Routledge
Taylor & Francis Group

Journal of New Music Research

ISSN: 0929-8215 (Print) 1744-5027 (Online) Journal homepage: http://www.tandfonline.com/loi/nnmr20

Dance Hit Song Prediction

Dorien Herremans, David Martens & Kenneth Sörensen

To cite this article: Dorien Herremans, David Martens & Kenneth Sörensen (2014) Dance Hit Song Prediction, Journal of New Music Research, 43:3, 291-302, DOI: 10.1080/09298215.2014.881888

To link to this article: http://dx.doi.org/10.1080/09298215.2014.881888

Journal of New Music Research, 2014
Vol. 43, No. 3, 291–302, http://dx.doi.org/10.1080/09298215.2014.881888

Dance Hit Song Prediction

Dorien Herremans[1], David Martens[2] and Kenneth Sörensen[1]

[1]*ANT/OR, University of Antwerp Operations Research Group, Prinsstraat, B-2000, Belgium;* [2]*Applied Data Mining Research Group, University of Antwerp, Prinsstraat, B-2000, Belgium*

(Received 3 July 2013; accepted 6 January 2014)

Abstract

Record companies invest billions of dollars in new talent around the globe each year. Gaining insight into what actually makes a *hit* song would provide tremendous benefits for the music industry. In this research we tackle this question by focussing on the *dance* hit song classification problem. A database of dance hit songs from 1985 until 2013 is built, including basic musical features, as well as more advanced features that capture a temporal aspect. A number of different classifiers are used to build and test dance hit prediction models. The resulting best model has a good performance when predicting whether a song is a 'top 10' dance hit versus a lower listed position.

Keywords: machine learning, databases, information retrieval, music analysis

1. Introduction

In 2011 record companies invested a total of 4.5 billion in new talent worldwide (IFPI, 2012). Gaining insight into what actually makes a song a hit would provide tremendous benefits for the music industry. This idea is the main drive behind the new research field referred to as 'Hit song science' which Pachet (2012) define as 'an emerging field of science that aims at predicting the success of songs before they are released on the market'.

There is a large amount of literature available on song writing techniques (Braheny, 2007; Webb, 1999). Some authors even claim to teach the reader how to write *hit* songs (Leikin, 2008; Perricone, 2000). Yet very little research has been done on the task of automatic prediction of hit songs or detection of their characteristics.

The increase in the amount of digital music available online combined with the evolution of technology has changed the way in which we listen to music. In order to react to new expectations of listeners who want searchable music collections, automatic playlist suggestions, music recognition systems etc., it is essential to be able to retrieve information from music (Casey et al., 2008). This has given rise to the field of Music Information Retrieval (MIR), a multidisciplinary domain concerned with retrieving and analysing multifaceted information from large music databases (Downie, 2003). Many MIR systems have been developed in recent years and applied to a range of different topics such as automatic classification per genre (Tzanetakis & Cook, 2002), cultural origin (Whitman & Smaragdis, 2002), mood (Laurier, Grivolla, & Herrera, 2008), composer (Herremans, Sörensen, & Martens, 2013), instrument (Essid, Richard, & David, 2006), similarity (Schnitzer, Flexer, & Widmer, 2009), etc. An extensive overview is given by Fu, Lu, Ting and Zhang (2011). Yet, as it appears, the use of MIR systems for hit prediction remains relatively unexplored.

The first exploration into the domain of hit science is due to Dhanaraj and Logan (2005). They used acoustic and lyric-based features to build support vector machines (SVM) and boosting classifiers to distinguish top 1 hits from other songs in various styles. Although acoustic and lyric data was only available for 91 songs, their results seem promising. The study does however not provide details about data gathering, features, applied methods and tuning procedures.

Based on the claim of the unpredictability of cultural markets made by Salganik, Dodds and Watts (2006), Pachet and Roy (2008) examined the validity of this claim on the music market. Based on a dataset they were not able to develop an accurate classification model for low, medium or high popularity based on acoustic and human features. They suggest that the acoustic features they used are not informative enough to be used for aesthetic judgements and suspect that the previously mentioned study (Dhanaraj & Logan, 2005) is based on spurious data or biased experiments.

Correspondence: Dorien Herremans, University of Antwerp, Engineering Management, Prinsstraat 13, Antwerp, 2000 Belgium.
E-mail: dorien.herremans@uantwerpen.be

Borg and Hokkanen (2011) draw similar conclusions as Pachet and Roy (2008). They tried to predict the popularity of music videos based on their YouTube view count by training support vector machines but were not successful.

Another experiment was set up by Ni, Santos-Rodríguez, McVicar and De Bie (2011), who claim to have proven that hit song science is once again a science. They were able to obtain more optimistic results by predicting if a song would reach a top 5 position on the UK top 40 singles chart compared to a top 30–40 position. The shifting perceptron model that they built was based on thus far novel audio features mostly extracted from The Echo Nest.[1] Though they describe the features they used on their website (Jehan & DesRoches, 2012), the paper is very short and does not disclose a lot of details about the research such as data gathering, preprocessing, detailed description of the technique used or its implementation.

In this research accurate models are built to predict if a song is a top 10 dance hit or not. For this purpose, a dataset of dance hits including some unique audio features is compiled. Based on this data different efficient models are built and compared. To the authors' knowledge, no previous research has been done on the dance hit prediction problem.

In the next section, the dataset used in this paper is elaborately discussed. In Section 3 the data is visualized in order to detect some temporal patterns. Finally, the experimental setup is described and a number of models are built and tested.

2. Dataset

The dataset used in this research was gathered in a few stages. The first stage involved determining which songs can be considered as hit songs versus which songs cannot. Secondly, detailed information about musical features was obtained for both aforementioned categories.

2.1 Hit listings

Two hit archives available online were used to create a database of dance hits (see Table 1). The first one is the singles dance archive from the Official Charts Company (OCC).[2] The Official Charts Company is operated by both the British Phonographic Industry and the Entertainment Retailers Association (ERA). Their charts are produced based on sales data from retailers through market researcher Millward Brown. The second source is the singles dance archive from Billboard (BB).[3] Billboard is one of the oldest magazines in the world devoted to music and the music industry.

The information was parsed from both websites using the Open source Java html parser library JSoup (Houston, 2013) and resulted in a dataset of 21,692 (7159 + 14,533) listings with four features: song title, artist, position and date. A very small number of hit listings could not be parsed and these were left out of the dataset. The peak chart position for each song was computed and added to the dataset as a fifth feature. Table 2 shows an example of the dataset at this point.

Table 1. Hit listings overview.

	OCC	BB
Top	40	10
Date range	10/2009–3/2013	1/1985–3/2013
Hit listings	7159	14,533
Unique songs	759	3361

2.2 Feature extraction and calculation

The Echo Nest[4] was used in order to obtain musical characteristics for the song titles obtained in the previous subsection. The Echo Nest is the world's leading music intelligence company and has over a trillion data points on over 34 million songs in its database. Its services are used by industry leaders such as Spotify, Nokia, Twitter, MTV, EMI and more (EchoNest, 2013). Bertin-Mahieux, Ellis, Whitman and Lamere (2011) used The Echo Nest to build The One Million Song dataset, a very large freely available dataset that offers a collection of audio features and meta-information for a million contemporary popular songs.

