# Automatic Classification of Latin Music

## Some Experiments on Musical Genre Classification

Miguel Alexandre Gaspar Lopes

Dissertation oriented by Fabien Gouyon
Developed at INESC Porto

Universidade do Porto
Faculdade de Engenharia
FEUP

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia Electrotécnica e de Computadores
Rua Roberto Frias, s/n, 4200-465 Porto, Portugal

**July 2009**

# Automatic Classification of Latin Music

## Some Experiments on Musical Genre Classification

Miguel Alexandre Gaspar Lopes

Estudante Finalista do Mestrado em Engenharia Electrotécnica e Computadores pela Faculdade de Engenharia da Universidade do Porto

Dissertação realizada no 2º semestre do 5º ano do Mestrado Integrado em Engenharia Electrotécnica e Computadores (Major Telecomunicações) da Faculdade de Engenharia da Universidade do Porto

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia Electrotécnica e de Computadores
Rua Roberto Frias, s/n, 4200-465 Porto, Portugal

July 2009

# Summary

The automatic information retrieval from music signals is a developing area with several possible applications. One of these applications is the automatic classification of music into genres. In this project, a description of the current methodology to perform such a task is done, with references to the state of the art in this area. Music is composed of several aspects such as the melody, the rhythm or the timbre. Timbre relates to the overall quality of sound. In the work developed in this project, an analysis of timbre and its ability to represent musical genres is done. Experiments are performed that tackle the problem of the automatic classification of musical genres using the following general procedure: characteristics related to timbre are extracted from music signals; these values are used to create mathematical models representing genres; finally these models are used to classify unclassified musical samples and their classification performance is evaluated.

There are several ways to perform the procedure described above: related to which characteristics from the musical signals are extracted or which algorithms (designated by classifiers) are used to create the genre models.

In this project there is a comparison of the classification performance obtained with several classification algorithms and with the use of different characteristics of the musical datasets being classified. One dataset characteristic that was evaluated was the use of an artist filter and its influence in the classification performance. An artist filter is a mechanism that prevents that songs from a certain artist are used to test the genre models created with the songs from that same artist (among other songs from other artists, obviously). It was concluded that the use of the artist filter reduces the classification performance, and should be used in classification experiments of this kind, so that the results obtained are as representative as possible.

Most experiments were carried out using the Latin Music Database, a musical collection composed of 10 different Latin music genres. Experiments were also made using a collection of 10 more general musical genres (such as “classical” or “rock”).

An assessment of the adequacy of the analysis of timbre to identify the genres of the Latin Music Database is done, along with a computation of the timbral similarities between genres.

# Abstract

*Timbre is a quality of music that might be hard to define: it is related to the overall sound of a musical piece and is defined by its instruments and production characteristics – it is the quality that distinguishes the sound of two different instruments playing the same musical note, in the same volume. The musical timbre is certainly a defining quality in the characterization of a musical genre. The automatic classification of musical signals into genres through the analysis of timbre is what was attempted in the development of this work. The majority of the realized experiments were done with the Latin Music Database, composed of 10 exotic musical genres, some of them likely to be unknown to the reader. An analysis of the timbre characteristics of these genres was done, with results interesting to look at and to compare with a personal evaluation of these musical genres. Experiments with different classification algorithms were made and the different performances achieved compared. The influence in the classification performance of several factors was assessed: related to the size and nature of the datasets used to create and test the genre models.*

# Acknowledgements

*I would like to thank Fabien Gouyon, that oriented this dissertation and guided me through the development of this work in the most helpful and encouraging way, Luís Gustavo Martins for the explanations on Marsyas, Luís Paulo Reis for providing the template on which this document was written, INESC Porto for the realization of this project in the most pleasant environment and finally, to my family and friends for the sharing of moments of leisure and relaxation, that balanced the moments of work, in a combination that made the last few months an experience of personal well-being.*

# Index

# Figure List

# Table List

# Glossary

*DFT – Discrete Fourier Transform*

*EM – Expectation-Maximization*

*EMD – Earth Mover's Distance*

*FFT – Fast Fourier Transform*

*GMM – Gaussian Mixture Model*

*HMM – Hidden Markov Models*

*MFCC – Mel Frequency Cepstral Coefficients*

*MIR – Music Information Retrieval*

*RNN – Recurrent Neural Networks*

*SPR – Statistical Pattern Recognition*

*STFT – Short Time Fourier Transform*

*SVM – Support Vector Machines*

# Chapter 1

# 1. Introduction

## 1.1 Context

The automatic information retrieval from music is a developing area with several possible applications: the automatic grouping of songs within genres, the calculation of song similarities, the description of the various components of music – such as the rhythm or timbre. And more ambitious goals can be foreseen: maybe an automatic labeling of music with subjective qualities? Quantifiers of an "emotional" level of the songs? Of a "danceable potential"? There could be also automatic previsions about the commercial success of songs and records (which would reduce the staff employed by record labels responsible for the finding of new talents – considering some artists that are phenomenally advertised, one is left thinking if the loss is that great). Of course, if this is to happen, it won't be in a near future, as the subject of the automatic musical information retrieval is just giving its first steps. The internet has become a huge platform for multimedia data sharing, where music plays a big role. Search engines are becoming more complex and descriptive. Mechanisms to automatically organize and label music in various ways, as complete and precise as possible, will surely be of great help in the future.

In this dissertation, experiments were made with the goal to correctly classify music into different genres. Two musical collections were used: one composed entirely of Latin Music, distributed over ten different genres ("axé", "bachata", "bolero", "forró", gaúcha", "merengue", "pagode", "salsa", "sertaneja" and "tango"), and the other composed of ten general musical genres ("blues", "classical", "country", "disco", "hip hop", "jazz", "metal", "pop", "reggae" and "rock"). Separate experiments were conducted with these two different datasets.

Music can be characterized in four main dimensions: melody, harmony, rhythm and timbre. Melody, harmony and rhythm are temporal characteristics as they are related to progressions over time. Timbre, on the other hand, relates to the aspects of sound.

The classification experiments made in this dissertation only considered the musical characteristic of timbre – the goal here is the assessment of the quality of musical genre classification based only on an analysis of the timbre of music.

The approach used can be explained in the summarized way: values characterizing the timbre of the different genres were extracted and models that represented these extracted values (that characterize timbre) were automatically computed for each genre. These models were then tested, as they were used in an attempt to correctly classify into a genre new and unidentified values, characterizing timbre. The values used to create the model are the training values, and the values used to test the model are the testing values. These values were obtained with the computation of signal characteristics of time segments, or frames, of very small duration.

## 1.2 Motivation

The motivation behind this project can be identified as being of two different natures: a personal motivation, driven by my curiosity on the subject of this dissertation – what are the mechanisms behind the automatic genre classification of music? Or more generally, what are the mechanisms behind the automatic retrieval of information from sounds and music? What characteristics can be extracted from a specific song? How are these characteristics extracted? How can the extraction of characteristics from music be used to implement a system that will automatically classify new and unidentified music? What are the possibilities of such systems? What is the state of the art in this subject right now?

The other nature of the motivation behind the realization of this project has to with the possibility to contribute, even if in the smallest degree, to the body of work that is being built around the subject of automatic classification of music. It is a new and exciting area, of possibilities of great interest to anyone that finds pleasure in the listening or study of music.

A part of the work here developed is related to the study of the characteristics of timbre of ten different Latin music genres, and there is an attempt to compute similarities between these genres. The motivation behind this task was a personal curiosity towards the Latin musical genres, of which I knew little about and the possibility to contribute to a study of these musical genres, their timbral characteristics, and how well they are defined by them - in the degree set by the characteristics of the performed classification procedures.

## 1.3 Goals

The experiments described in this dissertation were designed with specific goals in mind. These goals are the comparison of the performance obtained with different classification algorithms and the assessment of the influence on the classification performance of several factors, which are:

1) The nature of the datasets used in the classification procedures - a comparison is made between the classification performances obtained using the Latin music genres and the ten general genres.
2) The size of the datasets used – due to computational limitations, the datasets could not be used in their totality, therefore smaller samples of the datasets were used instead. A comparison was made between the results obtained with datasets of variable size.
3) The relation between the datasets used for training and for testing – the classification is performed with models representing genres; these models are created from the analysis of the songs belonging to that genres, in a designated "training" phase. These models are then tested with a "testing" dataset (the results of these tests are the results used to assess the classification performance). The training and testing datasets can be the same, or can be different – defined after a larger dataset is split into two, creating the datasets. The ratio of this split can vary. The influence of these characteristics in the classification performance was assessed.
4) The use of an artist filter in the definition of the training and testing datasets – as it was said above, the training and testing datasets can be created after a larger dataset is split in two. The split can be made in a random way or can be done respecting conditions. A condition was defined that no artist (or more precisely, its songs) can be present in both the training and testing datasets – this condition is the artist filter. If songs of a certain artist are contained in the training set, songs from that same artist cannot be contained in the testing set and vice-versa. The influence of the use of the artist filter in the classification results was assessed.

Another goal set is to compare the performance of the different Latin music genres and to compute genre similarities based on the characteristics of timbre.

## 1.4 Structure of the document

This dissertation is structured in five chapters, the first one being this introduction to the developed work.

The second chapter is composed of a more extensive introduction to the subject of automatic musical genre classification and a description of the approaches being taken in this area.

The third chapter is the description of the experimental methodology followed in the experiments reported in this dissertation: there is a description of the extraction of characteristics from the musical samples; of the algorithms used in the classification procedures; of the characteristics of the datasets used. Finally, there is a subchapter dedicated to the analysis of some experiments that are found in the literature available.

The fourth chapter is dedicated to the presentation and analysis of the results achieved in the performed experiences.

The last chapter contains the general conclusions of the developed work.

# Chapter 2

# 2. Automatic musical genre classification

## 2.1 Defining musical genres

The automatic identification of music is a task that can be of great use – the internet has become a huge and popular platform of multimedia data sharing and the trend is to become even bigger – the limits to what it can and will achieve are impossible to preview. Music is available through search engines that tend to become more complex and sophisticated and these search mechanisms of the music available require descriptions as complete as possible.

In this context, mechanisms to automatically label music (or other multimedia content) in a digital format are obviously helpful. The manual labeling of music can be a hard and exhausting job if the quantities to be labeled are of great size. The manual labeling of a catalog of music of a few hundred thousand titles into several objective (*e.g.* rhythm) and subjective (*e.g.* mood) categories, takes about 30-man years – in other words, this task would require the work of 30 musicologists during one year [1].

The most obvious and immediate property of a piece of music might be its genre. If one tries to organize music into main groups, these groups are probably the big musical genres – classical, jazz, pop/rock, world music, folk, blues or soul. It is natural that the field of science related to the automatic labeling of music would start with an attempt to label music with its genre. It is in this place where the subject of automatic labeling of music is right now – at the very beginning of its task. It is not hard to imagine that future developments on this area can lead to greater possibilities – for example, to automatically label music with subjective qualities such as the emotional effects caused in the listener.

Advances in this subject of computational music analysis can also lead to the automatic production of music and, just like machines now outperform humans in the playing of chess, maybe one day music-making will be a task reserved to machines (but no doubt that this idea seems far-fetched and I would be particularly curious to see how something like the *soul* of James Brown is recreated in an algorithm).

The subject of automatic information retrieval from music is commonly designated by MIR (an acronym for Music Information Retrieval). Advances in this area are related with subjects such as content-based similarity retrieval (to compute similarities between songs), playlist generation, thumb nailing or fingerprinting [1].

The problem of genre definition is not trivial. There are obvious questions that come up when one tries to map songs into specific musical genres: for example, should this mapping be applied to individual songs, albums or artists? The fact is that a considerable proportion of artists change the characteristics of their music along their career or at some point venture into musical genres that are not the ones that would characterize the majority of their work. Many musicians even escape categorization, when their music is a big melting pot of different and distant influences. And when defining a song's genre, should only the musical aspects of the song be taken into consideration? Or also socio-cultural aspects? The musical genres (or subgenres) "Britpop" and "Krautrock" (the word "Kraut" is used in English as a colloquial term for a German person) suggest a decisive geographical factor in its name, and "Baroque" designates the music made in a specific period of the history, the classical music made in Europe during the 17$^{th}$ and first part of the 18$^{th}$ century. It is important to note that there is not a general agreement over the definition and boundaries of musical genres. According to [2], if we consider three popular musical information websites (Allmusic, Amazon and Mp3), we see that there are differences in their definition of genres: Allmusic defines a total of 531 genres, Amazon 719 genres and Mp3 430 genres. Of these, only 70 genres were common to the three. The very nature of the concept of "musical genre" is something extremely flexible: new genres are created through expressions coined by musical critics that "caught fire" and got popular ("Post-rock", popularized by the critic Simon Reynolds is an example), or through the merging or separation of existing genres.

These considerations are addressed in [3], where two different concepts of "musical genre" are defined: an intentional concept and an extensional concept. In the intentional concept, genre is a linguistic tool which is used to link cultural meanings to music. These meanings are shared inside a community. The example given is the "Britpop" genre, which serves to categorize the music of bands like the Beatles and carries with it references to the 60's and its cultural relevance and context - things we are all more or less familiar with (it should be noted that the authors got it wrong: the common usage of the term "Britpop" refers to the music from a set of British bands of the 90's, such as Blur or Oasis, but the point is made). On the other hand, in the extensional concept, genre is a dimension, an objective characteristic of a music title, like its timbre or language of the lyrics.

The fact that these two concepts are many times incompatible is a problem in the attempt to achieve a consensual musical genre taxonomy.

## 2.2 The supervised and unsupervised approach

There are three general approaches to the task of automatic musical genre classification. The first approach consists on the definition of a precise set of rules and characteristics that define musical genres. A song would be analyzed and evaluated if it fell under the definition of a musical genre – if it respected the defined rules that characterize the genre. This approach is designated in [4] as "expert systems". No approach of this kind has been implemented yet – the reason is the complexity needed for these systems: an expert system would have to be able to describe high level characteristics of music such as voice and instrumentation and this is something that is yet to be achieved.

The unsupervised and supervised approach can be considered as the two parts of a more general category that could be named "machine-learning approach", for the mathematical characteristics that define a musical genre are automatically elaborated, during training phases. Various different algorithms exist for this purpose and they are called "classifiers". During a training phase, mathematical data that characterizes musical signals is given to the classifier, the classifier processes this data and automatically finds relations or functions that characterize subgroups of the data given - in this case, musical genres. There are two approaches to do this: an unsupervised and a supervised way.

As it was said above, the classifier processes mathematical data that characterizes musical signals; it does not process the musical signal itself. These characteristics are called "features" and are extracted from the musical signal. Each feature takes a numeric value.

In the unsupervised approach, the sets of features used in the training phase are not labeled. The algorithm is not given the information that a certain set of features is "rock", or that another one is "world music".

The labels (the classes) will emerge naturally: the classification algorithm will consider the differences between the given sets of features and define classes to represent these differences.

This approach has the advantage of avoiding the problematic issues, referred in chapter 2.1, related to the usage of a fixed genre taxonomy (genre taxonomy being the definition and characterization of the total universe of musical genres), such as the difficulties of assigning a genre to certain artists or the fuzzy frontiers between genres.

One could consider that this approach makes more sense: if one song's genre is going to be classified as a function of the song's extracted mathematical signal features, the definition of genre should only take into consideration these same features. But then, if this procedure is adopted, one has to expect that the new emerging "genres" will not correspond to the definition of genres we are all familiar with, that are subject not only to the musical signal characteristics, but also to cultural and social factors. Maybe a new taxonomy of genres could be adopted, one that reflected only the set of characteristics (reflected in the extracted features) of the musical signal, classified with a defined

classifier. People would adopt this new genre taxonomy when talking about musical genres and the task of automatic musical genre classification would become much easier. But that is highly unlikely to happen, as it is impossible to imagine a complete reconfiguration of the musical genre vocabulary used by common people, musicians and critics alike. And one that could go against some rules of common sense: if, by some extraordinary mathematical coincidence, it grouped in the same new genre such opposite artists as, let's say, the pop-oriented band The Clash and the schizophrenic and aggressive Dead Kennedys. Controversial at least.

The unsupervised approach requires a clustering algorithm – that decides how to group together the models of the audio samples. "K-means" is one of the simplest and most popular algorithms. (This algorithm will be described in a latter section of this work, although in a different context.)

In the supervised approach there is an existing taxonomy of genres, which will be represented by mathematical models, created after a training phase: sets of songs belonging to different genres are labeled with their genre, their features are extracted and an algorithm will automatically find models to represent each musical genre. After that, unlabeled songs can be classified into the existing genres.

## 2.3 Feature extraction

There are a variety of features that can be extracted from an audio signal. These features can be characterized through some properties. Four properties are distinguished in [5] that can be used to characterize features: "the steadiness or dynamicity of the feature", "the time extent of the description provided by the features", "the abstractness of the feature", and "the extraction process of the feature".

The first property, about "the steadiness or dynamicity of the feature", indicates if the extracted feature characterizes the signal at a specific time (or at a specific segment of time), or if it characterizes the signal over a longer period of time, that spans several segments of time: for example, each of these time segments will provide specific feature values that will then be used to calculate features characterizing the longer time segment, such as the means and standard deviations of the short-time segments feature values. In other words, in this latter case, the features are the mean and standard deviation values of consecutive sequences of extracted features.

The second property, about the "time extent of the description provided by the features", indicates the extent of the signal characterized by the extracted feature. The features are distinguished between "global descriptors" and "instantaneous descriptors" – the global descriptors are computed for the whole signal (and representative of the whole signal), while the instantaneous descriptors are computed for shorter time segments, or frames.

"The abstractness of the feature" is the third referred property. The example given to illustrate this property is the spectral envelope of the signal. Two different processes to extract features are used to represent the spectral envelope – "cepstrum" and "linear prediction". Cepstrum is said to be more abstract than linear prediction.

The last property, "the extraction process of the feature", relates to the process used to extract the features. For example, some features are computed directly on the waveform signal, such as the "zero crossing rate". Other features are extracted after performing a transform on the signal, such as the FFT (Fast Fourier Transform).

The features extracted may be related to different dimensions of music, such as the melody, harmony, rhythm or timbre. A description of these dimensions is done in the following chapter.

## 2.4 Melody, harmony, rhythm and timbre

Various definitions are used to characterize "melody". Some of these definitions are found in [6]: melody is "a pitch sequence", "a set of attributes that characterize the melodic properties of sound" or "a sequence of single tones organized rhythmically into a distinct musical phrase or theme". There is a certain difficulty in the attempt to precisely define melody, although melody is a familiar and understandable concept: it is what is reproduced in the whistling or humming of songs. A review of techniques used to extract features that can describe melody is found in [6].

Harmony is distinguished from melody in the following way: while melody is considered as a sequence of pitched events, harmony relates to the simultaneity of pitched events – or using a more common designation, "chords". It is common the idea that melody relates to a "horizontal" progression of music, and harmony relates to a "vertical" one. This idea is easily understood if we think of time as the horizontal (melodic) dimension – harmony will be a vertical dimension, related to the simultaneity of melodic progressions. Work related to the extraction of information related to harmony can be found in [7], where a method that estimates the fundamental frequencies of simultaneous musical sounds is presented, or in [8], where it is described a method to analyze harmonies/chords, using signal processing and neural networks, and analyzing sound in both time and frequency domains. A method to retrieve pitch information from musical signals is presented in [9].

Rhythm is related to an idea of temporal periodicity or repetition. The existence of temporal periodicity, and its degree of predominance, is what characterizes a piece of music as "rhythmic" (or not). This is a vague definition, but there is not one more precise, the notion of rhythm is something intuitive: it is agreeable that jazz music is more rhythm-oriented than classical music or certain types of experimental rock.

A report of a variety of approaches related to the subject of automatic rhythmic description is made in [10], with diverse applications being addressed such as "tempo induction, beat

tracking, quantization of performed rhythms, meter induction and characterization of intentional timing deviations". The concepts of metrical structure, tempo and timing are used in the analysis of rhythm: the metrical structure is related to the analysis of beats, which are durationless points in space, equally spaced in time. Tempo is defined as the rate of beats at a given metrical level. It is commonly expressed as the number of beats per minute or the time interval between beats. The definition of timing might be harder to make: timing changes are short time changes in the tempo (which remains constant after that variations), as opposed to tempo changes, which refer to long term changes in the tempo (the tempo or beat rate changes after the variation). The task of extraction features to characterize rhythmic properties is addressed in [10] and [11].

