

# Analysis and Genre Classification of Music in Digital Music Libraries

Jacob Grinstead  
Division of Science and Mathematics  
University of Minnesota, Morris  
Morris, Minnesota, USA 56267  
grins008@morris.umn.edu

## ABSTRACT

In what ways is digitally stored music classified by genre? What analysis process is used to allow for these classifications to be made? What affects the analysis of music? These questions are addressed in this research paper. This paper focuses on content-based classification algorithms, or in other words, algorithms that process the analysis results of the music's musical features like tempo or average loudness. The overarching goal of this research is to see just how effective algorithms have become in terms of classifying music. This problem is being addressed by the use of multiple algorithms that analyze the music's harmonics, patterns, stylistics (tempo, beats, etc. . . ), and more to classify them into different genres. These algorithms include Extreme Gradient Boosting, Extreme Random Trees, and a deep neural network. This paper highlights the importance of musical encoding in digital music and the effectiveness of different algorithms used for classifying the music. The XGBoost algorithm ended up being the best at classifying pieces of music by genre.

## Keywords

Encoded Music, Extreme Gradient Boosting, Extreme Random Trees, Music Analysis, Genre Classification, Neural Networks, XGBoost, ExtraTrees

## 1. INTRODUCTION

Digital music libraries are electronic libraries where music is stored. These range from music managers like Spotify or iTunes to independent music libraries like those that music organizations use to store their music. These digital libraries are continuing to rapidly increase in number and size, many of them offering hundreds of thousands of songs at least. As is the nature of music, all of these songs have their own distinct features that, when combined, can be used to classify them. The field of music analysis and classification is also continuously growing. New algorithms, implementations, and general ideas are replacing older, less effective versions as time goes on.

There are many challenges with using genre as a classification in music. These challenges come in both technical and practical forms. On the technical side, we see difficulties in

improving the algorithms that are used to analyze and classify. Fortunately, time has been kind to these challenges as continuous improvement is being made as time progresses. The practical difficulties we see are in the subjectivity of genres. Songs can be classified as many genres, and people also disagree on these genres that songs belong to. These practical challenges also relate to the importance of genre.

Using genre as a classification is important for many reasons. The subjectivity of genres allows for people to identify with them. For example, people culturally identify with jazz, rap, country, etc... People also are already accustomed to using genres when searching for music - evident in both online music stores as well as music collections in retail stores. Finally, people use genres more than any other criteria when searching for music [7]. All of these reasons show why it makes sense to continue classifying music with genres.

This paper provides an overview of musical terms and theory, *Essentia* (a library used for analyzing music), and machine learning. These are necessary to understand the following topics.

How music is encoded impacts its representation and thus impacts further analysis by algorithms. Section 3 covers the majority of the analysis portion of the paper. It focuses mainly on the basics of musical encoding, what the different encodings of the same piece of music look like and how they behave, and the results that these different encodings bring to light.

After analysis follows classification. This paper goes into a few content based algorithms that can be used to classify genres of songs in digital music libraries in section 4. It covers Extreme Gradient Boosting (XGBoost), Extreme Random Trees (ExtraTrees), and a deep neural network. Lastly, this paper provides conclusions about these algorithms and their importance in the field of musical analysis and classification.


















## 2. BACKGROUND

In order to completely understand musical encoding and the algorithms that are covered in the paper, it's important to know the basics of music and music theory. This background's first subsection will cover these basics, and will then go on to give a brief overview of *Essentia*, the library used for the analysis of the music. Finally, it will cover the basics of machine learning in preparation for the classification algorithms in section 4.

