EXAMENSARBETE INOM TEKNIK,
GRUNDNIVÅ, 15 HP
STOCKHOLM, SVERIGE 2018

Predicting Hit Songs with Machine Learning

MINNA REIMAN

PHILIPPA ÖRNELL

KTH
SKOLAN FÖR ELEKTROTEKNIK OCH DATAVETENSKAP

Predicting Hit Songs with Machine Learning

MINNA REIMAN, PHILIPPA ÖRNELL

Bachelor in Computer Science
Date: May 30, 2018
Supervisor: Ric Glassey
Examiner: Örjan Ekeberg
Swedish title: Förutsäga hitlåtar med maskininlärning
School of Electrical Engineering and Computer Science

Abstract

Exploring the possibility of predicting hit songs is both interesting from a scientific point of view and something that could be beneficial to the music industry. In this research we raise the question if it is possible to classify a music track as a hit or a non-hit based on its audio features. We investigated which machine learning algorithms could be suited for a task like this. Four different models were built using various algorithms such as Support Vector Machine and Gaussian Naive Bayes. The obtained results do not indicate that it is possible to predict hit songs on our particular dataset. This stands in contrast to some previous research within this field. We discuss the potential problem in using only audio features, and how this seems not to be sufficient information for predicting a hit.

Sammanfattning

Att förutsäga om en låt når topplistan skulle vara gynnsamt för musikindustrin samtidigt som det är intressant ur ett vetenskapligt perspektiv. I vår undersökning lyfter vi frågan om det är möjligt att klassificera en låt som hit eller icke-hit baserat på låtens ljudegenskaper. Vi utreder vilka algoritmer för maskininlärning som kan vara lämpliga för denna uppgift. Fyra olika modeller byggs utifrån algoritmer såsom Support Vector Machine och Gaussian Naive Bayes. De uppmätta resultaten indikerar att det inte är möjligt att förutsäga en hit på den utvalda datamängden. Dessa resultat motsäger en del av den tidigare forskningen gjord inom detta ämne. Vi diskuterar det potentiella problemet i att bara att analysera ljudegenskaper och hur detta inte tycks vara tillräcklig information för att identifiera en hit.

Contents

Chapter 1

Introduction

The increasing amount of digital music available online today and the progression of technology have changed the way we listen to and consume music. We expect to be able to search for specific music collections and even get automatic playlist recommendations. These ideas are covered by a research field called Music Information Retrieval (Kaminskas & Ricci 2012).

There is a diversity of research within this field, but there is not much research on the task of predicting hit songs and detection of its characteristics. Gaining insight into what makes a song popular would benefit the music industry. This subject is usually referred to as Hit Song Science which in 2012 was described by Pachet as *"an emerging field of investigation that aims at predicting the success of songs before they are released on the market"* (Pachet 2012).

The theory behind this science is that hit songs are related in the sense that they have a set of features that make them appealing to a majority of people. These features can be processed using machine learning where the aim is to find patterns in the data. If a pattern can be found, this could be used to predict if a song will become a hit (Ni, Santos-Rodriguez, Mcvicar & De Bie 2011).

1.1 Purpose

Some studies have tried to address the question whether or not it is possible to use machine learning techniques to predict hit songs. The purpose of this research is to further investigate this subject.

To be able to analyse music tracks, a dataset is compiled based on data from Spotify's API. The dataset consists of a number of audio features for each song. Afterwards, a number of machine learning algorithms is applied to this dataset. The accuracies of the different models are then compared. Thus the goal of this thesis is to examine if machine learning algorithms can be used to predict hit songs and with what accuracy this can be done.

1.2 Problem statement

The aim of this thesis is to investigate the question *is it possible to predict Hit Songs with Machine Learning using Audio Features*?

1.3 Scope and limitations

The scope of this research is to investigate if it is possible to predict a hit song based on 13 audio features of a track. Four different machine learning techniques are used: Logistic Regression, K-Nearest Neighbours, Gaussian Naive Bayes and Support Vector Machine. The focus of our research is to investigate if these models can be applied on our dataset and to compare their performances.

We are aware of the fact that there are different aspects that might affect if a track will become popular, e.g. social or economic. However, in this research we decided to focus only on the audio features of a music track. Because of time constraints we did not have the time to do any optimisation of the models. There was also a limitation in what metadata of the tracks we could obtain from the Spotify API. Unfortunately, the number of streams was not available from the API, which possibly could have provided value to the experiment. Another important parameter that was not accessible from the API was the popularity of an artist at a specific time. Ultimately, we used Scikit-learn for the machine learning which restricted what algorithms we could use.

1.4 Related work

Machine learning can be used for predictions of different types. Previous attempts have been made to address the question if it is possible to predict if a song will be popular. In this section, a number of related studies will be presented.

It can be worth mentioning what The Echo Nest is, since many of the related works used this for obtaining data about audio features. The Echo Nest is a leading music intelligence company that has over a trillion data points on over 38 million songs in its database (EchoNest 2018). In 2014, Spotify announced that they had acquired The Echo Nest (Mashable 2014). Some of the audio features derived by The Echo Nest for audio tracks are still available through the Spotify API but we noticed during this research that a great number of information that was earlier accessible unfortunately no longer is available for us to use.