Timbre is the quality in sound that differentiates two sounds with the same pitch and loudness. Timbre is what makes two different instruments, let's say, guitar and piano, playing the same notes in an equal volume, sound different. It is also designated by color or "tone quality".

The features characterizing the timbre analyze the spectral distribution of the signal and can be considered global in a sense that there is no separation of instruments or sources – all that "happens" at a certain time is analyzed. These features are "low-level" since usually they are computed for time segments (frames) of very small duration and do not attempt to represent higher level characteristics of music, such as melody or rhythm.

A classification of features characterizing timbre is made in [4]. These features are classified as: temporal features (that are computed from the audio signal frame, such as the "auto-correlation coefficients" or the "zero crossings rate"); energy features (that characterize the energy content of the signal, such as "root mean square of the energy frame" or the "energy of the harmonic component of the power spectrum"); spectral shape features (that describe the shape of the power spectrum of a signal frame, such as "centroid", "spread", "skewness" or the "MFCC"); and perceptual features (that simulate the human hearing process, such as the "relative specific loudness" or "sharpness").

The experiments made in the scope of this dissertation will extract and classify features only related to timbre. A detailed description of the features extracted will be done in the chapter 3.1.

## 2.5 The "bag of frames" approach

The majority of work related to the automatic classification of music uses features related to timbre. This approach is designated as the "bag of frames" (BOF) approach in [12].

The audio signal to be classified is split into (usually overlapping) frames of short duration (the duration of the frames and of the overlap percentage is variable, usually the duration and overlap percentage are around 50 ms and 50 %, respectively); for each frame features are extracted (necessarily related to timbre because the short duration of the frames makes

it impossible to extract features characterizing higher level attributes such as rhythm); the features from all frames are then processed by a classifier, which will decide which class better represents the majority of the frames of the signal to be classified.

These classifiers are trained during a training phase (with a training set that should be as representative as possible) that can be supervised or not, as it was described in chapter 2.2. The training should be done using the same features characterizing timbre and with the same characteristics of the features of the signal to be analyzed, obviously (the same frame duration and overlap percentage).

After the training phase, the classifiers can be used to perform classifications of new, unidentified signals. These new signals are classified as belonging to one of the pre-defined classes.

Some problems with the "bag of frames" approach are identified in [12], [13], [14] and [15]: these issues are named "glass ceiling", "paradox of dynamics" and "hubs".

"Glass ceiling" is a problem reported in [13] that relates to the fact that although variations on the characteristics of the features and on the classification algorithm can indeed improve the classification performance, this same performance fails to overcome an empirical "glass ceiling", of around 70 %. It is suggested that this limit of 70 % is a consequence of the nature of the features used and that, in order to improve this number, a new approach will have to be considered – using something more than only features characterizing timbre.

"Paradox of dynamics" is reported in [14] and addresses the fact that changes in the feature characteristics or in the classification algorithm in order to take into account the dynamic variations of the signal over time, do not cause an improvement of the classification results. Some of the strategies that take into account the time variations of the signal are referred: stacking together consecutive feature vectors to create larger feature vectors; the use of time derivatives of the extracted features; the averaging of the features over a certain time period (this method is the use of "texture windows", explained in section 2.6); the use of classifiers that take into account the dynamics of the features, such as Recurrent Neural Networks (R-NN) or Hidden Markov Models (HMM). After running a set of experiments it was concluded that these strategies do not improve the quality of the best static models if the data being classified is polyphonic music (music with several instruments) (on the contrary, performance decreases). However, when applying these strategies to non-polyphonic music (composed of clean individual instrument notes), there is an improvement of the classification performance. It should be noted that these results apply to methods considering features related to timbre only (the "bag of frames" approach).

It is reported in [15] the existence of a group of songs that are irrelevantly close to all other songs in the database and tend to create false positives. These songs are called "hubs" and the definition given is the following: "a hub appears in the nearest neighbours

of many songs in the data base; and most of these occurrences do not correspond to any meaningful perceptual similarity". This definition of "hub" does not necessarily cover songs which are close to most of the songs in the database if the database is composed of songs that resemble the style/genre of the analyzed songs. The example given is the fact that it could happen that a song from the Beatles ("A Hard Day's Night") would be found to occur frequently as a nearest neighbour to a great number of songs of a database composed only by Pop songs. However, in other databases, composed of different musical genres, this would not occur. Therefore, "A Hard Day's Night" should not be considered a "hub". It is found that different algorithms produce different "hub" songs (although there is a minority of songs that are "hubs" regardless of the algorithm used). It is concluded that the "hubness" of a song is a property of the algorithm used and its characteristics.

## 2.6 Analysis and texture windows

A technique to take into account the dynamics of the signal over time is through the use of texture windows. The concepts of analysis and texture windows are going to be introduced next.

The signal to be analyzed is broken into small, possibly overlapping, smaller time segments or frames. Each frame is analyzed individually and its features are extracted. The time length of these frames has to be small enough so that the magnitude of the frequency spectrum is relatively stable and the signal for that length of time can be considered stationary. The frame and its length is designated "analysis window".

The use of texture windows is a strategy to take into account the dynamics of the signal being studied. The features are extracted from short time samples of around 50 ms (the analysis window). When feeding these values to the classifiers, it is a common strategy to compute approximations of longer time segments – called the texture windows. The classifier will not evaluate all the features that correspond to all frames, individually, but will instead evaluate the distribution statistics of these features over the time of the texture window, through the parameters of mean and standard deviation.

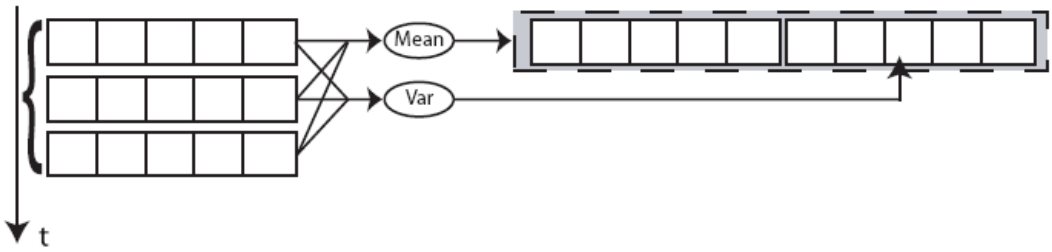


*Figure 1: Texture windows [16]*

For each analysis window there will be a correspondent texture window, composed of the features from the analysis window and from a fixed number of previous analysis windows.

This technique can be seen as perceptually more relevant, as frames of $10^{th}$ ms are obviously not long enough to contain any statistically relevant information for the human ear [4].

The ideal size of the texture window is defined by [9] as one that should correspond to the minimum time needed to "identify a particular sound or music texture", where "texture" is defined as a subjective characteristic of sound, "result of multiple short time spectrums with different characteristics following some pattern in time", and the example given is the vowel and consonant sections in speech.

Experiments run by [9] show that the use of texture windows effectively increases the performance of the classifiers. Using a single Gaussian classifier and a single feature vector characterizing music titles, if no texture window was used, the classification accuracy obtained was around 40 %. The use of texture windows increased greatly the accuracy and this accuracy improved with the size of the texture window used – until 40 analysis windows. After that, the classification accuracy stabilized. The value of 40 analysis windows corresponds to the time length of 1 s. The accuracy obtained with a texture window size of 40 was around 54 %.

Implementations of different time segmentations are made in [17], [18] and [19].

A new segmentation method, named "onset detection based segmentation" based on the detection of "peaks in the positive differences in the signal envelope" (onsets), which correspond to the beginning of musical events is purposed in [17]. This technique would ensure that a musical event is represented by a single vector of features and that "only individual events or events that occur simultaneously" contribute to a certain feature vector. It is argued that although the use of texture windows of 1 s that are highly overlapped "are a step in the right direction" because they allow the modeling of several distributions for a single class of audio, they also complicate the distributions because these windows are likely to capture several musical events.

The performance obtained using several types of segmentation is compared and the onset detection based segmentation yields the best results, although closely followed by the use of 1 s texture windows.

Three segmentation strategies in a database of 1400 songs distributed over seven genres were compared in [18]: using texture windows of 30 s long, using texture windows of 1 s long and using a different approach that consists on using vectors that are centered "on each beat averaging frames over the local beat period". The results are not conclusive, for the levels of performance achieved vary according to the genre being detected and the classifier used and one cannot proclaim that the "beat" time modeling achieves better results than the use of 1 s long texture windows.

[19] proposes “modeling feature evolution over a texture window with an autoregressive model rather than with simple low order statistics”, and this approach is found to improve the classification performance on the used datasets.

# Chapter 3

# 3. Experimental methodology

## 3.1 Feature extraction

The feature extraction process computes numerical values that characterize a segment of audio – in other words, it computes a numerical representation of that segment of audio (or frame) in the form of a vector consisting of the values of several particular features. This feature vector has typically a fixed dimension and can be thought of as a single point in a multi-dimensional feature space. The several steps of this procedure are described below:

A Time-Frequency analysis technique such as the Short Time Fourier Transform (STFT) is performed on the signal from which the features are to be extracted, to provide a representation of the energy distribution of the signal over a time-frequency plane. The signal is broken into small segments in time and the frequency content of each segment is calculated – for each time segment there is a representation of the energy as a function of the frequency.

The STFT can be calculated in a fast way using the Fast Fourier Transform (FFT). The FFT is a method to calculate the Discrete Fourier Transform (DFT) and its inverse. There are other approaches to perform these calculations, but the FFT is an extremely efficient one, often reducing the typical computation time by hundreds [20].

A frequency spectrum achieved after a STFT procedure represents perfectly the respective audio signal, but it contains too much information: a lot of the information represented in the spectrum is not important and feature vectors of limited dimensions (and as informative as possible) are more adequate to machine learning algorithms. For this reason, audio identification algorithms typically use a set of features that are calculated from the frequency spectrum signals.

Some of these features are calculated after the Short Time Fourier Transform is applied (which can be calculated using the Fast Fourier Transform (FFT) algorithm) such as the *The Spectral Centroid*, *The Rollof Frequency*, *The Spectral Flux* and the *Mel Frequency Cepstral Coefficients* (MFCC). The MFCC constitute the majority of the calculated features and their description is done in this chapter.

### 3.1.1 The Mel scale and the MFCC

The Mel scale was proposed by Stevens, Volkman and Newman in 1937 in the context of a study of human auditory perception in the field of psychoacoustics. The Mel is a unit that measures the pitch of a tone the way humans perceive it. This concept is easily understood with the following explanation of how the scale was developed: the frequency of 1000 Hz was chosen as a reference and designated to be equal to 1000 Mels. Listeners were then presented a signal and were asked to change its frequency until the perceived pitch was twice the reference, half the reference, ten times the reference, 1/10 the reference, and so on. From these experimental results, the Mel scale was constructed. The Mel scale is approximately linear below 1000 Hz and logarithmic above that frequency – the higher the frequency, the higher the frequency increment must be to produce a pitch increment [21]. The next figure represents the Mel Scale:

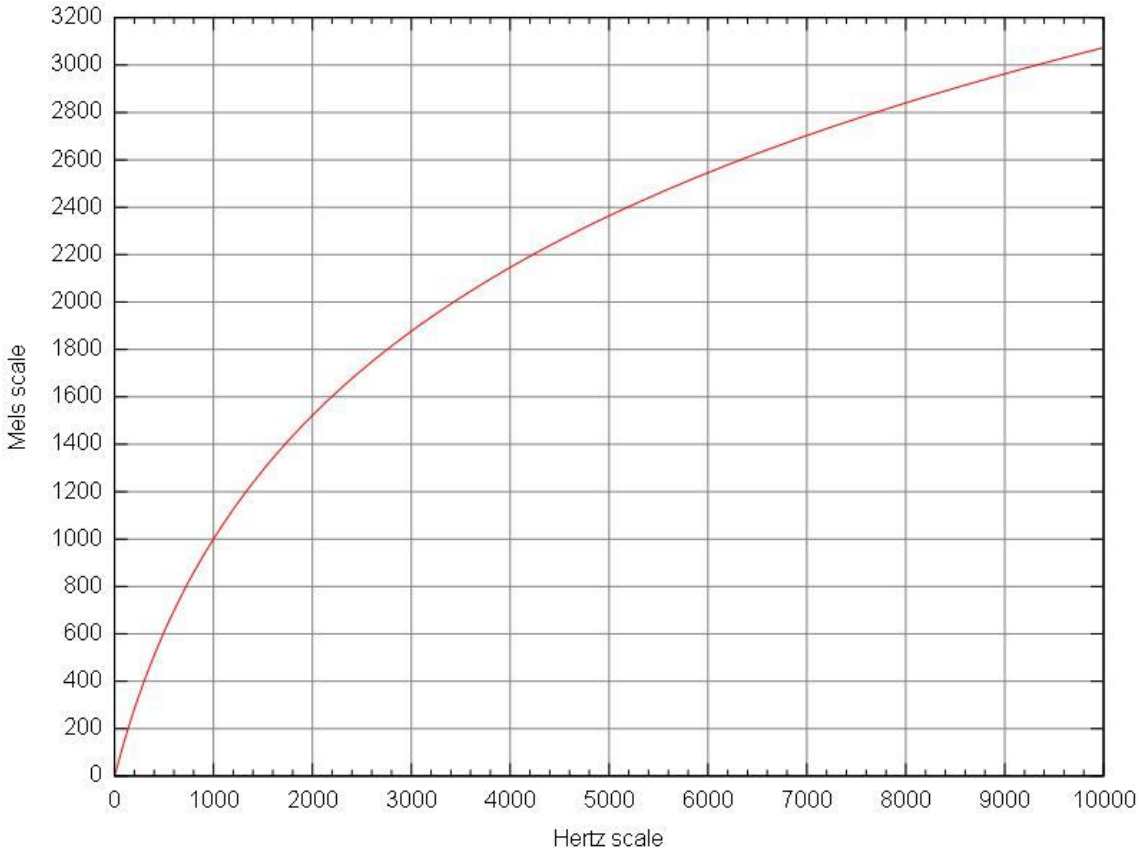


*Figure 2: The Mel Scale [22]*

The relation between the hertz unit (frequency) and the Mel unit is:

$m=1127.01048 \log(1+f/700)$ *Mel*

The inverse relation is:

$f=700(e^{(m/1127.01048)} - 1)$ *Hz*

When calculating the MFCC, the Mel scale is used to map the power of the spectrum obtained after the FFT is applied, using triangular overlapping windows.

The discrete cosine transform (DCT) expresses a signal in terms of a sum of sinusoids with different frequencies and amplitudes. It is used as the final step of the calculation of the MFCC: the signal obtained after the application of the Mel-mapping is approximated by a sum of a defined number of cosines whose frequency is always doubling. The coefficients of these cosines are the MFCC.

### 3.1.2 Marsyas and feature extraction

The extraction of the features of audio files was done using Marsyas. Marsyas stands for Music Analysis Retrieval and SYnthesis for Audio Signals. It is a work in progress and has been developed by George Tzanetakis and a community of developers. Tzanetakis started the Marsyas project in the end of the 90's and since then it has gone through major revisions. Marsyas 0.2 is the latest version of this software.

Marsyas is a framework for audio analysis, synthesis and retrieval. It provides a variety of functionalities to perform a number of audio analysis tasks. These functionalities include signal processing and machine learning tools.

This software is designed to be used by experts and beginners. Expert users will have the possibility to create more complex applications through the writing of new code. Beginner users will have the possibility to use the system through graphical user interfaces and high-level scripts [23]. The version of Marsyas used was the revision 3432.

The Marsyas tool "bextract" was used to perform the feature extraction from audio files. It uses as input a text file indicating the paths of the files whose features are going to be extracted; each path is labeled with the class the musical file belongs to (the musical genre).

The Marsyas tool "mkcollection" was used to create the labeled collection files. Its syntax is:

*mkcollection –c collectionfile.mf –l label /pathcollection*

Where *collectionfile.mf* is the output file, containing the path to the music files and their labels, *label* is the class (genre) of the analyzed files, and */pathcollection* is the path to the files. The musical files must be previously organized into genre paths. This procedure was done to all musical genre paths, resulting in a collection text files, one for each genre. These text files were copied into a unique larger text file - containing the path to all files of all genres to be classified.

The feature extraction is done with the tool "bextract", as it was said above. It uses as input the labeled text file with the path to the files to be considered and outputs a feature file. Its simplest syntax is:

*Bextract collectionsfile.mf –w featuresfile.arff*

*Collectionsfile.mf* is the text file with the labeled path to all the files to be considered and *featuresfile.arff* is the desired name of the output file, containing the extracted features.

There are many options to configure the feature extraction process. In the project, "bextract" was used in the following way:

*Bextract –ws 2048 –hp 1024 –m 100 –s 40 –l 30 collectionsfile.mf –w featuresfile.arff*

The option *-ws* indicates the size of the analysis windows, in the number of samples of the audio file being analyzed (the number of samples per second of the music file is defined by its audio sample rate, usually 44 kHz music files). Analysis windows of 2048 samples were used. The option *-hp* indicates the hop size, in number of samples. The hop size is the distance between the beginnings of consecutive analysis windows. If the hop size was the same as the analysis window, there would be no overlap between consecutive analysis windows. In this case, there is an overlap of 50 %, as the hop size is half the size of the analysis windows (1024 samples and 2048 samples).

The option *-m* indicates the size of the texture windows, in number of analysis windows. In this project, each texture window is composed of 100 consecutive analysis windows.

The option *-s* defines the starting offset in seconds of the feature extraction process in the audio files to be considered. The option *-l* defines the length in seconds of the segments from each sound file from where the features are to be extracted. If the *-s* and *-l* values are not set Marsyas extracts features from the whole audio files. There was a decision to extract features located in the middle of the songs (between the seconds 40 to 70). Many songs have beginnings that are not particularly representative – live records for example, have clapping and cheering sounds dominating the first seconds. The decision to consider only 30 s from each sound file was also made to limit the size of the output feature file.

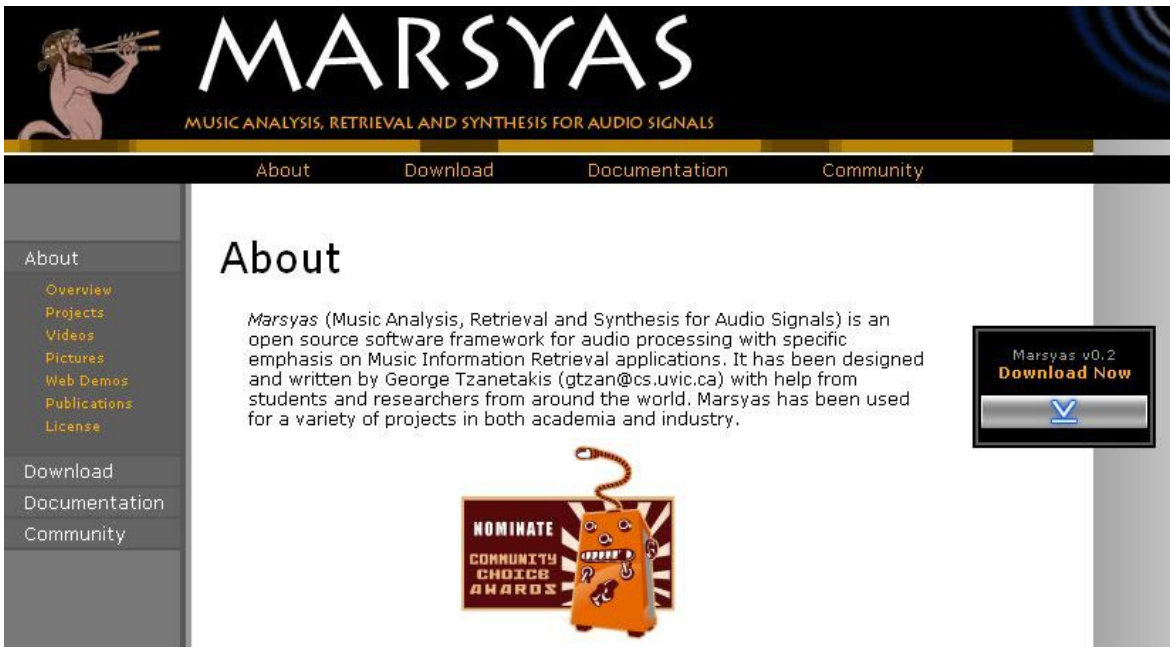


*Figure 3: Marsyas Official Webpage*

### 3.1.3 Description of the features used

16 different features were considered, and for each feature, two values were computed: the mean and the standard deviation, representing the values of the feature along the 100 individual analysis windows that constitute the texture window. This totals 32 features that are extracted from each different texture window, resulting in a feature vector composed of 32 features. This feature vector can be designated as "instance". Instances are single points in a multi-dimensional universe and are the objects used in the training and testing of classifiers.

