you what note values those beats are. Time signatures are what give a song its beat. The term "Four on the Floor" refers to dance music that will always be in 4/4 – four beats per measure because that's the best beat to dance to.

1.3 Paper Structure

The project is structured as follows:

II. *Dataset and Exploratory Data Analysis (EDA)*
III. *Feature Selection*
IV. *Model Definition*
V. *Conclusions and limitations*
VI. *References*

In chapter II, the dataset composition is outlined, and its characteristics are summarized. In chapter III, the fundamental steps to prepare the data for the analysis are run through as well as the process of selecting the features for an alternative analysis. In chapter IV, the machine learning models are presented along with the architecture choice, the hyperparameters tuning and the performances. In chapter V, the conclusions of the study are exposed with the results, the inherent limitations and the recommendations for future work.

2. Dataset and Exploratory Data Analysis (EDA)

In this section, the original structure of the dataset *song* is decomposed in detail. In the meanwhile, the findings of the Exploratory Data Analysis (EDA) and unsupervised learning are shared.

2.1 Dataset Overview

The song dataset contains around 12.500 records of songs with artist information and audio features. The variables considered for the project are listed and explained below:

- *hit:* whether a song is a hit or not
- *featuring:* whether there are other artists that contributed to a song
- *artist_name:* the name of the artist of a song
- *pop_artist:* the popularity of an artist ranked from 0 to 100
- *tot_followers:* number of followers on spotify
- *track_name:* name of the song
- *rel_date:* date in which a song is released
- *pop_track:* popularity of track ranked from 0 to 100 ***(it can differ from our definition of hit)***
- *avail_mark:* number of markets in which the song is present from 1 to 178

In addition, a variable for each feature is considered:

- *acousticness*
- *danceability*
- *duration_ms*
- *energy*
- *instrumentalness*
- *liveness*
- *loudness*
- *Speechiness*
- *tempo*
- *time_signature*
- *valence*

2.2 Exploratory Data Analysis

After the dataset was overviewed with a statistic summary, a check for outliers was perpetrated with negative results through box plots. Then, the data was normalized to have mean equal to 0 and standard deviation equal to 1.

2.2.1 Positive and Negative Imbalance

As shown in the song distribution over the years below, the random draw from the Million Song dataset led the dataset to be skewed to the left, meaning that the more than 80% of the songs was concentrated in recent years.

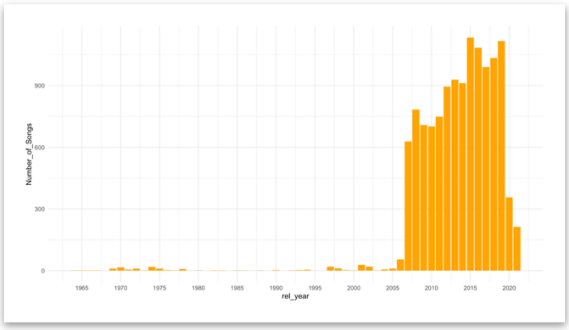

Figure 1 Distribution of songs over the considered time period

Additionally, hit songs were also present only after 2005. Therefore, keeping in mind the purpose of the study, the year 2007 was considered as the starting point.

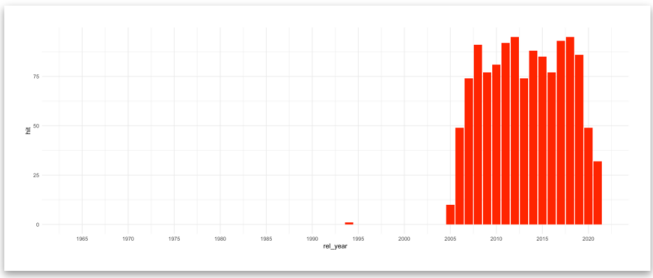

Figure 2 Distribution of hit songs over the considered time period

The figure below shows the distribution of the data after the adjustment.

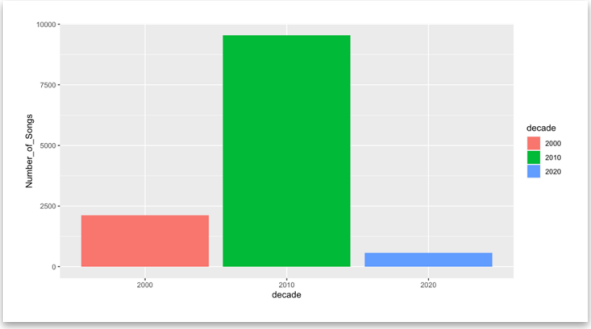

Figure 3 Distribution of songs after the adjustment

Furthermore, as it is highlighted by the hit distribution chart, dataset is representative of what the reality is: the hit songs are far less than the non-hit songs and thus, the balance of the data was not modified any further.