In this research The Echo Nest was used to build a new database mapped to the hit listings. The Open Source java client library jEN for the Echo Nest developer API was used to query the songs (Lamere, 2013). Based on the song title and artist name, The Echo Nest database and Analyzer were queried for each of the parsed hit songs. After some manual and java-based corrections for spelling irregularities (e.g. Featuring, Feat, Ft.) data was retrieved for 697 out of 759 unique songs from the OCC hit listings and 2755 out of 3361 unique songs from the BB hit listings. The songs with missing data were removed from the dataset. The extracted features can be divided into three categories: meta-information, basic features from The Echo Nest Analyzer and temporal features.

2.2.1 Meta-information

The first category is *meta-information* such as artist location, artist familiarity, artist hotness, song hotness etc. This is descriptive information about the song, often not related to the audio signal itself. One could follow the statement of IBM's Bob Mercer in 1985 'There is no data like more data' (Jelinek, 2005). Yet, for this research, the meta-information is discarded when building the classification models. In this way, the model can work with unknown songs, based purely on audio signals.

[1] echonest.com
[2] officialcharts.com
[3] billboard.com
[4] echonest.com

Table 2. Example of hit listings before adding musical features.

Song title	Artist	Position	Date	Peak position
Harlem Shake	Bauer	2	09/03/13	1
Are You Ready For Love	Elton John	40	08/12/12	34
The Game Has Changed	Daft Punk	32	18/12/10	32
…				

2.2.2 *Basic analyzer features*

The next category consists of *basic features* extracted by The Echo Nest Analyzer (Jehan & DesRoches, 2012). Most of these features are self-explanatory, except for *energy* and *danceability*, of which The Echo Nest did not yet release the formula.

Duration Length of the track in seconds.
Tempo The average tempo expressed in beats per minute (bpm).
Time signature A symbolic representation of how many beats there are in each bar.
Mode Describes if a song's modality is major (1) or minor (0).
Key The estimated key of the track, represented as an integer.
Loudness The loudness of a track in decibels (dB), which correlates to the psychological perception of strength (amplitude).
Danceability Calculated by The Echo Nest, based on beat strength, tempo stability, overall tempo, and more.
Energy Calculated by The Echo Nest, based on loudness and segment durations.

A more detailed description of these Echo Nest features is given by Jehan and DesRoches (2012).

2.2.3 *Temporal features*

A third category of features was added to incorporate the *temporal aspect* of the following basic features offered by the Analyzer:

Timbre A 12-dimensional vector which captures the tone colour for each segment of a song. A segment is a sound entity (typically under a second) relatively uniform in timbre and harmony.
Beatdiff The time difference between subsequent beats.

Timbre is a very perceptual feature that is sometimes referred to as tone colour. In The Echo Nest, 13 basis vectors are available that are derived from the principal components analysis (PCA) of the auditory spectrogram (Jehan, 2005). The first vector of the PCA is referred to as loudness, as it is related to the amplitude. The following 12 basis vectors are referred to as the timbre vectors. The first one can be interpreted as brightness, as it emphasizes the ratio of high frequencies versus low frequencies, a measure typically correlated to the 'perceptual' quality of brightness. The second timbre vector has to do with flatness and narrowness of sound (attenuation of lowest and highest frequencies). The next vector represents the emphasis of the attack (sharpness) (EchoNest, 2013). The timbre vectors after that are harder to label, but can be understood by the spectral diagrams given by Jehan (2005).

In order to capture the temporal aspect of timbre throughout a song Schindler and Rauber (2012) introduce a set of derived features. They show that genre classification can be significantly improved by incorporating the statistical moments of the 12 segment timbre descriptors offered by The Echo Nest. In this research the statistical moments were calculated together with some extra descriptive statistics: mean, variance, skewness, kurtosis, standard deviation, 80th percentile, min, max, range and median.

Ni, Santos-Rodríguez, McVicar and De Bie (2013) introduce a variable called Beat CV in their model, which refers to the variation of the time between the beats in a song. In this research, the temporal aspect of the time between beats (beatdiff) is taken into account in a more complete way, using all the descriptive statistics from the previous paragraph.

After discarding the meta-information, the resulting dataset contained 139 usable features. In the next section, these features were analysed to discover their evolution over time.

3. Evolution over time

The dominant music that people listen to in a certain culture changes over time. It is no surprise that a hit song from the 1960s will not necessarily fit in the contemporary charts. Even if we limit ourselves to one particular style of hit song, namely dance music, a strong evolution can be distinguished between popular 1990s dance songs and this week's hit. In order to verify this statement and gain insight into how characteristics of dance music have changed, the Billboard dataset (BB) with top 10 dance hits from 1985 until now was analysed.

A dynamic chart was used to represent the evolution of four features over time (Google, 2013). Figure 1 shows a screenshot of the Google motion chart[5] that was used to visualize the time series data. This graph integrates data mining and information visualization in one discovery tool as it reveals interesting patterns and allows the user to control the visual presentation, thus following the recommendation made by Shneiderman (2002). The x-axis shows the duration and the

[5]Interactive motion chart available at http://antor.ua.ac.be/dance

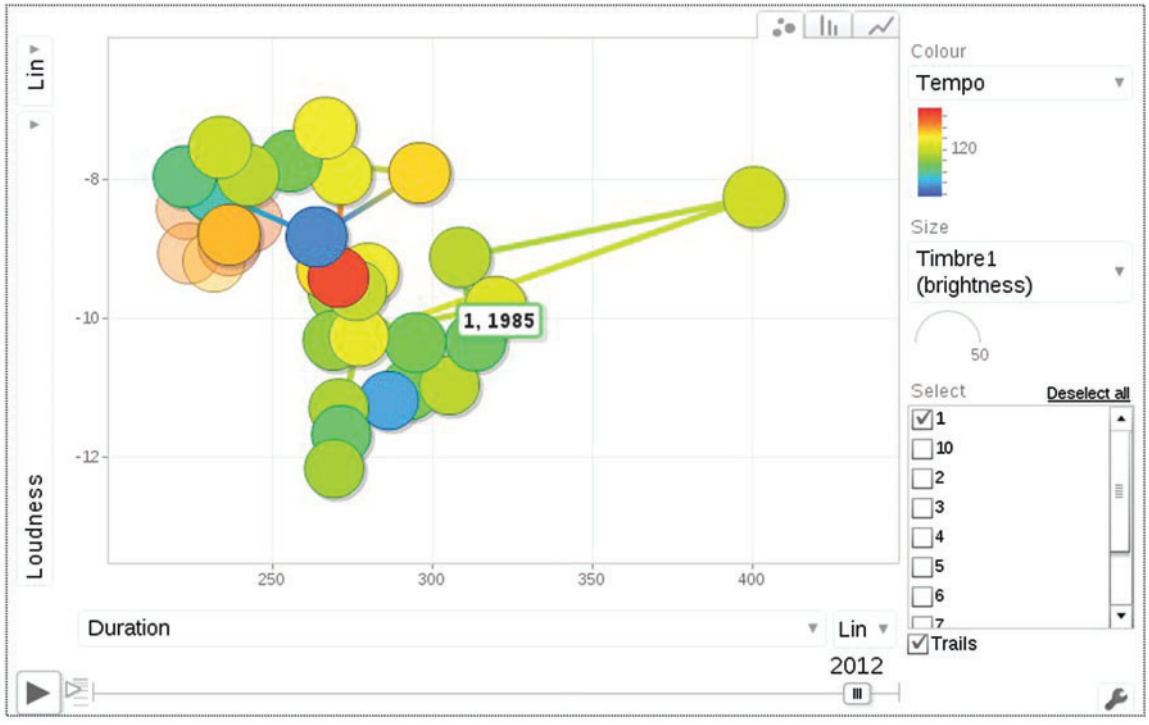

Fig. 1. Motion chart visualizing evolution of dance hits from 1985 until 2013[5].