All features but one (The Zero Crossings Rate) are extracted after a Fast Fourier Transform is applied to each frame. The extracted features are:

*The Zero Crossings Rate*: this is the rate at which the signal changes from positive to negative – or the rate at which it crosses the zero value.

*The Spectral Centroid*: the center of gravity of the frequency spectrum. The spectral centroid is the weighted mean of the frequencies, with the weights being the frequency magnitudes.

*The Rolloff Frequency*: the frequency below which 85 % of the magnitude distribution is concentrated.

*The Spectral Flux*: the squared difference between the normalized magnitudes of the spectal distributions of consecutive frames. This feature should make an estimate of the variation of the magnitudes of consecutive frames.

*The Mel Frequency Cepstral Coefficients* (MFCC): after the FFT transformation is applied to the audio segments, the resulting signals are smoothed according to the Mel scale. Then, a Discret Cosine Transform is performed to the resulting signals, to obtain the designated Mel-Frequency Ceptral Coefficients (MFCC), as it was described in chapter 3.1.1. The first 12 MFCC are considered and used as features.

### 3.1.4 The feature files

These feature files computed with Marsyas can be read by Weka (a platform to perform machine learning and classification experiments) and have an *.arff* extension. The beginning of the feature file contains the definition of the features or attributes. The attributes are defined in the lines beginning with "@attribute".

After the definition of the attributes, the file contains the values of these attributes for each instance or texture window. Each line has the values of all attributes, for one instance (in the same order of the attribution definition in the beginning of the file). The values are

separated by commas. The last attribute is the "output attribute", which indicates the class to which the instance belongs to.

Let's consider that a song has a 44 100 Hz sampling rate and that features are extracted from this song with an analysis window size of 2048 samples, a hop size of 1024 samples, and a texture window size of 100 analysis windows. What will the extracted feature vectors (or instances) represent?

The sampling rate is 44 100 Hz, so the distance in seconds between two consecutive samples is 1/44100 s. A window size of 2048 samples corresponds to 2048/44100 s, or 0.0464 s. A hop size of 1024 samples corresponds to 0.0232 s. The distance between the beginnings of two consecutive analysis windows is 0.0232 s (the hop size). The texture windows are composed of 100 consecutive analysis windows, and will therefore characterize 0.0232*100=2.32 s. It should be noted that a texture window is computed for each different analysis window – consecutive texture windows will share 99 of their 100 analysis windows, and of course, will be extremely similar.

Each line in the *.arff* file is composed of the feature vector of a different texture window. These lines are the instances that are used to train and test the classifiers.

## 3.2 Classification

Classification algorithms, as the name suggests, categorize patterns into groups of similar or related patterns. The definition of the groups of patterns (classes) is done using a set of patterns – the training set. This process is designated machine learning and can be supervised or not. If the learning is supervised each of the patterns that constitute the training set given to the system is labeled. The system will then define the characteristics of the given labels (or classes) based on the characteristics of the patterns associated to those labels. If the learning is not supervised, there is no prior labeling of the patterns and the system will define classes by itself based on the nature of the patterns. There are also machine learning algorithms that combine labeled and unlabeled training patterns – the learning is said to be semi-supervised.

Tzanetakis in [24] identifies the following three kinds of classification schemes: statistical (or decision theoretic), syntactic (or structural) and neural. A statistical classification scheme is based on the statistical characterizations of patterns and assumes that the patterns are generated by a probabilistic system – it does not consider the possible dependencies between the features of the pattern. On the contrary, a syntactic classification scheme takes in account the structural interrelationships of the features. A neural classification scheme is based on the recent paradigm of computing influenced by neural networks.

The basic idea behind the classification schemes is to learn the geometric structure of the features of the training data set, and from this data, estimate a model that can be generalized to new unclassified patterns.

A huge variety of classification systems has been proposed and a quest for the most accurate classifier has been going on for some time. [25] praises this activity as being extremely productive and the cause of today's existence of a wide range of classifiers that are employed with commercial and technical success in a great number of applications – from credit scoring to speech processing. However, no classifier has emerged as the best. A single classifier has yet to outperform all others on a given data set and the process of classifier selection is still a process of trial and error.

The existing classifiers differ in many aspects such as the training time, the amount of training data required, the classification time, robustness and generalization.

The quality of a classification process depends on three parameters: the quality of the features (how well they identify uniquely the sound data to be classified); the performance of the machine-learning algorithm; and the quality of the training dataset (its size and variety should be representative of the classes) [25].

### 3.2.1 Platforms used - Weka and Matlab

The platforms used for classification were Weka and Matlab. Weka stands for "Waikato Environment for Knowledge Analysis" and is a platform to perform machine learning and data mining tasks, such as classification or clustering. It is developed at the University of Waikato, New Zealand. It is free software and available under the GNU General Public License [26].

The feature files output by Marsyas (using the "bextract" option) are in a format that can be properly opened and processed by Weka.

Weka was used to perform classification experiments. Four different classifiers were used: Naive Bayes, Support Vector Machines, K-Nearest Neighbours and C4.5.

One Weka tool that was used is the "Resample Filter" option. It can happen that the feature files to be processed by the Weka classifiers are so large in size that the training and testing tasks take several hours and even days to perform. In these cases, sampling the dataset reduces the computational work made by the classifiers. The size of the sampled dataset is defined by the user (in the form of percentage of the total dataset), is randomly created, and can be biased so that the classes are represented by a similar number of instances.

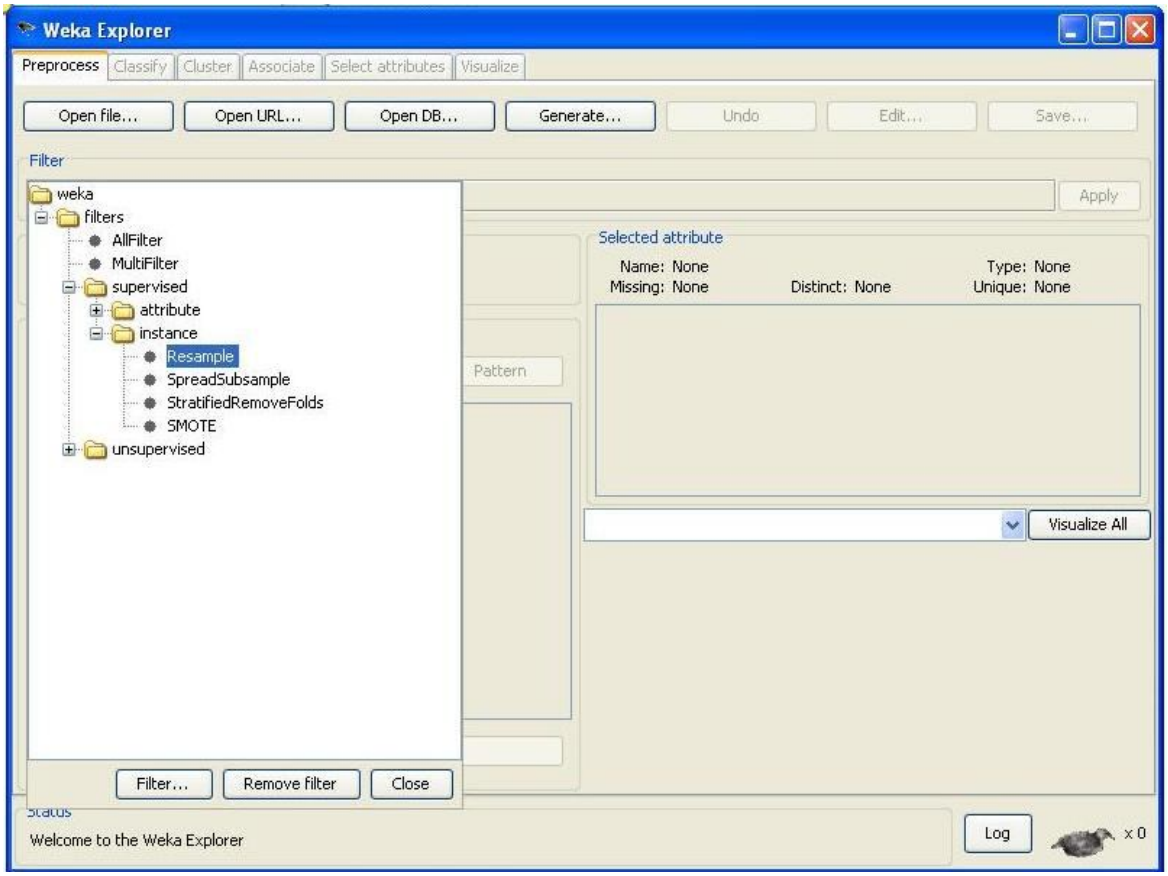


*Figure 4: Weka: "Resample"*

There are several ways to train and test datasets with Weka. Once the dataset is opened, the user can use all the dataset to train the classifier and use the same dataset to test the classifier. All the dataset's instances are used to train the classifier and the classifier's performance is tested with these same instances. There is the option to split the dataset into two smaller datasets: one dataset will be used to train the classifier and the other will be used to test it. The sizes of each dataset are defined by the user, in the form of percentages of the original dataset. This option is called "percentage split". Another way to train and test classifiers is through "*n*-fold cross-validation": the dataset is split into *n* folds, *n*-1 folds are used to train the classifier and the other fold is used to test it. The classification results are saved. Then, this process is repeated *n*-1 times, with the test fold changing every time. All the folds are used once as the test fold. The classification results are averaged. Finally, the user can train and test the classifier with separate dataset files.

The classification output in Weka has information about the quantity of correctly and incorrectly classified instances and detailed information about the classification of instances from each class. A Confusion Matrix is also output, it indicates how many instances belonging to each genre were classified as belonging to each genre.

Matlab was used to perform classification experiments with Gaussian Mixture Models. Matlab stands for "Matrix Laboratory"; it is a popular platform to perform numerical computation and provides also a programming language.

With the Weka classifiers there is a classification of instances: each instance is treated separately to train and test the classifier. The results that are ouput by Weka are relative to the classification performance of instances.

The Gaussian Mixture Models were used to perform a different kind of classification: of whole songs instead of instances. Songs were used to test the GMM, and the output of the test characterizes how were the individual songs classified (not instances).

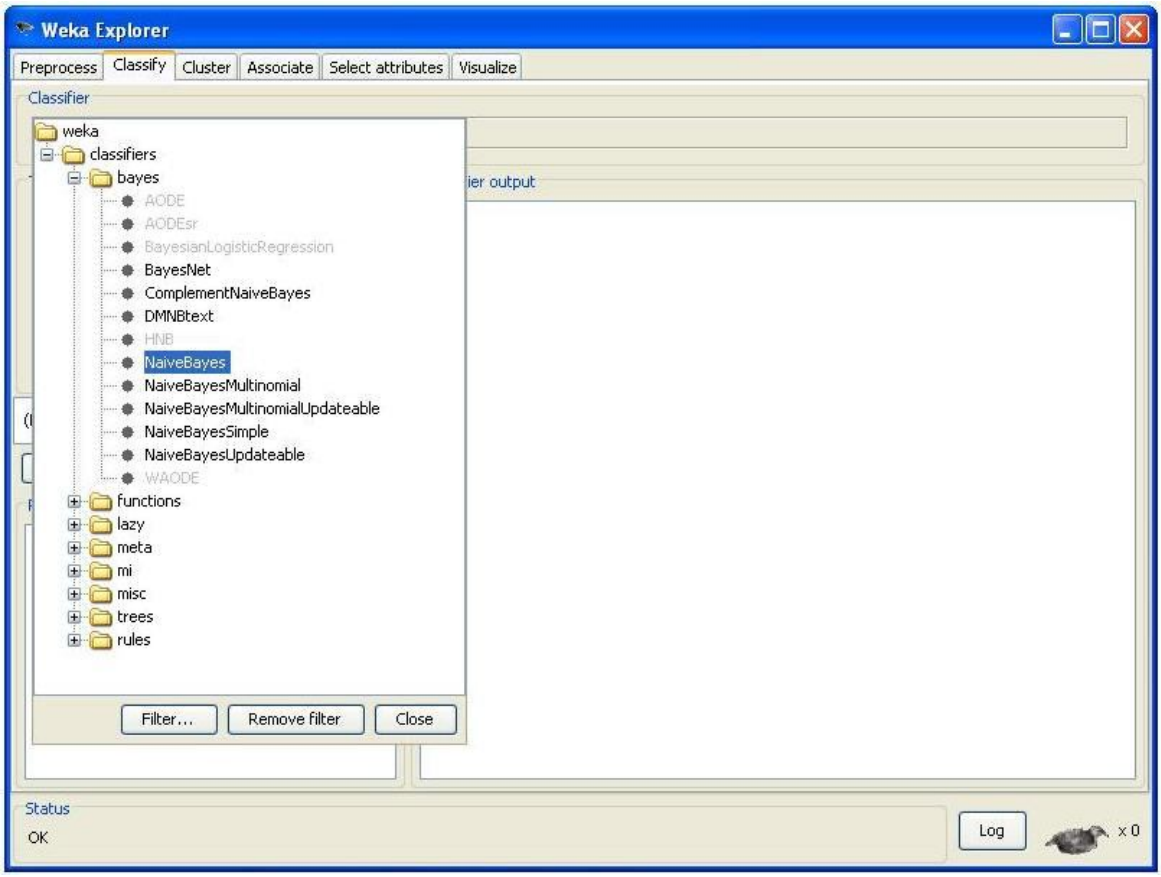


*Figure 5: Weka: Choosing a Classifier*

## 3.2.2 Instance classification with Weka classifiers

A description of the classifiers used with Weka is made below.

### 3.2.2.1 Naive Bayes

The Naive Bayes classifier assumes that there are no dependencies between the different features that constitute the feature vector. The probability of a feature vector or instance belonging to a certain class is equal to multiplying the probabilities of each individual feature belonging to that class. The probabilities of the feature vector belonging to each class are estimated and the most probable class is elected. The Naive Bayes classifier

yields surprisingly good results, comparable to the ones of other algorithms of greater complexity. The good results obtained with Naive Bayes are surprising because of the wrongful assumption that is made by this classifier: that the features are independent from each other. An explanation for this fact is that the probabilities estimated with the Naive Bayes classifier do indeed reflect this feature independency and thus are not precise, but the higher calculated probabilities usually do correspond to the correct classes, and so the Naive Bayes classifier performs well in classification tasks. Other purposed explanation is that although the classification of the Naives Bayes algorithm is affected with the dependency distributions along all attributes, there are times when the dependencies cancel themselves and the no-dependency assumption becomes accurate [27].

#### 3.2.2.2 Support Vector Machines

The concept behind Support Vector Machines is to find hyper planes that will constitute the frontiers between classes. In an *n*-dimensional universe, a hyper plane is *n*-1 dimensional and separates the universe in two. If the universe dimension is 1 (a line), the hyper plane will be a point; if the universe dimension is 2 (a plane), the hyper plane will be a line. Here, the dimension of the universe will be the number of different features extracted.

The frontier between two different classes will not be a single hyper plane, but two. These two hyper planes will constitute the margins of the frontier – inside those margins there are no class objects (see figure 6). When determining these hyper planes, the SVN classifier will take into consideration maximizing the distance between them (the margins), and minimizing errors. And what are these errors?

It is unlikely that the distribution of the (training set) data will be one in which classes are linearly separable. Considering a 2-dimensional universe, two classes belonging to that universe are linearly separable if one line can separate in one side the objects belonging to one class and on the other side the objects belonging to the other class. If the two classes are not linearly separable (which is the case most of the times), an error will occur if an object is placed on the wrong side of the line – the side of the other class.

The SVN approach has been generalized to take into account the cases where classes are not properly separated by linear functions. It is possible to separate classes with non-linear functions, such as circles (in a 2 dimensional universe). The concept remains the same: there will be two circles separating two classes, and when designing these circles there will be an effort to maximize the distance between the two circles and to minimize errors.

Further reading on support vector machines can be found in the documents [28] and [29], the former more comprehensive than the latter.

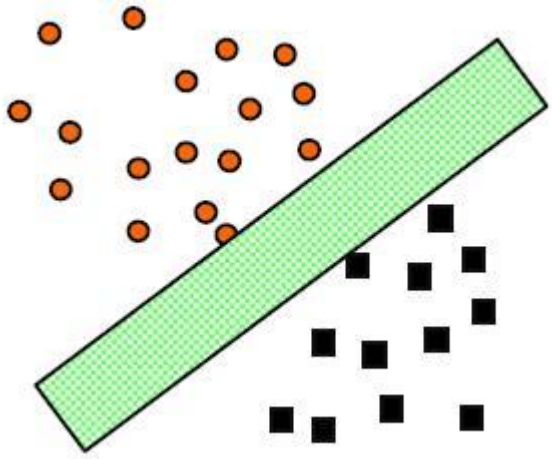

*Figure 6: SVM: frontier hyper-planes [29]*

#### 3.2.2.3 K-Nearest Neighbours

The K-Nearest Neighbours (KNN) classifier directly uses the training set for classification and does not use any mathematical form to describe classes in the feature space, such as probability density functions. With the KNN classifier, a new unclassified object is classified according to the position of its *K* nearest neighbours. This is better understood if we think of objects as points in the multi dimensional space. If *K* is one, the object will be classified as belonging to the class of its nearest neighbour. The "neighbours" are the instances of the training dataset, which are labeled. In the classification of an instance, the distances between this instance and all the labeled instances of the training set are computed, and the instance is classified with the class of its nearest training instance. A tutorial on K-Nearest Neighbours classification can be found in [30].

#### 3.2.2.4 J48

J48 is Weka's version of the C4.5 classifier, which is the latest public implementation of the decision tree approach popularized by J. R. Quinlan. The algorithm works as it is described below:

First, an attribute that best differentiates the different classes is chosen; tree branches are created for each value of the chosen attribute (or for subsets of the attribute's value); the instances are divided into subgroups (the created tree branches), reflecting their "chosen attribute's" value; if all the instances of a subgroup belong to the same class, the branch is labeled with that class; if there aren't attributes left that can distinguish instances belonging in the subgroup, the branch is labeled with the class of the majority of the subgroup's instances; if it isn't the case of any of these situations just described, the whole process is repeated (for each subgroup) [31].

### 3.2.3 Song classification with Gaussian Mixture Models

The Gaussian Mixture Model (GMM) classifier represents each class as a weighted mixture of a fixed-size number of Gaussian distributions. If a Gaussian Mixture Model is referred to as GMM3, this would mean that three Gaussian distributions are used. A Gaussian distribution is also frequently known as a Normal distribution. The used Gaussian distributions are multi-dimensional – the number of dimensions is the number of features. A GMM is characterized by the mixture weights and by the mean and covariance matrices (in plural, because the distributions are multi-dimensional) for each mixture component. These parameters are defined by the labeled training set during the training phase. Classes can be represented by a single Gaussian distribution – in this case, the classifier is simply called "Gaussian classifier".

The GMM classifier can be said to belong to a group of Statistical Pattern Recognition (SPR) classifiers, because a probability density function is estimated for the features extracted for each class [9].

