HIT SONG PREDICTION USING FEATURE-BASED MACHINE LEARNING

by

Anıl Orhan Çalışkol

B.S., Computer Engineering, Bilkent University, 2015

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2019

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis advisor Assoc. Prof. Arzucan Özgür for orienting, motivating and supporting me in the emergence of this thesis.

I would like to thank Assist. Prof. Emre Uğur and Prof. Olcay Taner Yıldız for accepting to participate in my thesis committee.

Finally, I am grateful to my family who have been with me in every difficult situation that brought me to these days and raised me diligently.

ABSTRACT

HIT SONG PREDICTION USING FEATURE-BASED MACHINE LEARNING

Music industry is making big investments every year to produce hit songs. The increasing number of songs available through digital platforms can enable the development of learning models for predicting hit songs and identifying their common features. This thesis investigates classifying a song as hit or non-hit by using various machine learning methods. Besides the basic musical features provided by Spotify, more complex features based on the chords and melody extracted from the music files by utilizing music theory information are designed. Chord based features are created using the important chord progressions based on tonal harmony, while the features based on melody are designed in an intuitive way. In addition, new benchmark datasets are created by using both hit and non-hit songs from dance and rock music genres. The results show that using chord and melody based features with the basic musical features may lead to an improvement in hit song prediction performance. For rock songs, the Random Forest classifier achieves a significant improvement on the results by using these features. It is also observed that using a specific feature combination with Support Vector Machine classifier increases the accuracy score of hit dance song prediction. Furthermore, all the features used in this study are analyzed in the last part of this study for each dataset.

ÖZET

ÖZNİTELİĞE DAYALI MAKİNE ÖĞRENMESİ İLE HİT ŞARKI TAHMİNİ

Müzik endüstrisi her yıl hit şarkıları üretmek için büyük yatırımlar yapıyor. Dijital platformlarda artan şarkı sayısı, şarkıları tahmin etmek ve ortak özelliklerini tanımlamak için öğrenme modellerinin geliştirilmesini sağlayabilir. Bu tez, çeşitli makine öğrenmesi yöntemleri kullanılarak bir şarkının hit veya hit olmayan olarak sınıflandırılmasını inceler. Spotify tarafından sağlanan temel müzikal özniteliklerin yanı sıra, müzik teorisinden yararlanılarak müzik dosyalarından elde edilen akor ve melodilere dayanan daha karmaşık öznitelikler tasarlandı. Akora dayalı öznitelikleri ton uyumuna dayalı önemli akor yürüyüşleri kullanılarak oluşturulurken, melodiye dayalı öznitelikler sezgisel bir şekilde tasarlanmıştır. Ayrıca, dans ve rock müzik türlerinden hem hit hem hit olmayan şarkılar kullanılarak yeni veri setleri oluşturulmuştur. Temel müzik özniteliklerinin akor ve melodi bazlı özniteliklerle birlikte kullanılmasıyla ortaya çıkan sonuçlar, bu özniteliklerin hit şarkı tahmin performansında iyileşmeye yol açabileceğini göstermektedir. Rock şarkıları için, Rassal Orman sınıflandırıcısı bu öznitelikleri kullanarak sonuçlarda istatistiksel olarak anlamlı bir gelişme sağlamıştır. Destek Vektör Makinesi sınıflandırıcısı ile belirli bir öznitelik kombinasyonunun kullanılmasının, hit dans şarkıları tahmininin doğruluk puanını arttırdığı gözlemlenmiştir. Ayrıca, bu çalışmada kullanılan tüm öznitelikler bu çalışmanın son bölümünde her veri kümesi için analiz edilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

IR	Informational Retrieval
MIR	Music Informational Retrieval
API	Application Programming Interface
RNN	Recurrent Neural Network
DBN	Deep Belief Network
MFCC	Mel Frequency Cepstral Coefficient
NNLS	Non-Negative Least Squares
HTML	HyperText Markup Language
DOM	Document Object Model
RBF	Radial Basis Function
SR	Support Vector Machine with RBF Kernel
SL	Support Vector Machine with Linear Kernel
SP	Support Vector Machine with Polynomial Kernel
LR	Logistic Regression
NB	Naive Bayes
DT	Decision Tree
RF	Random Forest
XG	XGBoost
BF	Basic Features
CF	Chord based Features
CoF	Common Progression Chord Feature
CiF	Circle Progression Chord Feature
SPFY	Songs from Spotify Playlists

1. INTRODUCTION

What are the main characteristics of hit songs? This issue has attracted the attention of music industry and become a curiosity for anyone who is interested in songwriting. Big investments are being made to promote and popularize music. "Hit Song Science" has emerged from researching hit music, revealing its secrets, and using them to predict the success of a song in advance of its release into the market [8].

Hit song science is an area of music information retrieval (MIR). MIR, which enables gathering information from music and analyzing the emerging information [9], is a multidisciplinary field comprising several areas such as psychology, musicology, music theory, signal processing, and machine learning. Although MIR was first declared as a programming language for extracting information from music in 1966 [10], with the rapid advances in technology and the increase in the amount of music in the digital environment, it has become an important research area.

In recent years, many applications of music information retrieval have been developed. Searching a song by humming or singing is one of the examples of content-based music retrieval [11], which enables people to reach information about a song when they remember only its melody. Also, classification of music is a very common problem in music information retrieval. Many types of classifications such as timbre [12], genre [13], mood [14], composer [15], instrument [16], cultural origin [17], and hit ranking [18] have been investigated.

Musical similarity is a study in the area of MIR. It tries to find the songs that have the most similar combination of audio features such as melody, rhythm, harmony, and instruments. Similarity can have different types like genre similarity, mood similarity, song similarity (different covers of song), composer similarity etc. [19]. These measures can be used in many applications.

With the increase of online music listening in the last decade, to improve user experience, music recommender systems are used in most platforms. Therefore, users enjoy a quality music experience while listening to their favorite songs and getting new suggestions related to them at the same time.

Another application of MIR is automatic music transcription which can be done by using multi-pitch detection, duration and tempo estimation, and instrument identification. As the number of instruments and layers in the song increases, it becomes more difficult to produce transcription. Many musicians can benefit from this practice.

Music identification or automatic content recognition is used to detect a track played from a media device or file and retrieve the metadata of it. In these systems, there is a database of acoustic fingerprints of songs. A music identification system searches for a matching fingerprint of the recorded audio from this database that was created before. For example, Shazam and Soundhound are the most known and used programs in this area.

Automatic music generation has also become popular in the recent years. For example, a system that listens to the music of Bach and generates music with a similar style by using deep neural networks has been developed [20].

The hit song science term is used for the research, which aims to predict whether a song will be popular, before it is released by using information extracted from music data. It investigates whether there are common features of successful songs, and aims to reveal them. After providing the song data, these systems try to find the possibility (or probability) of the song being a hit. Hit song science popularized after the dataset named Million Song Dataset was generated [21]. Many researchers who are interested in music started to analyze hit songs to find a formulation and model for them. When the commercial success of a hit song is taken into consideration, the music industry's interest in this area is inescapable [22].

In this research, in addition to the features used in the hit song prediction studies so far such as metadata, lyrics, duration, tempo, key, and mode of the song, new features extracted from chords and melody were designed based on music theory. In addition, various combinations of datasets were formed by using both hit and non-hit songs on dance and rock music genres. Besides, classifiers were trained to use the songs that had been released earlier than the songs used for hit song prediction, since the aim is to predict upcoming hit songs in advance. The results show that the top 10 hits have distinct qualities, since this group was classified more accurately compared to the other song groups.

1.1. Thesis Organization

This thesis is organised as follows. In Chapter 2, the topics and applications of music information retrieval are discussed and the machine learning methods used in this thesis are explained. In Chapter 3, basic music terms and the evaluation metrics used in the experiments are clarified. In Chapter 4, the creation, content, and organisation of the datasets are described. In Chapter 5, the basic features gathered from Spotify, the song chord and melody features are presented. In Chapter 6, the details of the experiments and the results are discussed. In Chapter 7, the conclusions we reach and the possible future work are summarized.

2. RELATED WORK

In [23], which is one of the first studies on hit song science, acoustic features and lyrics are used to separate hit and non-hit songs from each other, support vector machines are used as the classification method. Although it was one of the first attempts in this area and there was no a large dataset of songs, promising results were obtained.

After Salganik et al. reached the conclusion that the success of a song in cultural markets cannot be predicted, in 2008, another study concluded that the popularity of a song cannot be predicted by using acoustic or human features, because it is claimed that these features are not enough to determine subjective aesthetic judgements in songs [24, 25]. They reached the result that hit song science is not yet a science; however, with more experiments better results can be obtained.

In 2011, a research showed that when using relevant features from the million song dataset and the shifting perceptron algorithm as a state-of-the-art machine learning technique, hit songs in the top 5 and non-hit songs with a rank between top 30 and top 40 can be distinguishable and popularity of a track is not accidental [26]. Thus, hit song science once again came to agenda and it was seen as a field worth researching. However, the reported results were around 57% accuracy, with a lot of room for improvement.

In 2014, a research was conducted to predict hit dance songs by classifying the songs as top 10 hits or as a hit in a lower position, and they also used the million song dataset [15]. With this paper, it was shown that the popularity of dance songs can be predicted with audio features. The results of this study were more promising than the previous ones, which could be due to using more advanced temporal features, using recent songs and focusing on a certain music genre. We took this research as our base and used a similar dataset to predict hit dance songs by developing new song chord and melody features. In addition, we generated new benchmark datasets consisting of rock songs.

Chord extraction algorithms are necessary to extract the song chord features described in this thesis. Chord recognition systems use probabilistic models to predict chords progressions in songs. A chord recognition system based on recurrent neural networks (RNN) was developed in [27]. It includes both an acoustic model that uses audio signal components and a musicological model that uses a language model describing the temporal dependencies of chords according to music theory. It has obtained the best results with approximately 80% accuracy. In 2010, a chroma extraction method using a non-negative least squares (NNLS) algorithm that can reach 80% accuracy was proposed [28]. Since it performed well on chord extraction, a chord and harmony extraction library named Chordino was released. In 2016, an audio processing and music information retrieval library named Madmom, which uses convolutional neural networks for chord recognition was developed [29].

Audio melody extraction is a subject that enables pitches (fundamental frequencies) of predominant melody in a polyphonic audio recording to be extracted. In polyphonic music, multiple notes can be heard at the same time, which can also come from different instruments. There are four types of melody extraction methods: salience based, source separation, data driven and monophonic. A salience-based melody extraction vamp plug-in named MELODIA is designed in 2012 [30].

2.1. Machine Learning Methods

In the experiments, the Scikit-Learn python library, which is an extensive library for machine learning algorithms, is used [31]. We used the following machine learning methods for the hit song classification task.

2.1.1. Decision Tree

A decision tree uses a tree-like model including decisions and their possible consequences. It can be used for several applications such as maximizing profit or optimizing a cost for a user, calculating the best option for a problem, and indicating which outcome will actually happen in case of uncertainty. It can also be used for representing

an algorithm which includes only conditional control statements. Decision trees are used as a supervised machine learning method which are generally used for classification tasks; however, they can be used for regression in some cases. They can split the sample into two or more homogeneous sets using differentiator in input variables. Tree-based methods are more preferred than linear models in most cases because they can also handle non-linear relationships [32]. CART (Classification and Regression Trees) is used to represent decision tree algorithms.

Decision trees are similar to flowchart diagrams. They consist of three types of nodes such as root node which represents the entire population or sample, internal node which indicates an attribute and a decision about it, and a leaf node which has a class label showing the final decision. A branch is one of the sub-trees of an internal node. Branches indicate the paths for all possible conditions. A decision tree diagram can be seen in Figure 2.1.