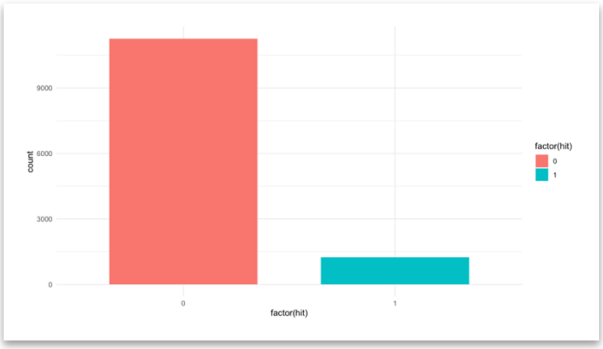

Figure 4 Dataset distribution of hit and non-hit

2.2.2 Unsupervised Learning

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis. Specifically, cluster analysis is the grouping of objects such that objects in the same cluster are more similar to each other than they are to objects in another cluster.

2.2.2.1 K-Means Clustering

In this method, data points are assigned to clusters in such a way that the sum of the squared distances between the data points and the centroid is as small as possible. Due to limitations in computational power, only an analysis of hit songs is performed. It is worth highlighting that each cluster will comprise songs from several and different music genres due to the cluster is not based on the genre attributes, but it is based on the song audio attributes, thus we may find some songs that share similar attributes -inter-genres- despite they were tagged into different genres. In order to know the optimal number of clusters, three different approaches were implemented: elbow method, average silhouette and gap statistics. While the gap statistics suggested 1 different cluster, the other two method suggested between 4 and 6 clusters. In the graph below, different Ks are compared.

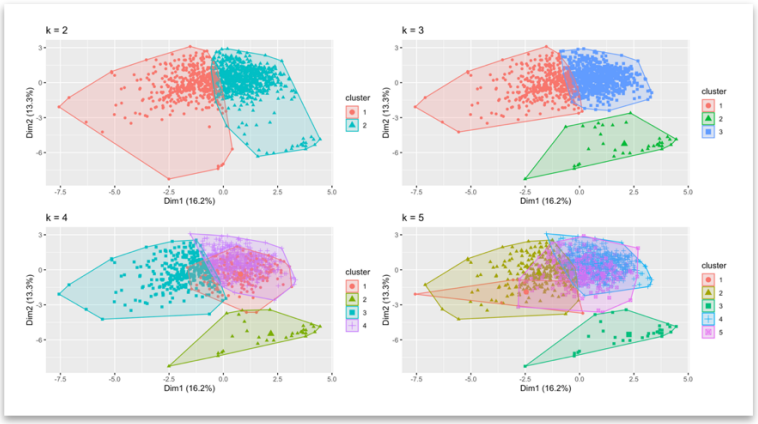

Figure 5 Cluster Comparison

It is worth noting that while the data is not clearly separable in k = 4 and k = 5, the graphs show that distinct groups can be formed with k = 2 and k = 3. It was decided to further analyze the 3-mean cluster.

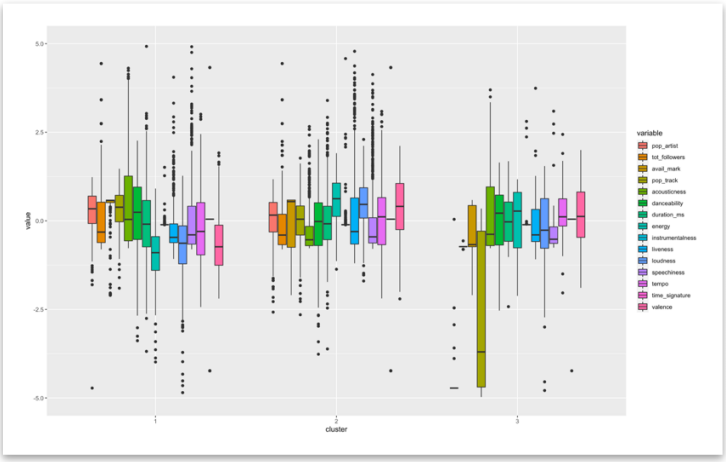

Figure 6 Feature distribution over clusters

The display a boxplot grouping the audio features by cluster label allows to understand what the result of clustering criteria was:

- *Cluster 1:* this cluster comprises songs available in few countries with low energy, loudness and valence. Acousticness and danceability are fairly high. In this category, we may find classic, dramatic rock and alternative indie songs. Generally, songs in this category have sad words as well as slow rhythm.
- *Cluster 2:* this cluster is quite different from the previous one. It has relatively low acousticness, and by contrast, fairly high energy. Loudness and valence are more than average as well. In this category, likely we may find songs from the '70, '80, '90 which are full of energy and happiness, they have high rhythm but also lyrics in contrast for example to house music.

- *Cluster 3:* this last cluster is characterized by unpopular songs by anonymous artists. Most of the features are close to the average, however some such as loudness and speechiness are below it. Songs created from remixes for example can be found here.