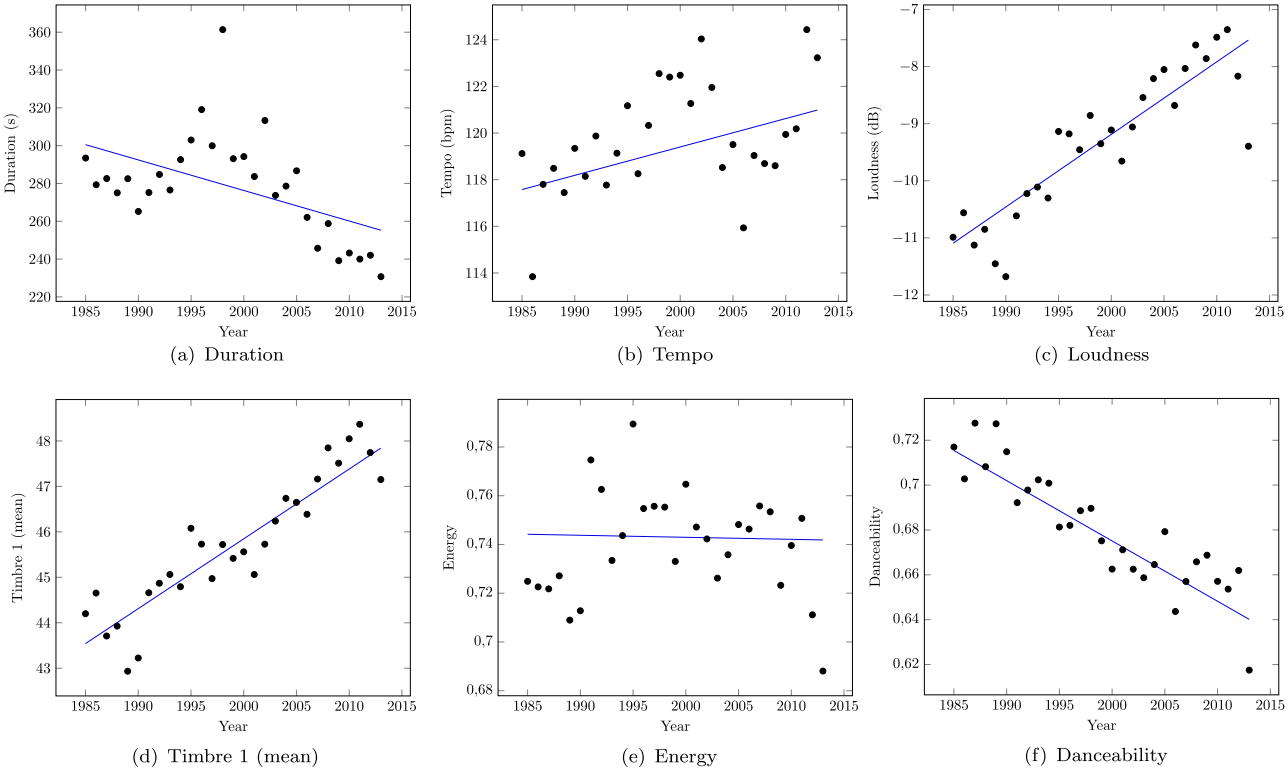

Fig. 2. Evolution over time of selected characteristics of top 10 songs.

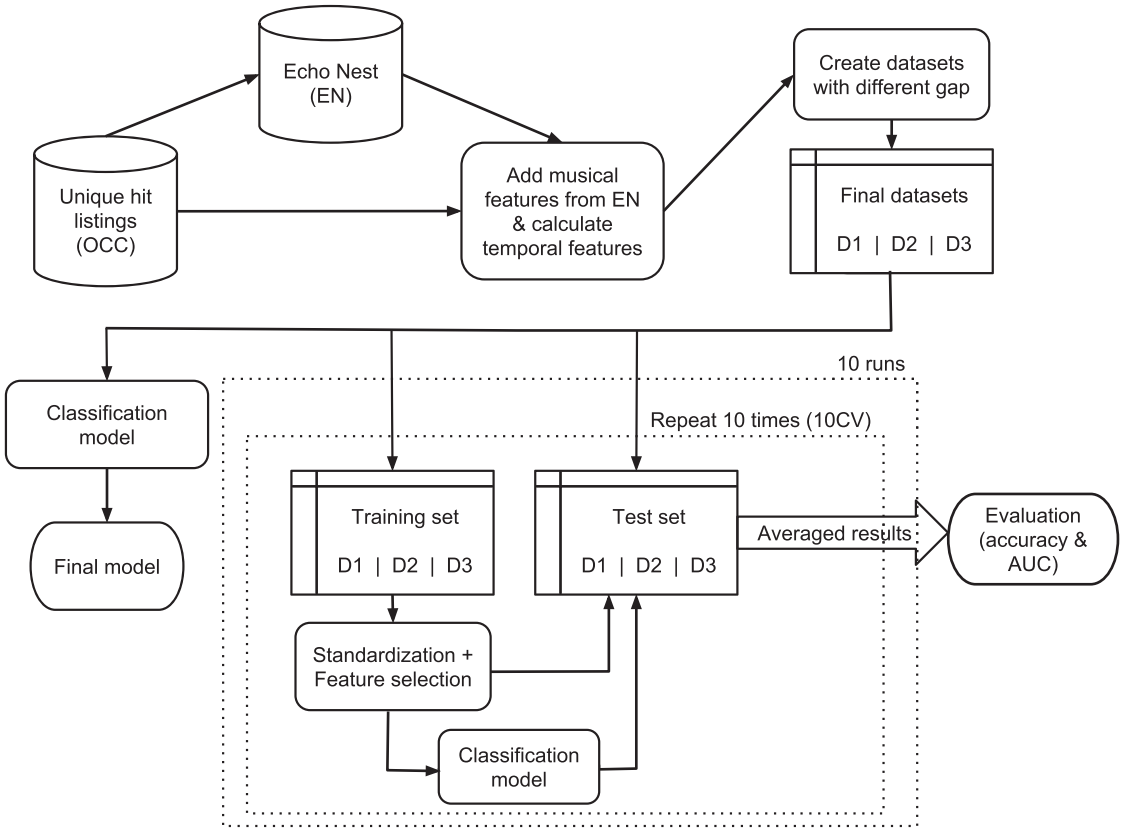

Fig. 3. Flow chart of the experimental setup.

y-axis is the average loudness per year in Figure 1. Additional dimensions are represented by the size of the bubbles (brightness) and the colour of the bubbles (tempo).

Since a motion chart is a dynamic tool that should be viewed on a computer, a selection of features were extracted to more traditional two-dimensional graphs with linear regressions (see Figure 2). Since the OCC dataset contains 3361 unique songs, the selected features from these songs were averaged per year in order to limit the amount of data points on the graph. A rising trend can be detected for the loudness, tempo and first aspect of timbre (brightness). The correlation between loudness and tempo is in line with the rule proposed by Todd (1992) 'The faster the louder, the softer the slower'. Not all features have an apparent relationship with time. Energy, for instance (see Figure 2(e)), doesn't seem to be correlated with time. It is also remarkable that the danceability feature computed by The Echo Nest decreases over time for dance hits. Since no detailed formula was given by The Echo Nest for danceability, this trend cannot be explained.

The next section describes an experiment which compares several hit prediction models built in this research.

4. Dance hit prediction

In this section the experimental setup and preprocessing techniques are described for the classification models built in Section 5.