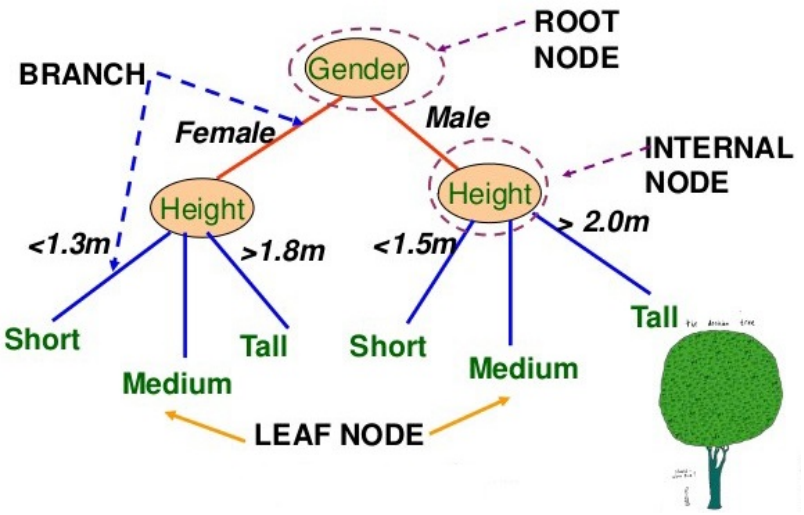

Figure 2.1. Decision Tree Diagram, adopted from [1]

Decision trees choose the best homogeneous split while constructing the tree. In order to decide which variable and value are used in the split, algorithms such as gini index, chi-square, and information gain are used. The purity of a node must be increased by using these algorithm to achieve the best results.

There are some advantages of decision trees such as being easy to understand, being able to handle non-linear problems and all data types, as well as being useful in data exploration. However, sometimes decision trees may lead to overfitting which arises from the failure to generalize the solution. Therefore, the small changes in data may cause building of completely different trees. It can be solved by pruning or setting constraints on model parameters. Also, bagging and boosting is helpful in order to solve this problem. These methods are used in Random Forest and XGBoost algorithms.

2.1.2. Naive Bayes

Naive Bayes is a type of classifier which belongs to supervised machine learning methods. Reason for using the word naive is that it is assumed that features are statistically independent in this method. With this assumption, this classifier is highly scalable and it can learn very fast numerous features using a small training dataset. Therefore, it can be used for many real world datasets such as speech, text, and image data. There are many modern applications of it such as spam filtering, vocal emotion recognition, and automatic medical diagnoses [33].

Naive Bayes is created based on Bayes' theorem. It is the conditional probability calculation formula found by Thomas Bayes in 1812. This theorem shows the relationship between conditional probabilities and marginal probabilities within a probability distribution for a random variable and the theorem denoted as follows:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.1}$$

Naive Bayes classifier predicts C_k class for given input vector x = $(x_1, \ldots, x_n)$ which represents features for each class k according to the probability $P(C_k|x_1, \ldots, x_n)$. By using Bayes' theorem, this probability can be written as follows:

$$P(C_k|x_1, x_2, \ldots, x_n) = \frac{P(x_1, x_2, \ldots, x_n|C_k) \cdot P(C_k)}{P(x_1, x_2, \ldots, x_n)} \tag{2.2}$$

Using the chain rule and the assumption that we have independent features, this probability can be converted to the following one:

$$P(C_k|x_1, x_2, \ldots, x_n) = \frac{P(C_k) \cdot \prod_{i=1}^{n} P(x_i|C_k)}{P(x_1, x_2, \ldots, x_n)} \tag{2.3}$$

Naive Bayes classifier must find the class k which maximizes the probability $P(C_k|x_1, x_2, \ldots, x_n)$ in order to classify the output for given input vector x. Since the denominator is constant, this probability is proportional to $P(C_k) \cdot \prod_{i=1}^{n} P(x_i|C_k)$. Therefore, only the numerator part of the probability can be considered to get class k.

In training part, the prior probabilities of each class $P(C_k)$ and, for each class and input value the likelihood probabilities $P(x_i|C_k)$ are calculated. These values are used to calculate posterior probability $P(C_k|x_1, x_2, \ldots, x_n)$ in order to make a prediction.

Considering the distribution of the features, likelihood probabilities can be calculated using density functions of the distributions. For example, when input values fits into Gaussian distribution, Gaussian density function can be used.

In multinomial naive bayes, feature vectors correspond to the frequencies that some events have been generated by a multinomial $(p_1, \ldots, p_n)$ where p_i is the probability that event i occurs.

2.1.3. Logistic Regression

Logistic regression is a binary classification method which uses the logistic function. The logistic function is an s-shaped curve that can map real-valued number into a value between 0 and 1. It is defined as follows:

$$y = \frac{1}{1 + e^{-x}} \tag{2.4}$$

In the above function, e indicates Euler's number and x is the input value. If there are n features, the linear equation which represents our model will be $b_0 + b_1 \cdot x_1 + \cdots + b_n \cdot x_n$ where b_0 is the bias and the values from b_1 to b_n are the coefficients for the input value x. Then, when we put the equation into logistic function, we get logistic regression equation as follows:

$$y = \frac{e^{b_0 + b_1 \cdot x_1 + \cdots + b_n \cdot x_n}}{1 + e^{b_0 + b_1 \cdot x_1 + \cdots + b_n \cdot x_n}} \tag{2.5}$$

In this equation, the y value which is between 0 and 1 is used to determine the output class. Values smaller than 0.5 may indicate the first output class, while values greater than or equal to 0.5 may indicate the second output class.

By using training data and minimizing the estimation error in the model, the coefficients and bias in the equation are estimated using maximum-likelihood estimation.

2.1.4. Support Vector Machines

Support vector machines, which are one of the supervised machine learning algorithm, were created by Vladimir Vapnik and Corrina Cortes [34] in 1995. They can be used for both classification and regression analysis but they are generally used for the classification task.

The idea behind the algorithm is that the input vectors are mapped into high dimensional feature space Z by non-linear mapping. In this space, this algorithm constructs a hyperplane which separates the data points from two classes so that classification with the least margin of error is made as seen in Figure 2.2. Support vectors are the ones closest to the red circles in the same figure. They can be seen as the border of the area of class data points. The goal of the algorithm is to make the distance between these support vectors maximized.

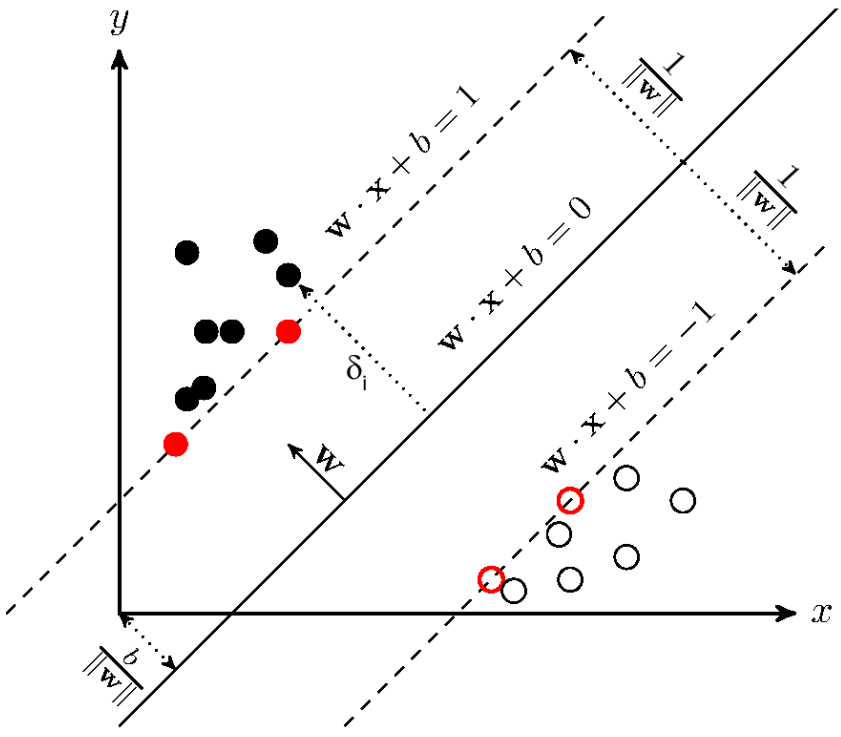

Figure 2.2. Support Vector Machines constructs a hyper-plane with the maximum margin between two classes, adopted from [2]

Consider a labeled training set consists of N data points (x_1, y_1) where x shows input vector, y shows the binary class labels for each data point and $y_i \in \{-1, 1\}$. If there is a vector w and a scalar b that provides the following inequalities for all elements in the training set then, this training set is said to be linearly separable.

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x} + b \geq 1 \quad &\text{if} \quad y_i = 1 \\ \mathbf{w} \cdot \mathbf{x} + b \leq -1 \quad &\text{if} \quad y_i = -1 \end{aligned} \tag{2.6}$$

These inequalities can be written in the form for all binary class labels as follows:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \tag{2.7}$$

It turns out to be an optimization problem since the main purpose of SVM is to maximize the margin between support vectors. Consider the following equation shows the optimal hyperplane which is the unique one maximizes the margin.

$$\mathbf{w}_0 \cdot \mathbf{x} + b_0 = 0 \tag{2.8}$$

The distance between the two support vectors can be calculated as follows:

$$\rho(\mathbf{w}_0, b_0) = \frac{2}{\mathbf{w}_0} = \frac{2}{\sqrt{\mathbf{w}_0 \cdot \mathbf{w}_0}} \tag{2.9}$$

Therefore, in order to reach the optimal hyperlane, $w \cdot w$ must be minimized in Equation 2.7. This optimization can be done by using quadratic programming.

When an optimal hyperplane does not exist for certain problems because of the linearity, to find the optimal hyperplane, more complex kernels can be used such as polynomial kernel and radial basis function kernel. In this research linear, polynomial and RBF kernels are used.

2.1.5. Random Forest

Random forest is a supervised ensemble learning method which can be used for both classification and regression [35]. Since it consists of a set of decision trees, it is named after forest as seen in Figure 2.3.

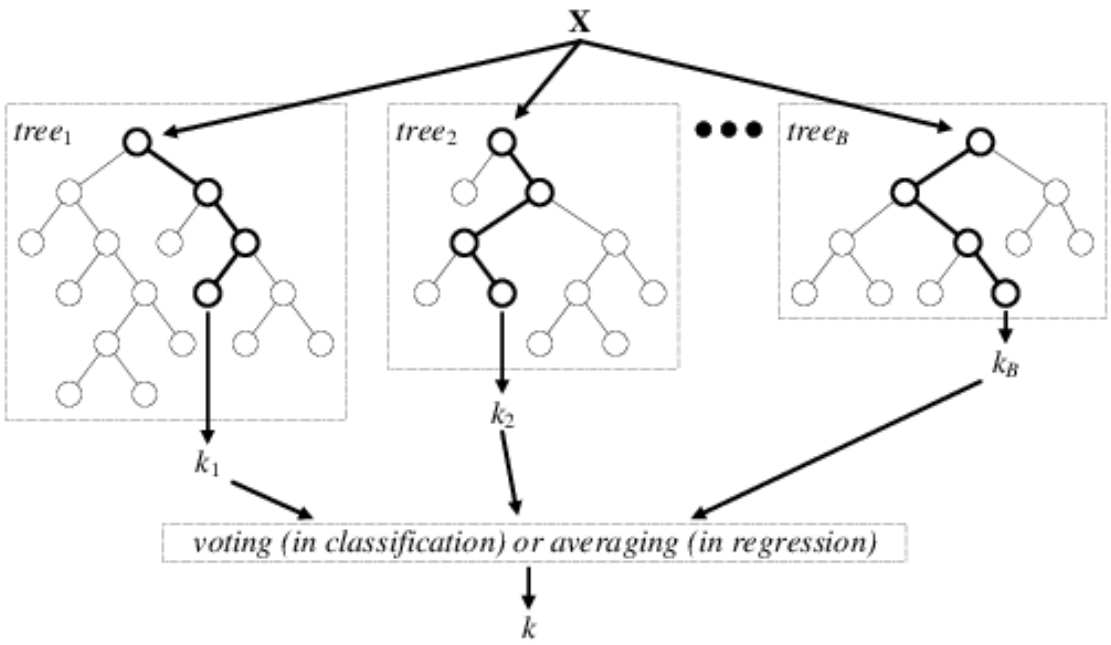

Figure 2.3. Architecture of the Random Forest Model, adopted from [3]

Decision trees which have high depth usually have low bias and high variance since they tend to learn very complex patterns. This situation causes the decision trees to overfit on the training set. Therefore, predictions are very sensitive to noise

and can be changed easily with small fluctuations.