2.2.2.2 Agglomerative Hierarchical Clustering

Also known as bottom-up approach or hierarchical agglomerative clustering (HAC). This clustering algorithm does not require us to prespecify the number of clusters. Bottom-up algorithms treat each data as a singleton cluster at the outset and then successively agglomerates pairs of clusters until all clusters have been merged into a single cluster that contains all data. After having assessed that the best distance metrics is *Ward's linkage*, the dendrogram is plotted and as it can be seen, we have 6 different clusters.

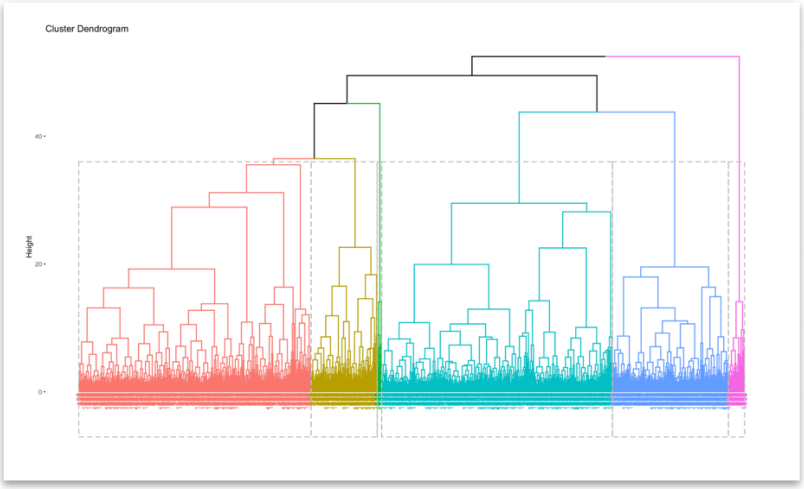

Figure 7 Dendrogram of agglomerative hierarchical cluster

From this graph, it can be analyzed how the clusters were composed and what characterizes each one of them. Additionally, the features can be compared across the clusters.