### 2.1 Music Theory

Figure 1 has the majority of the music theory terms needed

## Music Theory Basic Terms

 Quarter Note	$\frac{4}{4}$ Four-Four Time
 Half Note	$\frac{3}{4}$ Three-Four Time
 Whole Note	$\frac{2}{4}$ Two-Four Time
 Eighth Note	$\frac{6}{8}$ Six-Eight Time
 Beamed Eighth Notes	<b>C</b> Common Time
 Quarter Note	 Treble Clef
 Half Note	 Bass Clef
 Whole Note	 Sharp
 Eighth Note	 Flat
 Triplet	 Dotted Half
 Sixteenth Note	 Repeat sign

GROSSE *pointe*  
music academy

Figure 1: Some basic music theory terms [1]



Figure 2: An example of what sheet music looks like. It also shows a difference between a slur and a tie which are encoded differently, even though they look similar [9]

to understand how music is represented. Music is primarily made up of notes and rests, as seen in the top and bottom left sections of Figure 1, where notes represent different sounds, or pitches. Rests are where sound is not being made. Notes range from letters A through G, most of which can be sharp or flat - these symbols can be seen in the bottom middle section. Clefs make up one of the left-most elements on a musical staff, and relate to note range. Treble clefs represent higher pitches and base clefs represent lower pitches. The musical key of the piece is indicated by the amount of sharps or flats on the right side of the respective clef. Figure 2 has one flat on the right of the clefs, so we can get the major key of F. That's one example of a way an algorithm can detect information about a piece given only the music. Figure 2 also shows the subtle difference in appearance between a slur and a tie. The slur is shown to be extending downwards farther than the tie. Slurs can be used on two or more notes of the same or differing pitch (same or different notes) and make the slurred notes be played in a smooth and connected manner. Ties, on the other hand, can only be placed on two notes of the same pitch (same notes) and make the first note tied be played for the duration of the two tied notes. For example, in Figure 2, the Violin I part bows their instrument on each of the first two notes while the Violin II part bows once for the first quarter note and the following eighth note together.

## 2.2 Essentia

Essentia is an open-source library for music analysis [4], and is comprised of many different algorithms that allows it to do many different things. Some of these things are:

- Classify music based on computed audio features
- Detect the key of a song
- Estimate beat positions and tempo of a song
- Find if a song is happy, sad, aggressive, or relaxed based on its note progression
- Find the similarity between different songs

It is used in a study by Benjamin Murauer and Gunther Specht[8] to extract different musical features from songs. They use these features in their classification algorithms to help with the classification.

## 2.3 Machine Learning

Machine learning is a method of data analysis where computer systems rely on patterns and inference to do specific tasks rather than being given explicit instructions [10]. Algorithms that use machine learning build and update models based on sample data to make predictions without being explicitly programmed to do so [13]. There are many different branches of machine learning as well as ways that these algorithms learn. The one this paper will focus on is supervised learning.

Supervised learning includes inputs and desired outputs that both make up the training data, formed as a matrix. In this matrix of training data, there are vectors known as training examples, which are numerical representations of data. Through iterations, these algorithms learn the function that can be used to predict outputs based on the inputs of the training data. These outputs are labeled as desired, so if the machine gets it incorrect, someone can change it to the desired label and the program learns from that. This is one of the key features of supervised learning.

Relating to the material in this paper, the three classification algorithms covered in section 4 are machine learning algorithms at their core. Each of these algorithms use machine learning to build and predict mathematical models based on a training set of 25,000 songs, but does so in a different way. These ways are covered in their respective subsections. Representing these inputs and outputs from the training data is Table 1. It shows outputs from the pieces (used as inputs for training) as features extracted by Essentia from the training data.

## 3. MUSICAL ENCODING

Before seeing any algorithms that are used to analyze music, it is important to understand musical encoding and what effects it can have on analysis. There are two forms that music can take when encoded. The first way is in *image form*, otherwise known as sheet music - the music is scanned and uploaded to a digital music library or database. This is mainly used for sites that have sheet music on them like MuseScore or online sheet music stores, where the music is in sheet music form. Typically, these aren't optimal for analysis, as it's much more difficult to analyze an image than it is musical features. The second form is what this paper will focus on. It is a *content-based representation* - the staff, measures, notes, duration of notes, etc... [9]. This form takes

much longer to analyze than the image-based approach as it takes much less time to scan a piece of music rather than extracting features from it and analyzing them using a program like Essentia. However, content-based representations like this give rise to better analysis results, as we’ll see later in Section 4. For example, finding how many times a singular note appears or finding parts where a particular melody plays is easier and faster.