In ensemble methods, combination of machine learning techniques forms one predictive model so as to decrease variance by using bagging [36]. In bagging, by selecting random samples from the training set with replacements, different decision trees are trained [37]. Then, in order to perform a prediction on unseen samples, for regression process the average of the predictions from all the regression trees which are used to train these samples is used; however, for classification process, the predicted class is decided by the voting of all classification trees. However, only bagging is not enough. In order to perform a prediction by using random forest classifier, feature bagging is needed. In this process, random subsets of the features must be used during the training process. Since different decision trees specialize in different features and less correlated trees are formed, variance of the model decreases, while bias increases. The success of prediction increases by using numerous trees and as randomness increases.

2.1.6. XGBoost

XGBoost, which started as a research project within the scope of Distributed Machine Learning Community group, is a supervised ensemble learning algorithm that uses gradient boosting in order to optimize the loss function [38]. It is known as the algorithm used by many winning teams in various machine learning competitions [39]. This algorithm is powerful since it uses parallel and distributed computing, and also uses efficient memory.

Unlike the bagging techniques used in random forest, in boosting, decision trees are built sequentially and each of them is in charge of reducing the errors of the previous tree [40]. This way, trees with strong learning are generated instead of many weak trees like in bagging. Therefore, a decision tree which has lower variance and bias is created. Different from bagging method, trees with fewer splits are used in boosting. Moreover, because having many trees may cause overfitting, there must be a stopping criteria for boosting.

In gradient boosting, by computing the average gradient component and multiplying it by some factor, γ, for samples in each leaf of the tree, the next learner tree is generated. Basically, using the gradient of the lost function on the previous learner, the next learner is produced.

Moreover, there are many advantages of XGBoost such as regularization, cache awareness, and weighted quantile sketch.

3. PRELIMINARIES

3.1. Music Terms

3.1.1. Pitch and Octave

In music, an octave consists of 12 notes (C, C#, D, ... A#, B). In Figure 3.1, it is seen on the piano that octaves are separated from one another with different colors. Pitch corresponds to a frequency for a specific note in an octave. For example C4 describes C note in the fourth octave. The frequency of notes increases as you move to the right on the keyboard. Therefore, pitches also help us understand whether different notes are higher or lower than each other.

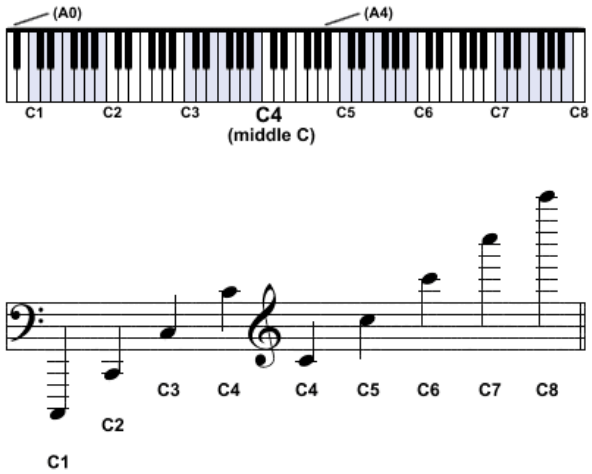

Figure 3.1. Octave naming and pitch notation, adopted from [4]

3.1.2. Chord

A chord is a term that consist of multiple notes. However, these notes must be heard simultaneously, i.e. on the keyboard, notes must be played at the same time. When they are sounding sequentially, then it is called arpeggio. For example, C major chord consists of C, E and G notes as seen in Figure 3.2.

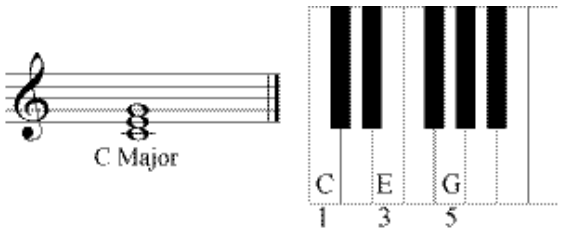

Figure 3.2. C major chord, adopted from [5]

3.1.3. Melody

Melody,is a sequential pitch collection which is perceived an entity. These pitches have a rhythmic value; however, in this work, we have not considered the rhythm in melodies.

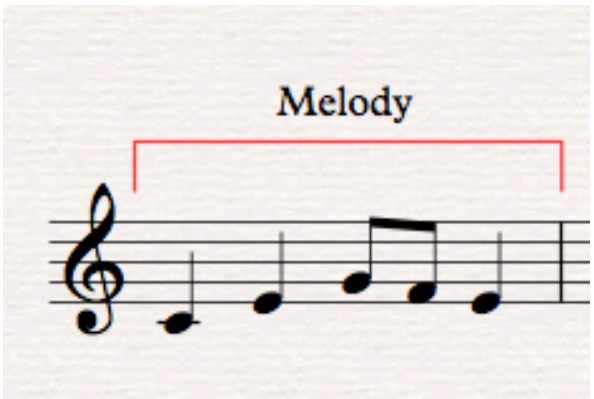

Figure 3.3. Melody example, adopted from [6]

3.2. Experiment Evaluation Metrics

3.2.1. Classification Metrics

In binary classification, there are some evaluation metrics that are used in order to evaluate the results. We can explain these metrics in our hit song prediction case. There are two classes (hit or non-hit) for songs and we want to predict the real class values of songs. If we predict a hit song as hit, then this result is true positive. If we predict a hit song as non-hit, its true condition is positive but prediction condition is

negative; therefore, this is a false negative condition. All the four conditions can be seen in Table 3.1.

Table 3.1. Classification confusion matrix

	True Condition Positive	**True Condition Negative**
Predicted Condition Positive	True Positive	False Positive
Predicted Condition Negative	False Negative	True Negative

Accuracy score is the fraction which describes how many songs we predicted with its correct class among all the songs. Precision score is the fraction which shows how many hit songs are predicted correctly among the songs which are predicted as hit songs. Recall score is the fraction which shows how many hit songs are predicted correctly among the songs which are really hits.

F-measure is a metric which uses both the precision and recall scores in order to calculate how accurate the classification is. It can be calculated as the harmonic mean of precision and recall as shown in Equation 3.1.

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.1}$$

4. SONG LISTS AND DATASET

In order to create datasets in this work, firstly, song lists are created by using reliable chart sources. Then, the songs on these lists are matched to the songs on Spotify. Finally, audio features of these songs are gathered from Spotify API to create the main datasets for dance and rock songs.

4.1. Spotify Web API

Spotify enables developers to search for playlists generated by both Spotify and its users, to reach song tracks in these playlists and to get information about tracks, artists, and albums. Also, every song has a unique track ID which consists of mixed alphanumeric characters. [41]

The API also provides audio features for all songs which are present on Spotify. These features are extracted by using the system developed by a music intelligence company called the Echo Nest. In 2011, a dataset called "The One Million Song" was built by Bertin-Mahieux, Ellis, Whitman and Lamere using the Echo Nest API [21]. In 2014, Spotify purchased Echo Nest and the Echo Nest API continues as a Spotify service [42].

The audio features are used as basic features in the experiments and will be explained in Chapter 5.

4.2. Song Lists

The song lists used in this research can be divided into two sections as hit songs and non-hit songs. Hit song lists are gathered from two sources which have large chart archives. On the other hand, non-hit songs are collected from Spotify playlists. After the lists are collected, the basic musical features are gathered from Spotify API.

Table 4.1. Song lists generated by using hit lists

	Hit Dance Songs	**Hit Rock Songs**
Source Web Site	Official Charts	Billboard Charts
Top	40	50
Date Range	10/2009 - 3/2013	07/2009 - 09/2017
Unique Songs	696	1681

4.2.1. Song list generated by using hit lists

Two different chart sources are used to gather the list of hit songs. Billboard magazine and Official Charts Company have an archive of these charts including various music genres [43, 44]. Official Charts Company collects information on the sales of downloads, CD's, vinly and other physical music formats in the United Kingdom provided by Millward Brown market research company in order to create their weekly Top 40 charts [45]. Billboard is an American magazine that publishes news, charts, reviews, and events focused on music industry. It is known by its charts such as Billboard Hot 100 which indicates the top-selling songs, and Billboard 200 which indicates the top-selling albums. Billboard charts feed from sales of songs from stores or internet, and radio airplay [46].

In order to collect dance hit songs list, Official Dance Singles Chart Top 40 is used like in the dance hit prediction paper by Herremans et al. [18]. On the other hand, Hot Rock Songs chart from Billboard magazine is preferred for hit rock songs list since it includes songs from the USA and other countries. By using Simple HTML DOM Parser library, song title and artist name, weekly chart position, and chart date information are parsed from these web sites [47]. After parsing, 7319 entries for dance songs and 21550 entries for rock songs having these four features are gathered. Finally, using these chart entries, unique song lists are created with their peak positions for both music genres. A peak position indicates the best chart position of a song during the time period when it is on the chart.

After the unique song lists are generated, the Spotify track ID of each song is obtained by using the search track endpoint in Spotify Web API.

4.2.2. Song list generated by Spotify playlists (Non-hits)

Playlists including rock songs are gathered by using the search in playlists endpoint in Spotify Web API. Those who are not listed in hit rock songs list were added to the non-hit songs list directly. 1368 songs whose release dates are between 2009 and 2018 years were added to this list with their metadata such as song title, artist, year, and Spotify track ID.

5. FEATURES

In this research, song features consisting of basic features, more complex song chord features, and melody features are used.

5.1. Basic Features

The audio features provided by Spotify API are used as basic features. These features are obtained by using the track IDs of songs from the API [41].

The audio features such as song popularity, artist popularity, duration, tempo, time signature, key, mode, loudness, danceability, and energy are used for dance hit songs, since these features have also been used in [18]. However, in addition to these features, additional features such as acousticness, insturmentalness, liveness, speechiness, and valence are also used for rock songs dataset. The descriptions of the basic features are provided below.

(i) Song popularity provides a value between 0 and 100 based on the total number of plays the track has had and how recent those plays are.
(ii) Artist popularity is a value between 0 and 100 that indicates the popularity of the artist, calculated from the popularity of all the artist's tracks.
(iii) Duration shows the length of the track in miliseconds.
(iv) Tempo is the average number of beats per minute, and it is proportional to the speed of the rhythm in the song.
(v) Time signature shows how many beats take part in one bar (or measure).
(vi) Key indicates which note the song is based on. It can take integer values from 0 to 11 such as C=0, C#=1 etc.
(vii) Mode is the modality of the track. It can take two integer values, 0 for minor and 1 for major mode.
(viii) Loudness is a float value showing the overall loudness of the track in decibels (dB).

(ix) Danceability is also a float value between 1.0 and 0.0, showing how appropriate the song is for dancing in terms of tempo, rhythm stability, beat strength, and overall regularity.

(x) Energy indicates how energetic the song is based on how fast, noisy and loud the track is.

(xi) Acousticness provides a probability of the track having an acoustic sound.

(xii) Instrumentalness shows whether the track includes vocals, the fewer vocals the higher the value of it. For example, in rap songs its value approaches to 0.0, while in instrumental songs it is closer to 1.0.

(xiii) Liveness is a feature revealing whether the track is performed live by detecting audience in the audio file. When it is higher than 0.8, the track is most probably live.

(xiv) Speechiness indicates whether the track has spoken words. While it starts to approach the value of 1.0, the possibility of that track being a recording of a speech such as talk show, audio book increases.

More detailed information about these features is provided in Spotify Web API.

5.2. Song Chord Features

5.2.1. Chord Extraction and Preprocessing

In order to extract song chord features, the song chords had to be extracted first. We used the madmom library in this study, since an article that evaluated the state-of-the-art chord extraction algorithms demonstrated that madmom [29], which is an audio and music signal processing library performed better than its competitors such as Essentia [48] and Chordino [49] with 89.63% score [50]. This library uses convolutional neural networks to extract chords and outputs the chords as well as the time intervals in seconds as shown in Table 5.1. Only the major and minor chords are extracted. In addition to these, the time intervals that do not have any chords (shown with the N symbol), are removed while preprocessing.

Table 5.1. Example song chords output

Start Time	**End Time**	**Chord**
0.0	0.2	N
0.2	1.6	F:maj
1.6	2.4	A:maj
2.4	4.1	D:min

The audio files of the songs in the dataset are collected and the chords of all songs are extracted as text files. In order to analyze the chords regardless of the key of a song, the chords are converted to roman numerals which indicate chord functions relative to a key. These roman numerals indicate scale degrees. Therefore, functions of chords can be analyzed independently from key [51, 52].