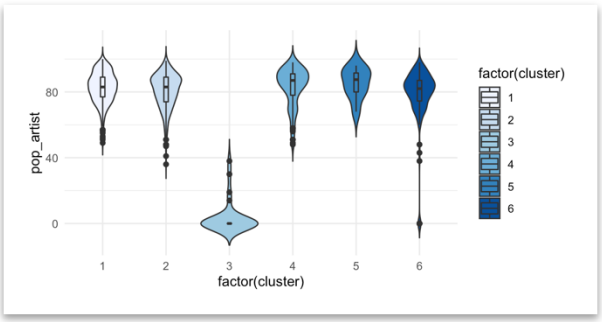

Figure 8 Box plot of artist popularity over clusters

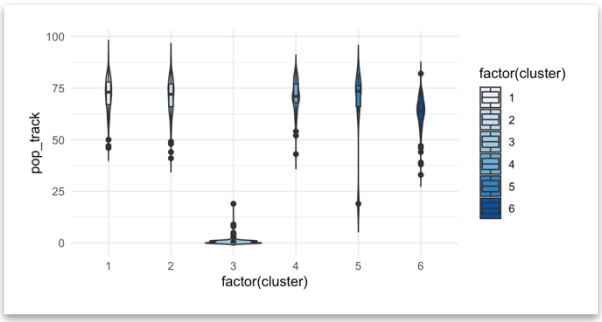

Figure 9 Box plot of track popularity over clusters

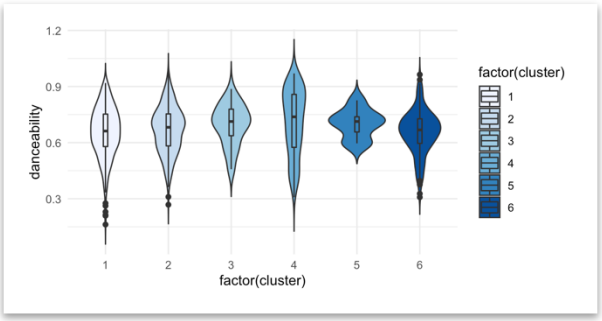

Figure 10 Box plot of danceability over clusters

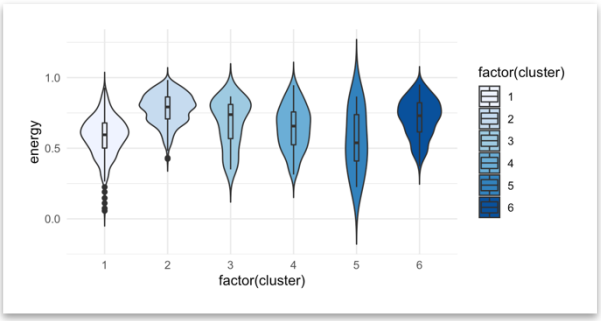

Figure 11 Box plot of energy over clusters

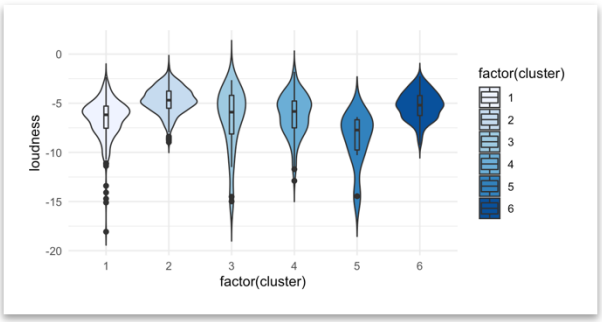

Figure 12 Box plot of loudness over clusters

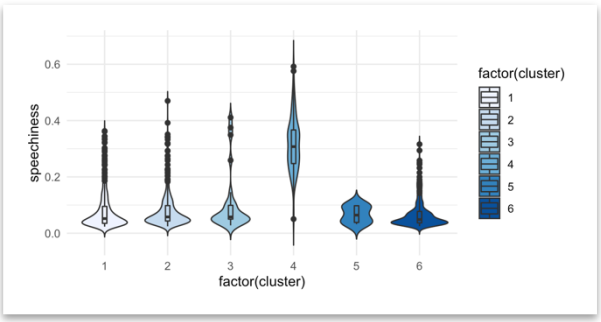

Figure 13 Box plot of speechiness over clusters

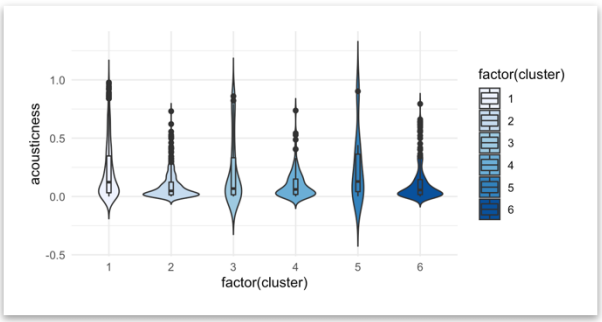

Figure 14 Box plot of acousticness over clusters

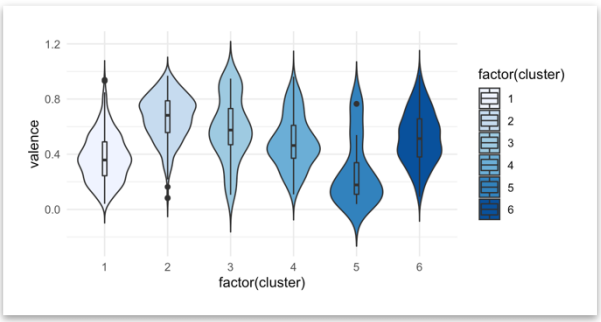

Figure 15 Box plot of valence over clusters

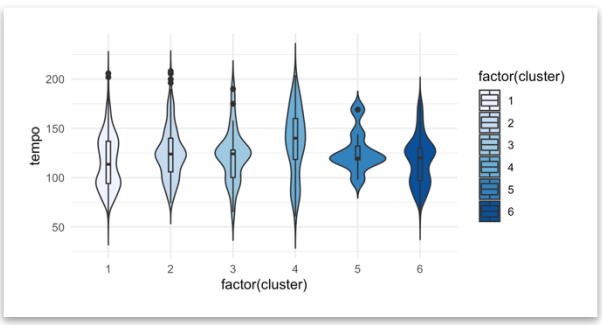

Figure 16 Box plot of tempo over clusters

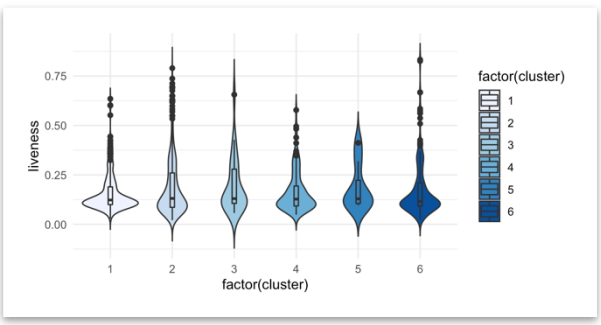

Figure 17 Box plot of liveness over clusters

For example, cluster 3 has the most unpopular songs by unpopular artists. On the other hand, cluster 5 has the broadest range of energy coupled with the smallest of danceability; similar reasoning can be applied to acousticness and tempo respectively.

2.2.2.3 Divisive Hierarchical Clustering

The divisive clustering algorithm is a top-down clustering approach, initially, all the points in the dataset belong to one cluster and split is performed recursively as one moves down the hierarchy. The result of divisive clustering with k = 5 is plotted below.

Figure 18 Plot of divisive hierarchical clusters

Cluster 1 is the biggest cluster, and it contains 96% of all songs. After performing several attempts with different Ks, no significant results were obtained. In this case, divisive clustering did not perform good enough to obtain significant insights.