In 2008, Pachet and Roy wanted to validate the hypothesis that popularity of songs can be predicted from acoustic or human features. Their research was based on a dataset of 32000 titles and 632 features, which can be seen as a massive number of features to consider. They were not able to develop a good classification model and claimed that the popularity of a song cannot be learnt by using state-of-the-art machine learning (Pachet & Roy 2008).

However, in 2011, a study made by Ni et al. provided a more optimistic result on the problem of predicting music popularity. They had the goal of distinguishing the top 5 from the bottom 10 in a top 40 hit list. Their dataset was based on UK charts during a time period of 50 years. 5947 unique songs were collected from the Official Charts Company (OCC), and the audio features were extracted from The Echo Nest. Their results indicated that it is possible to identify hits (Ni, Santos-Rodriguez, Mcvicar & De Bie 2011).

In 2011, Borg and Hokkanen investigated if they could predict the popularity of a song based on its audio features and Youtube view counts.

The features for audio tracks were obtained from The Echo Nest. For this task, a number of Support Vector Machines were built, and the achieved results were very modest. The Support Vector Machines, regardless of feature choice and parameters, never achieved more than 53% accuracy. They draw the conclusion that audio features alone do not seem to be good predictors of what makes a song popular. They suggest that popularity is likely driven by social forces (Borg & Hokkanen 2011).

In 2013, Fan and Casey at Dartmouth College compared prediction of UK hit songs with Chinese hit songs. The song tracks were collected from OCC for UK hits and ZhongGuoGeQuPaiHangBang for Chinese hits. They obtained the audio features from The Echo Nest and used the two machine learning algorithms Linear Regression and Support Vector Machine. In this study, they defined a hit as a song appearing on the top 20 of the top 40 chart, and a non-hit as the last 20 on the same list. The result of the study revealed that Chinese hit song prediction was more accurate than the UK hit song prediction. By using Linear Regression, the error rate for predicting Chinese songs was 41% whilst prediction of UK hits generated a 52 % error rate. The characteristics of a Chinese hit appeared to be significantly different from those of UK hits (Fan & Casey 2013).

Another related research carried out by Herremans et al. in 2014, focuses on classification of dance hit songs. This research also extracted audio features from The Echo Nest. The dataset used for the models was based on the OCC listings. They took top 40 songs and used the peak chart position of each song to determine if the song was a dance hit or not. They split the top 40 list into hits and non-hits. Three different datasets were created in which they varied what was considered hits and non-hits.

They created a dataset of dance hit songs from 2009 to 2013. Some of the machine learning algorithms they used where: Decision tree, Naive Bayes, Logistic Regression and Support Vector Machine. This study showed that the popularity of dance hit songs can indeed be predicted from analysing audio features. They concluded that the overall best algorithm was Logistic Regression, with an accuracy of 0.65 and a precision of 83% (Herremans et al. 2014).

A more recent study from 2016 conducted by Pham et al., at Stanford University, evaluated different machine learning algorithms and their ability to predict popularity of music tracks. Their data contained both audio features and metadata. To classify the data, they used Support Vector Machines, Neural Networks, and Logistic Regression etc. The data was obtained from The Million Song Dataset which is based on data from The Echo Nest. In their research, they used a subset of 2717 tracks and a hit was defined as a song with a high popularity-value. The results of the research showed that all models performed with similar accuracy, with a values ranging from 0.70 to 0.85 (Pham, Kyauk & Park 2016).

Many of the previous studies suggests that hit song prediction can be made but there is also research that disagrees with this. Therefore, we want to continue investigating this matter.

1.5 Definitions

Hit song: In this research a hit song is defined as a song track that has appeared on the Billboard Hot 100 Songs at any given time.
Non-hit song: A non-hit song is a song that has not appeared on the Billboard Hot 100 songs.

Chapter 2

Background

2.1 Machine learning

The amount of digitally recorded data is constantly increasing and there is undeniably useful information that can be extracted from these data archives. The complexity and volume of the data makes it hard for humans to process the information and give meaning to it. Machine learning can be used to find interesting and useful patterns in data in an automated way. It has become a common tool to be used in tasks that involves extracting information from large datasets (Shalev-Shwartz & Ben-David 2014).

Machine learning tools wants to give programs the ability to learn and adapt (Shalev-Shwartz & Ben-David 2014). Tom Mitchell provided a formal definition that says: *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E"* (Mitchell et al. 1997). In other words, a program is said to learn if the performance of a certain task advances with training.

The ability to assess a new situation based on earlier experiences is something recognizable in both human learning and machine learning. Computers can process historical data with different techniques in order to create algorithms that has an ability to generalize (Brink, Richards & Fetherolf 2016).

Machine learning can be divided into two different approaches: **Supervised** and **unsupervised** learning. Supervised learning is learning from a training set of labelled examples. Each example has a label, which tells the system what the correct output value is. The labels are used to identify what category the example belongs to. The task is to find patterns and infer a function or rule that maps the input to output. Unsupervised learning is learning from a training set of unlabelled examples. Thus no correct solution is known and the algorithm needs to identify underlying structures (Sutton & Barto 1998).

2.2 Machine learning algorithms

In this work we are focusing on supervised learning, since we are working with a labelled dataset. We know which examples in our data that are hits and we want to see if we can find a pattern that separates these from non-hits. The task is to construct models to predict the label of a song, if it is a hit or a non-hit. For this reason, we picked common machine learning techniques which is used in supervised learning.