This second form is one that matters in most algorithms that analyze music. Until music classification technologies, like the algorithms covered in Section 4, improve, the current best way to encode music in this form is to use a music notation software like *MuseScore* or *Finale*. Programs like these encode the music while it’s being written by the composer in the software, and use what is known as a *What You See is What You Get* model. All of these programs encode differently, though - a piece written in MuseScore will be encoded differently than the same piece written in Finale. The music notation software used isn’t the only thing that matters when it comes to encoding. Encodings can be different based on how the composers create their music. For example, two composers may use ties differently in their corresponding compositions of the same piece, thus causing differences.

Unfortunately, since composers create their scores differently, and there is no way for them to indicate their intentions within the score nor is there a standard to follow, encodings will always differ. However, one way to combat this slightly is the MusicXML file. This file type can be retrieved from several different programs and therefore can be a useful go between for comparison. We’ll see one way it’s used in the following section.

### 3.1 Different Encodings of Music

To show that pieces of music that are visually the same get encoded differently, we can look at the research done by Néstor Nápoles, Gabriel Vigliensoni, and Ichiro Fujinaga[9]. They found the same piece (Beethoven’s *Op.18 No.1*) on three different platforms where each instance of that piece was generated by unique individuals using different software. One was taken from MuseScore’s community website, and using the MuseScore software, was exported from a .mscz file to the MusicXML file format. The second piece was taken from gutenber.org, and using the Finale software, was exported from a .MUS file to a MusicXML file. The final piece was taken from tes.com, and using the Sibelius software, was exported from a .SIB file to a MusicXML file.

With each of the encoded pieces, Nápoles, Vigliensoni, and Fujinaga used the toolkit music21 and VIS framework for their music analysis. In their analysis, they looked for discrepancies between the three pieces. They looked for something they call note/rest onsets. By this, they mean looking at two notes or rests in the same spot in each encoding of the piece. These onsets are said to be matching when the notes or rests are the same. They were able to tell discrepancies when these onsets weren’t matching in their comparisons. Figure 3 shows the discrepancies from the three comparisons of the encoded pieces. The vertical axis shows each instrument in the piece, and following from them horizontally, the black areas represent where both pieces being compared had matching onsets. On the other hand, the white areas show the discrepancies, where the onsets didn’t equal each other. It’s worth noting that the authors don’t discuss the

feature name	exemplary value
low level average loudness	0.938
low level melbands skewness mean	2.246
low level spectral flux median	0.112
rhythm bpm	83.583
...	...
danceability	1.101
tonal key	'E'
tonal chord	'major'

Table 1: Numerical features extracted from a song with Essentia [8]

strengths or weaknesses of using onset matching compared to other possible options for comparing encodings.

### 3.2 Results of the Different Encodings

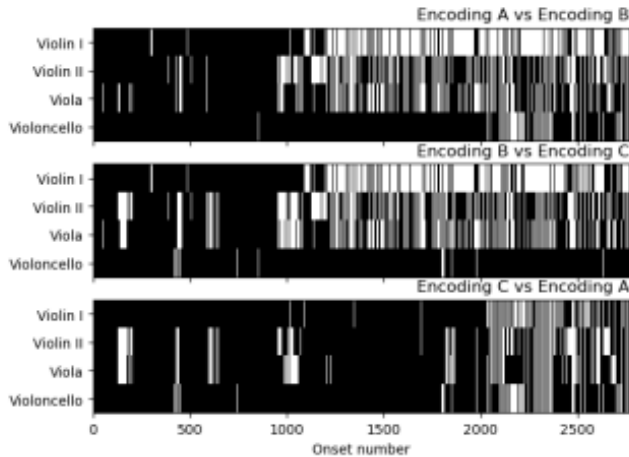
It’s unfortunate, but the use of "Encoding  $x$  vs Encoding  $y$ " shows the authors randomized their encodings so they couldn’t tell which encoding was for which piece. By doing this, we are unable to draw conclusions about which encoding provides the most consistency when compared with another. However, from the results shown in Figure 3, we can still make two observations. The first is that the same piece encoded in three different ways will have somewhat similar, but still different onsets. This tells us that depending on how a piece of music is encoded, the classification may be different since the encoding being analyzed is unique to that piece within the music software used.