5.2.2. Chord Progression Features

In music, composers use chords to maintain harmonic flow in songs. Therefore, chord sequences known as chord progressions are formed. By using functional harmony in music theory, progressions can be divided into two features as common progressions and circle progressions. These progressions were generated with the intention to create strong transitions between chords.

5.2.2.1. Common Progressions. In tonal harmony, certain chord successions are assumed to be more characteristic than the others. Some successions tend to sound like they are about to resolve, while some others create a feeling of incompleteness. By using this information in functional harmony, two charts can be formed for major and minor keys as shown in Figures 5.1 and 5.2, respectively [52]. The dotted arrow shows that there can be a transition from first degree to every scale degree.

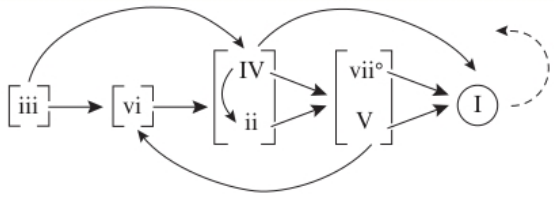

Figure 5.1. Common chord progression chart in major key

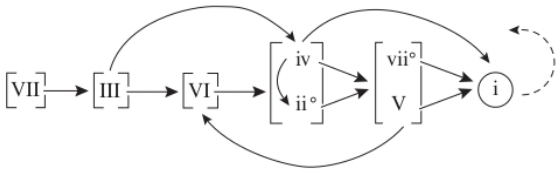

Figure 5.2. Common chord progression chart in minor key

5.2.2.2. Circle Progressions. In music theory, circle of the fifth is another important concept that gives a lot of information about harmony. The circle consists of consecutive fifths in counter-clockwise direction or fourths in clockwise direction. Circle progressions chart occurs when circle of the fifth is figured by using scale degrees. This type of progression is known as the strongest and the most common among all harmonic progressions. It is very common in classical music and it forms the basis of progressions in other music genres [53].

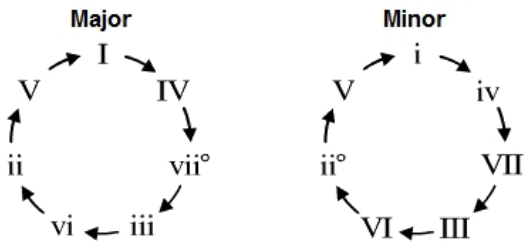

Figure 5.3. Circle progressions chart, adopted from [7]

5.2.3. Resolution

Resolution is the movement from a dissonant chord to consonant chord in tonal music theory. It is used to create musical interest. This feature is defined by counting the progressions from V chord to I chord and vii chord to I in major scale and V chord to i chord in minor scale.

5.2.4. Calculation of Chord Feature Values

In order to calculate the value of these three features from chords, we measured how much the chords fit into these patterns. All the transitions in the chord progression of a song are checked to determine whether they are appropriate for the given charts in terms of the mode of the song (major or minor). The count of transitions that fit into these patterns in terms of the key of the song is divided by the total transition count, whose square gives the feature score as shown in formula 5.1. Therefore, this feature score always takes a value between 0 and 1 inclusive.

$$FeatureScore = (\frac{t_{fits}}{t_{total}})^2 \tag{5.1}$$

For example, while the common progression score for a song in major mode with progression iii - vi - V - I is being calculated, it is noticed that 2 transitions fit into the chart out of a total 3 progressions. The score can be calculated as $(\frac{2}{3})^2$.

5.3. Song Melody Features

Firstly, the usage of the melody extraction plug-in and the preprocessing part are explained. Then, melody leap, melody pitch mean, consonance and dissonance features are defined and how to calculate them is explained.

5.3.1. Melody Extraction and Preprocessing

In order to extract song melody features, MELODIA plug-in is used [30]. It generates predominant pitch values with their corresponding time intervals. Melodia has four parameters in order to increase the correction of estimation. The first and second parameters determine the minimum and maximum frequency that is to be used to extract in hertz. We used minimum frequency as 100 Hertz and maximum frequency

as 500 Hertz in order to gather vocal melody in audio recordings. The third parameter is voicing tolerance which determines how many pitch contours are included in the final melody. The fourth parameter is minimum peak salience which contributes to analyze monophonic records by avoiding silent parts in audio turning into junk pitch contours. The third and fourth parameters were used as default values which are 0.2 and 0.0, respectively. In preprocessing, pitch values shorter than 0.25 second were removed in order not to extract wrong pitch values such as intermediate notes which do not exist in the scale.

5.3.2. Melody Leap

By using melody that is extracted from audio files, leaps in melody are detected and scored. Leaps are large intervals (difference in pitch) between two consecutive notes. Normally, minimum leap can be an interval greater than minor second (3 half steps); however, in this work intervals greater than major second (4 half steps) are considered. Moreover, only ascending leaps are included in this feature. Clearly, only leaps to higher notes are calculated. In order to calculate this feature, the average of intervals for each consecutive notes is found as half step in a melody sequence. For a sequence with N notes, when the interval between note i and note $i+1$ denoted as $I_{i,i+1}$, the average interval will be as follows:

$$\frac{\sum_{n=1}^{N-1} I_{i,i+1}}{N-1} \tag{5.2}$$

5.3.3. Melody Pitch Mean

Melody pitch shows the mean pitch value the melody is focused on. Pitches in melody sequence are assigned to numeric values according to its pitch value (e.g. 0 = C, 1 = C#/Db) and octave. For example, D3 pitch value can be converted to a scalar value by multiplying it by 12 (since there are 12 notes in an octave) and its octave

value 3 and adding pitch value 2 to it. For a song, pitch mean is calculated by taking the average of these scalar values of pitches. For a sequence with N notes, when the pitch value of a note i denoted as P_i, the mean pitch is calculated as follows:

$$\frac{\sum_{n=1}^{N} P_i}{N} \tag{5.3}$$

5.3.4. Consonance and Dissonance

Consonance and dissonance are important terms in music and also indicators of simultaneous or successive sounds. While consonant sounds are more pleasant and acceptable, dissonant sounds are more unpleasant and harsh and have unstable tone combination. It can also be explained with physics. The frequency ratios of sounds that have lower simple numbers are more consonant than those that are higher.

In this research, every pitch value in melody is associated to song key and their interval is assessed as consonant or dissonant. Perfect unison, fifth and fourth, minor and major third, minor and major sixth intervals are classified as consonant. Minor and major second, minor and major seventh, and tritone (diminished fifth or augmented fourth) intervals are classified as dissonant. For example, if song key is C and the pitch value is D, then the interval between C and D is major second. Therefore, this interval is considered as consonant. When the interval between the note i and song key denoted as IC_i, consonant and dissonant intervals can defined as follows:

$$IntervalClass = \begin{cases} Consonant, & \text{if } IC \in 0, 3, 4, 5, 7, 8, 9 \\ Dissonant, & \text{otherwise} \end{cases} \tag{5.4}$$

For each pitch value in a melody sequence, classes of these intervals are counted and it is divided by the total number of pitches in the sequence in order to find the average value.

6. EXPERIMENTS AND RESULTS

Several experiments are conducted on hit song prediction by using the Scikit-Learn library for implementing the machine learning models in this work.

First, the paper named "Dance Hit Song Prediction" by Herremans et al. has been replicated and explained with its results [18]. Next, it is decided that this method can be applied to rock hit songs. Therefore, experiment I is done by using four primary datasets different from those explained in Chapter 4. In Experiment II, both dance and rock hit songs datasets with chord features are used to predict hit and non-hit classes labeled by using peak positions of the songs. In Experiment III, the same datasets are used with experiment II; however hit and non-hit classes are labeled by using song popularity feature of the songs.

In the experiments, SVM with RBF kernel (SR), SVM with linear kernel (SL), SVM with polynomial kernel (SP), Logistic Regression (LR), Naive Bayes (NB), Decision Tree (DT), XGBoost (XG), and Random Forest (RF) are used as the classification methods.

6.1. Replication of Dance Hit Song Prediction

Hit songs between October 2009 and March 2013 are gathered from Official Charts Company website with their weekly positions. 730 unique songs are selected from previous data with their peak positions. By using Spotify API, various features are gathered for 697 of 730 songs from Official Charts Company data. These features are duration, tempo, time signature, mode, key, loudness, danceability, and the energy of song.

For this replication experiment, three datasets are generated from the song list by labeling songs as hits or non-hits by using their peak positions. These sets are listed on Table 6.1.

Table 6.1. Datasets used for replication

Dataset	**Hits**	**Non-hits**	**Size**
Dataset 1 (D1)	Top 10	Top 30-40	415
Dataset 2 (D2)	Top 10	Top 20-40	566
Dataset 3 (D3)	Top 20	Top 20-40	697

First of all, normalization is applied on features. There are two classes and different classification methods are used to predict whether a song will be hit or non-hit. Then, 10-fold cross validation technique is used to get accuracy scores for each dataset. Accuracy results are listed on Table 6.2.

Table 6.2. Dance hit songs prediction results with 10-fold cross validation

Classifier	**D1**	**D2**	**D3**
SR	0.63	0.58	0.60
SP	0.62	0.54	0.60
NB	0.63	0.54	0.57
LR	0.64*	0.58	0.59
DT	0.53	0.53	0.55
best score: *			

When replication results are compared with the results in the original paper, it is seen that the biggest difference between the results is around 10%. Therefore, different experiments will be conducted in the following sections. Logistic regression classifier obtains the best result with dataset 1 because of the margin between peak positions. Therefore, it can be concluded that margin between classes affects the accuracy of prediction as seen in dataset 1.

6.2. Experiment I. Initial Experiments on Rock Hit Song Prediction

Rock hit songs between July 2009 and April 2017 are gathered from Billboard's website and rock hit songs between September 1994 and April 2017 are gathered from Official Charts Company's (OCC) website with their weekly positions. From Billboard 1452 unique songs and from Official Charts Company 2057 unique songs are selected from previous data with their peak positions (minimum positions over all time).

By using Spotify API, these features are gathered for 1280 of 1452 songs from Billboard's data and 1289 of 2057 songs from Official Charts Company's data. The obtained features are duration, tempo, time signature, mode, key, loudness, danceability, and the energy of song.

Four datasets are generated from these two different groups of data. Dataset 1 and 2 are generated from Billboard data, whereas dataset 3 and 4 generated from OCC. The songs are labeled as hit or non-hit based on their peak positions, where a margin has been left while separating hit vs. non-hit songs.

Table 6.3. Datasets used for rock hit song prediction

Dataset	**Hits**	**Non-hits**	**Size**	**Source**
D1	Top 10	Top 30-40	978	Billboard
D2	Top 10	Top 20-40	1114	Billboard
D3	Top 10	Top 30-40	748	OCC
D4	Top 10	Top 20-40	1044	OCC

10 fold cross-validation is applied on all datasets. The accuracy scores are listed on Table 6.4.

In the hit song prediction experiments all of four individual classifiers, and their combination trained and validated using rock hit songs from two different sources; namely, Official Charts and Billboard, have shown a significant success in predicting whether a song would be a hit or not. Only in dataset 3 the classifiers have a low

Table 6.4. Rock hit song prediction results with 10-fold cross validation

Classifiers	**D1**	**D2**	**D3**	**D4**
SR	0.88*	0.88*	0.66	0.76
SP	0.88*	0.88*	0.66	0.76
NB	0.83	0.83	0.60	0.75
LR	0.88*	0.88*	0.65	0.76
DT	0.80	0.79	0.54	0.62
best score: *				

success rate, possibly due to the fact that the dataset is small.

The results show that SVM and logistic regression classifiers perform better on datasets gathered from Billboard; however, the margin does not affect the results on these datasets.

6.3. Experiment II. Hit Song Prediction using Labeling by Peak Positions

This section consists of three different experimental parts, namely prediction of hit dance songs, prediction of hit rock songs and prediction of mixed rock songs which are selected from both hit lists and Spotify playlists. In all of the experiments in this section, the songs are labeled by their peak positions.

6.3.1. Prediction of Hit Dance Songs

With dance songs gathered from OCC charts, three different datasets are created by using the peak positions of songs. While creating these datasets, the labeling criteria were defined based on the work in [18]. The aim is to observe the effects of changing the gap between hit and non-hit song groups. Since it is not clear which part of top 50 songs should be defined as hit, different combinations of labeling have been generated. The datasets are listed in Table 6.5.