In the experiments described in this dissertation, the Gaussian Mixture Models were created following the method presented in [32] and the functions available in the MA Toolbox, a collection of functions for Matlab [33]. These functions compute audio similarities. The installation of the Netlab toolbox [34] was required to perform some MA Toolbox functions.

The parameters of each Gaussian Mixture Model (weights, means and covariances of each Gaussian distribution) are initialized with the K-means algorithm.

A detailed explanation of the K-means algorithm is found in [35]. This algorithm is a process to group a universe of a finite number of objects into a number of groups or partitions (this number being *K*). These partitions will be characterized by the means of the objects that are inside them. The idea is to find these means, so that the distances between the mean of each partition and the objects that are inside of it are as small as possible. This is implemented by starting with *K* groups, each group consisting of a single random point. New points are added, and each one is added to the group whose mean is closest to the new point. After a point is added, the mean of the respective group is updated to take into account the new point.

Each of these partitions can be characterized by a probability density function, represented by its means and covariances. Applying this concept to Gaussian Mixture Models, the probability density function of each partition will be one of the Gaussian distributions that compose the GMM and the weights of the Gaussian distributions will be proportional to the number of objects of their respective partition.

The probability density function of each "partition" is estimated using the EM (Expectation-Maximization) algorithm. The EM algorithm is a method that estimates the

most probable parameters of a distribution (the maximum-likelihood) of a certain dataset, only considering a fraction of that dataset. Further reading on maximum-likelihood and the EM algorithm can be found in [36].

The classification with Gaussian Mixture Models requires methods to calculate "distances" between GMM – for example, if songs are to be classified, a GMM will be computed for each song to be classified and the song will be classified according to the distance between the song's GMM and the various genres' GMM. The song will be classified as belonging to the closest genre. This type of song classification was made with Gaussian Mixture Models, as opposed to the instance classification that was made with all other classifiers (using Weka).

The distances between Gaussian Mixture Models was calculated using the "Earth Mover's Distance" (EMD) algorithm, as presented in [37] and implemented with the MA Toolbox. The implementation of the EMD algorithm was made by Yossi Rubner [38] and can be found in [33].

The EMD algorithm calculates the distance between two Gaussian Mixture Models as the amount of "work" needed to transform one into the other. If we look at GMM as "piles of earth" centered on the means of each Gaussian distribution that constitute them, the distance between two Gaussian Mixture Models is proportional to the quantity of earth that has to be moved inside one GMM, so that it is transformed in the other GMM.

In the experiments performed, the Gaussian Mixture Models are composed of three clusters (three Gaussian distributions), their covariance type is diagonal and the distance between models is calculated with the EMD algorithm – these are the setup parameters of the computation of the GMMs using the MA Toolbox.

## 3.3 Datasets used

The classification experiments were run using two databases, the Latin Music Database [39] and the George Tzanetakis music collection, composed of 10 general music genres [9].

### 3.3.1 The George Tzanetakis collection

The George Tzanetakis Collection is divided in 10 genres: "blues", "classical", "country", "disco", "hip hop", "jazz", "metal", "pop", "reggae" and "rock". Each genre is represented with 100 audio samples from different songs, each of them 30 s long.

### 3.3.2 The Latin Music Database

The Latin Music Database consists of 10 genres, each of them composed of over than 300 songs. A brief description of each genre is going to be made below.

#### 3.3.2.1 Axé

The Axé genre originated in Salvador, Bahia (Brazil) during the 80's, in the Carnival festivities. Its origins are related with the evolutions of the "trio eléctrico", an attraction of the Carnival from the Bahia, where music was played with electric guitars on the top of a driving stage, a car or a truck. The "trio electrico" was born in 1950, when the musical duo Dodó and Osmar went out on that year's Carnival on the top of a restored old Ford playing electric guitars. The "trio eléctrico" popularity grew and developed into more grandiose manifestations, pushed by the musical experimentations of Morais Moreira, Caetano Veloso and others. An energetic mixture of fast paced drums, electric guitars, simple and cheerful melodies came to define the "axe" genre. Popular artists include Daniela Mercury, Babado Novo and Ivete Sangalo [39] [40].

#### 3.3.2.2 Bachata

Bachata originated in the poor communities living in the Dominican Republic in the early 1960's. The vocals have a dominant role, supported mainly by electric guitars (originally they were acoustic guitars), played fast with a string picking technique and a characteristic metallic high tone. The melodies are sung in an emotional way, in a high pitch, reflected by the content of the lyrics, usually sentimental declarations of love. The Bachata genre was initially considered to be too crude and vulgar and did not receive much attention by the middle and upper classes – this situation eventually changed (in the 80's and 90's), with the change to electric guitar and with the popularity of artists such as Luis Vargas, Antony Santos, Luis Guerra and Aventura [39].

*Figure 7: the bachata band "Aventura", the creative tour de force behind timeless classics such as "Amor de Madre", "Amor Bonito" or "Eso no Es Amor"*

#### 3.3.2.3 Bolero

Bolero has Afro-Cuban roots. There are two forms of this genre: a Spanish one and a Cuban one. The Spanish version of bolero originated in Spain in the late 18th century, and is a dance played in slow tempo, accompanied by music with voice, castanets and guitars.

The Cuban Bolero originated in Santiago de Cuba in the end of the 19$^{th}$ century, with itinerant musicians who earned their living by playing the guitar. Pepe Sanchez is credited to be the first *trovador* and the creator of the Cuban bolero - although he never wrote a single piece of music, he was the teacher of a number of musicians that followed. This musical genre spread throughout the rest of Latin America. Bolero composers include Rafael Hernandez (from Puerto Rico) and Agustín Lara (from Mexico). Bolero is sung in Spanish and is played with guitar, conga and bongos. It is a slow and relaxed music, with complex and subtle instrumental variations. The lyrics deal with themes related to love [39] [41].

#### 3.3.2.4 Forró

Forró originated in the Brazilian northeast. It designates a local dance with African origins, and the music that accompanies it. Forró is played (dance and music) during the Festa Junina, a Brazilian festival that celebrates the saints of the Catholic religion (namely St. John). There are some theories about the exact origin of the name "forró": as a derivative from "forróbodó" (meaning a great party) is the most popular explanation. Forró is played in different ways in different locations and has been subject to change with time. It is played mainly with three instruments: accordion, a metal triangle, and a *zamumba* bass drum).

Forró is very rhythmic, but in a slower way, where the repetitive, mellow and somewhat exuberant sound of the accordion is dominant. The melodies, conducted by the voice, are simple but joyful. There is an inspiring delicateness that makes this genre pleasant to listen to. If Axé is the soundtrack to intense physical manifestations, Forró celebrates a more relaxed attitude towards life. Forró musicians include Luiz Gonzaga, Trio Nordestino or Falamansa [39] [42].

#### 3.3.2.5 Gaúcha

Gaúcha comes from the southern part of Brazil, and designates several musical styles played in that region. During the 1970's several festivals helped popularize this musical genre. The lyrics deal with the themes of love and respect towards woman, the nature and the culture. There are some similarities between Gaúcha and Forró – the dominant role of the accordion is probably the main one. On the other hand, Gaúcha is played faster and is more repetitive. The melodies lack the subtleties of Forró and are less interesting – there is an emphasis on the danceable rhythm [39] [43].

#### 3.3.2.6 Merengue

The genre Merengue comes from the Dominican Republic, and was created in the middle of the 19th century. There are discussions about the exact origin of Merengue, but its popularity is in great part due to the embrace of this musical genre by Rafael L. Trujillo (president and dictator of the country from 1930 to 1961) since the 1930's. This genre has been used to spread news, gossip and political propaganda. Merengue is easy to recognize: it is uniquely rhythmic and supported by a mixture of instruments playing in a breathless, superfast, mechanical way. The instruments played are the accordion, bass guitar, guira, guitar, saxophone, tambora, trumpet, trombone or the tuba. There are some different types of merengue such as the "merengue cibaeño" or the "merengue palo eacho" [39] [44].

#### 3.3.2.7 Pagode

Pagode originated in the Rio de Janeiro (Brazil), as a samba sub-genre, in the 1970's. People started to get together for parties where samba was played – in these meetings the *tan-tan* was introduced, a cylindrical hand drum that replaced the *surdo*, the large bass drum played in samba music – this happened for practical reasons, as the *tan-tan* was easier to transport. Other instruments were added, such as the banjo and the *repique de mão*, a percussive instrument. The resulting sound was more intimate than the traditional samba, and had new rich percussive textures. Popular pagode artists are Zeca Pagodinho or Grupo Revelação [39].

*Figure 8: Zeca Pagodinho, the author of "Àgua da Minha Sede", the memorable song he wrote when he was thirsty*

#### 3.3.2.8 Salsa

Salsa is a style that was developed by Puerto-Rican and Cuban musicians, immigrants in New York, in the 1960's and 1970's. It can be used to describe a great number of Cuban-derived musical genres that are played throughout Latin American. A description of the sound of salsa was made by Ed Morales: "extravagant, clave-driven, Afro-Cuban-derived songs anchored by piano, horns, and rhythm section and sung by a velvety voiced crooner in a sharkskin suit". Salsa has a relaxed feel to it, and it has a rhythmic and danceable nature – the listener is taken to the middle of Havana, where musicians are playing in the streets. It is played with a variety of instruments, such as the piano, conga, trumpet, trombone, bass guitar, claves, cowbell, maraca, double bass, timbales, flute, guitar, bongos, saxophone, drum kit, vibraphone, guiro and the violin.

There are some salsa sub-genres, such as the *salsa erotica*, *salsa gorda* and the *salsa romantic*. Popular artists include Celia Cruz and Willie Colón [39] [45].

#### 3.3.2.9 Sertaneja

Sertaneja is a popular musical genre from the interior of Brazil and that has its origins in the called "caipira" music – simple rural country music, played with handmade traditional instruments. The musician Cornélio Pires is responsible for popularizing the "caipira" music in the 1920's and is considered to be a pioneer of the Sertaneja music. The 1980's saw a growth in the popularity of the Sertaneja genre, when several artists started playing a more commercial and popular version of it. Among these most successful commercial Sertaneja artists are Chitãozinho & Xororó and Leandro & Leonardo. Most sertaneja artists are duos that play guitars and ten-string violas. The lyrics focus on the usual romantic themes and on the rural life. The vocals are dominant in this genre, conducting the melody (usually "countryish", with inebriated melancholy) [39] [46].

#### 3.3.2.10 Tango

The word "Tango" characterizes a dance and its corresponding musical style. Tango originated in Argentina (in Buenos Aires) and in Uruguay (Montevideo). The origins of the word "tango" are unclear – there are indications that it came from the African languages brought to South America by the migration of slaves.

The dance expression of Tango has origins in dances from Cuba, Uruguay and from the African community in Buenos Aires. The music is influenced by European music. In the end of the 1890's, the word tango started to be used in Argentina. In the beginning of the 1900's the dance of Tango started to be included in balls and theatres and became extremely popular throughout the society. Tango developed influenced by immigrants of different ethnicities and cultures. Musicians like Roberto Fripo and Julian de Caro gave tango more classical and elegant outfits during the 1910's and 1920's. The popularity of tango survived until today and the greatest tango composers include Carlos Gardel and Ástor Piazzola [39] [47].

## 3.4 Experimental setup

### 3.4.1 Testing, training and validation datasets

The Latin Music Database was separated in two sets, one for a main set of experiments (which included training and testing) and the other for a later set of validation experiments. The main dataset (for training and testing) was composed of 160 songs from each genre, totaling 1600 songs. The validation dataset was composed of 40 songs from each genre, totaling 400 songs.

The separation of songs in the main and validation datasets was made with an artist filter (*i.e.* songs by a specific artist are either in the main dataset or in the validation dataset). The main dataset was also divided in two datasets, one for training and the other for testing. An artist filter was also applied. In the results section below, we assess the relevance of using an artist filter in this data separation.

Having the songs separated for each dataset, the features from the songs were computed using Marsyas. Only the features from 30 s from the middle of each song were computed – this was done to decrease the size of the feature file and to achieve a better musical representation, as many songs have non-characteristic beginnings (with silence, or in the case of live recordings, applause and crowded sounds).

Another transformation was applied with Weka to the feature files: as the feature files are generally very large in size, samples from the feature files were computed. Samples composed of 1 %, 5 % and 10 % of the train and test datasets were made. These samples are biased so that the numbers of instances from each genre are roughly the same. Various experiments were run using different sizes of the samples of the train and test datasets. An influence in the classification performance of the size of the datasets was looked for.

Classification experiments were made using the training dataset for training and the testing dataset for testing; using the training set to both train and test the classifier; and splitting randomly the training set instances into two datasets – one for training and the other for testing. In the former case, the artist filter is applied, in the other cases it is not. This was made with the goal to assess the influence on the classification performance of the artist filter.

The classification using Gaussian Mixture Models (with Matlab) was done in the following way: a dataset of 160 songs per genre (1600 songs total) was divided into three smaller datasets, using the artist filter – songs from a certain artist can only be contained in one of these three datasets. With these three datasets, a 3 fold cross validation classification was made: two datasets were used for creating the genre Gaussian Mixture Models and the other dataset was used for testing. This process was repeated 3 times (using a different dataset for testing each time) and the results averaged. For each genre in the training set a Gaussian Mixture Model was computed, and for each song in the test set

a Gaussian Mixture Model was computed as well. A song in the test set is classified as belonging to a certain genre if the distance between the song's GMM and that genre's GMM is smaller than the distance between the song's GMM and all other genres' GMM.

As the George Tzanetakis music collection files are not labeled with the songs' names and artists, there is no way to apply the artist filter to it. This collection, contrary to the LMD, was not separated into training, testing and validation datasets. A single feature file was computed for all collection. Due to the big size of the feature file, samples of 20 % and 25 % were used. As it happened with the Latin Music Database feature files, the samples were biased so that the genres are represented by an approximately equal number of instances. The classification was done in two ways: splitting randomly the sampled feature file into two datasets – one for training and the other for testing; and using a 10 fold cross validation procedure. In the 10 fold cross validation the dataset is split into 10 folds, 9 folds are used for training and the other is used for testing. This is repeated 9 times, each time using a different fold for testing and the results are averaged.

### 3.4.2 Analysis measures

Some measures were adopted to analyze the classification results: the True Positive (TP) Rate, the False Positive (FP) Rate, Precision and the F-Measure.

The TP Rate of a genre A is the proportion of instances (or songs) that were correctly classified as belonging to genre A, among all instances that truly belong to genre A. The FP Rate of a genre A is the proportion of instances which were classified as belonging to genre A, among all the instances that do not belong to genre A. The Precision of a genre A is the proportion of instances that do indeed belong to genre A, among all the instances that were classified as belonging to genre A. The F-Measure measures both the TP Rate and the Precision, and is given by the formula *F-Measure = 2 * Precision * TP Rate / (Precision + TP Rate)*. These measures are adopted from the Weka terminology. The average results (from all genres) of the TP Rate, FP Rate, Precision and F-Measure were used to measure the performance of classifiers – particularly the average TP Rate, which is the rate of correctly classified instances (or songs).

The baseline for all these measures is 0.1: the baseline for the TP Rate is 0.1 because if there was a random classification being made, the rate of correctly classified songs, or instances, would be 1/10 (there are ten genres being considered). This rate should grow as the classification improves. The baseline for the FP Rate is 0.1 because if there are 10 songs per genre being classified and ten genres total, a random classification would result in 9 songs being classified as belonging to a genre, out of the 90 songs that do not belong to that genre. This rate should decrease as the classification improves (there will be less songs being incorrectly classified as a certain genre). The baseline for the Precision is 0.1 because, using the example above, a random classification will classify 10 songs as belonging to a genre, and from those 10 songs, only 1 will indeed belong to that genre.

The baseline for the F-Measure is 0.1, as it is calculated in the formula: *baseline (F-Measure) = 2 * baseline (Precision) * baseline (TP Rate) / baseline (Precision) + baseline (TP Rate).*

### 3.4.3 The "artist filter"

From the Latin Music Database, two datasets were created: one dataset for the main experiments and other dataset for later validation experiments. The main dataset is composed of 160 songs per genre (1600 songs total) and the validation dataset is composed of 40 songs per genre (4000 songs total). An artist filter was used.

A Matlab script was written to achieve this dataset separation: songs were picked up randomly from the Latin Music Database and filled in the two datasets until their size reached the numbers of 160 songs per genre and 40 songs per genre, for the main dataset and validation dataset, respectively. At the same time, an artist list was created, that indicated to which dataset did the artists from the picked songs belonged to. A brief description of the algorithm is made below:

1. One song is picked up randomly;
2. If the song has already been picked, go to 1;
3. If the song's artist is present in the main dataset, go to 1;
4. The song is written in the validation dataset; auxiliary variables are updated;
5. One song is picked up randomly;
6. If the song has already been picked, go to 5;
7. If the song's artist is present in the validation dataset, go to 5;
8. The song is written in the main dataset; auxiliary variables are updated;
9. Steps 5-9 are repeated four times;
10. Repeat 1-9 until 2000 songs are written (1600+400).

First, a song in the validation dataset is written and then four songs in the main dataset are written. This procedure is repeated until the main dataset reaches the size of 160 songs per genre and the validation dataset reaches the size of 40 songs per genre.

The main dataset was also separated into two smaller datasets of training and testing, also with an artist filter.

A different algorithm was used here: for each genre an artist list was created, indicating the number of songs per artist. The songs from the "less popular" artists (the artists with less number of songs) were picked and filled in the testing dataset until the number of songs per genre contained in this dataset reached the minimum number of 16 songs (one tenth of the total songs per genre in the main dataset, which is 160). It is better to illustrate

this with an example: let's imagine the less popular artists in the Axé genre have only 3 songs: there are 5 artists with 3 songs each and a lot more artists with 4 songs each, 5 songs each and so on. The songs from the five 3-song artists will be picked, totaling 15 songs. This number is less than 16, so one 4-song artist will be picked and its songs filled in the testing dataset, totaling 19 songs. These 19 Axé songs will be put in the testing dataset and the other Axé songs (160-19=141, in the example given) will be put in the training dataset.

In the experiences with the Gaussian Mixture Models, there was also a creation of 3 datasets with the artist filter. From a main dataset of 160 songs per genre (1600 songs total), 3 datasets of the same size were created. The algorithm to do so was the following: the number of artists and the number of songs of each artist was counted; then, artists were put into the three datasets, starting with the most "popular" artists (the artists with the greater number of songs) and then with the artists with the least number of songs. The algorithm was the following:

1. Pick the artist with the greater number of songs, that has not been chosen yet;
2. Place the artist's songs into the dataset with the least number of songs (if there are two or three datasets with an equal number of songs, just choose one dataset);
3. Update the number of songs of the dataset and auxiliary variables;
4. Repeat the procedure until all artists were picked.

With this method three datasets were created, each with either 53 or 54 songs per genre, and without artists present in two datasets at the same time.

# Chapter 4

# 4. Results and analysis

## 4.1 Results

Four classifiers were used with the platform Weka: KNN, J48, SVM and Naive Bayes. These classifiers were used with both the Latin Music Database and the George Tzanetakis music collection. Classification with Gaussian Mixture Models was only made with the Latin Music Database.

### 4.1.1 Results using the GTZAN collection

With this collection two different testing methods were conducted: on the first one, a sample of 25 % of the total instances was created and this sample was split into two datasets, one for training the classifiers, the other for testing – the size of the training dataset was 90 % of the created sample, and the size of the testing dataset was 10 %. The other classification experiments were made using a sample of 20 % of the total instances and using the 10 fold cross validation method to conduct classification experiments. The results (percentage of correctly classified instances or TP Rate) of these experiments are shown in table 1. The number of instances considered in the experiments is shown in table 2.

| **Resample** | **25%** | **20%** |
|---|---|---|
| **Test option:** | **Split 90%/10%** | **10 fold cross validation** |
| **Naive Bayes** | 51.6% | 51.5% |
| **KNN** | 99.9% | 99.9% |
| **J48** | 94.7% | 93.6% |
| **SVM** | 84.4% | 83.3% |

*Table 1: Results obtained with the GTZAN collection*

| **Sample Percentage** | **Number Of Instances** |
|---|---|
| **25%** | 161 749 |
| **20%** | 129 399 |

*Table 2: Number of instances used in the GTZAN experiments*

### 4.1.2 Results using the LMD collection

On the Latin Music Database, classification tests were conducted with and without the artist filter on datasets of different sizes with different ways of testing and using the Weka classifiers and also Gaussian Mixture Models (with Matlab).