2.2.1 Logistic Regression

Logistic regression is a mathematical model which can be used to describe the relationship between one or more independent variables and one dependent binary variable (Kleinbaum, Klein & Pryor n.d.). This model is therefore used for problems where the outcome can be classified in one of two categories. When applying the estimated logistic model to new cases of a test dataset, it provides a prediction of success probability, which is a number between 0 and 1. The logistic regression provides a rule for classifying the test data with a cut-off on the predicted success probability (Ledolter 2013).

2.2.2 K-Nearest Neighbours

K-Nearest Neighbours classification implements learning based on the K nearest neighbours of each query point. This method is a type of non-generalizing learning since it only stores instances of the training data, it does not create an internal model for generalizing the data. The classification is evaluated from a majority vote of the nearest neighbours of each query point, and each point is assigned to the class which is the most common among its neighbours. The choice of the value of k is dependent on the dataset: if k is set to a low value, the point is assigned to a class with only a few neighbours, and if k is a high value, then the effect of noise is suppressed and the classification is of a more general form (ScikitLearn 2017).

2.2.3 Gaussian Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms (Learn 2017). Bayes' theorem are used in these methods with the naive assumption that each feature is independent of every other feature. This is called conditional independence (Zhang 2004). The Naive Bayes methods must be applied to discrete variables but can be extended to continuous variables, most commonly by assuming a Gaussian distribution. This extension is called Gaussian Naive Bayes (Moraes & Machado 2009).

2.2.4 Support Vector Machine

A Support Vector Machine is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In a two dimensional space this hyperplane is a line dividing a plane in two parts with one class on each side of the line. In a 3 dimensional space, this separator is instead a plane separating the different classes. The Support Vector Machine attempts to find a separating hyperplane with a margin that is as large as possible from the nearest data points (OpenCV 2017).

2.3 Spotify API and the dataset

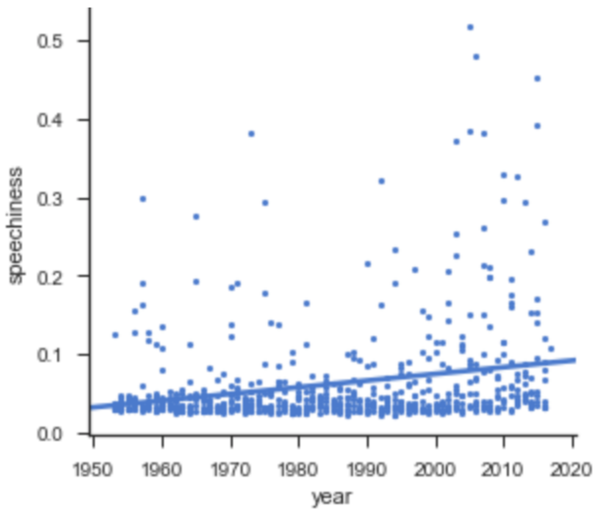
The dataset of audio features used in this research is fetched from Spotify's Web API. Below is a table of the numeric audio features and their official descriptions defined by Spotify:

Feature	**Type**	**Description**
Danceability	float	Describes how suitable a track is for dancing. This value is based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is the least danceable and a value of 1.0 is the most danceable.
Energy	float	A value representing a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud and noisy. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy. Energy is described as a value between 0.0 and 1.0.
Key	int	The key the track is in. Integers map to pitches using standard Pitch Class notation. (`https://en.wikipedia.org/wiki/Pitch_class`)
Loudness	float	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). This value usually ranges between -60 and 0 dB.
Mode	int	Describes the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by the value 1 and minor is represented by the value 0.
Speechiness	float	Detects the presence of spoken words in a track. The more exclusively speech like the recording is, the closer the value is to 1.0. If the speechiness ranges between 0.66 and 1.0, the track is probably made entirely of spoken words (such as audio books, poetry etc.). Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered. Values below 0.33 most likely represent music and other non-speech-like tracks.
Acousticness	float	A confidence measure between 0.0 and 1.0 of how acoustic a track is.
Instrumentalness	float	Predicts whether a track contains no vocals. Sounds like "*Ooh*" and "*Aah*" are treated as instrumental in this context, while rap or spoken words are clearly "vocal". The attribute ranges between 0.0 and 1.0 and the closer to 1.0 the value is; the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks.
Liveness	float	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track was performed live.
Valence	float	Describes the musical positiveness conveyed by a track. The value ranges between 0.0 and 1.0. Tracks with high valence sound more positive (happy, cheerful etc.), while tracks with low valence sound more negative (sad, depressed etc.).
Tempo	float	The overall estimated tempo of a track in beats per minute (BPM). Tempo is the speed or pace of a given piece and derives directly from the average beat duration.
Duration	int	The duration of a track in milliseconds.
Time signature	int	An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).