Table 6.5. Datasets of dance songs created by hit song lists

Dataset	**Hits**	**Size**	**Non-hits**	**Size**	**Total Size**
DD1	Top 10	260	Top 30-40	155	415
DD2	Top 10	260	Top 20-40	306	566
DD3	Top 20	406	Top 21-40	290	696

Cross validation with 10-fold is applied on these three datasets and accuracy scores are calculated. For each dataset, basic features (BF) such as duration, tempo, time signature, mode, key, loudness, danceability, energy and chord progression features (CF) are used. Chord features are common progression and circle progression features. The accuracy scores are given in Table 6.6.

Table 6.6. Classification results of 10-fold CV

	DD1		**DD2**		**DD3**	
Classifier	BF	BF + CF	BF	BF + CF	BF	BF + CF
SR	0.629	0.607	0.578*	*0.583**	0.605*	0.603*
SL	0.636	*0.639**	0.578*	0.573	0.582	*0.583*
SP	0.624	*0.631*	0.544	0.539	0.592	0.586
LR	0.641*	0.631	0.574	*0.580*	0.582	0.578
NB	0.627	0.568	0.539	*0.564*	0.561	0.552
DT	0.537	*0.583*	0.532	0.530	0.546	0.546
best score: *, increase in score with CF: italic						

SVM with RBF kernel performed better than the other classifiers in most cases. The best results are obtained on the DD1 dataset, since while labeling songs as hit or not, a margin between the two classes provides better separation of hit and non-hits. The new features affect the accuracy scores positively for half of the classification methods over the DD1 and DD2 data sets.

6.3.2. Prediction of Hit Rock Songs

Rock songs gathered from Billboard charts have been divided into three datasets in terms of their peak positions as shown in Table 6.7. This chart includes only songs ranked at the top 50. The training set which consists of songs released between 2009 and 2015 is nearly 73% of the dataset. The rest of the dataset, which consists of songs released between 2016 and 2017, is divided into two equal sized parts that are used as validation and test sets, respectively.

Table 6.7. Datasets of rock songs created by hit song lists

Dataset	**Hits**	**Size**	**Non-hits**	**Size**	**Total Size**
RD1	Top 10	293	Top 30-50	710	1003
RD2	Top 10	293	Top 20-50	1083	1376
RD3	Top 20	631	Top 21-50	1050	1681

Using only hit rock songs, combination of basic features, and two chord features are used to classify a song as hit or not. Since rock songs datasets are unbalanced, F_1 score metric was chosen to report classification results. The obtained F_1 scores are shown in Table 6.8 for three datasets.

Table 6.8. Classification results of rock songs from hit lists

	RD1		**RD2**		**RD3**	
Classifier	BF	BF + CF	BF	BF + CF	BF	BF + CF
SR	0.342	0.267	0.263	*0.337**	0.469	0.398
SL	0.344	*0.413**	0.057	*0.243*	0.486	0.400
SP	0.233	*0.333*	0.298	0.289	0.398	0.305
LR	0.344	0.333	0.286	0.279	0.498*	*0.505**
NB	0.361*	0.341	0.206	*0.220*	0.355	*0.375*
DT	0.306	*0.337*	0.331*	0.317	0.394	*0.413*
RF	0.319	0.281	0.103	*0.164*	0.409	0.392
best score: *, increase in score with CF: italic						

The results obtained on the RD3 dataset are better, since it is more balanced compared to the other datasets and the total size of the dataset is also larger than them. The results show that the chord based features result in classification performance improvement for at least half of the algorithms over the three datasets.

6.3.3. Prediction of Mixed Rock Songs

Another experiment is performed by using both hit song lists from Billboard charts and non-hit songs gathered from Spotify playlists. The datasets are generated as shown in Table 6.9. Songs from Spotify playlists are denoted by SPFY.

The training set, which consists of songs released between 2009-2016, is nearly 70% for all datasets. The rest of the dataset is allocated for validation and test set by dividing into two pieces.

Table 6.9. Datasets of rock songs created by hit song lists and Spotify playlists

Dataset	**Hits**	**Size**	**Non-hits**	**Size**	**Total Size**
MD1	Top 10	293	Top 30-50 + SPFY	2078	2371
MD2	Top 20	631	Top 30-50 + SPFY	2078	2709
MD3	Top 20	631	Top 21-50 + SPFY	2418	3049
MD4	Top 50	1681	SPFY	1368	3049

Table 6.10 shows the F_1 scores of the classification algorithms obtained over the datasets that contain rock songs from the hit song lists as well as non-hit songs from Spotify playlists. These datasets are more realistic compared to the ones that contain only songs from the top 40 or top 50 of the hit listings. In real life, there are many songs that are not ranked at all in such lists. As the results demonstrate, the algorithms obtain much lower classification F_1 scores on the more realistic datasets that include a small number of hit songs and a larger number of non-hit songs compiled from the lower ranked songs in the hit lists and the songs in Spotify playlists that had not entered such lists at all. The chord based features result in better performance for over half of the algorithms for the three datasets.

Table 6.10. Classification results of rock songs from hit listings and Spotify playlists

	MD1		**MD2**		**MD3**	
Classifier	BF	BF + CF	BF	BF + CF	BF	BF + CF
SR	0.088	0.086	0.186	0.185	0.154	*0.171*
SL	0.170*	0.104	0.191	*0.193**	0.183	*0.185**
SP	0.095	*0.132*	0.182	*0.185*	0.188*	0.161
LR	0.112	0.094	0.188	0.186	0.182	*0.173*
NB	0.147	*0.157**	0.119	*0.180*	0.048	*0.080*
DT	0.053	*0.107*	0.246*	0.164	0.139	*0.140*
RF	0.097	0.074	0.224	0.115	0.127	*0.145*
best score: *, increase in score with CF: italic						

Table 6.11 shows the individual effect of the two types of chord based features over the MD4 dataset. CoF indicates common chord progressions, and CiF indicates circle progressions feature.

Table 6.11. Detailed classification results of MD4 dataset

Classifier	**BF**	**BF + CoF**	**BF + CiF**	**BF + CoF + CiF**
SR	0.435	*0.441*	*0.465**	*0.444*
SL	0.391	*0.407*	*0.436*	0.356
SP	0.430	0.409	0.430	*0.451*
LR	0.464	0.442	0.461	0.462
NB	0.386	0.384	0.384	*0.401*
DT	0.468*	0.395	0.413	0.394
RF	0.459	*0.479**	0.439	*0.479**
best score: *, increase in score with CF: italic				

In MD4 dataset, tree based machine learning methods performed better than the others. The common chord progression feature increased the performance for half of the methods. When using both chord features, F_1 scores are increased for more than half of the classification algorithms.

6.4. Experiment III. Hit Song Prediction using Labeling by Song Popularity

In all experiments in this section, songs are labeled as hit or non-hit by their song popularity feature gathered from Spotify. Experiments are conducted on hit dance and rock songs. Moreover, statistical significance tests are conducted for the best feature combinations.

6.4.1. Prediction of Hit Dance Songs

An experiment is carried out on dance songs gathered from OCC charts. The aim is to observe whether there is a correlation between features and song popularity on Spotify. The test set consists of songs released in 2012 or later. On the other hand, training set consists of songs released before 2012. Songs are labeled based on their "song popularity" scores ranging between 0 and 100. For class distinction, the song popularity distinction point is set to 50. Songs with song popularity equal to 50 are removed (14 songs). Songs with song popularity less than 50 are labeled as non-hit, more than 50 are labeled as hit as shown in Table 6.12.

Table 6.12. Datasets of hit dance songs

	Hits ($>$ 50)	**Non-hits ($<$ 50)**	**Total Size**
Training Set ($<$ 2012)	109	336	445
Test Set ($\geq$ 2012)	42	194	236
All Sets	151	530	681

For each dataset, basic features (BF), chord progression features (CF), melody leap (ML), resolution (R), mean pitch (MP), consonance (C) and dissonance (D) features are used. To evaluate these features, 50 repetitions of 10-fold cross validation are done over the training set and means of F_1 scores are given in Table 6.13. If a classifier classified all songs as hit then the accuracy would be 0.245 and F_1 score would be 0.394. If a classifier classified all songs as non-hit, then the accuracy would be 0.760 and F_1 score would be 0.864.

Before the experiments the hypermeters of the classifiers are tuned by using grid search algorithm in the Scikit-Learn library. For SVM with RBF kernel, C is 100 and gamma is 10^{-4}. For XGBoost maximum depth is 5, minimum child weight is 5 and colsample bytree is 1. For random forest, 200 estimators are used with maximum depth 1 and maximum features 8, and the minimum sample split is 1.

Table 6.13. Classification results of 50 runs 10-fold CV for hit dance songs

Clsf.	BF	BF+CF	BF+ML	BF+R	BF+MP	BF+C	BF+D	BF+All
SR	0.513*	*0.518**	0.511*	0.511*	*0.517**	0.505	0.506	*0.521**
XG	0.331	*0.345*	*0.339*	*0.333*	0.322	*0.341*	*0.340*	*0.343*
RF	0.506	0.494	*0.507*	*0.507*	*0.507*	*0.507**	*0.507**	0.493
best score: *, increase in score with new features: italic								

Support vector machines with radial basis function kernel performed better than the other classifiers in most cases. New features generally affect the F_1 scores positively with all the classifiers. Since SVM performed better among the classifiers, the best combination of features is investigated using SVM classifier among all the combinations of features. As a result of this investigation, it is reached that the best feature combination consists of basic features, melody leap, mean pitch and dissonance. Detailed results for basic features, basic features with the best combination, and all features are shown in Table 6.14.

Table 6.14. Detailed classification results of 50 runs 10-fold CV for hit dance songs

Features	**Accuracy**	**Precision**	**Recall**	**F_1 Score**
BF	0.675	0.408	0.721	0.513
BF + Best Combination	*0.686**	*0.422**	*0.755**	*0.533**
BF + All Features	*0.679*	*0.414*	*0.736*	*0.521*
best score: *, increase in score with new features: italic				

Cross validation results shows that features have a notable effect on the results. Especially, using mean pitch, melody leap, and dissonance features together with basic features result in a 2% increase in F_1 score. It is observed that dissonance and

consonance features, used with other features, have a negative impact on F_1 score.

6.4.1.1. Significance Test for the Best Combination of Features. In order to evaluate the significance of the results statistically, hypothesis testing must be carried out on two related variables such as F_1 score using only basic features and F_1 score using basic features and the best combination of features. Since we have 500 samples for each score, paired samples t-test is decided to be performed. In this test, the null hypothesis is that the mean difference between the two F_1 scores is equal to 0. On the other hand, alternative hypothesis is F_1 score with basic features and best combination of features is greater than the other F_1 score.

Some assumptions should be examined first, in order for the test to be reliable. The dependent variable which is the difference between two conditions is continuous. Moreover, the dependent variable should be approximately normally distributed. Differences of F_1 scores are nearly normally distributed as shown in the Figures 6.1 and 6.2.

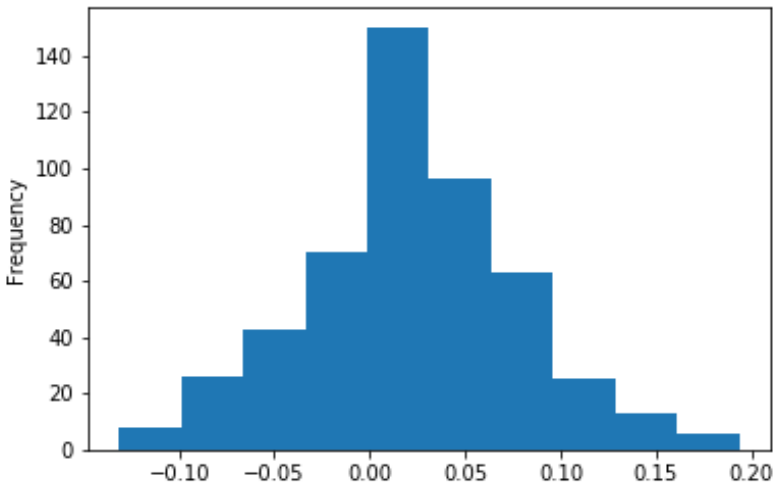

Figure 6.1. Dance Songs F1 Difference Histogram

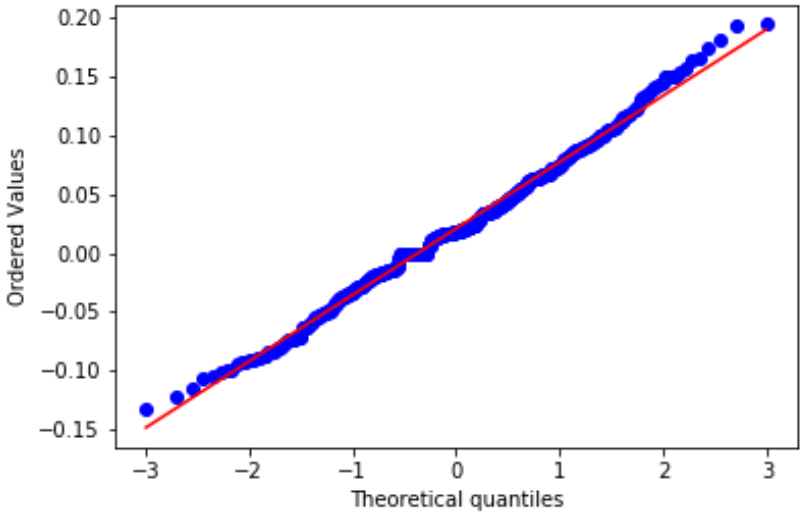

Figure 6.2. Dance Songs F1 Difference Q-Q Plot

When the box plots of the F_1 scores are considered for both conditions, there are some significant outliers in the dependent variable as shown in Figure 6.3.