The second comes in the form of a question. What causes the discrepancies that we see, exactly? The authors used the graph as a guide for finding the discrepancies in the encodings. They were able to narrow it down to two main factors: discrepancies caused by the music notation software (i.e. MuseScore) and those caused by human error (the composer of the new piece him/herself). From these two categories, they found discrepancies caused by incomplete measures, non-closed ties, using slurs instead of ties, use of repeated notes vs. musical symbol meaning to repeat the note, and more. Whether these issues were the fault of the software or the composer, they all caused the encoding to be symbolically different, thus causing the discrepancies. From these results, we can see that how pieces get encoded is important to the results we may get from subsequent analysis or classification.

## 4. ANALYSIS AND CONTENT BASED ALGORITHMS FOR CLASSIFICATION

After seeing how encodings of digital music can affect how they are represented, some algorithms that can be used to analyze these representations will be covered. The following three machine learning algorithms, XGBoost, ExtraTrees, and a deep neural network, were used in a study done by Benjamin Murauer and Günther Specht from Innsbruck University to classify genres of music [8]. In the study, 25,000 songs were used for training and 35,000 songs were used for the testing of their algorithms. Each algorithm used supervised learning to train on the 25,000 pieces of music. The majority of the songs came as standard mp3 files from a free musical data set of 100,000 songs, created for purposes like

## Encoding Matters



**Figure 3: Summary of the discrepancies between the three different encodings. The horizontal axis represents where notes or rests were the same and started in the same spot in the music (in black) and where different notes or the same notes that started at a different time in the music (in white) [9]**

this study, that they then extracted musical features from using *Essentia*. Table 2 shows the genres used in the training data set, but the artist, song title, and genre were taken out of the metadata for the testing data set. The majority of the tracks were 30 seconds long.

Instead of classifying the genre for each of the 35,000 songs in the testing data set, they gave the probability of a song being in a genre. For example, the chance of a song being rock and roll might be .9, pop at .06 and hip-hop at .04 [8]. To see how accurate their algorithms are at predicting genres, a predefined mean log loss equation was used:

$$L = -\frac{1}{N} \sum_{n=1}^N \sum_{c=1}^C y_{nc} \ln(p_{nc}) \quad (1)$$

In this equation,  $N$  is the number of songs (35,000),  $C$  is the number of distinct genres,  $y_{nc}$  is a binary label showing that the  $n^{th}$  sample belongs to a class  $c$ , and  $p_{nc}$  is the provided probability for that specific song.  $L$ , then, is the mean log loss score. A log loss score measures the performance of a classification model’s prediction accuracy. The closer this score is to 0, the more accurate the predictions are. Murauer and Specht matched the predicted genres with the actual genre, using Equation 1. An example they use in their paper is a song having the probabilities  $p(\text{Rock})=.9$ ,  $p(\text{Electronic})=.06$ ,  $p(\text{Hip-Hop})=.04$ , where rock is the selected predicted genre since its probability is highest for that piece of music [8]. Thus the  $p_{nc}$  for this piece would be the value for the Rock genre, which they don’t go into specifics over either. Figure 4 represents this well, where the vertical axis is the log loss score and the horizontal axis is the probability. For example, having a log loss of 1.5 is roughly a probability of .27 (27 percent chance of correctly predicting genre) or having a log loss of .5 is a probability closer to .6

genre	# of songs
Rock	7,103
Electronic	6,314
Experimental	2,251
Hip-Hop	2,201
Folk	1,519
Instrumental	1,350
Pop	1,186
International	1,018
Classical	619
Old-Time / Historic	510
Jazz	384
Country	178
Soul-RnB	154
Spoken	118
Blues	74
Easy Listening	21
<b>total</b>	<b>25,000</b>

**Table 2: Distribution of genres for the training data set [8]**

(60 percent chance of correctly guessing genre).