4.1 Experiment setup

Figure 3 shows a graphical representation of the setup of the experiment described in Section 6.1. The dataset used for the hit prediction models in this section is based on the OCC listings. The reason for this is that this data contains top 40 songs, not just top 10. This will allow us to create a 'gap' between the two classes. Since the previous section showed that the characteristics of hit songs evolve over time it is not representable to use data from 1985 for predicting contemporary hits. The dataset used for building the prediction models consists of dance hit songs from 2009 until 2013.

The peak chart position of each song was used to determine if they are a dance hit or not. Three datasets were made with each having a different gap between the two classes (see Table 3). In the first dataset (D1), hits are considered to be songs with a peak position in the top 10. Non-hits are those that only reached a position between 30 and 40. In the second dataset (D2), the gap between hits and non-hits is smaller, as

Table 3. Datasets used for the dance hit prediction model.

Dataset	Hits	Non-hits	Size
D1	Top 10	Top 30–40	400
D2	Top 10	Top 20–40	550
D3	Top 20	Top 20–40	697

songs reaching a top position of 20 are still considered to be non-hits. Finally, the original dataset is split in two at position 20, without a gap to form the third dataset (D3). The reason for not comparing a top 10 hit with a song that did not appear in the charts is to avoid doing accidental genre classification. If a hit dance song would be compared to a song that does not occur in the hit listings, a second classification model would be needed to ensure that this non-hit song is in fact a dance song. If not, the developed model might distinguish songs based on whether or not they are a dance song instead of a hit. However, it should be noted that not all songs on the dance hit lists are in fact the same type of dance songs, there might be subgenres. Still, they will probably share more common attributes than songs from a random style, thus reducing the noise in the hit classification model. The sizes of the three datasets are listed in Table 3, the difference in size can be explained by the fact that songs are excluded in D1 and D2 to form the gap. In the next sections, models are built and we compare the performance of classifiers on these three datasets.

The Open Source software Weka was used to create the models (Witten & Frank, 2005). Weka's toolbox and framework is recognized as a landmark system in the data mining and machine learning field (Hall et al., 2009).

4.2 Preprocessing

The class distribution of the three datasets used in the experiment is displayed in Figure 4. Although the distribution is not heavily skewed, it is not completely balanced either. Because of this the use of the accuracy measure to evaluate our results is not suited and the area under the receiver operating curve (AUC) (Fawcett, 2004) was used instead (see Section 6).

All of the features in the datasets were standardized using statistical normalization and feature selection was done (see Figure 3), using the procedure CfsSubsetEval from Weka with GeneticSearch. This procedure uses the individual predictive ability of each feature and the degree of redundancy between them to evaluate the worth of a subset of features (Hall, 1999). Feature selection was done in order to avoid the 'curse of dimensionality' by having a very sparse feature set. McKay and Fujinaga (2006) point to the fact that having a limited amount of features allows for a thorough testing of the model with limited instances and can thus improve the quality of the classification model. Added benefits are the improved comprehensibility of a model with a limited amount of highly predictive variables (Hall, 1999) and better performance of the learning algorithm (Piramuthu, 2004).

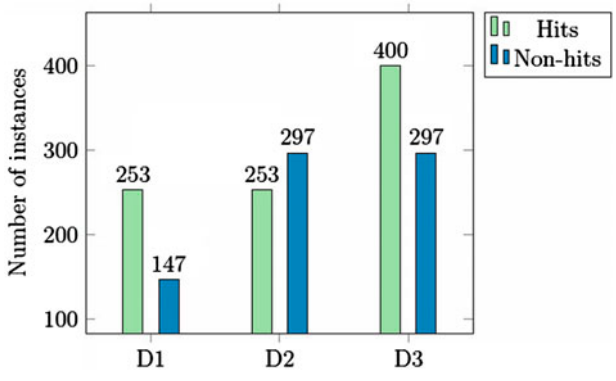

Fig. 4. Class distribution.

Table 4. The most commonly occurring features in D1, D2 and D3 after FS.

Feature	Occurrence	Feature	Occurrence
Beatdiff (range)	3	Timbre 1 (mean)	2
Timbre 1 (80 perc)	3	Timbre 1 (median)	2
Timbre 1 (max)	3	Timbre 2 (max)	2
Timbre 1 (stdev)	3	Timbre 2 (mean)	2
Timbre 2 (80 perc)	3	Timbre 2 (range)	2
Timbre 3 (mean)	3	Timbre 3 (var)	2
Timbre 3 (median)	3	Timbre 4 (80 perc)	2
Timbre 3 (min)	3	Timbre 5 (mean)	2
Timbre 3 (stdev)	3	Timbre 5 (stdev)	2
Beatdiff (80 perc)	2	Timbre 6 (median)	2
Beatdiff (stdev)	2	Timbre 6 (range)	2
Beatdiff (var)	2	Timbre 6 (var)	2
Timbre 11 (80 perc)	2	Timbre 7 (var)	2
Timbre 11 (var)	2	Timbre 8 (Median)	2
Timbre 12 (kurtosis)	2	Timbre 9 (kurtosis)	2
Timbre 12 (Median)	2	Timbre 9 (max)	2
Timbre 12 (min)	2	Timbre 9 (Median)	2

The feature selection procedure in Weka reduces the data to 35–50 attributes, depending on the dataset. The most commonly occurring features after feature selection are listed in Table 4. Interesting to note is that the features *danceability* and *energy* both disappear from the reduced datasets, except for danceability which stays in the D3 dataset. This could be explained by the fact that these features are calculated by The Echo Nest based on other features.

5. Classification techniques

A total of five models were built for each dataset using diverse classification techniques. The two first models (decision tree and ruleset) can be considered as the easiest to understand classification models due to their linguistic nature (Martens, 2008). The other three models focus on accurate prediction. In the following subsections, the individual algorithms are briefly discussed together with their main parameters and settings, followed by a comparison in Section 6. The AUC values mentioned in this section are based on 10-fold cross-validation performance (Witten & Frank, 2005). The shown models are built on the entire dataset.

5.1 C4.5 tree

A decision tree for dance hit prediction was built with J48, Weka's implementation of the popular C4.5 algorithm (Witten & Frank, 2005).

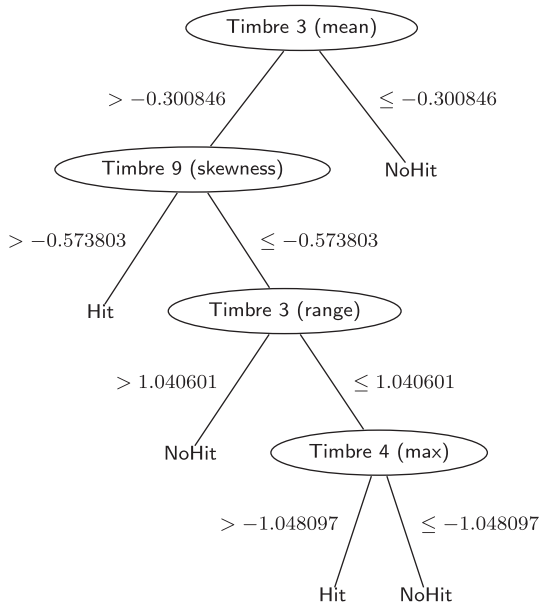

Fig. 5. C4.5 decision tree.

The tree data structure consists of decision nodes and leaves. The class value is specified by the leaves, in this case hit or non-hit, and the nodes specify a test of one of the features. When a path from the node to a leaf is followed based on the feature values of a particular song, a predictive rule can be derived (Ruggieri, 2002).