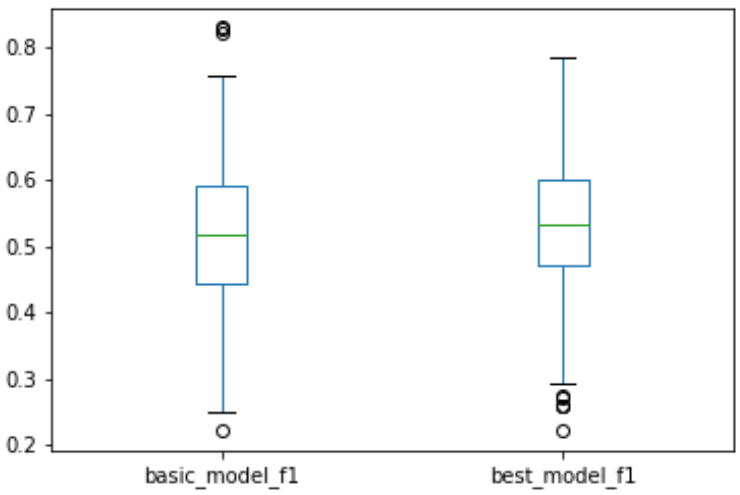

Figure 6.3. Box Plot of Dance Songs F_1 Scores

Since one assumption is violated, in addition to the paired samples t-test, it is decided to perform Wilcoxon Sign-Ranked test on these two conditions. Scipy library is used to conduct these experiments. For paired samples t-test p-value is found to be 2.096×10^{-15} Since it is less than 0.05, the null hypothesis can be rejected in support

of the alternative. Also, when Wilcoxon Sign-Ranked test is done, the p-value is found to be 1.455×10^{-14} It also shows that null hypothesis can be rejected, since it is less than 0.05. Therefore, it is concluded that the results are statistically significant.

6.4.1.2. Evaluation on the test set. So far the test set is not used. When basic features, best features and all features are trained with the songs released before 2012 and tested with songs released since 2012 using SVM with RBF kernel, F_1 scores are listed in Table 6.15.

Table 6.15. Classification results on the Dance Songs test set

Features	**Accuracy**	**Precision**	**Recall**	**F1 Score**
BF	0.737*	0.372*	0.690*	0.483*
BF + Best Combination	0.703	0.333	0.667	0.444
BF + All Features	0.716	0.342	0.643	0.446
best score: *				

The results show that new features generally have a negative effect on the F_1 scores on this test set. This may be due to the lack of suitable features for this test set.

Since hit songs are much less than non-hit songs, in the following set of experiments, classification is done by considering only top k songs in terms of the confidence scores of the predictions and song popularity, respectively. The songs are ordered by the probability of being a hit song given by the classifier and songs in top 1, top 5, top 10 and top 15 in this list are classified by using each feature, the accuracy scores are listed in Table 6.16

In top 10 and top 15 experiments, addition of chord progression and mean pitch features to the basic features one by one causes the classifier to predict the songs better. When the songs are ordered by the song popularity feature and the accuracy scores for top k songs are listed in Table 6.17.

Table 6.16. Classification results of top k dance songs sorted by confidence

	BF	BF+CF	BF+ML	BF+R	BF+MP	BF+C	BF+D	BF+All
Top 1	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Top 5	0.800	0.800	0.800	0.800	0.800	0.800	0.800	0.600
Top 10	0.500	0.500	0.500	0.500	*0.600*	0.500	0.500	0.400
Top 15	0.400	*0.467*	0.400	0.400	*0.533*	0.400	0.400	*0.467*
increase in score with new features: italic								

Table 6.17. Classification results of top k dance songs sorted by song popularity

	BF	BF+CF	BF+ML	BF+R	BF+MP	BF+C	BF+D	BF+All
Top 1	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Top 5	1.000	1.000	1.000	0.800	0.800	1.000	0.800	1.000
Top 10	0.900	0.900	0.800	0.800	0.800	0.900	0.800	0.800
Top 15	0.800	0.800	0.733	0.733	0.733	0.800	0.733	0.733

It can be stated that the classifier generally predicts mega hit songs well. Especially, for top 1 and top 5 songs, in more than half of the experiments the prediction takes place without error.

When the new features are examined, dissonance, melody leap and mean pitch features maximize F_1 scores together. Dissonance is effective because the music is ordinary and stable without dissonance. Therefore, it is important to use dissonant pitches in melody. Moreover, ascending leaps are also significant and necessary in melody in order to make a strong impact on the listeners. When the pitch is suddenly increased, this experience creates excitement over them. Dance songs have this kind of melodies. Furthermore, high average of pitch values in melody (mean pitch) are fascinating to listeners because high pitch means high frequency, high frequency means high energy. Dance songs should be energetic to be popular since the purpose of these songs is to make people dance. Therefore, it can be observed that melody-based features are more important than chord-based features considering dance songs. It can be concluded that melody is more significant to make dance songs popular due to the

frequent use of keyboard and synthesizer and prominence of them in dance songs.

6.4.2. Prediction of Hit Rock Songs

An experiment is performed by using both hit lists from Billboard charts and non-hit songs gathered from Spotify playlists. This dataset consists of features of 3049 songs. The test set consists of songs from 2017 and training set consists of songs released before 2017. Songs are labeled based on their "song popularity" score ranging between 0 and 100. For class distinction, the song popularity distinction point is set to 50. Songs with song popularity equal to 50 are removed (78 songs). Songs with song popularity less than 50 are labeled as non-hit, more than 50 are labeled as hit as shown in Table 6.18.

Table 6.18. Dataset of hit rock songs

	Hits ($>$ 50)	**Non-hits ($<$ 50)**	**Total Size**
Training Set ($<$ 2017)	971	1184	2155
Test Set ($\geq$ 2017)	280	536	816
All Sets	1251	1720	2971

Using only hit rock songs, the basic features (BF) as well as the basic features, chord progression features (CF), melody leap (ML), resolution (R), mean pitch (MP), consonance (C) and dissonance (D) features are used. 50 repetitions of 10-fold cross validation are done on the training set in order to evaluate these chord and melody features. Means of F_1 scores are given in Table 6.19. If a classifier classified all songs as hit, then accuracy would be 0.451 and F_1 score would be 0.622. If a classifier classified all songs as non-hit, then accuracy would be 0.550 and F_1 score would be 0.710. Our best results are greater than both baseline scores.

Before the experiments the hypermeters of the classifiers are tuned by using grid search algorithm in the Scikit-Learn library. For SVM with RBF kernel, C is 100 and gamma is 10^{-3}. For XGBoost maximum depth is 3, minimum child weight is 2 and colsample bytree is 0.65. For random forest, 200 estimators are used with maximum

depth 7 and maximum feature 4, and minimum sample split is 2.

Table 6.19. Classification results of 50 runs 10-fold CV for hit rock songs

Clsf.	BF	BF+CF	BF+ML	BF+R	BF+MP	BF+C	BF+D	BF+All
SR	0.700	0.700	0.698	0.700	0.698	0.696	0.696	0.697
XG	0.699	*0.701*	0.696	0.696	0.697	0.697	0.697	*0.701*
RF	0.717*	*0.723**	*0.719**	*0.718**	*0.718**	*0.718**	*0.718**	*0.721**
best score: *, increase in score with new features: italic								

Random forest performed better than the other classifiers in most cases. Chord progression features obtain the most successful scores. The new features generally affect the F_1 scores positively with random forest classifier. Since random forest performed better compared to other classifiers, the best combination of features is investigated among all the combinations of features using random forest classifier. It is found that the best feature combination consists of basic features, chord progression, resolution and melody leap. Detailed results for basic features, basic features with the best combination, and all features are shown in Table 6.20.

Table 6.20. Detailed classification results of 50 runs 10-fold CV for hit rock songs

Features	**Accuracy**	**Precision**	**Recall**	**F1 Score**
BF	0.712	0.643	0.813	0.717
BF + Best Combination	*0.717**	*0.645**	*0.827**	*0.724**
BF + All Features	*0.714*	0.642	*0.825*	*0.721*
best score: *, increase in score with new features: italic				

When the results are examined, it is seen that basic features with the best combination of features performed well by using the random forest classifier. Also, it is seen that F_1 score reached with all features is lower than the F_1 score of the features with the best combination.

6.4.2.1. Significance Test for the Best Combination of Features. As the previous section on dance songs, significance of the results statistically are carried out on two

related variables such as F_1 score using only basic features and F_1 score using basic features and the best combination of features.

When the assumptions are examined, the differences of F_1 scores are nearly normally distributed as shown in the Figures 6.4 and 6.5.

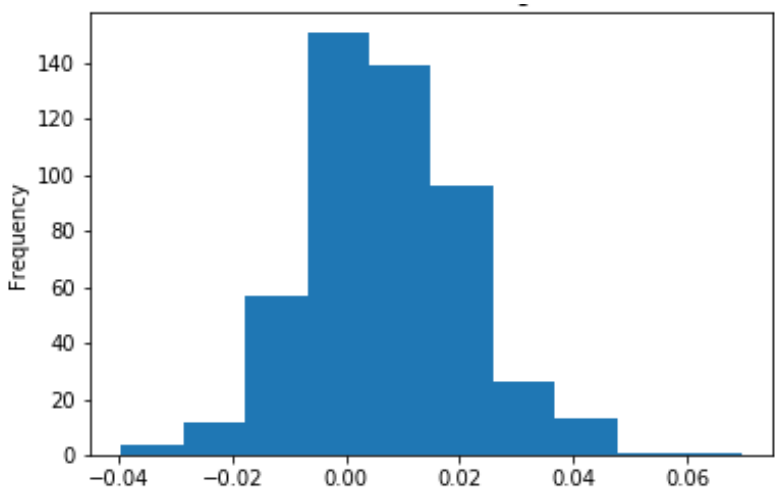

Figure 6.4. Rock Songs F1 Difference Histogram

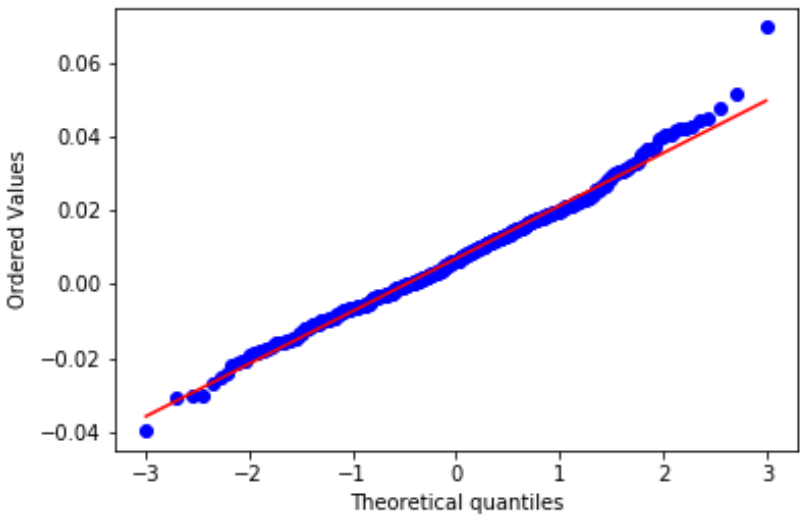

Figure 6.5. Rock Songs F1 Difference Q-Q Plot

Additionaly, there is not any significant outlier in the dependent variable as shown in Figure 6.6.