The results obtained without the artist filter are presented below. In these experiments only the training dataset was used – the testing was done with it or with a fraction of it.

The table 3 shows the results (TP Rate) obtained using resample percentages of 1 %, 5 % and 10 % of the training set. The classification was done using the same instances to train and test the classifiers (testing is done with the training set); and splitting randomly the instances into training and testing datasets (90 % for training and 10 % for testing).

The table 4 shows the results (TP Rate) obtained using a fixed resample percentage of 5 %. Three different proportions between the number of the testing and training instances were considered: 50%/50%, 70%/30% and 90%/10%.

| **Resample** | **1%** | **1%** | **5%** | **5%** | **10%** |
|---|---|---|---|---|---|
| **Test** | **Training Set** | **Split 90%/10%** | **Training Set** | **Split 90%/10%** | **Split 90%/10%** |
| **Naive Bayes** | 46% | 46.2% | 46.6% | 46.2% | 46.1% |
| **KNN** | 91.5% | 76.6% | 99.1% | 96.2% | 99% |
| **J48** | 93.1% | 54.7% | 95.9% | 75.7% | 83.6% |
| **SVM** | 63.4% | 59.6% | 69.6% | 67.6% | 71.4% |

*Table 3: Results with the LMD, no artist filter*

| **Resample** | **5%** | **5%** | **5%** | **5%** |
|---|---|---|---|---|
| **Test** | **Training Set** | **Split 90%/10%** | **Split 70%/30%** | **Split 50%/50%** |
| **Naive Bayes** | 46% | 46.2% | 46.3% | 46.3% |
| **KNN** | 99.1% | 96.2% | 94.4% | 90.6% |
| **J48** | 95.9% | 75.7% | 72.5% | 68.1% |
| **SVM** | 69.6% | 67.6% | 66.6% | 64.9% |

*Table 4: Results with the LMD, no artist filter, resample fixed*

The results obtained with the artist filter are presented in the table 5. In this set of experiments, two different datasets were used, one for training and the other for testing. These datasets were created with an artist filter, as described in the chapter 3.4.3. Samples of 1 %, 5 % and 10 % of the training and testing instances were used. The results (TP Rate) are presented in table 5 for different combinations of the sizes of the training and testing datasets.

Table 6 shows the number of instances considered in both the training and testing datasets, for the different used sample percentages.

| **Train Resample** | **1%** | **1%** | **5%** | **5%** | **10%** |
|---|---|---|---|---|---|
| **Test Resample** | **1%** | **5%** | **1%** | **5%** | **10%** |
| **Naive Bayes** | 40.0% | 39.5% | 41.1% | 41.1% | 40.0% |
| **KNN** | 45.2% | 45.6% | 45.9% | 45.8% | 45.2% |
| **J48** | 38.8% | 37.5% | 40.1% | 39.0% | 34.0% |
| **SVM** | 43.2% | 43.6% | 43.7% | 45.1% | 44.8% |

*Table 5: Results with the LMD with the artist filter*

| **Sample Percentage** | **Training Set** | **Testing Set** |
|---|---|---|
| **10%** | 177 147 | 19 574 |
| **5%** | 88 573 | 9787 |
| **1%** | 17 714 | 1957 |

*Table 6: Number of instances used in the LMD experiments*

The results obtained with the Gaussian Mixture Models are represented in the tables 7 and 8. In this set of experiments, the testing was made using full songs – instead of instances. Table 7 is a confusion matrix; it shows how the songs from the ten LMD genres were classified. Table 8 shows the TP Rate, the FP Rate, the Precision, and the F-Measure, for each genre.

| | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **<-classified as** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 160 | 95 | 2 | 2 | 5 | 0 | 6 | 13 | 1 | 35 | 1 | **a=axe** |
| 160 | 2 | 110 | 5 | 2 | 6 | 7 | 14 | 3 | 11 | 0 | **b=bachata** |
| 160 | 3 | 0 | 84 | 5 | 2 | 2 | 32 | 6 | 25 | 1 | **c=bolero** |
| 160 | 22 | 3 | 4 | 13 | 23 | 13 | 35 | 5 | 42 | 0 | **d=forro** |
| 160 | 46 | 0 | 5 | 20 | 25 | 9 | 22 | 0 | 33 | 0 | **e=gaucha** |
| 160 | 10 | 1 | 1 | 2 | 8 | 118 | 9 | 5 | 6 | 0 | **f=merengue** |
| 160 | 34 | 3 | 10 | 6 | 3 | 0 | 66 | 1 | 36 | 1 | **g=pagode** |
| 160 | 31 | 3 | 20 | 10 | 7 | 10 | 35 | 13 | 31 | 0 | **h=salsa** |
| 160 | 47 | 1 | 12 | 3 | 0 | 0 | 23 | 4 | 70 | 0 | **i=sertaneja** |
| 160 | 34 | 0 | 36 | 2 | 8 | 1 | 10 | 2 | 25 | 42 | **j=tango** |
| | 324 | 123 | 179 | 68 | 82 | 166 | 259 | 40 | 314 | 45 | |

*Table 7: Confusion Matrix obtained using classification with GMM*

| **TP Rate** | **FP Rate** | **Precision** | **F-Measure** | |
|---|---|---|---|---|
| 0.594 | 0.159 | 0.293 | 0.392 | **Axe** |
| 0.688 | 0.00903 | 0.894 | 0.778 | **Bachata** |
| 0.525 | 0.0660 | 0.469 | 0.495 | **Bolero** |
| 0.0812 | 0.0382 | 0.191 | 0.114 | **Forro** |
| 0.156 | 0.0396 | 0.305 | 0.206 | **Gaucha** |
| 0.738 | 0.0333 | 0.711 | 0.724 | **Merengue** |
| 0.412 | 0.134 | 0.255 | 0.315 | **Pagode** |
| 0.0812 | 0.0188 | 0.325 | 0.130 | **Salsa** |
| 0.438 | 0.169 | 0.223 | 0.296 | **Sertaneja** |

| | | | | |
|---|---|---|---|---|
| 0.262 | 0.00208 | 0.933 | 0.409 | **Tango** |
| 0.398 | 0.0669 | 0.46 | 0.386 | **Average** |

*Table 8: Classification performance by class using GMM*

A comparison between the TP Rate, FP Rate, Precision and F-Measure obtained with the used classifiers is represented in the table 9. The results of the Weka classifiers were obtained with the artist filter, using 10 % samples of both the training and testing datasets. The GMM results are the "average" results of the table 8.

| | **TP Rate** | **FP Rate** | **Precision** | **F-Measure** |
|---|---|---|---|---|
| **Naive Bayes** | 0.400 | 0.066 | 0.395 | 0.382 |
| **KNN** | 0.452 | 0.061 | 0.452 | 0.449 |
| **J48** | 0.340 | 0.074 | 0.341 | 0.330 |
| **SVM** | 0.448 | 0.061 | 0.452 | 0.438 |
| **GMM** | 0.398 | 0.067 | 0.460 | 0.386 |

*Table 9: Classification performance comparison*

## 4.2 Classifier comparison

The classification performance using the George Tzanetakis collection is great for all classifiers except Naïve Bayes. An absurdly high classification TP Rate was obtained with the KNN classifier (99.9 %). The TP Rate using J48 is around 94 % and using SVM, around 84 %. With Naive Bayes the results are more modest, in the order of 51 %. There are not significant classification differences using for testing 10 % of the training set, or using 10 fold cross validation.

On the Latin Music Database, without the artist filter, in all experiences but one, KNN achieves the highest TP Rate, J48 comes in second, SVM comes in third, and Naive Bayes achieves the worst results. In one experience (when the resample is 1 %, and there is 90%/10% split between training and testing), KNN comes in first, then SVM, J48 and finally Naive Bayes.

When using the artist filter, KNN achieves the best results between 45.2 % and 45.9 % of correctly classified instances; SVM achieves results in the range between 43.2 % and 45.1 %; Naive Bayes achieves results between 39.5 % and 41.1 %; and J48 achieves results between 34 % and 40.1 %. Using the Gaussian Mixture Models, the percentage of correctly classified songs (the average TP Rate) is 39.8 %.

## 4.3 On the influence of the artist filter

The results using the artist filter are considerably worse than the ones obtained without it. The best classifications results are achieved with KNN and still its TP Rate is no better

than 50 % (it is around 46 %). SVM achieves a TP Rate of between 43 % and 45 %, J48 achieves between 35 % and 40 %, and Naive Bayes' results are around 40%.

On the other hand, the classification results obtained without the artist filter are much better: a TP Rate of 99 % was achieved using KNN, of 84 % with J48, and of 72 % with SVM.

The big influence of the artist filter is expected in a way: the nature of the instances classified is one where consecutive instances from the same songs are very similar; and without the application of the artist filter it is likely to happen that the classifiers are trained and tested with similar instances, from the same songs. This fact is the reason behind the high classification results achieved when the artist filter was not used.

## 4.4 On the influence of the dataset nature

The comparison between the results using the GTZAN collection and the LMD must be made using the results achieved with the LMD without the artist filter since the artist filter was not applied to the GTZAN collection. The results obtained with the GTZAN collection are usually better: comparing the results obtained using the GTZAN collection (using a sample 25 % of the total features file, totaling 161 749 instances, from which 90 % are used for training and 10 % are used for testing) with the results obtained using the LMD collection (using a sample of 10 % of the training set features, totaling 177 147 instances, from which 90 % are used for training and 10 % are used for testing), we see that:

- Naive Bayes performs slightly better with the GTZAN dataset than with the LMD dataset (51.6 % of correctly classified instances, to 46.1 %);
- KNN performance is as outstanding as it is suspicious, for both datasets (99.9 % correctly classified instances with the GTZAN dataset; and 99 % with the LMD);
- J48 performs better with the GZAN dataset (94.7 % to 83.6 %);
- SVM performs better with the GTZAN dataset (84.4 % to 71.4 %).

The fact that the results obtained with the GTZAN music collection are better than the ones obtained with the LMD collection can be explained by the nature of the collections considered: the George Tzanetakis collection is composed of ten "general" genres ("blues", "classical", "metal", "reggae", etc), and the Latin Music Database is composed of ten genres that can be considered as being part of a larger "Latin" genre, as some of the genres share a great deal of similarities between them, and are certainly hard to distinguish by an untrained ear. On the other hand, the genres of the GTZAN collection are much easier to distinguish.

## 4.5 On the influence of the testing mode

The influence of the testing mode can be seen in the table 4. The sample percentage used is fixed (5 %), and the testing is done in two ways: using the training set for testing (all instances are used for training and for testing) and splitting the dataset into training and testing sets of different proportions – 90%/10%; 70%/30%; 50%/50% . The classification results (as in percentage of correctly classified instances, or TP Rate) are shown below, in the figure 9:

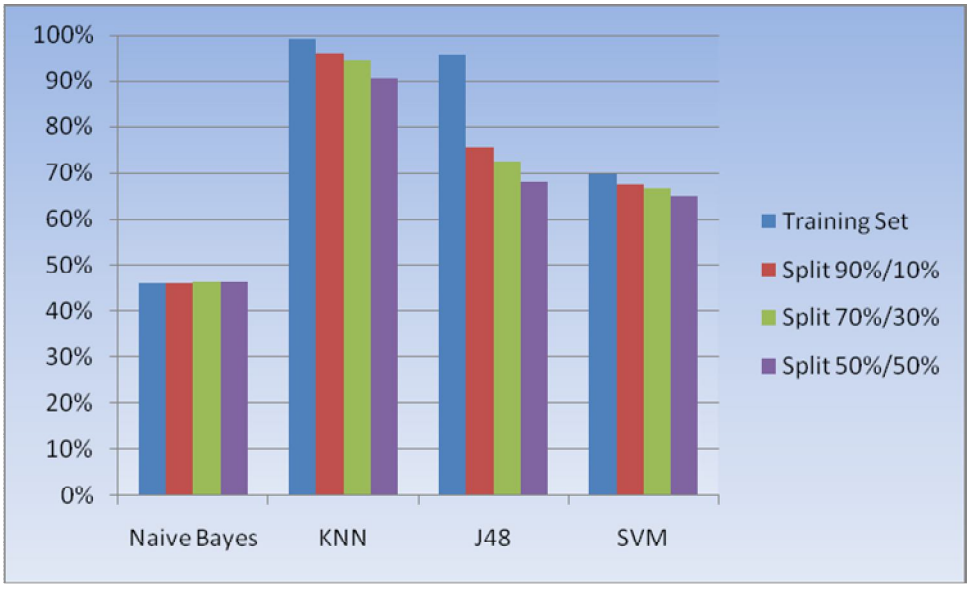


*Figure 9: Influence on classification of the testing/training percentage*

One of the first things that comes to mind when looking at figure 9 is the fact that the performance of Naive Bayes is indifferent to variations in the training and testing dataset sizes (it is constant around 46 %). To all other classifiers (KNN, J48 and SVM), the performance increases gradually when the proportion between training/testing dataset sizes goes in the direction of a larger training set and a smaller testing set. Still, the best results are achieved when the training and testing sets are the same. This result is expected: if the instances used for testing are the same instances that are used for training, one should expect better classification results than if the testing instances are not used for training.

With the GTZAN collection, classification was made with two testing methods: using a percentage split (90 % used for training and 10 % used for testing), and using a 10 fold cross validation. The results can be seen in table 1 and in the figure 10. The results obtained using the percentage split and the 10 fold validation are very similar, for all classifiers. It can be concluded that the 10 fold cross validation procedure might not be justifiable (it takes 10 times the time of the 90/10 percentage split), if the dataset used is

large enough – over 160 000 instances were used in the percentage split classification, obtained after a random resample of 25 % of a larger instance dataset.

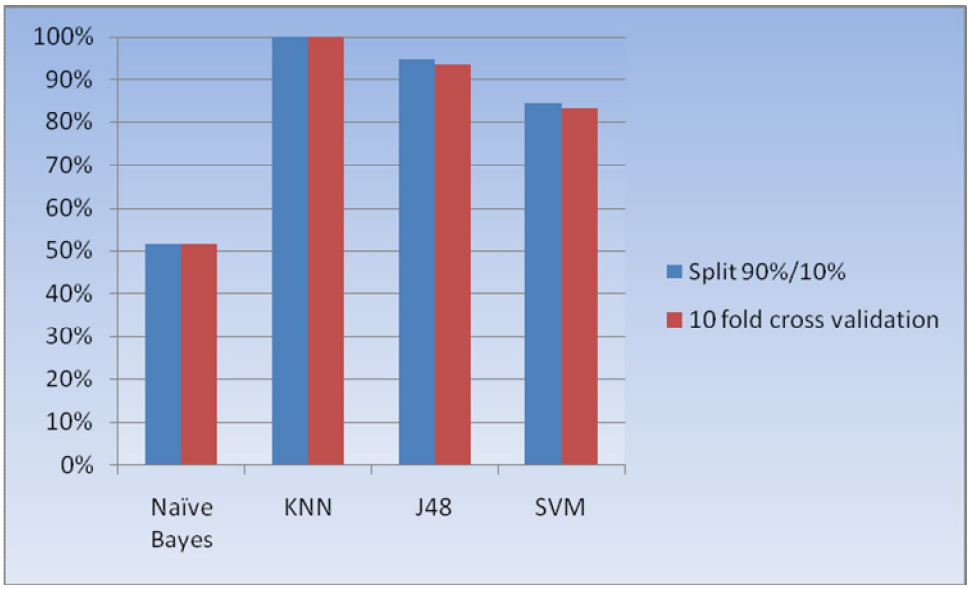


*Figure 10: Split 90/10 Vs 10 fold cross validation (GTZAN)*

## 4.6 On the influence of the dataset size

The influence of the dataset size can be seen in figures 11 and 12. On the figure 11, the testing was always made using the training set. Experiments were made using a sample of 1 % and of 5 % of the instances of the training dataset. A sample of 1 % of the training file corresponds to a number of 17 714 instances, and a sample of 5 % corresponds to a number of 88 573 instances. It can be seen that the classification performance is better when using a resample of 5 %, for all classifiers. This result is expected as the quality of the training should improve as the dataset used for it grows in size. Also, it has to be noted that the testing was made with the training set – the same instances used to train the classifier are then used to test it - and this is why the classification results are extremely high.

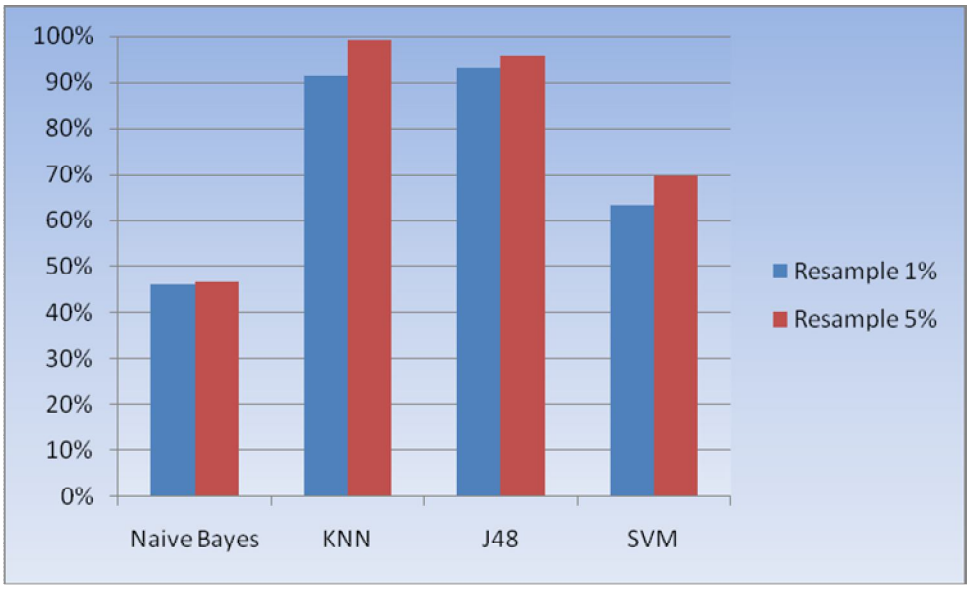


*Figure 11: Comparison between different dataset sizes (LMD, training set)*

Figure 12 represents experiments using resamples of 1 %, 5 % and 10 % of the instances of training dataset. The testing was made always using a percentage split of 90%/10%. On all experiments but Naive Bayes, all classifiers improve their performance as the training/testing dataset grows larger. As in the previous situation, this result could be explained with the fact that the quality of the training improves with the size of the training data set. Other way to explain these results is to understand that there was no artist filter applied here and a growth in the size of the training and testing datasets will improve the odds that very similar instances (from the same artists and from the same songs) are found in the two datasets. This hypothesis is confirmed in the figure 13.