A 'divide and conquer' approach is used by the C4.5 algorithm to build trees recursively (Quinlan, 1993). This is a top down approach, in which a feature is sought that best separates the classes, followed by pruning of the tree (Wu et al., 2008). This pruning is performed by a subtree raising operation in an inner cross-validation loop (three folds by default in Weka) (Witten & Frank, 2005).

Decision trees have been used in a broad range of fields such as credit scoring (Hand & Henley, 1997), land cover mapping (Friedl & Brodley, 1997), medical diagnosis (Wolberg & Mangasarian, 1990), estimation of toxic hazards (Cramer, Ford, & Hall, 1976), predicting customer behaviour changes (Kim, Song, Kim, & Kim, 2005) and others.

For the comparative tests in Section 6 Weka's default settings were kept for J48. In order to create a simple abstracted model on dataset D1 (FS) for visual insight in the important features, a less accurate model (AUC 0.54) was created by pruning the tree to depth four. The resulting tree is displayed in Figure 5. It is noticeable that time differences between the third, fourth and ninth timbre vector seem to be important features for classification.

5.2 RIPPER ruleset

Much like trees, rulesets are a useful tool to gain insight in the data. They have been used in other fields to gain insight in diagnosis of technical processes (Isermann & Balle, 1997), credit scoring (Baesens, Setiono, Mues, & Vanthienen, 2003), medical diagnosis (Kononenko, 1995), customer relationship management (Ngai, Xiu, & Chau, 2009) and more.

In this section JRip, Weka's implementation of the propositional rule learner RIPPER (Cohen, 1995), was used to inductively build 'if-then' rules. The 'Repeated Incremental Pruning to Produce Error Reduction algorithm' (RIPPER), uses sequential covering to generate the ruleset. In a first step of this algorithm, one rule is learned and the training instances that are covered by this rule are removed. This process is then repeated (Hall et al., 2009).

The ruleset displayed in Table 5 was generated with Weka's default parameters for number of data instances (2) and folds (3) (AUC = 0.56 on dataset D1, see Table 7, Section 6). It's notable that the third timbre vector is an important feature again. It would appear that this feature should not be underestimated when composing dance songs.

5.3 Naive Bayes

The naive Bayes classifier estimates the probability of a hit or non-hit based on the assumption that the features are conditionally independent. This conditional independence assumption is represented by Equation 1 given class label y (Tan, Steinbach, & Kumar, 2007).

$$P(\mathbf{x}|Y = y) = \prod_{j=1}^{M} P(x_j|Y = y), \tag{1}$$

whereby each attribute set $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$ consists of M attributes.

Because of the conditional dependence assumption, the class-conditional probability for every combination of $\mathbf{x}$ does not need to be calculated. Only the conditional probability of each x_i given Y has to be estimated. This offers a practical advantage since a good estimate of the probability can be obtained without the need for a very large training set.

Naive Bayes classifies a test record by calculating the posterior probability for each class Y (Lewis, 1998):

$$P(\mathrm{Y}|\mathbf{x}) = \frac{P(Y) \cdot \prod_{j=1}^{M} P(x_j|Y)}{P(\mathbf{x})} \tag{2}$$

Although this independence assumption is generally a poor assumption in practice, numerous studies prove that naive Bayes competes well with more sophisticated classifiers (Rish, 2001). In particular, naive Bayes seems to be particularly resistant to isolated noise points, robust to irrelevant attributes, but its performance can degrade by correlated attributes (Tan et al., 2007). Table 7 (see Section 6) confirms that Naive Bayes performs very well, with an AUC of 0.65 on dataset D1 (FS).

5.4 Logistic regression

The SimpleLogistic function in Weka was used to build a logistic regression model (Witten & Frank, 2005).

Table 5. RIPPER ruleset.

(T1mean ≤ -0.020016) and (T3min ≤ -0.534123) and (T2max ≥ -0.250608) $\Rightarrow$ NoHit
(T880perc ≤ -0.405264) and (T3mean ≤ -0.075106) $\Rightarrow$ NoHit
$\Rightarrow$ Hit

Table 6. Results with 10-fold validation (accuracy).

Accuracy (%)	D1		D2		D3	
	–	FS	–	FS	–	FS
C4.5	*57.05*	*58.25*	*54.95*	*54.67*	*54.58*	*54.74*
RIPPER	*60.95*	*62.43*	*56.69*	*56.42*	*57.18*	*56.41*
Naive Bayes	**65**	**65**	*60.22*	*58.78*	59.57	*59.18*
Logistic regression	**64.65**	**64**	**62.64**	**60.6**	**60.12**	*59.75*
SVM (Polynomial)	**64.97**	**64.7**	**61.55**	**61.6**	**61.04**	**61.07**
SVM (RBF)	**64.7**	**64.63**	*59.8*	*59.89*	**60.8**	**60.76**

FS = feature selection, $p < 0.01$: italic, $p > 0.05$: bold, best: bold.

Table 7. Results for 10 runs with 10-fold validation (AUC).

AUC	D1		D2		D3	
	–	FS	–	FS	–	FS
C4.5	*0.53*	*0.55*	*0.55*	*0.54*	*0.54*	*0.53*
RIPPER	*0.55*	*0.56*	*0.56*	*0.56*	*0.54*	*0.55*
Naive Bayes	**0.64**	**0.65**	*0.64*	**0.63**	**0.6**	*0.61*
Logistic regression	**0.65**	**0.65**	**0.67**	**0.64**	**0.61**	**0.63**
SVM (Polynomial)	*0.6*	*0.59*	*0.61*	*0.61*	*0.58*	*0.58*
SVM (RBF)	*0.56*	*0.56*	*0.59*	*0.6*	*0.57*	*0.57*

FS = feature selection, $p < 0.01$: italic, $p > 0.05$: bold, best: bold.

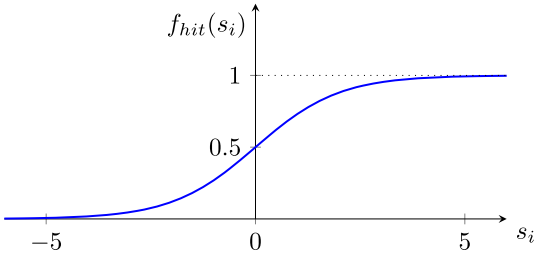

Fig. 6. Probability that song i is a dance hit.

Equation 3 shows the output of a logistic regression, whereby $f_{\text{hit}}(s_i)$ represents the probability that a song i with M features x_j is a dance hit. This probability follows a logistic curve, as can be seen in Figure 6. The cut-off point of 0.5 will determine if a song is classified as a hit or a non-hit. With AUC = 0.65 for dataset D1 and AUC=0.67 for dataset D2 (see Table 7, Section 6), logistic regression performs best for this particular classification problem.

$$f_{\text{hit}}(s_i) = \frac{1}{1+e^{-s_i}} \quad \text{whereby } s_i = b + \sum_{j=1}^{M} a_j \cdot x_j \qquad (3)$$

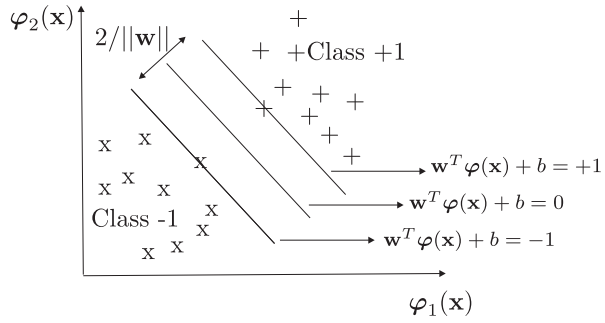

Fig. 7. Illustration of SVM optimization of the margin in the feature space.