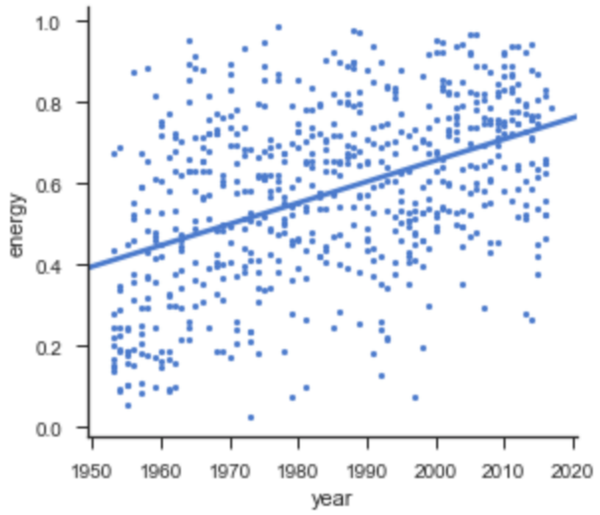
Table 2.1: Description of the audio features

2.4 Audio features change over time

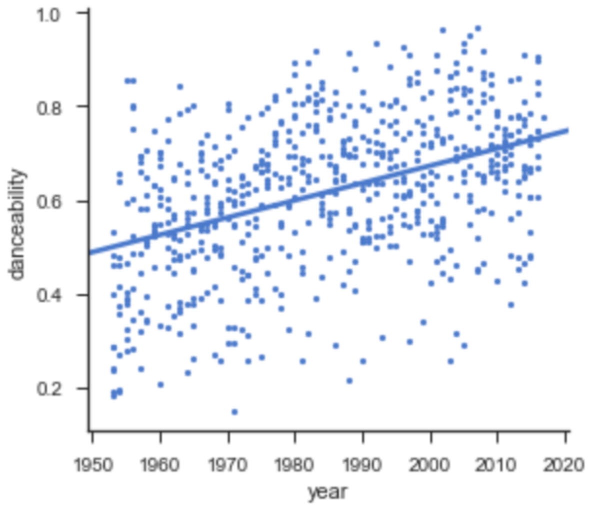
The audio features of hit songs differ over the decades. We think this is an appropriate thing to keep in mind when we are selecting what data to be used for model building. To visualize this, we created diagrams in which we plotted how the different features for song tracks has changed over time. The data is based on the Billboard Hot 100 tracks from 1951 until today, and the audio features are fetched from the Spotify API.