3. Feature Selection

This section explores the tools used to study underlying patterns in the data such as the *Correlation Matrix* and *Principal Components Analysis*.

3.1 Correlation Matrix

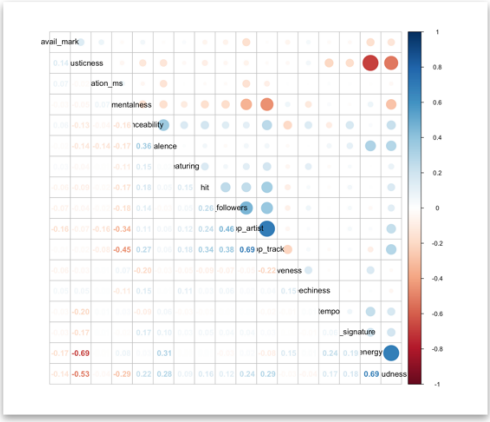

Figure 19 Correlation Matrix

As it shown in the plot above, pop_artist is highly correlated with pop_track. This naturally is because the more the artist is popular, the higher the odds that a song will be popular too. Energy is highly

correlated with loudness because of nature of the sound. The intensity, or quantity of energy, in sound waves determines the loudness of the sound. The decibel is a measurement of intensity (dB). Sound waves get more intense, and noises become louder as decibel levels rise. Loudness increases by ten times for every ten-decibel increase in sound intensity. Acousticness seems to be negatively strong correlated with energy and by extension with loudness. Instrumentalness seems to negatively influence the popularity of a track as well. Nevertheless, the variables mentioned above are too important to be removed for the analysis. Thus, a different approach was carried out.

3.2 Principal Components Analysis (PCA)

Principal Component Analysis (PCA) is performed to have an additional perspective. From 16 variables, the dimensions were reduced to 10 with an explanation of about 90% of the variance.

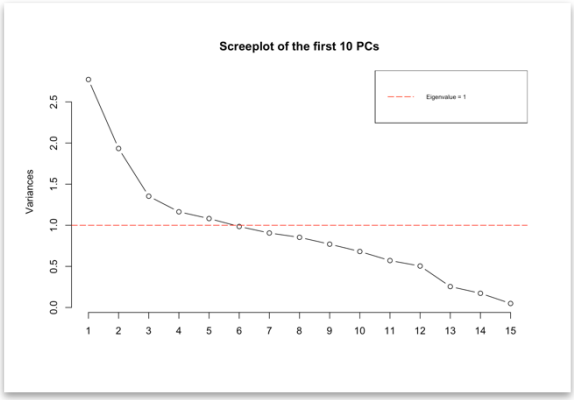

Figure 20 Plot of Elbow Method for PCA

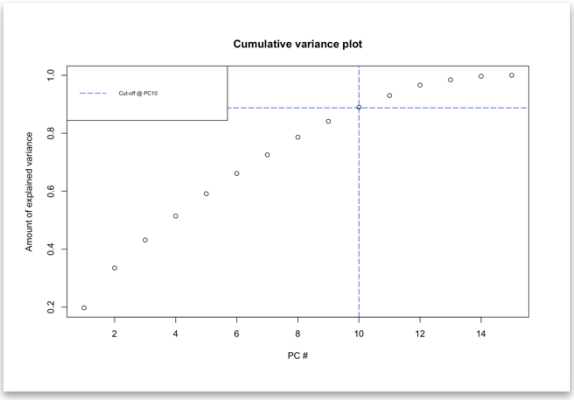

Figure 21 Plot of amount of explained variance

The graph below shows the different contribution made by each variable to the explanation of the variance. The biggest ones are the *pop_artist* and the *energy* with *acousticness* following right behind.

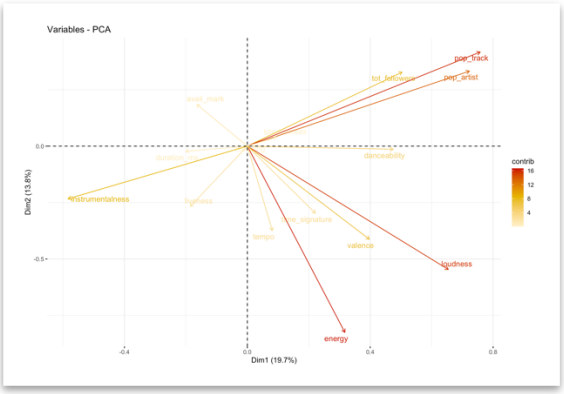

Figure 22 Graph of most important variables in PCA

4. Model Definition

This section gives an overview of the models used on the different training sets. First, the *Logistic Regression* is performed and then regularized through *Lasso.* Then *Discriminant Analysis* is run linear and then quadratic. Additionally, *Decision Tree* classifier is trained and then boosted through *Random Forest*. Lastly, *Support Vector Machine (SVM)* classifier is run and optimized through cross-validation.