Logistic regression models generally require limited computing power and are less prone to overfitting than other models such as neural networks (Tu, 1996). Like the previously mentioned models, they are also used in a number of domains, such as the creation of habitat models for animals (Pearce & Ferrier, 2000), medical diagnosis (Kurt, Ture, & Kurum, 2008), credit scoring (Wiginton, 1980) and others.

5.5 Support vector machines

Weka's sequential minimal optimization algorithm (SMO) was used to build two support vector machine classifiers. The support vector machine (SVM) is a learning procedure based on the statistical learning theory (Vapnik, 1995). Given a training set of N data points $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ with input data $\mathbf{x}_i \in \mathbb{R}^n$ and corresponding binary class labels $y_i \in \{-1, +1\}$, the SVM classifier should fulfill following conditions (Cristianini & Shawe-Taylor, 2000; Vapnik, 1995):

$$\begin{cases} \mathbf{w}^T\varphi(\mathbf{x}_i) + b \geq +1, & \text{if } \; y_i = +1, \\ \mathbf{w}^T\varphi(\mathbf{x}_i) + b \leq -1, & \text{if } \; y_i = -1, \end{cases} \qquad (4)$$

which is equivalent to

$$y_i[\mathbf{w}^T\varphi(\mathbf{x}_i) + b] \geq 1, \qquad i = 1, ..., N. \qquad (5)$$

The non-linear function $\varphi(\cdot)$ maps the input space to a high (possibly infinite) dimensional feature space. In this feature space, the above inequalities basically construct a hyperplane $\mathbf{w}^T\varphi(\mathbf{x}) + b = 0$ discriminating between the two classes. By minimizing $\mathbf{w}^T\mathbf{w}$, the margin between both classes is maximized (see Figure 7).

In primal weight space the classifier then takes the form

$$y(\mathbf{x}) = \text{sign}[\mathbf{w}^T\varphi(\mathbf{x}) + b], \qquad (6)$$

but, on the other hand, is never evaluated in this form. One defines the convex optimization problem:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \mathcal{J}(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \tag{7}$$

subject to

$$\begin{cases} y_i[\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b] \geq 1 - \xi_i, & i = 1, ..., N, \\ \xi_i \geq 0, & i = 1, ..., N. \end{cases} \tag{8}$$

The variables ξ_i are slack variables which are needed to allow misclassifications in the set of inequalities (e.g. due to overlapping distributions). The first part of the objective function tries to maximize the margin between both classes in the feature space and is a regularization mechanism that penalizes for large weights, whereas the second part minimizes the misclassification error. The positive real constant C is the regularization coefficient and should be considered as a tuning parameter in the algorithm.

This leads to the following classifier (Cristianini & Shawe-Taylor, 2000):

$$y(\mathbf{x}) = \text{sign}\left[\sum_{i=1}^{N}\alpha_i\, y_i\, K(\mathbf{x}_i, \mathbf{x}) + b\right], \tag{9}$$

whereby $K(\mathbf{x}_i, \mathbf{x}) = \boldsymbol{\varphi}(\mathbf{x}_i)^T\boldsymbol{\varphi}(\mathbf{x})$ is taken with a positive definite kernel satisfying the Mercer theorem. The Lagrange multipliers α_i are then determined by optimizing the dual problem. The following kernel functions $K(\cdot, \cdot)$ were used:

$$K(\mathbf{x}, \mathbf{x}_i) = (1 + \mathbf{x}_i^T\mathbf{x}/c)^d, \quad \text{(polynomial kernel)}$$
$$K(\mathbf{x}, \mathbf{x}_i) = \exp\{-\|\mathbf{x} - \mathbf{x}_i\|_2^2/\sigma^2\}, \quad \text{(RBF kernel)}$$

where d, c and σ are constants.

For low-noise problems, many of the α_i will typically be equal to zero (sparseness property). The training observations corresponding to non-zero α_i are called support vectors and are located close to the decision boundary.

As Equation 9 shows, the SVM classifier with non-linear kernel is a complex, non-linear function. Trying to comprehend the logics of the classifications made is quite difficult, if not impossible (Martens, Van Gestel, & Baesens, 2009; Martens & Provost, 2014).

In this research, the Polynomial kernel and RBF kernel were used to build the models. Although Weka's default settings were used in the previous models, the hyperparameters for the SVM model were optimized. To determine the optimal settings for the regularization parameter C (1, 3, 5,…, 21), the σ for the RBF kernel ($\frac{1}{\sigma^2}$ = 0.00001, 0.0001,…10) and the exponent d for the polynomial kernel (1,2), GridSearch was used in Weka. The choice of hyperparameters to test was inspired by settings suggesting by Weka (2013a). GridSearch performs two-fold cross-validation on the initial grid. This grid is determined by the two input parameters (C and σ for the RBF kernel, C and d for the polynomial kernel). 10-fold cross-validation is then performed on the best point of the grid based on the weighted AUC by class size and its adjacent points. If a better pair is found, the procedure is repeated on its neighbours until no better pair is found or the border of the grid is reached (Weka, 2013b). This hyperparameter optimization is performed in the 'classification model' box in Figure 3. The resulting AUC-value is 0.59 for the SVM with polynomial and 0.56 for the SVM with RBF kernel on D1 (FS) (see Table 7, Section 6).

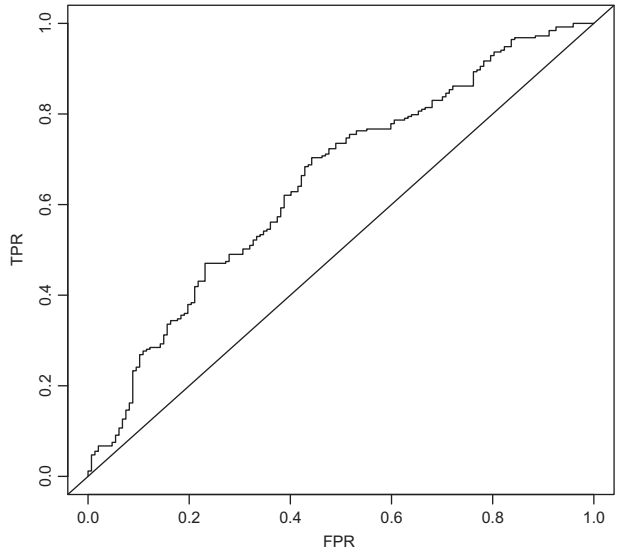

Fig. 8. ROC for Logistic regression.

6. Results

In this section, two experiments are described. The first one builds models for all of the datasets (D1, D2 & D3), both with and without feature selection. The evaluation is done by taking the average of 10 runs, each with a 10-fold cross-validation procedure. In the second experiment, the performance of the classifiers on the best dataset is compared with an out-of-time test set.

6.1 Full experiment with cross-validation

A comparison of the accuracy and the AUC is displayed in Tables 6 and 7 for all of the above-mentioned classifiers. The tests were run 10 times, each time with stratified 10-fold cross-validation (10CV), both with and without feature selection (FS). This process is depicted in Figure 3. As mentioned in Section 4.2, AUC is a more suited measure since the datasets are not entirely balanced (Fawcett, 2004), yet both are displayed to be complete. During the cross-validation procedure, the dataset is divided into 10 folds. Nine of them are used for model building and one for testing. This procedure is repeated 10 times. The displayed AUC and accuracy in this subsection are the average results over the 10 test sets and the 10 runs. The resulting model is built on the entire dataset and can be expected to have a performance which is at least as good as the 10CV performance. A total of 10 runs were performed with the 10CV procedure and the average results

Table 8. Results for 10 runs on D1 (FS) with 10-fold cross-validation compared with the split test set.