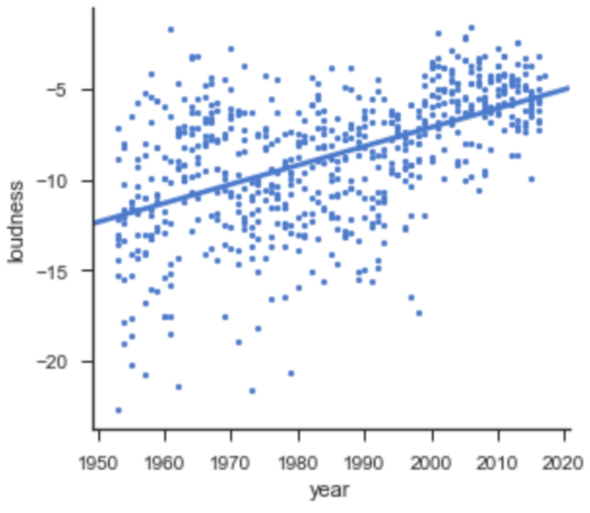
(a) Changes in Speechiness



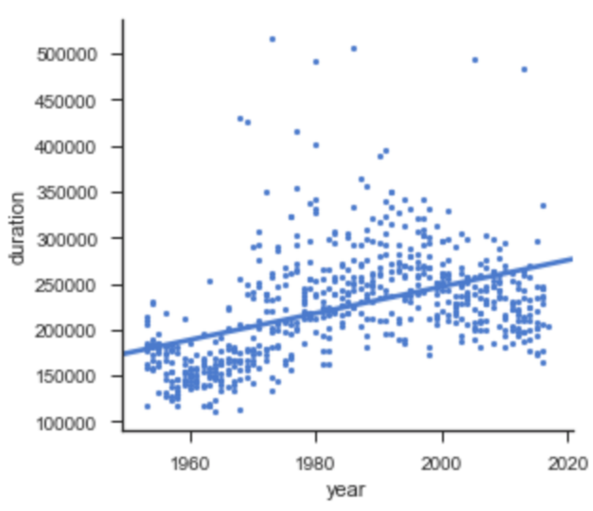
(b) Changes in Energy



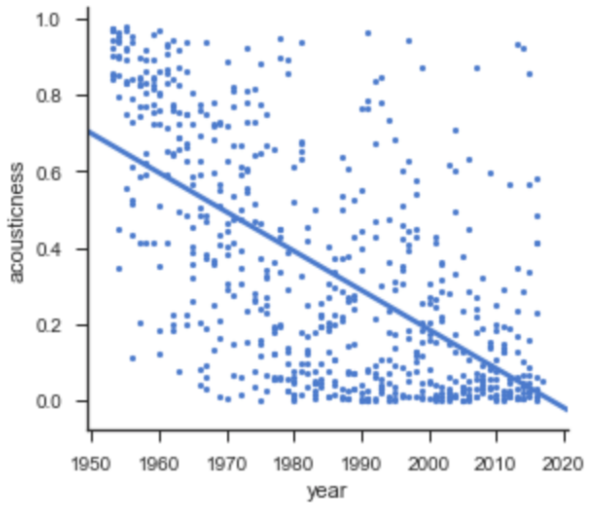
(a) Changes in Danceability



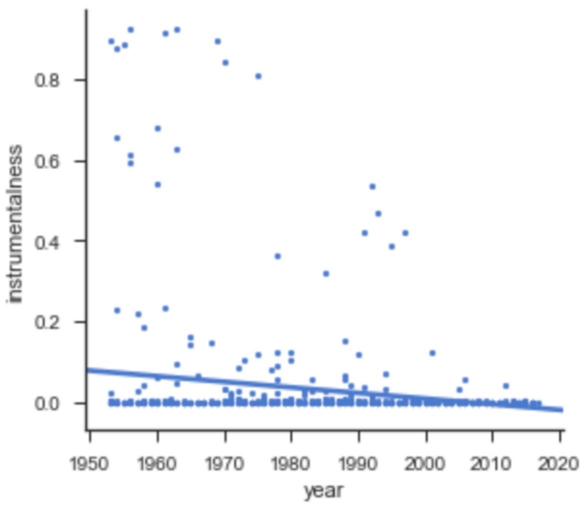
(b) Changes in Loudness



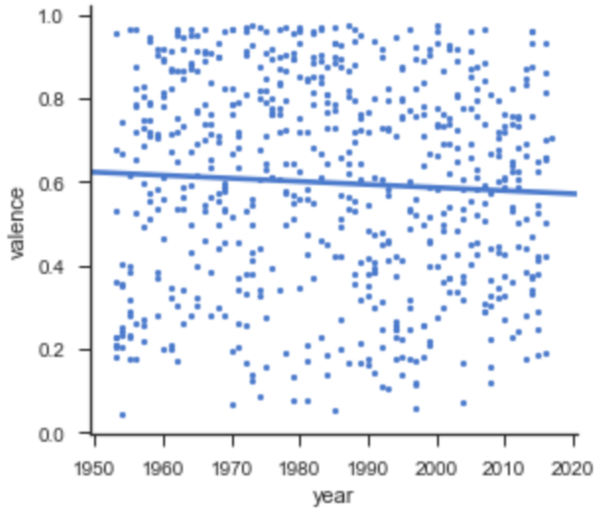
(c) Changes in Duration



(d) Changes in Acousticness



(e) Changes in Instrumentalness



(f) Changes in Valence

Chapter 3

Method

The machine learning workflow is separated in five parts: extract data, preprocess data, model building, evaluation and prediction (Brink, Richards & Fetherolf 2016). The workflow is described below:

3.1 Extract data

The dataset used in this research consists of both hit songs and non-hit songs. The hit songs were retrieved from the Billboard Hot 100, between the years 2016 and 2018. The hit dataset consists of 287 tracks (duplicates were removed). The non-hits songs were obtained by gathering 25 songs from 13 different genres, resulting in a set of 322 tracks, after removing the tracks that had appeared on the Billboard Hot 100 between 2016 to 2018. The total dataset resulted in a size of 609 tracks.

To obtain the audio features of the tracks we used Spotify's open Web API. To retrieve the data from the Spotify API a lightweight Python library was used, called Spotipy. The data used in the research contains 13 numeric audio features for each music track, described in section 2.3.

3.2 Preprocess data

In most cases, the preprocessing part of the workflow is cleaning the data, for instance remove null values and values that differ a lot from the rest. To avoid this issue, we decided not to use complete datasets, but instead compile our own datasets directly from Billboard and Spotify. In this way, we could make sure our datasets only contained the metadata that were relevant for our project. We also made sure to only select the audio features which values were in the same range, and did not contain any null values. The only thing we had to make sure of was that there were no duplicates in the dataset, since each datapoint should be unique for the classification.

The dataset, consisting of both hits and non-hits, was split up in two sets: a training set and a test set. The training set is the greatest subset, consisting of 80 per cent of the data. It is used to train the model so it can make predictions on the remaining 20 per cent of the data, i.e. the test data.

3.3 Model building

To build the machine learning models we used The Jupyter Notebook (`http://jupyter.org`). Jupyter Notebook is an open-source web application that can be used to implement statistical modelling, data visualisation, machine learning etc. We wrote the machine learning code in Python, which was executed in the iPython kernel (the computational engine) in the Jupyter Notebook. The library we used was Scikit learn.

Due to the fact that we were unsure how the data could be classified, we used four different algorithms. The algorithms were Logistic Regression, K-Nearest Neighbours, Gaussian Naive Bayes, and Support Vector Machine which are described in detail in section 2.5. The selected algorithms are a mix of linear and nonlinear models. To ensure that the evaluation of each of these algorithms was performed with the exact same data splits, we reset the number of seed before each run. By doing so, we could assure that the accuracy of each model were directly comparable.

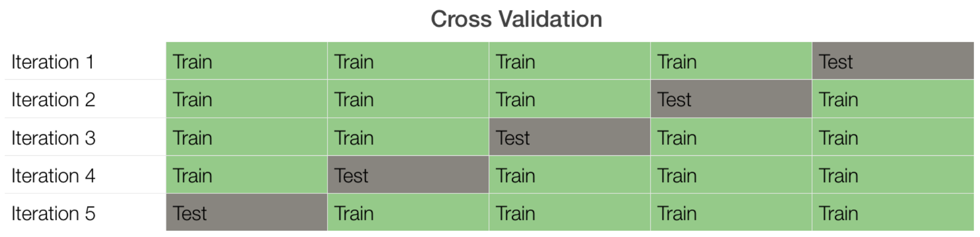
The models were built to classify the data into categories- in our research we wanted to classify if a song is a hit or not. To build the model, we used the training dataset described above. Based on the training data, the algorithm created a classification rule. This rule makes the model able to classify new examples and to predict whether or not a song is a hit.