Training set
16 features scaled 10 features reduced with PCA

4.1 Logistic Regression and Lasso Regression

4.1.1 Logistic Regression

Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In binary classification, a point with probability greater than 1 is categorized as hit, otherwise it is a non-hit. In logistic regression, a logit transformation is applied on the odds, that is the probability of success divided by the probability of failure. This is also commonly known as the log odds, or the natural logarithm of odds, and this logistic function is represented by the following formulas:

$$Logit(pi) = 1/(1+ exp(-pi))$$

$$ln(pi/(1-pi)) = Beta\_0 + Beta\_1*X\_1 + \ldots + B\_k*K\_k$$

In this logistic regression equation, logit(pi) is the dependent or response variable and x is the independent variable. All of these iterations produce the log likelihood function, and logistic regression seeks to maximize this function to find the best parameter estimate. Once the optimal coefficient (or coefficients if there is more than one independent variable) is found, the conditional probabilities for each observation can be calculated, logged, and summed together to yield a predicted probability.

4.1.2 LASSO Regression

Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters). This particular type of regression is well-suited for models showing

high levels of muticollinearity or when you want to automate certain parts of model selection, like variable selection/parameter elimination.

The acronym "LASSO" stands for Least Absolute Shrinkage and Selection Operator. Lasso regression performs L1 regularization, which adds a penalty equal to the absolute value of the magnitude of coefficients. This type of regularization can result in sparse models with few coefficients; Some coefficients can become zero and eliminated from the model. Larger penalties result in coefficient values closer to zero, which is the ideal for producing simpler models.

$$loss = sum\ i=0\ to\ n\ (y\_i - yhat\_i)\text{^}2$$
$$l1\_penalty = sum\ j=0\ to\ p\ abs(beta\_j)$$

4.2 Discriminant Analysis

4.2.1 Linear Discriminant Analysis

Linear discriminant analysis (LDA) is a type of linear combination, a mathematical process using various data items and applying functions to that set to separately analyze multiple classes of objects or items. It is used for modelling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space. It relies on the separability which is the distance between the different classes. This is called between-class variance. It computes then distance between the mean and sample of each class. It is also called the within-class variance. construct the lower-dimensional space which maximizes the between-class variance and minimizes the within-class variance. P is considered as the lower-dimensional space projection, also called Fisher's criterion.

$$Mean = Sum(x)/Nk$$
$$\Sigma^2 = Sum((x - M)^2)/(N - k)$$
$$\delta k(x) = xT\Sigma - 1\mu k - 12\mu Tk\Sigma - 1\mu k + log\pi k$$

4.2.2 Quadratic Discriminant Analysis

Quadratic discriminant analysis is quite similar to Linear discriminant analysis except it relaxes the assumption that the mean and covariance of all the classes were equal. QDA is a variant of LDA in which an individual covariance matrix is estimated for every class of observations. QDA is particularly useful if there is prior knowledge that individual classes exhibit distinct covariances.

$$\delta k(x) = -12log|\Sigma k| - 12(x-\mu k)T\Sigma - 1k(x-\mu k) + log\pi k.\delta k(x) = -12log|\Sigma k| - 12(x-\mu k)T\Sigma k - 1(x-\mu k) + log\pi k.$$

4.3 Decision Tree and Random Forest

4.3.1 Decision Tree

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed based on features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. In this study, Binary Classification Trees were considered.

4.3.1 Random Forest

Random forest is a commonly-used machine learning which combines the output of multiple decision trees to reach a single result. Random Forest is a so-called ensemble method. Ensemble learning methods are made up of a set of classifiers (e.g. decision trees) and their predictions are aggregated to identify the most popular result. It is an extension of the bagging method as it utilizes both bagging and feature randomness to create an uncorrelated forest of decision trees. Feature randomness, also known as feature bagging or "the random subspace method, generates a random subset of features, which ensures low correlation among decision trees. This is a key difference between decision trees and random forests. While decision trees consider all the possible feature splits, random forests only select a subset of those features. By accounting for all the potential variability in the data, we can reduce the risk of overfitting, bias, and overall variance, resulting in more precise predictions.

The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of a data sample drawn from a training set with replacement, called the bootstrap sample. Of that training sample, one-third of it is set aside as test data, known as the out-of-bag (oob) sample Another instance of randomness is then injected through feature bagging, adding more diversity to the dataset and reducing the correlation among decision trees. a classification task, a majority vote (i.e. the most frequent categorical variable) will yield the predicted class. Finally, the oob sample is then used for cross-validation, finalizing that prediction.