	AUC		accuracy (%)	
	split	10CV	split	10CV
C4.5	0.62	*0.55*	62.50	*58.25*
RIPPER	0.66	*0.56*	**85**	*62.43*
Naive Bayes	0.79	**0.65**	77.50	**65**
Logistic regression	**0.81**	**0.65**	80	**64**
SVM (Polynomial)	0.729	*0.59*	**85**	**64.7**
SVM (RBF)	0.57	*0.56*	82.5	**64.63**

$p < 0.01$: italic, $p > 0.05$: bold, best: bold.

are displayed in Tables 6 and 7. A Wilcoxon signed-rank test is conducted to compare the performance of the models with the best performing model. The null hypothesis of this test states: 'There is no difference in the performance of a model with the best model'.

As described in the previous section, decision trees and rulesets do not always offer the most accurate classification results, but their main advantage is their comprehensibility (Craven & Shavlik, 1996). It is rather surprising that support vector machines do not perform very well on this particular problem. The overall best technique seems to be the logistic regression, closely followed by naive Bayes. Another conclusion from the table is that feature selection seems to have a positive influence on the AUC for D1 and D3. As expected, the overall best results when taking into account both AUC and accuracy can be obtained using the dataset with the biggest gap, namely D1.

The overall best model seems to be logistic regression. The receiver operating curve (ROC) is displayed in Figure 8. The ROC curve displays the trade-off between true positive rate (TPR) and false negative rate (FNR) of the logistic classifier with 10-fold cross-validation for D1 (FS). The model clearly scores better than a random classification, which is represented by the diagonal through the origin.

The confusion matrix of the logistic regression shows that 209 hits (i.e. 83% of the actual hits) were accurately classified as hits and 47 non-hits classified as non-hits (i.e. 32% of the actual non-hits). Yet overall, the model is able to make a fairly good distinction between classes, which proves that the dance hit prediction problem can be tackled as realistic top 10 versus top 30–40 classification problem with logistic regression.

6.2 Experiment with out-of-time test set

A second experiment was conducted with an out-of-time test set based on D1 with feature selection. The instances were first ordered by date, and then split into a 90% training and 10% test set. Table 8 confirms the good performance of the logistic regression. A peculiar observation from this table is that the model seems to be able to predict better for newer songs (AUC: 0.81 versus 0.65). This can be due to coincidence, different class distribution between training and test set (see Figure 9) or the structure of the dataset. One speculation of the authors

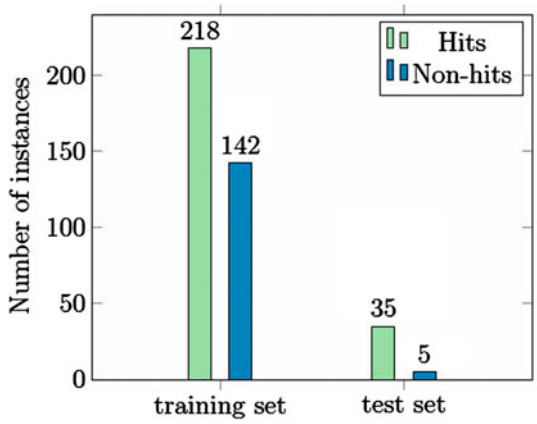

Fig. 9. Class distribution of the split training and test sets.

Table 9. Confusion matrix logistic regression.

a	b	← classified as
209	44	a = hit
100	47	b = non-hit

is that the oldest instances of the dataset might be 'lingering' hits, meaning that they were top 10 hits on a date before the earliest entry in the dataset, and were still present in a low position in the used hit listings. These songs would be falsely seen as non-hits, which might cause the model to predict less good for older songs.

7. Conclusion

Multiple models were built that can successfully predict if a dance song is going to be a top 10 hit versus a lower positioned dance song. In order to do this, hit listings from two chart magazines were collected and mapped to audio features provided by The Echo Nest. Standard audio features were used, as well as more advanced features that capture the temporal aspect. This resulted in a model that could accurately predict top 10 dance hits.

This research proves that popularity of dance songs *can* be learnt from the analysis of music signals. Previous less successful results in this field speculate that their results could be due to features that are not informative enough (Pachet & Roy, 2008). The positive results from this paper could indeed be due to the use of more advanced temporal features. A second cause might be the use of 'recent' songs only, which eliminates the fact that hit music evolves over time. It might also be due to the nature of dance music or that by focussing on one particular style of music, any noise created by classifying hits of different genres is reduced. Finally, by comparing different classifiers that have significantly different results in performance, the best model could be selected.

This model was implemented in an online application where users can upload their audio data and get the probability of

it being a hit.[6] An interesting future expansion would be to improve the accuracy of the model by including more features such as lyrics, social network information and others. The model could also be expanded to predict hits of other musical styles. In the line of research being done with automatic composition systems (Herremans & Sörensen, 2013), it is also interesting to see if the classification models from this paper could be included in an optimization function (e.g. a type of fitness function) and used to generate new dance hits or improve existing ones.

Funding

This research has been partially supported by the Interuniversity Attraction Poles (IUAP) Programme initiated by the Belgian Science Policy Office (COMEX project).

References

Baesens, B., Setiono, R., Mues, C., & Vanthienen, J. (2003). Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Science, 49*(3), 312–329.

Bertin-Mahieux, T., Ellis, D., Whitman, B., & Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)* (pp. 591–596). ???: International Society for Music Information Retrieval.

Borg, N., & Hokkanen, G. (2011). What makes for a hit pop song? What makes for a pop song? Retrieved from http://cs229.stanford.edu/proj2011/BorgHokkanen-WhatMakesForAHitPopSong.pdf

Braheny, J. (2007). *Craft and Business of Songwriting* (3rd ed.). New York: F & W Publications.

Casey, M., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., & Slaney, M. (2008). Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE, 96*(4), 668–696.

Cohen, W. (1995). Fast effective rule induction. In A. Prieditis, & S. Russell (Eds.), *Proceedings of the 12th International Conference on Machine Learning* (pp. 115–123). Tahoe City, CA: Morgan Kaufmann.

Cramer, G., Ford, R., & Hall, R. (1976). Estimation of toxic hazard: A decision tree approach. *Food and Cosmetics Toxicology, 16*(3), 255–276.

Craven, M., & Shavlik, J. (1996). Extracting tree-structured representations of trained networks. *Advances in Neural Information Processing Systems, 8*, 24–30.

Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. New York: Cambridge University Press.

Dhanaraj, R., & Logan, B. (2005). Automatic prediction of hit songs. In *Proceedings of the international conference on music information retrieval* (pp. 488–491). London: Queen Mary, University of London.

Downie, J. (2003). Music information retrieval. *Annual Review of Information Science and Technology, 37*(1), 295–340.

EchoNest (2013). The Echo Nest. Retrieved from http://echonest.com

Essid, S., Richard, G., & David, B. (2006). Musical instrument recognition by pairwise classification strategies. *IEEE Transactions on Audio, Speech, and Language Processing, 14*(4), 1401–1412.

Fawcett, T. (2004). ROC graphs: Notes and practical considerations for researchers. *Machine Learning, 31*, 1–38.