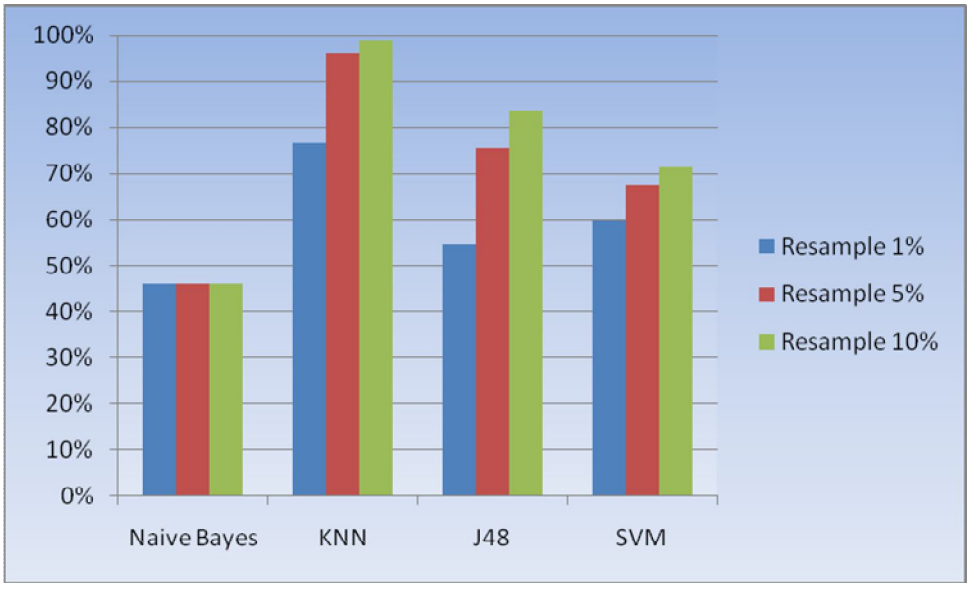


*Figure 12: Comparison between different dataset sizes (LMD, percentage split)*

Figure 13 shows the results obtained with different dataset sizes, when using the artist filter. The dataset sample values are for both the training and testing datasets.

Using the artist filter, there is not a big difference on the classification results obtained when using different training and testing dataset sizes. In figure 13 the experiments shown were made using dataset samples of 1 %, 5 % and 10 %.

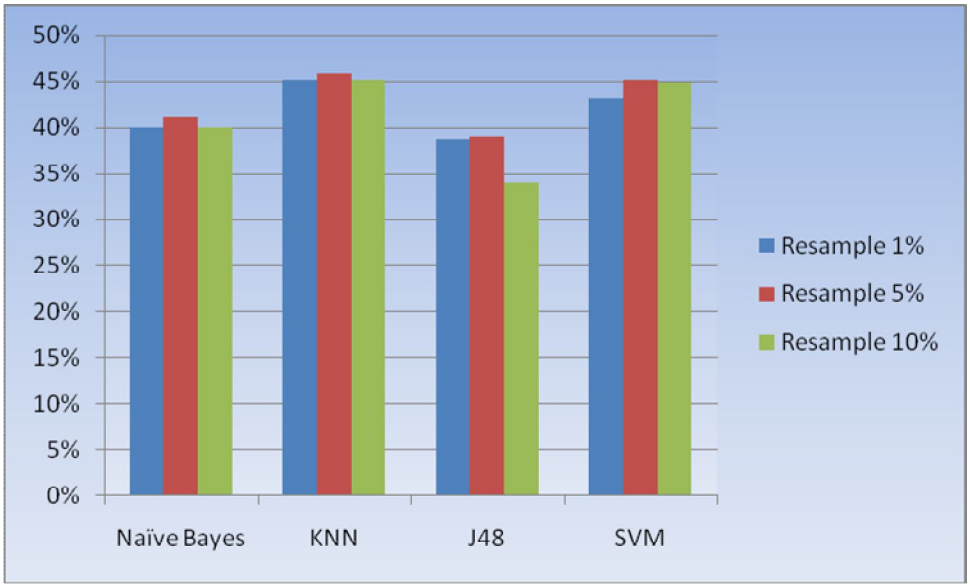


*Figure 13: Comparison between different dataset sizes (LMD, artist filter)*

The classification performance obtained with the artist filter is much worse than the performance obtained without it and unlike the no-artist filter situation, there is not an increase of the classification performance as the datasets grow larger.

It could be said that the increase in the classification performance as the size of the dataset becomes larger, when the artist filter is not used, is due to the fact that as the training and testing datasets increase in size, they are more likely to share very similar instances (from the same artists or songs), which will improve the obtained classification results.

## 4.7 Analysis of confusion matrices and genre similarities

The confusion matrix obtained with the classification with Gaussian Mixture Models is analyzed below. The classification was made with a dataset of 160 songs per genre in a procedure described in the chapter 3.4.1. The details of the classification of the 160 genre songs, using GMM, are represented in the figures 14 to 23.

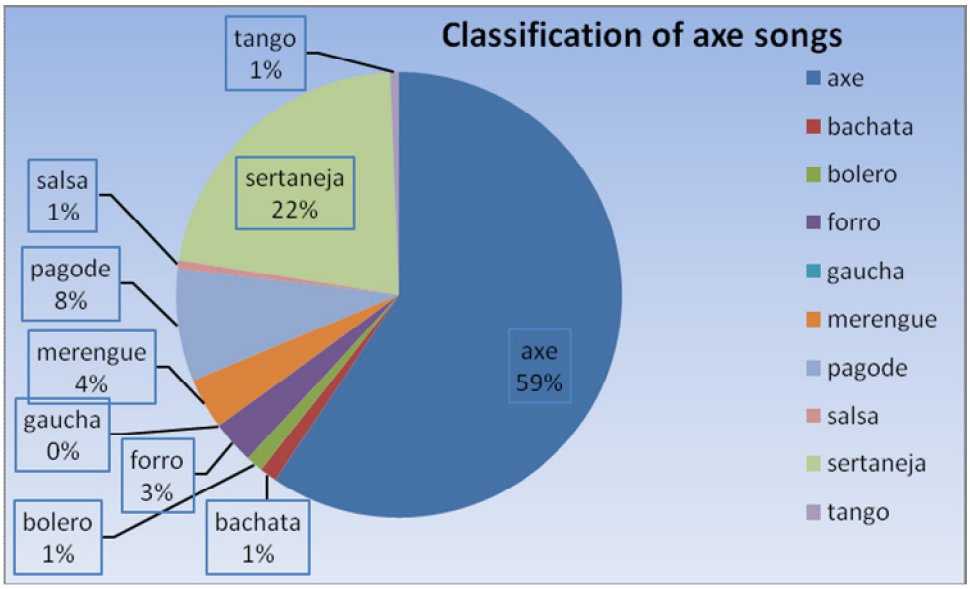


*Figure 14: Classification of Axé songs*

The classification of the Axé songs is illustrated in the figure 14: 59 % of the songs (95 out of 160) were correctly classified as Axé. Then, 22 % of songs (35 songs) were classified as Sertaneja, 8 % of the songs were classified as Pagode, 4 % as Merengue and 3 % as Forró. The rest of the genres got classification results of 0 % and 1 %. These results suggest a certain similarity between Axé and Sertaneja (and perhaps Pagode), but also the existence of a distinctiveness between Axe and several genres, such as Gaúcha and Bachata.

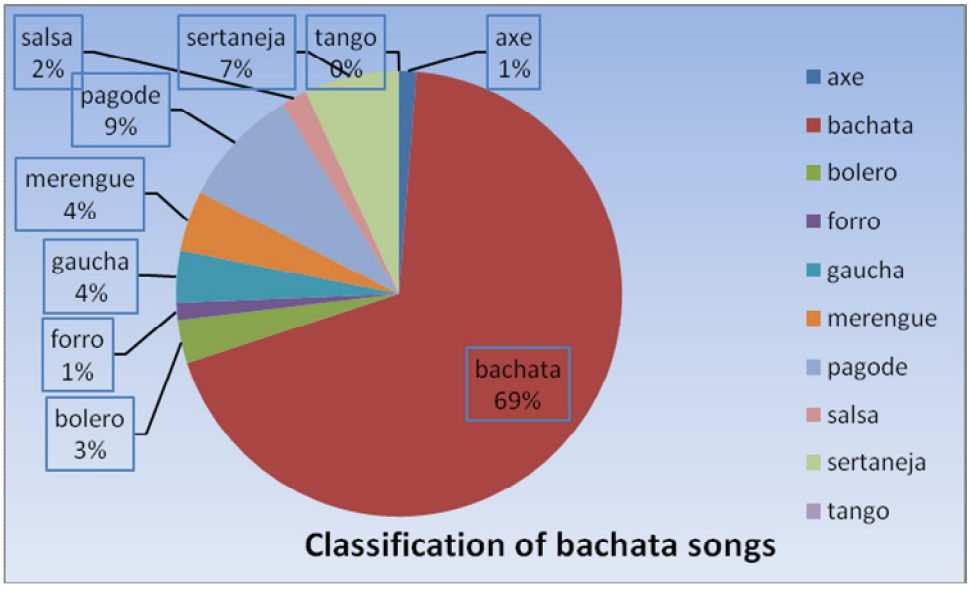


*Figure 15: Classification of Bachata songs*

Figure 15 illustrates the classification of the Bachata songs: 69 % of the Bachata songs are correctly classified as Bachata (110 songs out of 160). 9 % of the songs are classified as Pagode and 7 % of the songs are classified as Sertaneja. The rest of the genres achieve

classifications in the range between 0 % and 4 %. These results suggest that Bachata is well defined in the universe of the Latin Music Database.

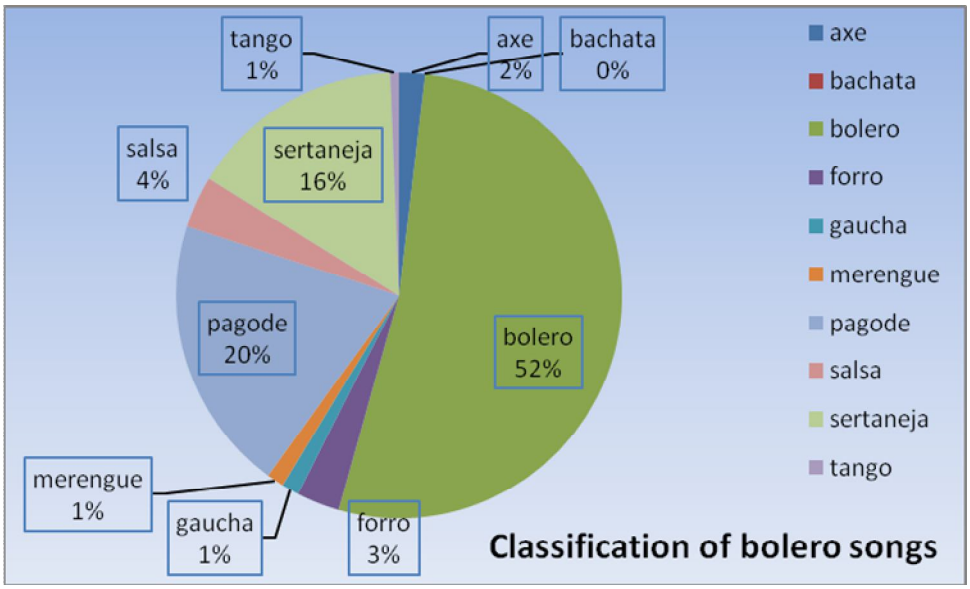


*Figure 16: Classification of Bolero songs*

Figure 16 illustrates the classification of the Bolero songs: 52 % of the songs (84 songs out of 160) were correctly classified as Bolero. 20 % of the songs were classified as Pagode and 16 % of the songs were classified as Sertaneja. The rest of the genres achieved classifications in the range between 0 % and 4 %. 88 % of the Bolero songs were classified as Bolero, Pagode and Sertaneja. This suggests a closeness between these three genres, and an apparent distinctiveness between Bolero and the rest of the genres.

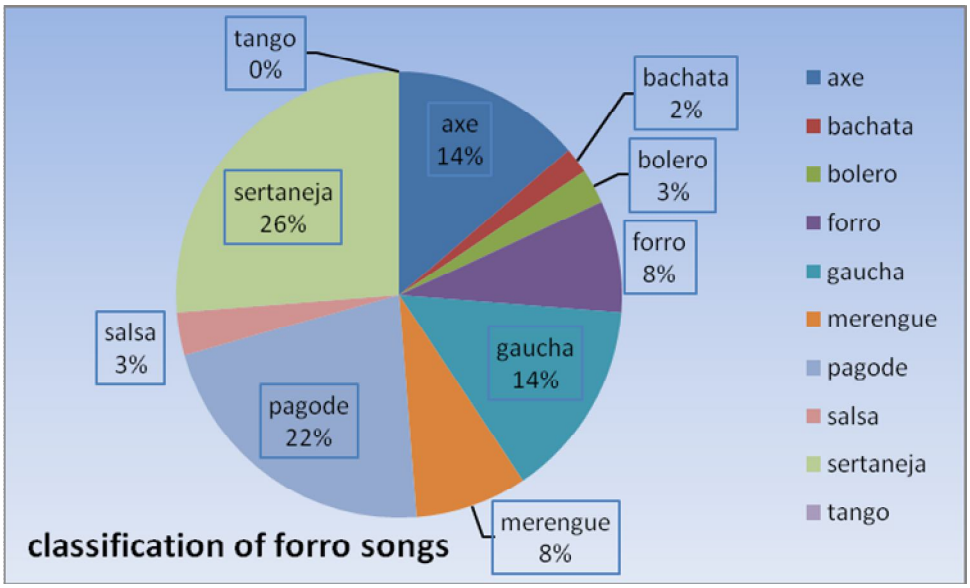


*Figure 17: Classification of Forró songs*

The classification of the Forró songs is illustrated in figure 17. Only 8 % of the songs (13 songs) were correctly classified as Forró. 26 % of the songs are classified as Sertaneja, 22 % of the songs are classified as Pagode, 14 % of the songs are classified as Axé and Gaúcha and 8 % of the songs are classified as Forró and Merengue. The rest of the genres achieve classifications between 0 % and 3 %. This suggests that Forró has timbral characteristics that are shared by other genres – namely Sertaneja, Axé, Pagode and Gaúcha.

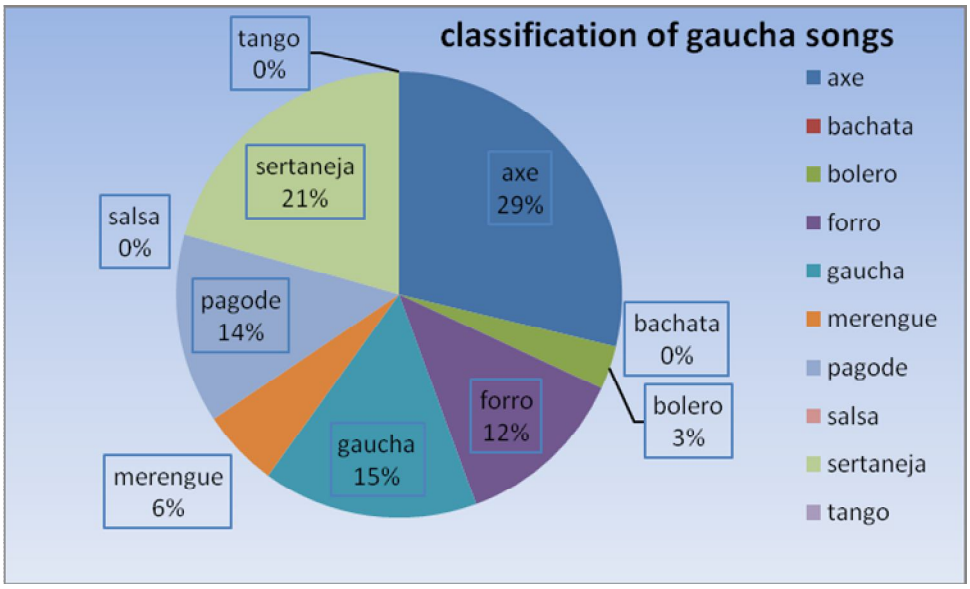


*Figure 18: Classification of Gaúcha songs*

Figure 18 illustrates the classification of Gaúcha songs: only 15 % of the songs (25 songs out of 160) are correctly classified as Gaúcha. 29 % of the songs are classified as Axé, 21 % of the songs are classified as Sertaneja, 14 % of the songs are classified as Pagode, 12 % of the songs are classified as Forró and 6 % of the songs are classified as Merengue. The rest of the genres achieved classifications between 0 % and 3 %. The majority of the songs are classified as Axé, Sertaneja, Pagode, Gaúcha and Forró. The results are relatively similar to the ones obtained when classifying the Forró songs – this might suggest a timbral similarity between Forró and Gaúcha. Also, one can see that Sertaneja and Pagode achieve high classifications in all genres so far (Axé could be mentioned as well – a great deal of the Forró and Gaúcha songs were classified as Axé, but that did not happen in the classification of Bachata and Bolero songs).

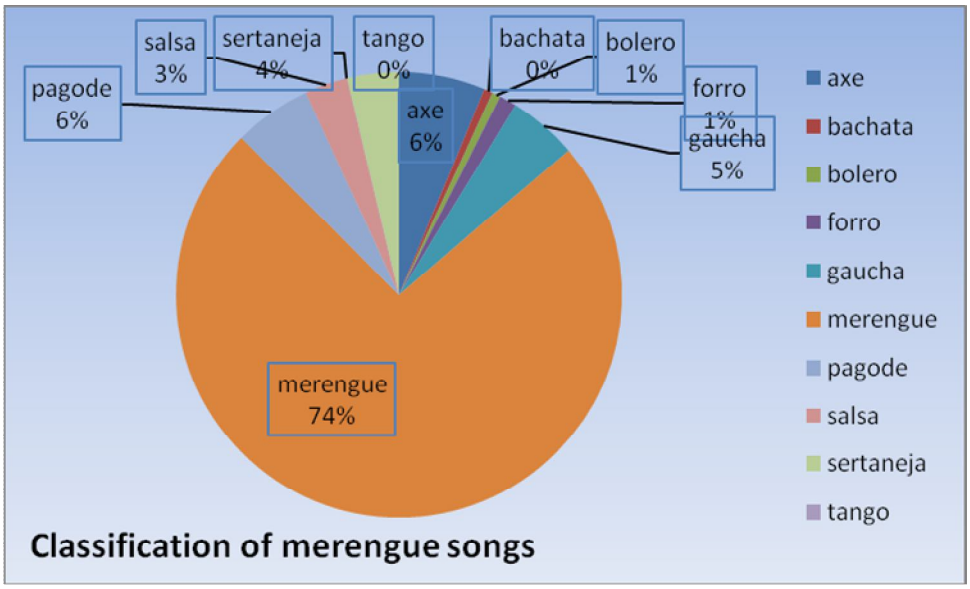


*Figure 19: Classification of Merengue songs*

Figure 19 illustrates the classification of Merengue songs: 74 % of the songs (118 out of 160) are correctly classified as Merengue. Then, 6 % of the songs are classified as Pagode, another 6 % of the songs are classified as Axé, 5 % are classified as Gaúcha, 4 % are classified as Sertaneja and 3 % are classified as Salsa. The other genres achieve classifications of 0 % and 1 %. These results suggest that Merengue is very well defined in its timbre among all other LMD genres.

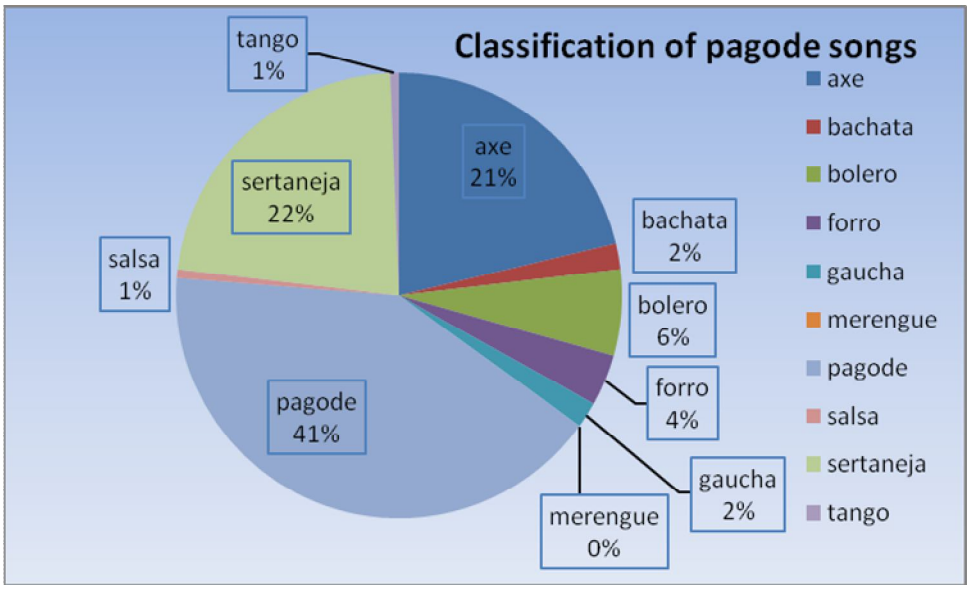


*Figure 20: Classification of Pagode songs*

Figure 20 illustrates the classification of Pagode songs: 41 % of the songs (66 songs) were correctly classified as Pagode. 22 % of the songs were classified as Sertaneja and 21 % of the songs were classified as Axé. The rest of the genres achieve results between 0 % and 6

%. This suggests a timbral closeness between Sertaneja, Pagode and Axé (84 % of the songs are classified into one of these three genres).