Using the *Essentia* library to analyze the music, they extracted numerical features from each song. A subset of these features can be seen in Table 1. Some of the features were categorical instead of numerical, take the tonal key for example, and so these categorical features were transformed into a form suitable for machine learning, known as a one-hot encoding (see Murauer and Specht’s study for further details)[8]. After the features were extracted in *Essentia* and approximately pre-processed, they were put through three classifier algorithms: XGBoost, ExtraTrees, and a Deep Neural Network.

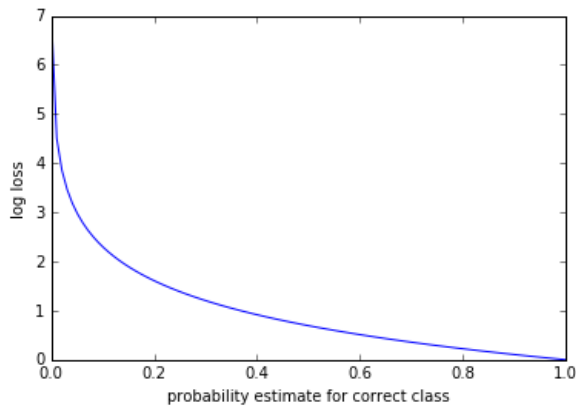
## 4.1 XGBoost Classifier

The XGBoost classifier is a scalable tree boosting system that is shown to be able to solve real-world scale problems with minimal resource usage [3]. It was developed with efficiency in mind and uses a gradient boosting algorithm for its classification [2]. This algorithm produces an ensemble of weak prediction models and generalizes them with a differentiable loss equation - like equation 1 [12].

### 4.1.1 Gradient Boosting

The purpose of gradient boosting algorithms is to support predictive modeling problems for classification. They do so by producing models based on decision trees that the algorithm creates. These models predict and improve upon errors of previous models to continuously better themselves, using machine learning. The algorithm used for XGBoost continuously produces models of different probabilities of a piece based on the decision trees that were formed for that piece. It does this to find the most likely probability.

The use of decision trees as their structure allows the algorithm to go from observations in the branches of the trees to conclusions in the leaves [11]. Gradient boosting also implements a gradient descent algorithm to minimize the loss between prediction models. This algorithm is used for



**Figure 4: Graph of log loss function. The vertical axis shows the log loss scores, and thus following a straight line right and then down to the horizontal axis once connecting with the function’s line, gives the probability related to the log loss score [6]**

finding the minimum of functions, which is what helps with minimizing the loss.

#### 4.1.2 XGBoost Results

As seen in Table 3, the mean loss log scores,  $L$ , for XGBoost resulted in a low of .82 and a high of .85. The parameter `n_estimator` represents how many decision trees were used and max depth is the maximum depth each tree could be. Thus, we can see that with fewer trees and a smaller depth, the XGBoost classifier was more efficient at predicting the correct genre of a piece at about .57.

## 4.2 ExtraTrees Classifier

The ExtraTrees classifier is a variant of the random forest classifier. It uses an extreme random trees algorithm for classification. Similar to the XGBoost classifier, ExtraTrees uses averaging of the decision trees from the extreme random tree algorithm to make its predictions of probability. By using extreme random trees, it differs from other tree-based classifiers in two ways [5]. The first is that it splits its nodes by choosing a cutting point completely at random. This means that each decision tree will have different nodes in different locations within the tree. The second difference is that it uses the whole learning sample rather than a subset of it. In other words, the ExtraTrees classifier used all 25,000 pieces in the training data set to learn from, where other popular tree-based algorithms would not.