Friedl, M., & Brodley, C. (1997). Decision tree classification of land cover from remotely sensed data. *Remote Sensing of Environment, 61*(3), 399–409.

Fu, Z., Lu, G., Ting, K., & Zhang, D. (2011). A survey of audio-based music classification and annotation. *IEEE Transactions on Multimedia, 13*(2), 303–319.

Google. (2013). Google Charts – Visualisation: Motion Graph. Retrieved from https://developers.google.com/chart/interactive/docs/gallery/motionchart

Hall, M. (1999). *Correlation-based feature selection for machine learning* (PhD thesis). The University of Waikato, Hamilton, New Zealand.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009). The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter, 11*(1), 10–18.

Hand, D., & Henley, W. (1997). Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society: Series A (Statistics in Society), 160*(3), 523–541.

Herremans, D., & Sörensen, K. (2013). Composing fifth species counterpoint music with a variable neighborhood search algorithm. *Expert Systems with Applications, 40*(16), 6427–6437.

Herremans, D., Sörensen, K., & Martens, D. (2013). Classification and generation of composer specific music. *Working Paper*. Belgium: University of Antwerp.

Houston, P. (2013). *Instant jsoup How-to*. Birmingham: Packt Publishing Ltd.

IFPI. (2012). *Investing in music* International Federation of the Phonographic Industry. Retrieved from http://www.ifpi.org/content/library/investing_in_music.pdf

Isermann, R., & Balle, P. (1997). Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice, 5*(5), 709–719.

Jehan, T. (2005). *Creating music by listening*. (PhD thesis). Massachusetts Institute of Technology, Cambridge, MA.

Jehan, T., & DesRoches, D. (2012). EchoNest Analyzer Documentation. Retrieved from http://developer.echonest.com/docs/v4/_static/AnalyzeDocumentation.pdf

Jelinek, F. (2005). Some of my best friends are linguists. *Language Resources and Evaluation, 39*(1), 25–34.

Kim, J., Song, H., Kim, T., & Kim, H. (2005). Detecting the change of customer behavior based on decision tree analysis. *Expert Systems, 22*(4), 193–205.

Kononenko, I. (1995). Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine, 23*(1), 89–109.

[6] http://antor.ua.ac.be/dance

Kurt, I., Ture, M., & Kurum, A. (2008). Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. *Expert Systems with Applications, 34*(1), 366–374.

Lamere, P. (2013). jEN-API – A java client for the EchoNest. Retrieved from http://code.google.com/p/jen-api/

Laurier, C., Grivolla, J., & Herrera, P. (2008). Multimodal music mood classification using audio and lyrics. In *Seventh International Conference on Machine Learning and Applications, ICMLA'08* (pp. 688–693).

Leikin, M. (2008). *How to Write a Hit Song* (5th ed.). Winona, MN: Hal Leonard.

Lewis, D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In *Machine learning: ECML-98* (pp. 4–15). Berlin: Springer.

Martens, D. (2008). Building acceptable classification models for financial engineering applications. *SIGKDD Explorations, 10*(2), 30–31. Retrieved from http://dl.acm.org/citation.cfm?id=1540285

Martens, D., & Provost, F. (2014). Explaining data-driven document classifications. *MIS Quarterly, 38*(1), 73–99.

Martens, D., Van Gestel, T., & Baesens, B. (2009). Decompositional rule extraction from support vector machines by active learning. *IEEE Transactions on Knowledge and Data Engineering, 21*(2), 178–191.

McKay, C., & Fujinaga, I. (2006). jSymbolic: A feature extractor for MIDI files. In *Proceedings of the International Computer Music Conference* (pp. 302–305), New Orleans, LA.

Ngai, E., Xiu, L., & Chau, D. (2009). Application of data mining techniques in customer relationship management: A literature review and classification. *Expert Systems with Applications, 36*(2), 2592–2602.

Ni, Y., Santos-Rodríguez, R., McVicar, M., & De Bie, T. (2011). Hit song science once again a science?

Ni, Y., Santos-Rodríguez, R., McVicar, M., & De Bie, T. (2013). Score a Hit - Documentation. Retrieved from http://www.scoreahit.com/Documentation

Pachet, F. (2012). Hit song science. In T. Li, G. Tzanetakis, & M. Ogihara (Eds.), *Music Data Mining* (pp. 305–326). Boca Raton, FL: Chapman & Hall/CRC Press.

Pachet, F., & Roy, P. (2008). Hit song science is not yet a science. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008)* (pp. 355–360), Philadelphia, PA.

Pearce, J., & Ferrier, S. (2000). Evaluating the predictive performance of habitat models developed using logistic regression. *Ecological Modelling, 133*(3), 225–245.

Perricone, J. (2000). *Melody in songwriting: Tools and techniques for writing hit songs (Berklee Guide)*. Boston: Berklee Press.

Piramuthu, S. (2004). Evaluating feature selection methods for learning in data mining applications. *European Journal of Operational Research, 156*(2), 483–494.

Quinlan, J. (1993). *C4. 5: Programs for machine learning* (Vol. 1). Burlington, MA: Morgan Kaufmann.

Rish, I. (2001). An empirical study of the naive Bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence* (Vol. 1, pp. 41–46).

Ruggieri, S. (2002). Efficient C4. 5 [classification algorithm]. *IEEE Transactions on Knowledge and Data Engineering, 14*(2), 438–444.

Salganik, M., Dodds, P., & Watts, D. (2006). Experimental study of inequality and unpredictability in an artificial cultural market. *Science, 311*(5762), 854–856.

Schindler, A., & Rauber, A. (2012). Capturing the temporal domain in echonest features for improved classification effectiveness. *Proceedings on Adaptive Multimedia Retrieval (Oct. 2012)*.

Schnitzer, D., Flexer, A., & Widmer, G. (2009). A filter-and-refine indexing method for fast similarity search in millions of music tracks. In *Proceedings of the 10th International Conference on Music, Information Retrieval (ISMIR09)*.

Shneiderman, B. (2002). Inventing discovery tools: combining information visualization with data mining. *Information Visualization, 1*(1), 5–12.

Tan, P., Steinbach, M., & Kumar, V. (2007). *Introduction to data mining*. Delhi: Pearson Education India.

Todd, N. (1992). The dynamics of dynamics: A model of musical expression. *The Journal of the Acoustical Society of America, 91*, 3540–3550.

Tu, J. (1996). Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of Clinical Epidemiology, 49*(11), 1225–1231.

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing, 10*(5), 293–302.

Vapnik, V. (1995). *The nature of statistical learning theory*. New York: Springer-Verlag.

Webb, J. (1999). *Tunesmith: Inside the art of songwriting*. New York: Hyperion.

Weka. (2013a). *Optimizing parameters*. Hamilton, New Zealand: The University of Waikato. Retrieved from http://weka.wikispaces.com/Optimizing+parameters

Weka. (2013b). *Weka documentation, class GridSearch*. Retrieved from http://weka.sourceforge.net/doc.stable/weka/classifiers/meta/GridSearch.html

Whitman, B., & Smaragdis, P. (2002). Combining musical and cultural features for intelligent style detection. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)* (pp. 47–52).

Wiginton, J. (1980). A note on the comparison of logit and discriminant models of consumer credit behavior. *Journal of Financial and Quantitative Analysis, 15*(03), 757–770.

Witten, I., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Burlington, MA: Morgan Kaufmann.

Wolberg, W., & Mangasarian, O. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the National Academy of Sciences, 87*(23), 9193–9196.

Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.-H., Steinbach, M., Hand, D.J., & Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems, 14*(1), 1–37.