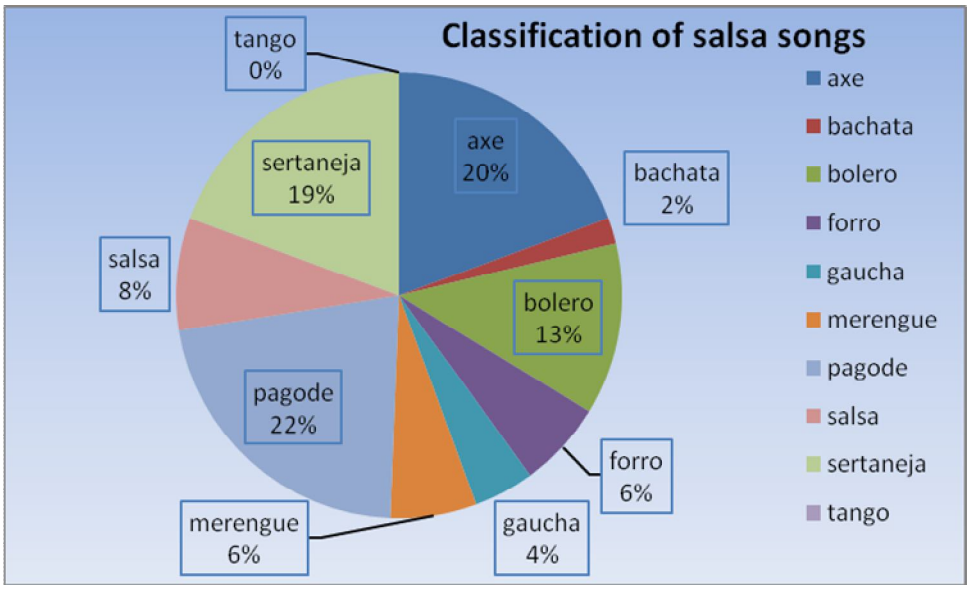


*Figure 21: Classification of Salsa songs*

The classification of the Salsa songs is illustrated in figure 21: 22 % of the songs are classified as Pagode, 20 % of the songs are classified as Axé, 19 % of the songs are classified as Sertaneja, 13 % of the songs are classified as Bolero and only 8 % of the songs are classified as Salsa. The rest of the genres achieve results between 0 % and 6 %. These results suggest that Salsa is not very well defined in the characteristics of timbre, among the LMD genres. The high classification of Axé, Sertaneja and Pagode follows the trend of the previous genres. The number of songs classified as Bolero (13 %) could suggest a timbral closeness between this genre and Salsa.

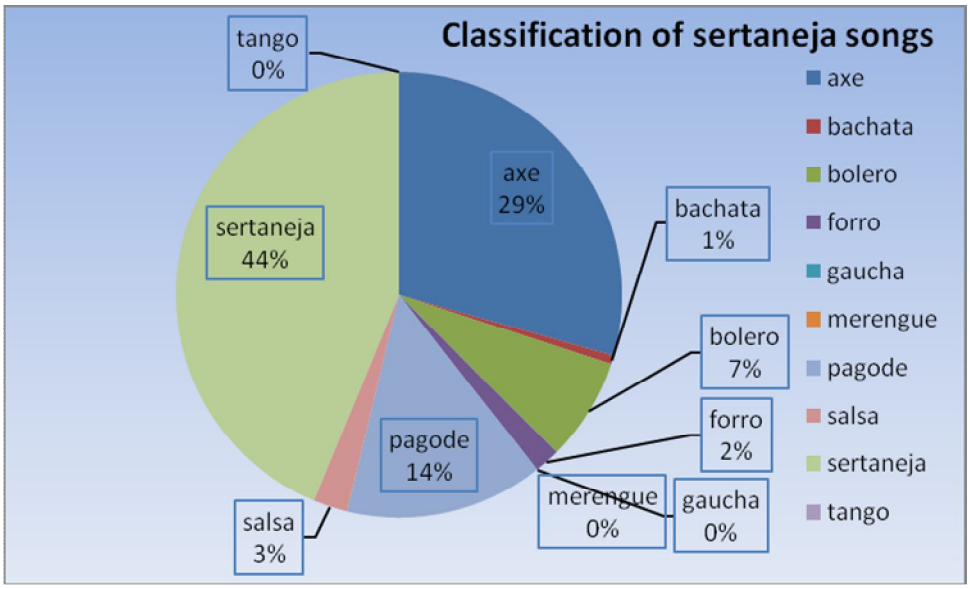


*Figure 22: Classification of Sertaneja songs*

The classification of the Sertaneja songs is illustrated in the figure 22: 44 % of the songs were correctly classified as Sertaneja, 29 % of the songs were classified as Axé, 14 % of the songs were classified as Pagode, and 7 % of the songs were classified as Bolero. The rest of the genres achieved results between 0 % and 3 %. As in previous results, a timbral closeness between Sertaneja, Axé and Pagode is suggested.

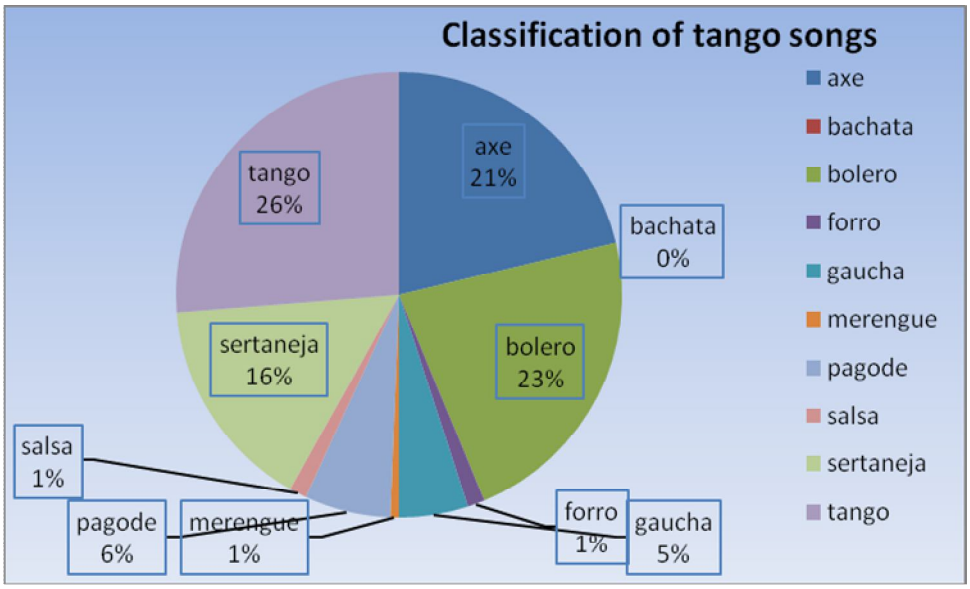


*Figure 23: Classification of Tango songs*

The classification of the Tango songs is illustrated in the figure 23: 26 % of the songs are correctly classified as Tango, 23 % of the songs are classified as Bolero, 21 % of the songs are classified as Axe and 16 % of the songs are classified as Sertaneja. The rest of the genres have classifications in the range between 0 % and 6 %.

Some things should be said after analyzing these results. Some genres were many times in the highest classification spots, namely: Sertaneja had results over 15 % in 8 out of ten genres (all but Bachata and Merengue); Pagode had results over 6 % in all genres (and results over 14 % in 6 genres – all but Axé, Bachata, Merengue and Tango); Axé had results over 20 % in 6 genres – all but Bachata, Bolero, Forró and Merengue. This suggests that these genres have timbral characteristics similar to a great number of other genres. This is confirmed by the FP Rate in table 5: Axé, Sertaneja and Pagode achieved the highest values.

On the other hand, there are some genres that can be considered to be abnormally far away from all the others: only 45 songs (out of the all genre total of 1600 songs) were classified as Tango (only 3 of them were not tango); only 40 songs were classified as Salsa (here, on the contrary, only 13 of them were indeed Salsa). Forró and Gaúcha also had low values: only 68 songs were classified as Forró, and only 82 songs were classified as Gaúcha.

The genres Bachata and Merengue should be addressed: these two genres achieved the highest percentage of correctly classified songs (69 % and 74 % respectively), and the incorrectly classified songs were distributed among several other genres (no other genre achieved a double digits percentage). This could mean that Bachata and Merengue are well defined in their timbral characteristics, among the LMD genres.

The distances between the genre Gaussian Mixture Models were calculated, in the same way the distances between genre GMM and song GMM were calculated before. These values give indications of the similarities between the genre models. These distance values range from 0 to 337.8 (this highest distance is between Merengue and Bolero). These distance values are represented in the table 10:

| | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** |
|---|---|---|---|---|---|---|---|---|---|---|
| **a=axe** | 0 | 188,2 | 175,8 | 66,8 | 31,7 | 84,3 | 67,6 | 42,1 | 48,6 | 138,0 |
| **b=bachata** | 188,2 | 0 | 155,2 | 74,1 | 119,2 | 111,8 | 88,6 | 90,8 | 140,5 | 202,3 |
| **c=colero** | 175,8 | 155,2 | 0 | 97,9 | 161,3 | 337,8 | 62,2 | 110,4 | 74,2 | 84,3 |
| **d=forro** | 66,8 | 74,1 | 97,9 | 0 | 32,5 | 77,3 | 22,4 | 43,6 | 40,1 | 147,5 |
| **e=gaucha** | 31,7 | 119,2 | 161,3 | 32,5 | 0 | 75,5 | 53,2 | 37.0 | 43,6 | 153,1 |
| **f=merengue** | 84,31 | 111,8 | 337,8 | 77,3 | 75,5 | 0 | 106,2 | 69,2 | 131,3 | 299,9 |
| **g=pagode** | 67,6 | 88,6 | 62,2 | 22,4 | 53,2 | 106,2 | 0 | 43,2 | 33,1 | 120,7 |
| **h=salsa** | 43,1 | 90,8 | 110,4 | 43,6 | 37.0 | 69,2 | 43,2 | 0 | 39,5 | 136,9 |
| **i=sertaneja** | 48,6 | 140,5 | 74,2 | 40,1 | 43,6 | 131,3 | 33,1 | 39,5 | 0 | 126,3 |
| **j=Tango** | 138 | 202,3 | 84,3 | 147,5 | 153,1 | 299,9 | 120,7 | 136,9 | 126,3 | 0 |

*Table 10: Distance between genres GMM*

The distances between the genre Gaussian Mixture Models are represented in the figures 24-29. There is a figure for each genre, that represents the distances between that genre to all other genres. To make it easier the comparison between figures, the same scale is used for all genres (the maximum value is 200) – for this reason some "points" are out of the scale, but their distance is perceptible. These images were created to to characterize an estimate of the the distances between genres.

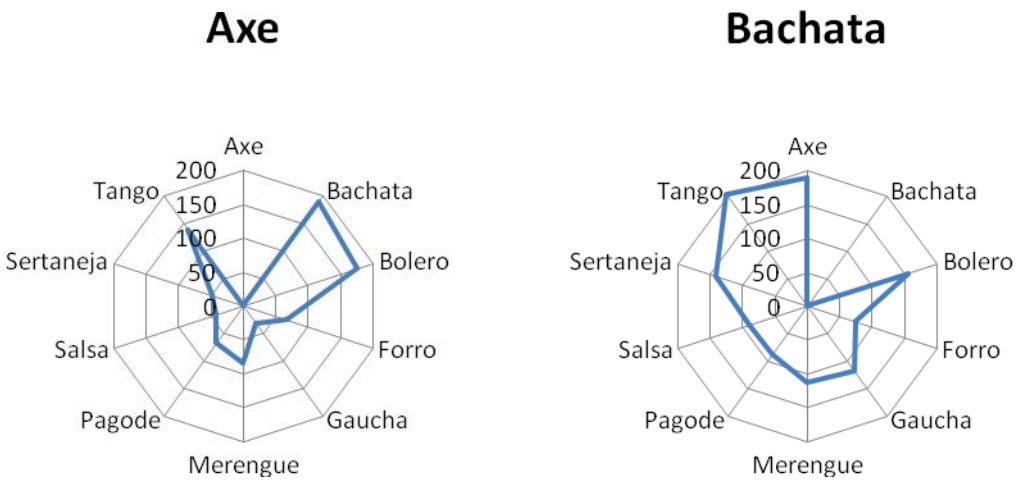


*Figure 24: Distance between genres: Axé and Bachata*

Figure 24 illustrates the distances between Axé and the other genres and between Bachata and the other genres.

When it comes to Axé, there are three distant genres – Tango, Bachata and Bolero. The other genres are close to Axé (distances less than 100).

The genre Bachata can be said to be generaly more separated from the other genres. Bachata's most distant genres are Axé, Tango, Sertaneja, and Bolero. The closest genres are Salsa, Pagode and Forró (distances less than 100).

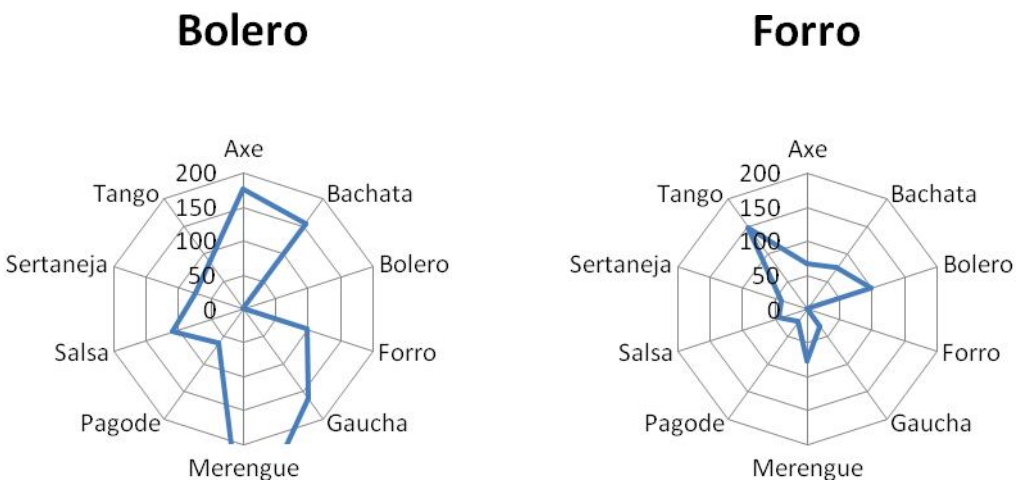


*Figure 25: Distance between genres: Bolero and Forró*

Figure 25 illustrates the distances between Bolero and the other genres and between Forró and the other genres.

Bolero's most distant genres are Axé, Bachata, Gaúcha and Merengue. The closest genres are Forró, Pagode, Sertaneja and Tango.

Forró can be said to be close to the majority of the genres. There is only one genre that has a distance over 100 – Tango. Salsa and Sertaneja are particularly close to Forró.

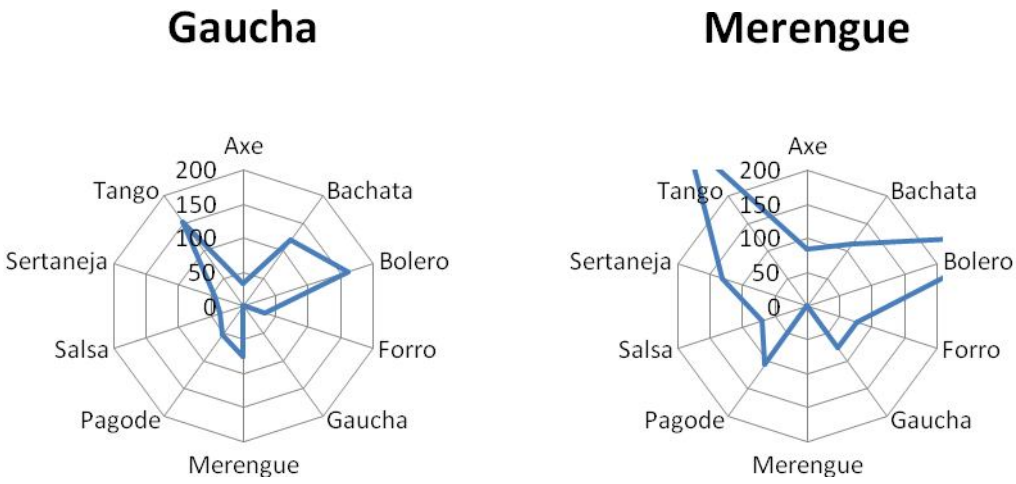


*Figure 26: Distance between genres: Gaúcha and Merengue*

Figure 26 illustrates the distances between Gaúcha and the other genres and between Merengue and the other genres.

Gaúcha is close to a great number of genres – the distance between Gaúcha and 6 genres is below 100. The three exceptions are Bachata, Bolero and Tango. The distance to Salsa, Sertaneja, Forró, and Axé is below 50.

When it comes to Merengue, four genres can be said to be relatively close to this genre – Axé, Salsa, Gaúcha, and Forró (still, these distances are between 50 and 100, not that close). All other genres are distant from Merengue, with Bachata and Tango being much further away.

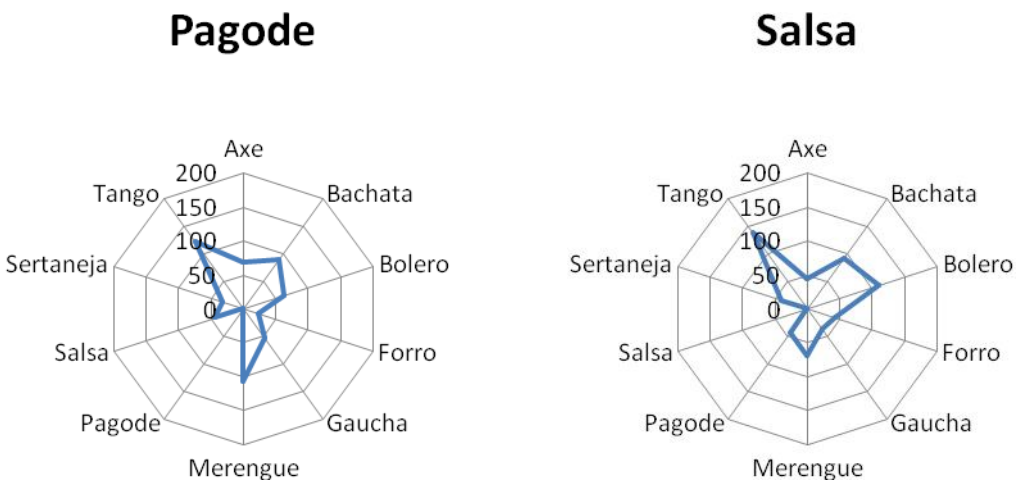


*Figure 27: Distance between genres: Pagode and Salsa*

Figure 27 illustrates the distances between Pagode and the other genres and between Salsa and the other genres.

Pagode is close to the majority of genres: only Tango and Merengue are distant more than 100 units. Forró is particularly close to Pagode (the distance between them is 22.4).

Salsa is close to the majority of genres – only Bolero and Tango are distant more than 100 units.