#### 4.2.1 Extreme Random Trees

Like XGBoost, this algorithm builds an ensemble of decision trees as models. However, it randomly chooses where the splitting of the trees occurs. This random splitting gives the extreme random trees algorithm a lower variance than other tree-based algorithms like the random forest classifier. In clearer words, the chances of the algorithm assigning a different probability to a piece of music similar to a piece of music used in the training set are lower. It’s important to note, however, that this randomness increased its overall bias as well. This means that when it’s compared to other algorithms, the chance of some of its results being systematically prejudiced based on the machine learning process is

higher.

#### 4.2.2 ExtraTrees Results

As seen in table 3, the mean loss log scores,  $L$ , for ExtraTrees resulted in a .92 where the number of trees was either 1,000 or 2,000. Like XGBoost, the `n_estimator` parameter represents the number of trees used. We ignore the balanced ExtraTrees classifier as none of the other runs were balanced and it performed worse. A .92 mean log loss score isn’t great, coming out to be around a chance of .5 for correct classification.

## 4.3 Deep Neural Network Classifier

#### 4.3.1 Neural Networks

Neural networks are part of one branch of machine learning. They are a set of algorithms that are designed to recognize patterns [10]. These patterns are numerical in nature, being stored in vectors, after being translated from its real-world data. A common use of neural networks is to help with classification. In other words, they help to group different data inputs.

There are three general layers to a neural network, the input layer, hidden layer, and output layer. In terms of a musical classification example, the input layer takes in the numerical feature data from *Essentia*. The authors used one-hot encoders to turn the features like tonal key into numerical form. All of this input was then pushed through a hidden layer, where these numerical features were transformed into a specific output using activation functions. They were then pushed out to the output layer, where one last function is performed to normalize everything. This output is then used to calculate the probability of the piece of music belongs to a certain genre.

This paper [8] covers a deep neural network that was created by Murauer and Specht as one of their classification algorithms. The biggest difference between a neural network and a deep neural network is that deep neural networks have a greater depth. In other words, they have a greater number of hidden layers that provide better results for classification.

#### 4.3.2 Deep Neural Network

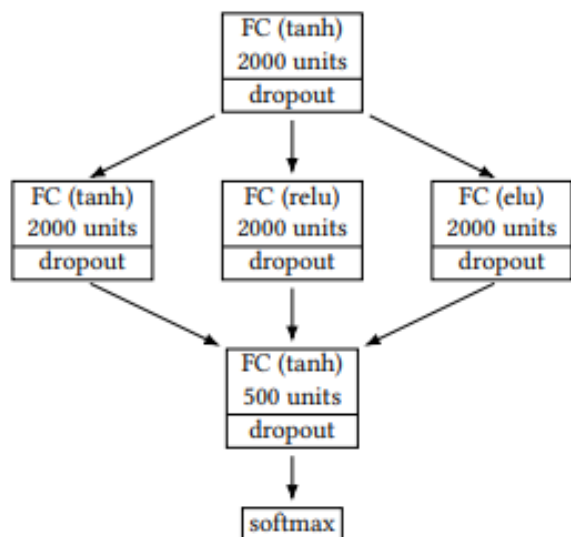
The deep neural network classifier that they designed had several layers with functions that calculated the different features extracted by *Essentia*. After moving between these layers, it went to the final layer, what they called the softmax layer, where everything is normalized, or, transformed into a range from 0 to 1. They then used these scores as the  $p_{nc}$  from the log loss function in Equation 1 to calculate the final genre probabilities belonging to each piece. The architecture of the deep neural network is shown in Figure 5.

#### 4.3.3 Deep Neural Network Results

In Table 3, the deep neural network classifier is labeled as *DNN*. We can see that the mean loss log score,  $L$ , for the deep neural network is 1.44. This means that it had a fairly low prediction accuracy.