3.4 Evaluation of the models

To validate that the models worked as expected, we used cross validation. Cross validation is used to estimate the performance of a predictive model. It is a process for creating a distribution of pairs of training and test sets out of a one single dataset (Springer 2017).

We used K-fold cross validation in which the training data is partitioned into K subsets of equal size. Each of these subsets are called folds. A learning algorithm is trained on K-1 of the subsets and tested on the remaining one. This is done K times, and each time a different partition is used as the test set, as seen in figure 3.1. An average of the performance can be used as an estimate of how well the model will perform in the future (Daumé III 2012).

In our research we used 10-fold cross validation to estimate the accuracy of our models. The accuracy metric is a ratio of the number of correctly predicted instances divided by the total number of correct instances in the dataset, and then multiplied by 100 to give a percentage. All four models are validated using this 10-fold cross validation.


Cross Validation

Iteration 1	Train	Train	Train	Train	Test
Iteration 2	Train	Train	Train	Test	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Test	Train	Train	Train
Iteration 5	Test	Train	Train	Train	Train

Figure 3.1: Cross validation visualized

3.5 Prediction

When the models are created, we could start making predictions on new examples/tracks. To be able to make predictions, we used the test dataset that consisted of data never previously seen by the model. The model building and prediction can be visualised with in the following way:

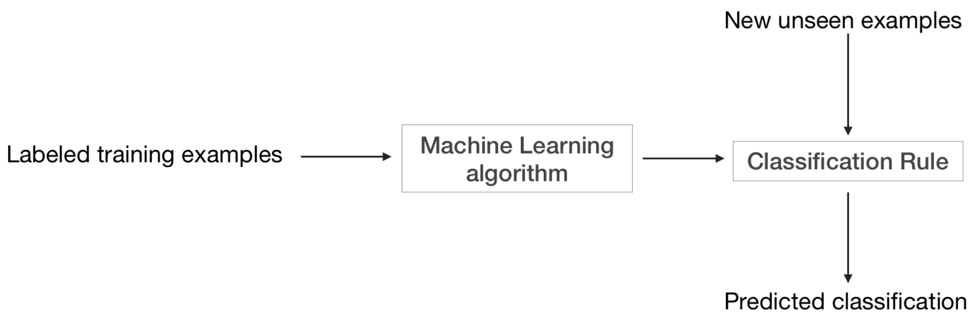


Figure 3.2: Supervised Learning

3.6 Evaluation of the predictions

To evaluate the accuracy of the models, we used confusion matrices. In the Encyclopedia of Machine Learning and Data Mining a confusion matrix is described as a matrix which summarises the performance of a classifier with respect to some test data (Ting 2017).

As seen in figure 3.3 our confusion matrix is represented by two classes, positive (hit) and negative (non-hit). The four different outcomes are *true negatives* (TN), *true positives* (TP), *false negatives* (FN), and *false positives* (FP) (Ting 2017). In our models, two errors may occur: classifying a hit falsely as a non-hit (FN) or classifying a non-hit falsely as a hit (FP).

	NEGATIVE	**POSITIVE**
NEGATIVE	True Negative	False Positive
POSITIVE	False Negative	True Positive

Figure 3.3: A confusion matrix

In classification the measures of *precision* and *recall* are widely used. Precision is what percentage of tuples that are labelled as positive are actually such. This can be seen as a measure of exactness. Recall is what percentage of positive tuples are labelled as such, which can be seen as a measure of completeness (Han, Pei & Kamber 2011). These measures can be computed with the formulas:

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{P}. \tag{3.2}$$

Chapter 4

Results

In the following chapter the result of our work will be presented. A total of 4 different models were built on our dataset using the classification techniques described in 2.1. The evaluation of the models was made using 10-fold cross validation. To evaluate the prediction made by the models we used confusion matrices.

4.1 Accuracy of the models

The accuracy of the models was as follows:

Model Comparison	
Model	Accuracy
Logistic Regression	54.38 %
K-Nearest Neighbours	51.96 %
Gaussian Naive Bayes	60.17 %
Support Vector Machine	52.16 %

Table 4.1: Comparison of the results

4.2 Prediction accuracy of the models

In this section we present the confusion matrices, and the recall and precision scores of each model.

4.2.1 Logistic Regression

Table 4.2 displays the confusion matrix of the Logistic Regression model.

	$false$	$true$
$false$	36	31
$true$	27	28

Table 4.2: Confusion Matrix of Logistic Regression

Table 4.3 displays the precision and recall score, as well as the number of tracks belonging to each class (support). The average precision of Logistic Regression is 53%.