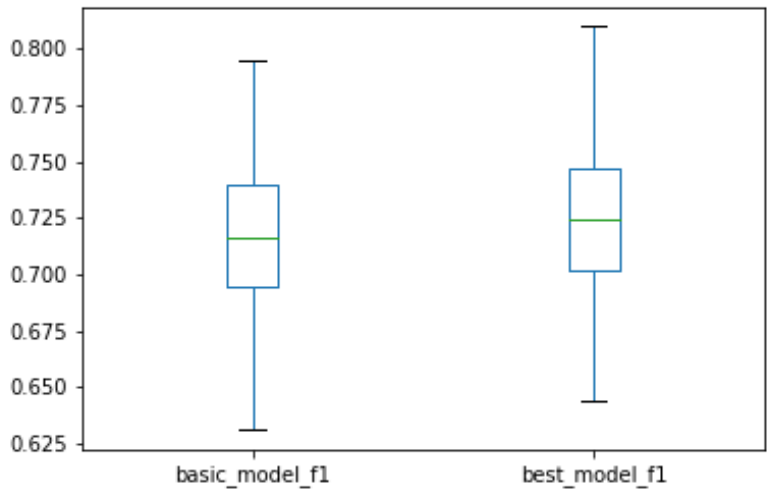

Figure 6.6. Box Plot of Rock Dataset F1 Scores

Since no assumption is violated, it is decided that paired samples t-test can be performed on these two conditions. By using Scipy library to conduct this experiment, p-value is found to be 6.397×10^{-25} . Since it is less than 0.05, the null hypothesis can be rejected in support of the alternative. Therefore, the results are statistically significant.

6.4.2.2. Evaluation on the test set. So far, the test set which has songs from 2017 year is not used. When the basic features, best features and all features are trained with the songs before 2017 and tested with songs from 2017 using random forest classifier, F_1 scores are listed in Table 6.21.

Table 6.21. Classification results on the Rock Songs Test Set

Features	**Accuracy**	**Precision**	**Recall**	**F_1 Score**
BF	0.777*	0.643*	0.786	0.707*
BF + Best Combination	0.776	0.641	*0.789*	0.707*
BF + All Features	0.775	0.638	*0.793**	0.707*
best score: *, increase in score with new features: italic				

New features have no effect on F_1 scores on the experiment with the test set. This may be due to the lack of suitable features for the test set of 2017 songs.

Since hit songs are much less than non-hit songs, the classification is made by considering only top k songs in terms of the confidence scores of the predictions and song popularity, respectively. The songs are ordered by the probability of being a hit song given by the classifier and songs in top 1, top 5, top 10 and top 15 in this list are classified by using each feature, the accuracy scores are listed in Table 6.22

Table 6.22. Classification results of top k rock songs sorted by confidence

	BF	BF+CF	BF+ML	BF+R	BF+MP	BF+C	BF+D	BF+All
Top 1	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Top 5	0.800	0.800	0.800	*1.000*	*1.000*	*1.000*	*1.000*	0.600
Top 10	0.800	*0.900*	0.800	0.800	*1.000*	*0.900*	0.800	0.700
Top 15	0.800	0.800	0.800	*0.867*	0.800	*0.933*	0.800	0.667
increase in score with new features: italic								

Looking at the 5 songs that the classifier predicts most likely, addition of resolution, mean pitch, consonance and dissonance features to the basic features causes the classifier to predict all the songs correctly, instead of four songs. When the songs are ordered by the song popularity feature and the accuracy scores for top k songs are listed in Table 6.23.

Table 6.23. Classification results of top k rock songs sorted by song popularity

	BF	BF+CF	BF+ML	BF+R	BF+MP	BF+C	BF+D	BF+All
Top 1	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Top 5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Top 10	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Top 15	1.000	1.000	1.000	1.000	1.000	0.950	1.000	0.950

It can be stated that the classifier generally predicts mega hit songs well. For example, when top 1 and the 5 songs with the best popularity are considered, classifi-

cation is done without error.

When the new features are examined, chord progression, melody leap and resolution features maximize F_1 scores together. It can be argued that all the chord-based features are important for hit rock songs in terms of popularity. Resolution and chord progression features are effective because by using the chords that are played by the electric guitar in rock music, listeners can easily be directed from one place to another. Moreover, ascending leaps are also effective for rock songs due to the same reasons as in dance songs. Therefore, it can be observed that chord-based features are generally more important than melody-based features considering rock songs. To conclude, chords transitions are very important because they can easily change the course of the song.

6.5. Feature Analysis of Hit Songs

6.5.1. Hit Dance Songs

Correlation matrix of important features for hit dance songs set are shown in Figure 6.7. When we examine the correlation between features, it can be stated that loudness and energy have a positive correlation about 0.63. When acousticness is high, energy and loudness are low due to the importance of drums. Also, it can be argued that mean pitch is positively correlated with energy and loudness features because when pitch is getting higher, energy of song may increase. Acousticness is also negatively correlated with mean pitch. Consonance and dissonance seem to be the opposite of each other as we expected. When we look at song popularity, the most important factor is artist popularity, acousticness and danceability have also a little impact on it. Furthermore, resolution is correlated with circular chord progressions since this progression includes a chord transition which is also stated in resolution feature.

6.5.2. Hit Rock Songs

Correlation matrix of important features for hit rock songs set are shown in Figure 6.8. It can be argued that loudness and energy has a positive correlation about 0.79. Acousticness has a negative correlation with energy and loudness because of the lack of drums. Also, it can be stated that mean pitch is positively correlated with energy and loudness features about 0.52 because when pitch is getting higher, energy of song may increase. Also, acousticness is negatively correlated with mean pitch. Consonance and dissonance seem to be the opposite of each other like in dance songs. Song popularity has a strong correlation with artist popularity, acousticness and danceability have also a little impact on it.

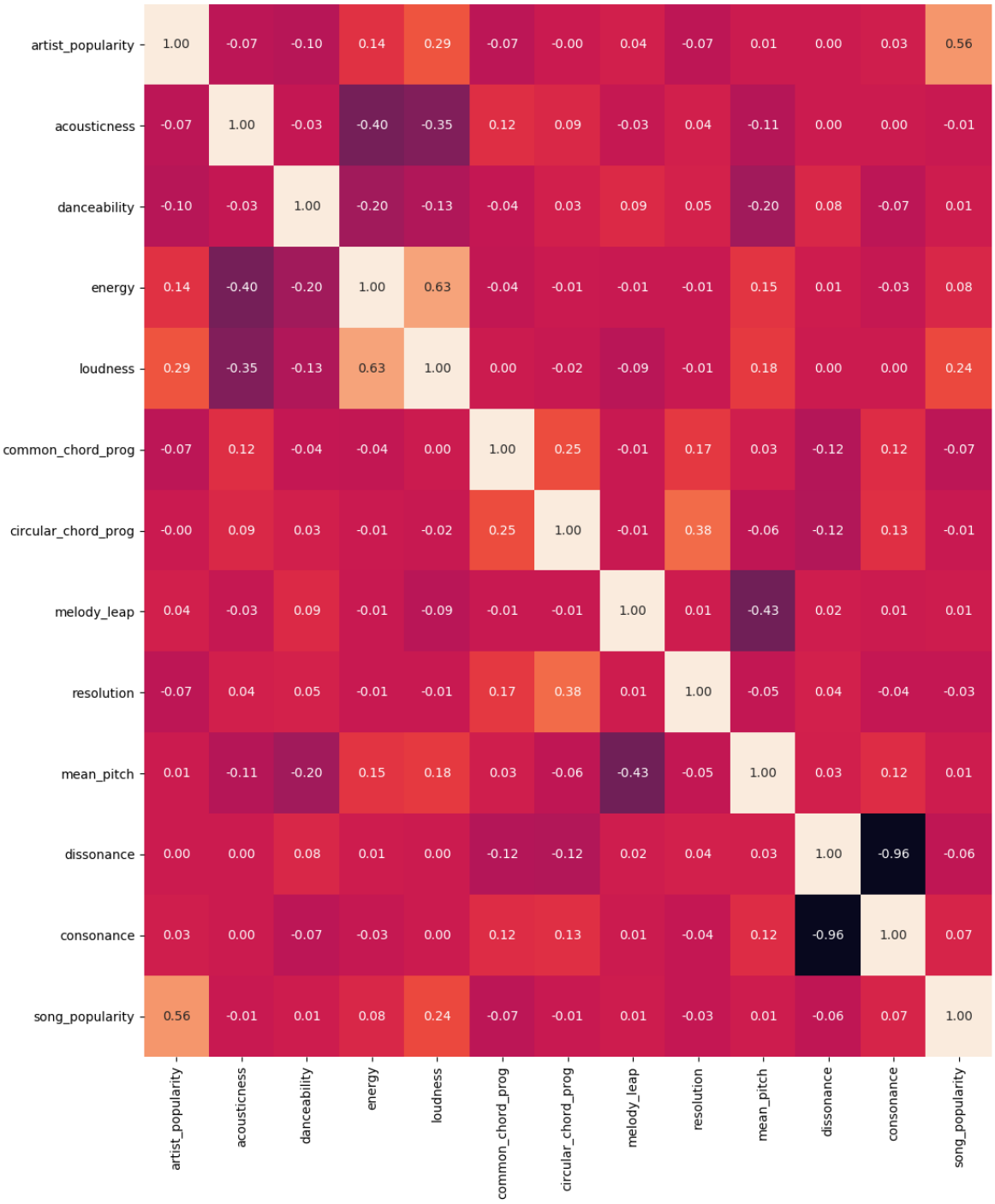

Figure 6.7. Hit Dance Songs Correlation Matrix

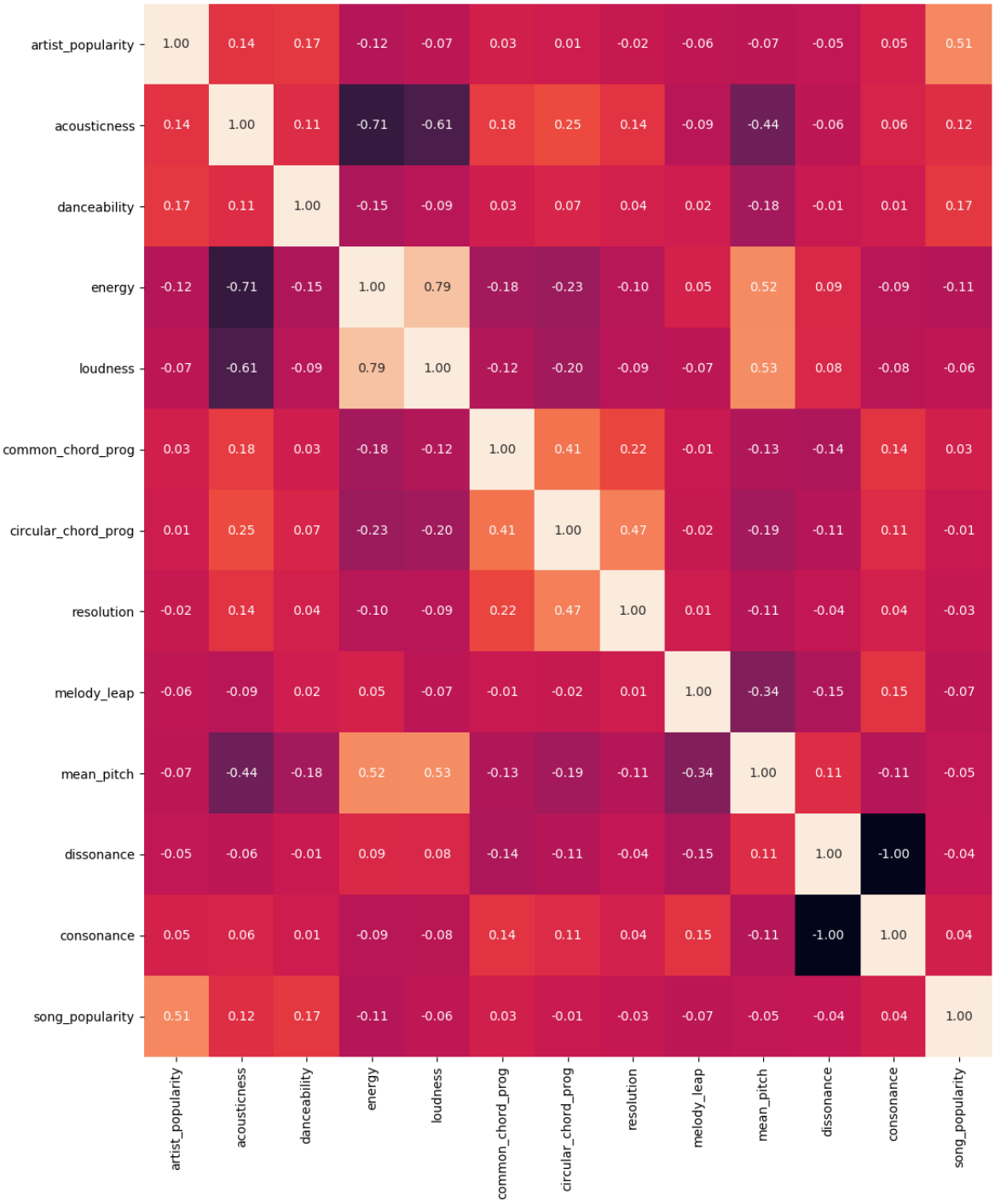

Figure 6.8. Hit Rock Songs Correlation Matrix

7. CONCLUSION

In this research, we addressed the task of hit song prediction. Besides the basic musical features that were used by most of the prior studies, we designed new chord based and melody based features based on music theory and utilized these features with supervised classification algorithms. We performed several experiments using the dance and rock songs which are selected from hit lists. In all experiments, it is examined that the new features may lead to improved hit song prediction performance. In order to perform evaluations in a scenario that models real life cases better, we compiled a new benchmark dataset for rock songs, which can be used in future studies.