## 5. CONCLUSIONS

Analyzing and classifying music in digital music libraries is not an easy task. Based on Murauer and Specht’s study [8], we can see from Table 3 the mean loss log scores,  $L$ , of the three classifier algorithms. We’ll ignore the CNN classifier,



**Figure 5: Architecture of the deep neural network.** FC denotes a fully connected layer with the different functions in parentheses [8]

classifier	parameters	$L$
XGBoost	n_estimators=1,000, max_depth=3	<b>0.82</b>
XGBoost	n_estimators=3,000, max_depth=5	0.85
ExtraTrees	n_estimators=1,000	0.92
ExtraTrees	n_estimators=2,000	0.92
ExtraTrees	n_estimators=2,000, balanced weights	0.96
CNN	*	1.65
DNN	*	1.44

**Table 3: Numerical features extracted from a song with Essentia [8]**

which is a convolutional neural network, because it isn't an algorithm that uses the content-based encodings. We see that XGBoost is better at classifying music than ExtraTrees, as it has a lower mean log loss score. ExtraTrees is then shown to be better than deep neural networks. In a field where neural networks are used heavily, this research says a lot about the future prospects of classifying songs in digital music libraries.

Based on the main study from Section 3 [9], we can infer that since pieces are interpreted and thus encoded differently, there could be different mean log loss scores if the pieces of music were analyzed with something other than Essentia. Whether they would be higher or lower is impossible to say. What we can say, however, is that the highest chance of getting a correct genre classification of around 60 percent is not ideal. Thus, there is plenty of room for improvement. Until further improvement is made, however, manual classification of genres will have to continue to be done to music in digital music libraries.

## Acknowledgments

A huge thank you to my advisor, Kristin Lamberty. Your insight, help, and patience was invaluable throughout the research and writing processes. Another thank you to Michelle

Adamo for her general critiques and help with organization and grammar. Finally, thanks to my alumni reader, Scott Steffes, for his great feedback. This paper wouldn't have been possible without the three of them.

## 6. REFERENCES

- [1] G. P. M. Academy. Music Theory. <https://www.gossepointmusicacademy.com/private-lessons/music-theory/>. Accessed: 2019-10-21.
- [2] J. Brownlee. A Gentle Introduction to XGBoost for Applied Machine Learning. <https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/>. Accessed: 2019-10-29.
- [3] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.
- [4] Essentia. Essentia open-source library and tools for audio and music analysis, description, and synthesis. <https://essentia.upf.edu/documentation/>. Accessed: 2019-10-21.
- [5] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, Apr 2006.
- [6] D. Maust. Log Loss is Greater Than 1. <https://stackoverflow.com/questions/35013822/log-loss-output-is-greater-than-1>. Accessed: 2019-10-29.
- [7] C. McKay and I. Fujinaga. Musical genre classification: Is it worth pursuing and how can it be improved? In *ISMIR*, 2006.
- [8] B. Murauer and G. Specht. Detecting music genre using extreme gradient boosting. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pages 1923–1927, Republic and Canton of Geneva, Switzerland, 2018. International World Wide Web Conferences Steering Committee.
- [9] N. Nápoles, G. Vigiensoni, and I. Fujinaga. Encoding matters. In *Proceedings of the 5th International Conference on Digital Libraries for Musicology*, DLFM '18, pages 69–73, New York, NY, USA, 2018. ACM.
- [10] Skymind. A Beginner's Guide to Neural Networks and Deep Learning. <https://skymind.ai/wiki/neural-network>. Accessed: 2019-10-21.
- [11] Wikipedia. Decision Tree Learning. [https://en.wikipedia.org/wiki/Decision\\_tree\\_learning](https://en.wikipedia.org/wiki/Decision_tree_learning). Accessed: 2019-10-29.
- [12] Wikipedia. Gradient Boosting. [https://en.wikipedia.org/wiki/Gradient\\_boosting](https://en.wikipedia.org/wiki/Gradient_boosting). Accessed: 2019-10-21.
- [13] Wikipedia. Machine Learning. [https://en.wikipedia.org/wiki/Machine\\_learning](https://en.wikipedia.org/wiki/Machine_learning). Accessed: 2019-10-29.