4.4 Support Vector Machine (SVM)

The goal of the Support Vector Machine algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence the name. SVM can be of two types: linear or non-linear. In this case, both models were implemented, in other words it is assumed that the dataset cannot be classified by using a straight line. The dimensions of the hyperplane depend on the features present in the dataset. The Gaussian Radial Basis Function (RBF) is used as kernel.

$$exp(-\gamma \ ||x - x'||^2)$$

4.5 Training Set Results

Models	Train Accuracy (Original)	Train Accuracy (PCA)
Logistic Regression	0.9099728	0.9041962
Discriminant Analysis (Linear)	0.8973297	0.8947139
Discriminant Analysis (Quadratic)	0.6843597	0.7025613
Lasso Regression	0.9100817	0.9044142
Random Forest	0.9998910	0.9998910
Support Vector Machine (Linear)	0.9027793	0.9027793
Support Vector Machine (Non-Linear)	0.9101907	0.9027793

As reported in the table above, the best performer was by far the Random Forest. This is because it is the most efficient and precise of the ones tested in this project. It should be noticed that all the other models, except for the *QDA,* performed very well with similar results. Additionally, *Lasso* regularization slightly increased the performances of the *Logistic Regression.* PCA obtained more or less the same results but with less complexity using only 10 variables however at the expense of interpretability.

5. Conclusions

This last section presents the results obtained on the various test sets by the best performing models. Then are outlined the main limitations of this paper as well as the recommendations for future works.

5.1 Test Results

The test set was created randomly using *sample.split* function from R package.

The Random Forest once again performed better for the classification. The overall results are reported below:

Models	Test Accuracy (Original)	Test Accuracy (PCA)
Logistic Regression	0.9061171	0.9031730
Discriminant Analysis (Linear)	0.8953222	0.8920510
Discriminant Analysis (Quadratic)	0.6830226	0.7091920
Lasso Regression	0.9067713	0.9038273
Random Forest	0.9142951	0.9025188
Support Vector Machine (Linear)	0.9028459	0.9028459
Support Vector Machine (Non-Linear)	0.9057900	0.9028459

All the models predicted the test set with an incredibly high consistency, showing almost no overfitting over the training dataset. Considering the training set, the same conclusions can be drawn here.

The focus was mainly on the accuracy of results, but the precision and recall for the best models are reported as well since false positive predictions may be costly when a music label invests in a song that is actually unlikely to become a hit.

Models	Hit Test Precision	Hit Test Recall
Logistic Regression	0.5925926	0.1077441
Lasso regression	0.6153846	0.1077441
Random Forest	0.6842105	0.2188552

From all the models it is possible to gather the most relevant features in order to predict whether song might be a hit or not:

- *featuring:* the presence of a famous artist can give popularity to the song

- *tot_followers:* if an artist is already famous, the probability that the song will be a hit arises
- *avail_mark:* if a song is available in more country, the probability arises
- *pop_track:* if a song is popular on Spotify, the probability arises
- *danceability:* the more a song is inclined to be danced, the more a song is likely to be a hit
- *instrumentalness:* the more a song has few vocal contents, the less likely a song will be a hit
- *liveness:* it is the least significant variable. It is a proxy to know if a song was recorded not in a studio/professionally. Thus, if it high then the song is less likely to be a hit
- *loudness:* it has one of the strongest influences for prediction. The more a song is loud, the more is likely to be a hit
- *time_signature:* it has medium relevance. It indicates what "type" of music it is. For example, a 4/4 is very good for dancing.

Taking the Logistic Regression as example, the ROC Curve and the Confusion Matrix are plotted below. The An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

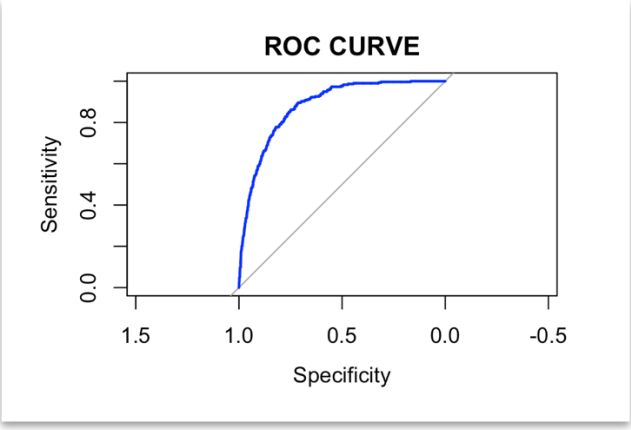

Figure 23 ROC Curve plot for Logistic Regression

A confusion matrix in R is a table that will categorize the predictions against the actual values. It includes two dimensions; among them one will indicate the predicted values and another one will represent the actual values.