In the classification that is done on the songs labeled by peak positions, it is observed that the margin between the two classes is an important factor. For example, in the experiment using the datasets containing both songs from hit song lists and songs from Spotify playlists that have not been listed in hit song lists, the results are lower when some less popular songs are labeled as non-hit compared to when all hits are labeled as hit and all songs not included in hit song lists are labeled as non-hit.

In the experiments performed using labeling with song popularity, the classification accuracy of dance songs is improved using the features that are designed based on melody when used with the Support Vector Machine algorithm, while it is argued that mostly chord based features are effective on the classification results of rock songs when used with the Random Forest algorithm. There are several reasons why melody-based features are effective in dance songs. The first reason may be that in dance songs the melody is in the foreground, since keyboard and synthesizer are frequently used in these songs. Also, when dance songs are examined, it can be seen that the chorus parts of these songs have no lyrics, but only the main melody of the song is played. On the other hand, the reason why chord-based features have an impact on rock songs may be due to the fact that use of electric guitar in rock songs can lead the listeners with chords.

Considering all the experiments which are conducted on, we propose chord-based features with basic features for predicting the success of rock songs, and melody-based features with basic features to predict the success of dance songs.

In the feature analysis section, it is observed that the popularity of a song is mostly affected by artist popularity. Therefore, it is implicated that the general success of the artists may affect their success on the new songs.

As a future work, we will design more features based on chords, melody and lyrics in order to increase the classification scores, especially in the real-world scenario. Also, deep neural networks can be used to reveal common chord progressions patterns that are used in songs.

REFERENCES

1. Hamid, M. N. A., *Classification Using Decision tree*, 2015, https://www.slideshare.net/knottisme/classification-using-decision-tree-53984611, accessed at June 2019.

2. Haltuf, M., *Support Vector Machines for Credit Scoring*, 2017, http://svm.michalhaltuf.cz/support-vector-machines, accessed at June 2019.

3. Verikas, A., E. Vaiciukynas, A. Gelzinis, J. Parker and M. C. Olsson, "Electromyographic Patterns during Golf Swing: Activation Sequence Profiling and Prediction of Shot Effectiveness", *Sensors*, Vol. 16, p. 592, 04 2016.

4. About, L., *Octave Naming and Pitch Notation*, 2019, https://www.liveabout.com/pitch-notation-and-octave-naming-2701389, accessed at June 2019.

5. PianoGuideLessons, *What is a chord? - Piano Chord Chart*, 2019, https://pianoguidelessons.com/what-is-a-chord, accessed at June 2019.

6. MusicTheoryAcademy, *Sequences*, 2019, https://www.musictheoryacademy.com/composing-music/sequences, accessed at June 2019.

7. 8Notes, *Circle Progressions (Music Theory Lesson)*, 2018, https://www.8notes.com/school/theory/circle_progressions.asp?show=all, accessed at June 2019.

8. Pachet, F. and C. Sony, "Hit song science", *Music Data Mining*, pp. 305–26, 2012.

9. Downie, J. S., "Music information retrieval", *Annual review of information science and technology*, Vol. 37, No. 1, pp. 295–340, 2003.

10. Kassler, M., "Toward musical information retrieval", *Perspectives of New Music*,

pp. 59–67, 1966.

11. Ghias, A., J. Logan, D. Chamberlin and B. C. Smith, "Query by humming: musical information retrieval in an audio database", *Proceedings of the third ACM international conference on Multimedia*, pp. 231–236, ACM, 1995.

12. Cosi, P., G. De Poli and G. Lauzzana, "Auditory modelling and self-organizing neural networks for timbre classification", *Journal of New Music Research*, Vol. 23, No. 1, pp. 71–98, 1994.

13. Tzanetakis, G. and P. Cook, "Musical genre classification of audio signals", *IEEE Transactions on speech and audio processing*, Vol. 10, No. 5, pp. 293–302, 2002.

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

15. Herremans, D., K. Sörensen and D. Martens, "Classification and generation of composer-specific music using global feature models and variable neighborhood search", *Computer Music Journal*, Vol. 39, No. 3, pp. 71–91, 2015.

16. Joder, C., S. Essid and G. Richard, "Temporal integration for audio classification with application to musical instrument classification", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 17, No. 1, pp. 174–186, 2009.

17. Whitman, B. and P. Smaragdis, "Combining Musical and Cultural Features for Intelligent Style Detection.", *ISMIR*, 2002.

18. Herremans, D., D. Martens and K. Sörensen, "Dance hit song prediction", *Journal of New Music Research*, Vol. 43, No. 3, pp. 291–302, 2014.

19. Wiering, F., "Can humans benefit from music information retrieval?", *International Workshop on Adaptive Multimedia Retrieval*, pp. 82–94, Springer, 2006.

20. Hadjeres, G. and F. Pachet, "DeepBach: a Steerable Model for Bach chorales generation", *CoRR*, Vol. abs/1612.01010, 2016.

21. Bertin-Mahieux, T., D. P. Ellis, B. Whitman and P. Lamere, "The Million Song Dataset", *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

22. Bischoff, K., C. S. Firan, M. Georgescu, W. Nejdl and R. Paiu, "Social knowledge-driven music hit prediction", *International Conference on Advanced Data Mining and Applications*, pp. 43–54, Springer, 2009.

23. Dhanaraj, R. and B. Logan, "Automatic Prediction of Hit Songs.", *ISMIR*, pp. 488–491, 2005.

24. Salganik, M. J., P. S. Dodds and D. J. Watts, "Experimental study of inequality and unpredictability in an artificial cultural market", *science*, Vol. 311, No. 5762, pp. 854–856, 2006.

25. Pachet, F. and P. Roy, "Hit Song Science Is Not Yet a Science.", *ISMIR*, pp. 355–360, 2008.

26. Ni, Y., R. Santos-Rodriguez, M. Mcvicar and T. De Bie, "Hit song science once again a science", *4th International Workshop on Machine Learning and Music*, Citeseer, 2011.

27. Boulanger-Lewandowski, N., Y. Bengio and P. Vincent, "Audio Chord Recognition with Recurrent Neural Networks.", *ISMIR*, pp. 335–340, Citeseer, 2013.

28. Mauch, M. and S. Dixon, "Approximate Note Transcription for the Improved Identification of Difficult Chords", *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, 2010.

29. Böck, S., F. Korzeniowski, J. Schlüter, F. Krebs and G. Widmer, "madmom: a

new Python Audio and Music Signal Processing Library", *Proceedings of the 24th ACM International Conference on Multimedia*, pp. 1174–1178, Amsterdam, The Netherlands, 10 2016.

30. Salamon, J. and E. Gómez, "Melody Extraction from Polyphonic Music Signals using Pitch Contour Characteristics", *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 20, No. 6, pp. 1759–1770, Aug. 2012.

31. Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830, 2011.

32. Brid, R. S., *Decision Trees - A simple way to visualize a decision*, 2018, `https://medium.com/greyatom/decision-trees-a-simple-way-to-visualize-a-decision-dc506a403aeb`, accessed at July 2019.

33. Androutsopoulos, I., J. Koutsias, K. Chandrinos, G. Paliouras and C. Spyropoulos, "An Evaluation of Naive Bayesian Anti-Spam Filtering", *CoRR*, Vol. cs.CL/0006013, 01 2000.

34. Cortes, C. and V. Vapnik, "Support-vector networks", *Machine Learning*, Vol. 20, No. 3, pp. 273–297, Sep 1995.

35. Ho, T. K., "Random Decision Forests", *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95, pp. 278–, IEEE Computer Society, Washington, DC, USA, 1995, `http://dl.acm.org/citation.cfm?id=844379.844681`.

36. Smolyakov, V., *Ensemble Learning to Improve Machine Learning Results*, 2017, `https://blog.statsbot.co/ensemble-learning-d1dcd548e936`, accessed

at June 2019.

37. Ho, T. K., "The Random Subspace Method for Constructing Decision Forests", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 20, No. 8, pp. 832–844, Aug. 1998, http://dx.doi.org/10.1109/34.709601.

38. Chen, T. and C. Guestrin, "XGBoost: A Scalable Tree Boosting System", *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, ACM, New York, NY, USA, 2016, http://doi.acm.org/10.1145/2939672.2939785.

39. DMLC, *XGBoost - ML winning solutions (incomplete list)*, 2017, https://github.com/dmlc/xgboost/tree/master/demo#machine-learning-challenge-winning-solutions, accessed at June 2019.

40. Sundaram, R. B., *Understanding the Math behind the XGBoost Algorithm*, 2018, https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost, accessed at June 2019.

41. Spotify, *Spotify Web API*, 2016, https://developer.spotify.com/web-api/, accessed at January 2018.

42. Etherington, D., *Spotify Acquires The Echo Nest, Gaining Control Of The Music DNA Company That Power Its Rivals*, 2014, http://techcrunch.com/2014/03/06/spotify-acquires-the-echo-nest, accessed at March 2018.

43. OfficialChartsCompany, *Official Singles Chart Top 100*, 2019, https://www.officialcharts.com/charts, accessed at January 2018.

44. Billboard, *Charts — Billboard*, 2019, https://www.billboard.com/charts, accessed at January 2018.

45. OfficialChartsCompany, *OCC Information Pack*, 2004, https://web.archive.org/web/20080413212134/http://www.theofficialcharts.com/docs/Information_Pack_June_2004.pdf, accessed at January 2018.

46. Godfrey, D., *Historical dictionary of American radio*, Greenwood Press, Westport, Conn, 1998, p. 45.

47. Chen, S. C. and J. Schlick, *PHP Simple HTML DOM Parser download*, 2019, https://sourceforge.net/projects/simplehtmldom/, accessed at January 2018.

48. MTG, *Essentia 2.1-5beta5-dev documentation*, 2019, https://essentia.upf.edu/documentation, accessed at January 2018.

49. Mauch, M. and C. Cannam, *Chordino and NNLS Chroma*, 2010, http://www.isophonics.net/nnls-chroma, accessed at January 2018.

50. Fabra, U. P., *State of the Art Audio Chord Estimation algorithms evaluation.*, 2017, https://musicinformationretrieval.wordpress.com/2017/03/06/state-of-the-art-audio-chord-estimation-algorithms-evaluation, accessed at June 2017.

51. Meredith, D., *Computational Music Analysis*, Springer Publishing Company, Incorporated, 1st edn., 2015, p. 36.

52. Müller, M., *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*, Springer Publishing Company, Incorporated, 1st edn., 2015, p. 250-253.

53. Benward, B., *1: Music: In Theory and Practice : Spiral*, McGraw-Hill College, 2003, p. 178.