	$precision$	$recall$	$support$
$non - hit$	0.57	0.54	67
hit	0.47	0.51	55
$avg/total$	0.53	0.52	122

Table 4.3: Precision and recall scores

4.2.2 K-Nearest Neighbours

Table 4.4 displays the confusion matrix of the K-Nearest Neighbours model.

	false	*true*
false	37	20
true	33	22

Table 4.4: Confusion Matrix of K-Nearest Neighbours

Table 4.5 displays the precision and recall score, as well as the number of tracks belonging to each class. The average precision of K-Nearest Neighbours is 48%.

	precision	*recall*	*support*
$non - hit$	0.53	0.55	67
hit	0.42	0.40	55
$avg/total$	0.48	0.48	122

Table 4.5: Precision and recall scores

4.2.3 Gaussian Naive Bayes

Table 4.6 displays the confusion matrix of the Gaussian Naive Bayes model.

	false	*true*
false	26	41
true	3	52

Table 4.6: Confusion Matrix of Gaussian Naive Bayes

Table 4.7 displays the precision and recall score, as well as the number of tracks belonging to each class. The average precision of Gaussian Naive Bayes is 74%.

	precision	*recall*	*support*
$non - hit$	0.90	0.39	67
hit	0.56	0.95	55
$avg/total$	0.74	0.64	122

Table 4.7: Precision and recall scores

4.2.4 Support Vector Machine

Table 4.8 displays the confusion matrix of the Support Vector Machine model.

	$false$	$true$
$false$	66	1
$true$	55	0

Table 4.8: Confusion Matrix of Support Vector Machine

Table 4.9 displays the precision and recall score, as well as the number of tracks belonging to each class. The average precision of Support Vector Machine is 30%. We can see that the Support Vector Machine classifies all tracks except one as non-hits.

	$precision$	$recall$	$support$
$non - hit$	0.55	0.99	67
hit	0.00	0.00	55
$avg/total$	0.30	0.54	122

Table 4.9: Precision and recall scores

Chapter 5

Discussion

In this chapter a discussion regarding the dataset, the results, and the method will be presented. In our research, we investigated four different machine learning models and their abilities to predict hit songs. Our results show that the machine learning models we selected were not successful on predicting hit songs on this specific dataset. The most accurate model was Gaussian Naive Bayes, with an accuracy of 60%. The other algorithms were no better than random chance. These results open up for discussion.

5.1 Discussion of the dataset

5.1.1 The tracks selected

An important aspect of our research is to consider what kind of data we have chosen to include in our dataset. We believe it is of importance to discuss what characterises a non-hit since previous research defined this differently than we did in our work. We found it a hard task of defining what a non-hit really is.

As stated in section 1.4, other researchers defined non-hits in different ways. Most of the studies only used top lists and made the distinction of hits from non-hits by cutting the list at a certain chart position. As previously mentioned, Fan and Casey separated a top 40 list into two parts, were the top 20 were considered hits and the bottom 20 were considered as non-hits. We found this approach questionable in many ways, partly because it is unlikely to consider the track on position 21 in the top list as non-hit. The song have still been played a great number of times and the track still made it to the top list, which must be seen as somewhat successful. Furthermore, since a song could be in the top 10 one month and in the consecutive month in the bottom 20, we did not think it was suitable to use just tracks from the hit list as the dataset.

To avoid this issue, we decided to compile our own dataset of non-hits. We wanted a great diversity of tracks in the dataset. Therefore, we included a number of arbitrary tracks from 13 different genres. In this way, we somewhat randomised the data we used as non-hits.

The size of the dataset might also affect the prediction accuracy of the models. To decide what size we wanted to use in our research we looked at how much data similar studies have used. Both Herreman and Fan used a dataset of similar size as us, while other studies had a much greater set. The distribution of the data is also being taken into account. We chose to keep the two datasets of hits and non-hits balanced, with an even number of entries in each class.

5.1.2 The features analysed

It is also of importance to discuss what audio features we included in our dataset. The kind of music making it to the top list is constantly changing. The audio features of top hits have changed over time as we can see in the plots in section 2.4. Some values, such as energy, loudness and danceability have increased over the decades, and features such as instrumentalness and acousticness have decreased.

This is not totally unanticipated as the style of popular music is constantly evolving. The timespan of which our machine learning models can predict hit songs is therefore limited. For this reason, it is not suitable to use data from previous decades when predicting contemporary hits. Therefore, we found it appropriate to train the models with exclusively recent hit songs. With this in consideration we chose to build our dataset of hit songs from 2016 to 2018.

It is also important to be aware that the features we selected can impact the models ability to predict. We did not do any further analysis after the models were built on which audio features were most relevant. This is definitely something that we would have done if we had more time for data analysis. We also wanted to include other features such as label and genre, but we did not find any good way to combine strings and numerical values in our machine learning algorithms.

5.2 Discussion of the results

The accuracy values were not as good as we hoped for, since some previous research produced more promising results. Herremans et. al found that with feature selection, the classification models built using Naive Bayes and Logistic Regression both had an accuracy of 65%, and the Support Vector Machine had 59% accuracy. Herremans evaluated the prediction performance on the method that performed best, which was Logistic Regression. The confusion matrix of logistic regression showed that 83% of the actual hits were accurately classified as hits and 32% of the non-hits classified as non-hits. This suggests that it is possible to predict hit songs.

In addition, the research by Pham et.al showed that all models performed with an even better accuracy, with values ranging from 70% to 85%.

Both these studies stand in contrast to the results that Pachet/Roy and Borg/Hokkanen achieved. Borg/Hokkanen used several Support Vector Machines and they did not get an accuracy higher than 53%.