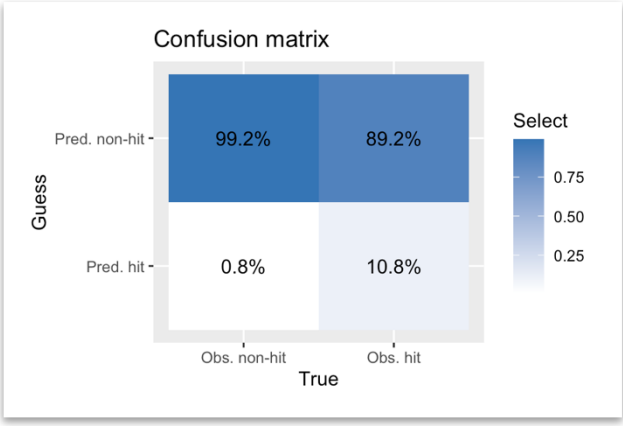

Figure 24 Confusion Matrix for Logistic regression

In conclusion, regarding the Logistic Regression, the model created is a model very good in predicting non-hit songs but not as much in predicting hits. This could be the consequence of the predominance of a category over the other.

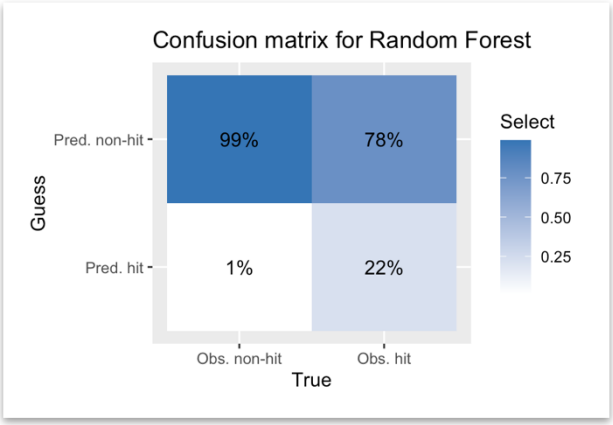

Figure 25 Confusion Matrix for Random Forest

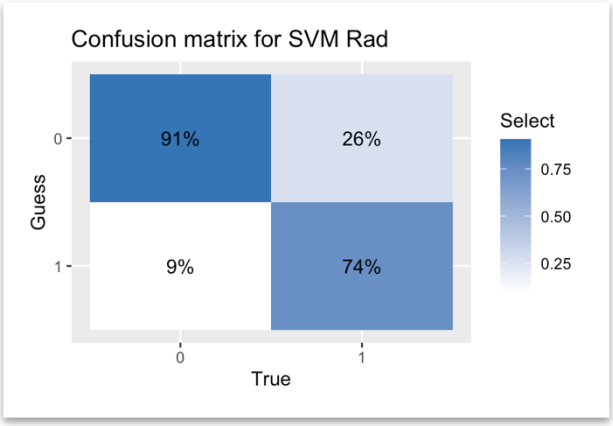

Figure 26 Confusion Matrix for SVM

As it can be seen from the confusion matrix above, a label company should use *Radial Support Vector Machine* because it has the predictive power for hit songs (74% correct rate).

5.2 Limitations and Recommendations for future work

This study aimed to predict whether a song will be a hit or not using features from Spotify API as a base. The first limitations come in play here. The definition of what a hit is inherently has constraints. This affects the composition of the dataset as well. A broader definition of hit could impact positively the research. In addition, the features are limited to the ones that Spotify API makes available. Other features for example of sound can be inserted such as bit depth and amplitude. Feature extraction and spectrogram analysis from the song audio files could be another successful approach. Subjective preferences, seasonality and time period might be taken in account as well. As instance, certain songs or genre such as Latin music are more likely to become hits in summer than in winter. Furthermore, particular characteristics in songs that made songs famous in the past are not a good guarantee for the future. While rock music dominated in the decades before 2000, nowadays electronic music took hold. Lyrics of a song could be considered to see if there are recurrencies, patterns or words that boost listening. In this paper, models from previous research are tested but it can be interesting to see how different algorithms would perform.

6. References

[1] Spotify API. *https://developer.spotify.com/documentation/web-api/*

[2] Spotipy!. *https://spotipy.readthedocs.io/en/master/*

[3] Elena Georgieva, Marcella Suta, and Nicholas Burton. Hitpredict: Predicting hit songs using spotify data. 2018

[4] Spotify for Developers. Song popularity, 2019

[5] Dorien Herremans, David Martens, and Kenneth Sörensen. Dance hit song prediction. Journal of New Music Research, 43(3):291–302, 2014

[6] Billboard. (2018). Billboard Hot 100 Chart. Retrieved from: *https://www.billboard.com/charts/hot-100*

[7] Guo, A. Python API for Billboard Data. Github.com. Retrieved from: *https://pypi.org/project/billboard.py/*

[8] Mauch, M., MacCallum, R. M., Levy, M., and Leroi, A. M. (2015). The Evolution of Popular Music: USA 19602010. R. Soc. open sci

[9] Sander Dieleman and Benjamin Schrauwen, Multiscale approaches to music audio feature learning, in Proc. Int. Soc. Music Information Retrieval Conf., 2013, pp. 116121

[10] The Million Song Dataset. *http://millionsongdataset.com/*

[11] Gareth James Daniela Witten Trevor Hastie Robert Tibshirani , An Introduction to Statistical Learning, 2017