Our results show that Gaussian Naive Bayes has an accuracy of 60% and the other classifiers Support Vector Machine, Logistic Regression and K-Nearest Neighbours have an accuracy slightly above 50%. Therefore, our work agrees with the research made by Borg and Hokkanen and with the work by Pachet and Roy, which suggest that prediction of hit songs is not possible only by analysing the audio features of the tracks.

The reason our models performed differently compared to the other studies can be explained in various ways. One major difference was the dataset we used. Since we chose to use a dataset containing several different genres, the dataset might have been to complex to generalise and classify. Another difference was that we did not do any features selection or model optimisation.

As seen in the related work section, the task of music prediction has been taken on with various outcomes. Some have drawn the conclusion that hit song prediction is possible. Although, we have to be careful when comparing these studies since they all have approached the question differently and made different assumptions. As discussed in the previous section, none of the research we investigated have defined what a non-hit is in the same way. This accounts for contrasting results with various success rates.

The different researches we looked at used a number of different machine learning models and despite our increasing but yet modest knowledge within this field we had difficulties to critically analyse their models and results.

Our results reveal that it is not possible to successfully predict whether or not a track is a hit song, using the dataset we compiled. Gaussian Naive Bayes had an accuracy of 60% but we do not see this as a result that suggests that it is possible to predict hit songs.

5.3 Possible improvements and future work

There are a number of improvements that would be interesting to consider for future research within this field. One improvement would be to use more features than we did in our research. We believe that including more metadata such as genre, label, lyrics, artist name, and artist popularity would improve the accuracy of the models considerably. Another improvement would be to do feature selection and model optimisation. Lastly, it would be favorable to define what a non-hit is in a more explicit way.

Chapter 6

Conclusion

The aim of this work was to further investigate whether or not it is possible to use machine learning techniques to predict hit songs. Our work could not confirm that it is possible, although other research has shown more promising results. The reason to our results are due to the dataset we selected and the models we built. The dataset we used for our non-hit data is very diverse thus possibly making it difficult for the models to generalise and classify the data. A future expansion of this work could be to use more metadata about the artist and the track, which we believe will yield higher accuracy and better results.

Bibliography

Borg, N. & Hokkanen, G. (2011), 'What makes for a hit pop song? what makes for a pop song?', *Unpublished thesis, Stanford University, California, USA* .

Brink, H., Richards, J. & Fetherolf, M. (2016), *Real-world machine learning*, Manning Publications Co.

Daumé III, H. (2012), 'A course in machine learning', *Publisher, ciml. info* pp. 5–73.

EchoNest, T. (2018), 'The echo nest', `http://the.echonest.com`.

Fan, J. & Casey, M. A. (2013), Study of chinese and uk hit songs prediction, *in* 'Proceedings of International Symposium on Computer Music Multidisciplinary Research', pp. 640–652.

Han, J., Pei, J. & Kamber, M. (2011), *Data mining: concepts and techniques*, Elsevier.

Herremans, D., Martens, D. & Sörensen, K. (2014), 'Dance hit song prediction', *Journal of New Music Research* **43**(3), 291–302.

Kaminskas, M. & Ricci, F. (2012), 'Contextual music information retrieval and recommendation: State of the art and challenges'.

Kleinbaum, D. G., Klein, M. & Pryor, E. (n.d.), 'Logistic regression: a self-learning text. 2002'.

Learn, S. (2017), 'Naive bayes', `http://scikit-learn.org/stable/modules/naive_bayes.html`.

Ledolter, J. (2013), *Data mining and business analytics with R*, John Wiley & Sons.

Mashable (2014), 'Spotify aquires music data company the echo nest', `https://mashable.com/2014/03/06/spotify-acquires-echo-nest/#228yuCje2sqZ`.

Mitchell, T. M. et al. (1997), 'Machine learning. wcb'.

Moraes, R. M. & Machado, L. S. (2009), 'Gaussian naive bayes for online training assessment in virtual reality-based simulators', *Mathware & Soft Computing* **16**(2), 123–132.

Ni, Y., Santos-Rodriguez, R., Mcvicar, M. & De Bie, T. (2011), 'Hit song science once again a science?'.

OpenCV (2017), 'Introduction to support vector machines', `https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html`.

Pachet, F. (2012), 'Hit song science', *Music data mining* pp. 305–26.

Pachet, F. & Roy, P. (2008), 'Hit Song Science Is Not Yet a Science.', *Ismir* pp. 355–360.

Pham, J., Kyauk, E. & Park, E. (2016), 'Predicting song popularity', *nd): n. pag. Web* **26**.

ScikitLearn (2017), 'Nearest neighbors', `http://scikit-learn.org/stable/modules/neighbors.html`.

Shalev-Shwartz, S. & Ben-David, S. (2014), *Understanding machine learning: From theory to algorithms*, Cambridge university press.

Springer (2017), 'Confusion matrix', `https://link-springer-com.focus.lib.kth.se/referenceworkentry/10.1007%2F978-1-4899-7687-1_50`.

Sutton, R. S. & Barto, A. G. (1998), *Reinforcement learning: An introduction*, Vol. 1, MIT press Cambridge.

Ting, K. M. (2017), *Confusion Matrix*, Springer US, Boston, MA, pp. 260–260.

Zhang, H. (2004), 'The optimality of naive bayes', *AA* **1**(2), 3.