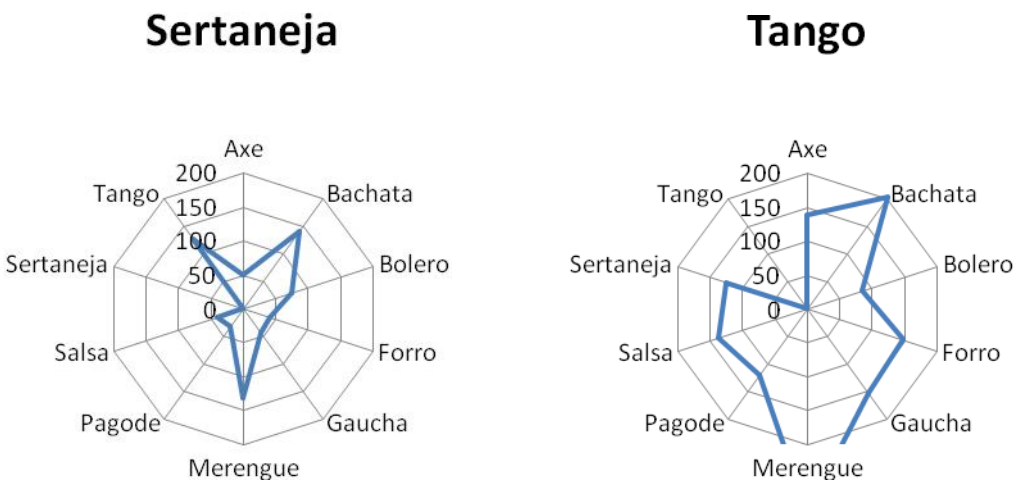


*Figure 28: Distance between genres: Sertaneja and Tango*

Figure 28 illustrates the distances between Sertaneja and the other genres and between Tango and the other genres.

Sertaneja can be said to be close to all genres but three: Bachata, Merengue and Tango, being particularly close to Salsa.

Tango can be said to be distant to all genres: only Bolero is relatively close (85 units of distance).

These relations between genres were obtained with the song classification using Gaussian Mixture Models and can be compared with the results obtained with the Weka classifiers – namely KNN (the Weka classifier that yielded the best results). The results considered are the ones obtained with a 10 % resample of both datasets (as the greater number of instances are considered). It should be noted again that the results obtained with the Gaussian Mixture Models are song classification results, and the results obtained with the Weka classifiers are instance classification results.

Table 11 and 12 shows the the genre confusion matrix and detailed accuracy obtained with the KNN classifier, respectively.

A comparison between the genres TP and FP Rates obtained with the KNN classifier and with the GMM classification is made in table 13.

| | **a** | **b** | **c** | **d** | **e** | **f** | **G** | **h** | **i** | **J** | **<-classified as** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1898 | 666 | 42 | 34 | 181 | 164 | 168 | 224 | 169 | 194 | 56 | **a=axe** |
| 2018 | 45 | 1320 | 52 | 111 | 11 | 222 | 84 | 108 | 55 | 10 | **b=bachata** |
| 1941 | 201 | 80 | 788 | 97 | 104 | 33 | 185 | 175 | 194 | 84 | **c=bolero** |
| 1994 | 304 | 60 | 34 | 820 | 231 | 92 | 172 | 127 | 141 | 13 | **d=forro** |
| 1936 | 258 | 51 | 82 | 178 | 788 | 148 | 92 | 154 | 162 | 23 | **e=gaucha** |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1913 | 142 | 171 | 23 | 109 | 83 | 1025 | 124 | 219 | 17 | 0 | **f=merengue** |
| 1812 | 274 | 157 | 119 | 118 | 135 | 108 | 512 | 202 | 157 | 30 | **g=pagode** |
| 2024 | 313 | 117 | 108 | 116 | 159 | 289 | 138 | 668 | 97 | 19 | **h=salsa** |
| 2000 | 456 | 72 | 224 | 101 | 286 | 54 | 163 | 148 | 455 | 41 | **i=sertaneja** |
| 1930 | 11 | 8 | 2 | 27 | 4 | 18 | 4 | 48 | 11 | 1797 | **j=tango** |
| | 2670 | 2078 | 1466 | 1858 | 1965 | 2157 | 1698 | 2018 | 1483 | 2073 | |

*Table 11: Confusion Matrix obtained using classification with KNN*

| TP Rate | FP Rate | Precision | F-Measure | |
|---|---|---|---|---|
| 0.351 | 0.113 | 0.249 | 0.292 | **Axe** |
| 0.654 | 0.044 | 0.633 | 0.643 | **Bachata** |
| 0.404 | 0.038 | 0.538 | 0.461 | **Bolero** |
| 0.411 | 0.065 | 0.419 | 0.415 | **Forro** |
| 0.407 | 0.067 | 0.401 | 0.404 | **Gaucha** |
| 0.536 | 0.064 | 0.475 | 0.504 | **Merengue** |
| 0.268 | 0.067 | 0.302 | 0.284 | **Pagode** |
| 0.330 | 0.077 | 0.331 | 0.331 | **Salsa** |
| 0.228 | 0.058 | 0.307 | 0.261 | **Sertaneja** |
| 0.931 | 0.016 | 0.867 | 0.898 | **Tango** |
| 0.452 | 0.061 | 0.452 | 0.449 | **Average** |

*Table 12: Classification performance by class, using KNN*

| | GMM TP Rate | GMM FP Rate | KNN TP Rate | KNN FP Rate |
|---|---|---|---|---|
| **Axé** | High (0.594) | High (0.159) | Average (0.351) | High (0.113) |
| **Bachata** | High (0.688) | Very Low (0.00903) | High (0.654) | Average Low (0.044) |
| **Bolero** | High (0.525) | Average (0.0660) | Average (0.404) | Average Low (0.038) |
| **Forró** | Very Low (0.0812) | Average Low (0.0382) | Average (0.411) | Average (0.065) |
| **Gaúcha** | Low (0.156) | Average Low (0.0386) | Average (0.407) | Average (0.067) |
| **Merengue** | High (0.738) | Average Low (0.0333) | High (0.536) | Average (0.064) |
| **Pagode** | Average (0.412) | High (0.134) | Low (0.268) | Average (0.067) |
| **Salsa** | Very Low (0.0812) | Low (0.0188) | Average (0.330) | Average (0.077) |
| **Sertaneja** | Average (0.438) | High (0.169) | Low (0.228) | Average (0.058) |
| **Tango** | Low (0.262) | Very Low (0.00208) | Very High (0.931) | Low (0.016) |

*Table 13: Comparison between TP and FP Rates obtained with GMM and KNN*

The TP Rate was classified as being “Very High” if it was higher than 0.9, “High” if it was higher than 0.5, classified as “Average” if it was between 0.3 and 0.5, classified as “Low” if it was below 0.3, and classified as “Very Low” if it was below 0.1.

The FP Rate was classified as being “High” if it was higher than 0.1, “Average” if it was between 0.05 and 0.1, “Average Low” if it was between 0.03 and 0.05, “Low” if it was below 0.03, and “Very Low” if it was below 0.01.

One thing that stands out when looking at table 13 is that the genres TP and FP Rates obtained with the instance KNN classifier are much more similar between them: for example, except 2 cases, the FP Rate varies from 0.038 to 0.077. The exceptions are 0.113 obtained with Axé and 0.016 obtained with Tango. On the other hand, with the GMM classification, the FP Rate varies from 0.00208 to 0.169, being distributed along this range of values.

The TP Rate values vary between 0.228 and 0.654, if Tango is excluded, which achieved an extremely high result (0.931). On the other hand, with the GMM classification, the TP Rate varies from 0.0812 to 0.738 (bigger range of values).

A way to look at this difference is with the labels given to the results:

With GMM, the TP Rate registered 2 “Very Low” values; 2 “Low” values, 2 “Average” values and 4 “High” values.

With KNN, the TP Rate registered 2 “Low” values, 5 “Average” values, 2 “High” values and 1 “Very High” value.

With GMM, the FP Rate registered 2 “Very Low” values, 1 “Low” value, 3 “Average Low” value, 1 “Average” value and 3 “High” values.

With KNN, the FP Rate registered 1 “Low” value, 2 “Average Low” values, 6 “Average” values and 1 “High” value.

It can be seen that in the KNN results, the FP and TP Rates are more centrally distributed, with more “Average” results, than with the FP and TP Rates obtained with using GMMs.

When it comes to single genre characteristics, the similarities between the song classification using GMM and the instance classification using KNN are found in the high TP Rate achieved by Bachata and Merengue; the high FP Rate achieved by Axé; and the “Low” and “Very Low” values achieved by Tango.

The reason for the differences found in the two classification procedures might have to do with the nature of the objects being classified. With the Gaussian Mixture Models it were songs that were being classified (160 songs per genre, 1600 total); with the Weka KNN classification, it were instances the objects being classified (around 2000 instances per genre). An instance cannot be as representative of a genre, as an a whole song, obviously: the instances classified characterize 2,32 s (as it was said in chapter 3.14). Specifically, with the KNN instance classification, a test instance is classified according to its closest instances of the training set. For example, it is much more likely that a specific Axé instance will have as its nearest neighbour an instance from a random genre; than a GMM of a whole song will have as its closest genre a random genre GMM. Said in other words:

it is likely to find very similar instances in songs from different genres (they are only 2.32 s long); this variation will be harder to happen in the song classification using GMM – there are less songs, and each song contains much more information, which will define the song better. This can be an explanation for the fact that the instance classification using KNN is much more uniformly distributed along the genres.

## 4.8 Validation results

In the validation experiments, the training was done with 198 451 instances randomly selected from the instances of the dataset used in the main experiments (the instances selected are 10 % of the total instances).The testing with the Weka classifiers was done with 49 419 instances randomly selected from the instances of the validation dataset (10 % of the instances were selected). With the Gaussian Mixture Models experiments, all the instances from the main dataset (160 songs per genre) were used to create genre GMM, and the validation dataset songs (40 per genre) were classified.

Table 14 represents the performance (TP Rate, FP Rate, Precision and F-Measure) achieved with the Weka classifiers (Naive Bayes, KNN, J48 and SVM) and with the Gaussian Mixture Models.

| | **TP Rate** | **FP Rate** | **Precision** | **F-Measure** |
|---|---|---|---|---|
| **Naive Bayes** | 0.367 | 0.070 | 0.370 | 0.356 |
| **KNN** | 0.418 | 0.065 | 0.426 | 0.419 |
| **J48** | 0.364 | 0.071 | 0.380 | 0.370 |
| **SVM** | 0.477 | 0.058 | 0.474 | 0.474 |
| **GMM** | 0.402 | 0.066 | 0.459 | 0.429 |

*Table 14: Classifiers performance (validation dataset)*

The results are a bit different (although not much) from the ones obtained in the main dataset – these results should be compared with the ones obtained with the experiments using the artist filter.

Naive Bayes achieves a worse performance (36.7 %) compared to the numbers around 40 % achieved in the main experiments. KNN also performs worse (41.8 %), compared to the performance of between 45 % and 46 % of correctly classified instances, achieved in the main experiments. J48 achieves a performance of 36.4 %, which is in the range of values achieved with this classifier in the main experiments (between 34 % and 40.1 %). With SVM, the percentage of correctly classified instances is 47.6 % (the highest result), which is better than the results achieved in the main experiments (between 43.2 % and 45.1 %).

These differences may be result of the difference in the quantities of the training instances and testing instances in the main and validation experiments.

In the main experiments, when samples of 10 % were used, the training was done with 177 147 instances and the testing was done with 19 574 instances. In the validation experiments, the training was done with 198 451 instances and the testing was done with 49 419 instances. The difference in these experiments is in the number of testing instances (the number of testing instances in the validation experiments is more than the double) as the number of training instances is similar. Therefore, the results obtained in the validation experiments should be taken as slightly more representative of the genre identification ability of the classifiers, as there were more instances tested.

The confusion matrix obtained with the GMM classification is represented in table 15. The classification performance per class is represented in table 16.

| | **a** | **b** | **b** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **<-classified as** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 40 | 25 | 0 | 1 | 0 | 0 | 4 | 1 | 0 | 9 | 0 | **a=axe** |
| 40 | 0 | 27 | 3 | 2 | 0 | 1 | 5 | 1 | 1 | 0 | **b=bachata** |
| 40 | 1 | 0 | 25 | 0 | 1 | 0 | 4 | 2 | 4 | 3 | **c=bolero** |
| 40 | 14 | 0 | 0 | 1 | 9 | 2 | 2 | 1 | 10 | 1 | **d=forro** |
| 40 | 16 | 0 | 1 | 0 | 16 | 1 | 0 | 0 | 5 | 1 | **e=gaucha** |
| 40 | 7 | 0 | 1 | 0 | 5 | 26 | 0 | 0 | 1 | 0 | **f=merengue** |
| 40 | 17 | 0 | 0 | 1 | 6 | 0 | 7 | 0 | 9 | 0 | **g=pagode** |
| 40 | 14 | 0 | 7 | 1 | 3 | 3 | 1 | 4 | 7 | 0 | **h=salsa** |
| 40 | 12 | 0 | 6 | 2 | 1 | 0 | 3 | 2 | 14 | 0 | **i=sertaneja** |
| 40 | 14 | 1 | 5 | 0 | 0 | 0 | 2 | 0 | 2 | 16 | **j=tango** |
| 400 | 120 | 28 | 49 | 7 | 41 | 37 | 25 | 10 | 62 | 21 | **Total** |

*Table 15: Confusion Matrix obtained using classification with GMM (validation dataset)*

| **TP Rate** | **FP Rate** | **Precision** | **F-Measure** | |
|---|---|---|---|---|
| 0,625 | 0,264 | 0,208 | 0,312 | **Axe** |
| 0,675 | 0,00278 | 0,964 | 0,794 | **Bachata** |
| 0,625 | 0,0667 | 0,510 | 0,562 | **Bolero** |
| 0,0250 | 0,0167 | 0,143 | 0,0426 | **Forro** |
| 0,400 | 0,0694 | 0,390 | 0,395 | **Gaucha** |
| 0,650 | 0,0306 | 0,703 | 0,675 | **Merengue** |
| 0,175 | 0,0500 | 0,280 | 0,215 | **Pagode** |
| 0,100 | 0,0167 | 0,400 | 0,160 | **Salsa** |
| 0,350 | 0,133 | 0,226 | 0,274 | **Sertaneja** |
| 0,400 | 0,0139 | 0,762 | 0,524 | **Tango** |
| 0,402 | 0,0663 | 0,459 | 0,429 | **Average** |

*Table 16: Classification performance by class using GMM (validation dataset)*

The highest TP Rate is achieved by Axé, Bachata, Bolero and Merengue, with TP Rate values over 0.6. The highest TP Rate achieved in the main experiments is achieved also by these four genres, with TP Rate values over 0.5.

The lowest TP Rate is achieved by Forró and Salsa (values of 0.025 and 0.1, respectively). These genres also are the ones with the lowest TP Rate in the main experiments (values of 0.0812 for both genres).

There are some differences in the other four genres. Gaúcha achieves a TP Rate of 0.4 (in the main experiments it achieves 0.156); Pagode achieves a TP Rate of 0.175 (in the main experiments it achieves 0.412); Sertaneja achieves a TP Rate of 0.35 (in the main experiments it achieves 0.438); Tango achieves a TP Rate of 0.4 (in the main experiments it achieves 0.262).

The overall TP Rate in the validation experiments is 0.4025, very similar to the overall TP Rate in the main experiments – 0.398.

When it comes to the FP Rate, Axé and Sertaneja achieve the highest values (0,264 and 0,133 respectively). These genres also achieve the highest FP Rate in the main experiments (0.159 for Axé and 0.169 for Sertaneja). Bachata, Forró, Salsa and Tango achieve the lowest FP Rate. (Bachata, Salsa and Tango achieve the lowest FP Rate in the main experiments). The overall FP Rate is 0.066, very similar to the one achieved in the main experiments (0.067).

Bachata, Merengue and Tango achieve the highest Precision in both the validation and main experiments (over 0.7). Axé, Forró, Pagode and Sertaneja achieve the lowest Precision in both the validation and main experiments (below 0.3). The overall Precision is the same in both experiments (0.46).

The validation results can be said that confirm the trends indicated in the main experiments.

## 4.9 Implementations in literature

Classification experiments were made with Support Vector Machines in [18] on a database of 1700 songs representing 7 genres (200 songs per genre). The features extracted characterize timbre and rhythm. Three implementations of texture windows were made: texture windows of 1 s long, of 30 s long and texture windows centered on beats and with the duration of these beats. The classification results vary from genre to genre, with the best results being for the “Classical” genre (around 90 %), and the worst for the “Electronic” genre (from 30 % to 50 %).

In [48] classification experiments using GMM and KNN are made, using four datasets with different characteristics (of size and variety). The features extracted characterize timbre and there is an extraction of “fluctuation patterns” (that describe “loudness fluctuations in frequency bands”). Two features are extracted from these Fluctuation Patterns (they are named “Focus” and “Gravity”). The features from songs are extracted and each song is modeled as a Gaussian Mixture Model. Similarity between Gaussian

Mixture Models is made using the Monte Carlo sampling algorithm [16]. Classification is made using a KNN algorithm ($K$=1). The results obtained vary with the datasets used and with the weight given to the timbral features and fluctuation patterns features. Results vary from very low values of around 10 %, when the only features considered are the ones related to the fluctuation patterns, to values around 65 %. These values relate to the ratio between the number of correctly classified titles and the total number of titles.

An attempt at artist identification is made in [49], where experiments are conducted over a dataset of 1200 pop songs from 18 artists. Features are extracted at a song level (representing the whole song) and a SVM classifier is used. A classification accuracy of 69 % was achieved when the training and testing songs came from different albums and an accuracy of 84 % correctly classified songs was achieved when the songs were randomly distributed between cross validation sets.

An analysis of the adequacy of features related to timbre to characterize musical genres is made in [32]. Features characterizing timbre are extracted and, for each song, a Gaussian Mixture Model is computed. The distance between songs is made with Monte Carlo sampling. The dataset used is composed of 17075 music titles from different genres. The distances between each song in the database and all the others are computed and the results are considered disappointing: considering the closest neighbour of all songs, an average of 0.46 songs belong to the same genre of the song (this result is not that bad actually); considering the closest 5 neighbours, an average of 1.43 songs belong to the same genre; considering the closest 10 neighbours, the average is 2.56 songs; considering the 20 closest neighbours, the average is 4.61 songs and considering the 100 closest neighbours, the average is 18.09 songs.

A comparison between GMM and KNN classifiers is made in [9]. The features extracted characterize timbre, rhythm and pitch. Three datasets were used: “genres” consisting of ten different musical genres (classical, country, disco, hip hop, jazz, rock, blues, reggae, pop and metal), “classical” composed of four subgenres (choir, orchestra, piano and string quartet) and “jazz” composed of six subgenres (big band, cool, fusion, piano, quartet and swing). Each of these 20 classes was trained with 100 excerpts of 30 s long. Each of these excerpts was represented by a single feature vector (with features related to timbre, rhythm and pitch). The classification was made using a 10 fold cross validation procedure. For the “genres” dataset, the classification results, in accuracy mean/standard deviation (in percentage), using KNN (with $K$ being 1, 3 and 5) were: 59/4 for $K$=1, 60/4 for $K$=3 and 56/3 for $K$=5. Using GMM, the results, in accuracy mean/standard deviation (in percentage), were: 59/4 for a single Gaussian classifier, 60/4 for GMM2, and 61/4 for GMM3, GMM4 and GMM5.

With the “jazz” dataset, the accuracy results using KNN range from 56 % to 59 %, and using GMM they range from 59 % (GMM5) to 68 % (GMM3). With the “classical” dataset, the accuracy results using KNN range from 70 % (KNN5) to 78 % (KNN3), and

using GMM they range from 77 % (single Gaussian classifier) to 88 % (GMM3, GMM4 and GMM5).

A hierarchical classification scheme using GMM is presented in [50]. The dataset used is composed of 17 audio classes, organized into subgenres, in a tree-like structure: there are three main classes, “speech”, “music” and “background”. “Background” is a single class, “speech” is composed of 3 sub-classes and “music” is organized into “classical” and “non-classical”, each of these two containing other subclasses. The total number of classes (a class being the end of a branch, without subclasses) is 17. There is an extraction of a variety of features that characterize timbre, rhythm, and also features as defined in the MPEG-7 standard. A total of 58 features were chosen from a larger group of 90 features, after a process that evaluated the “power” of each feature to discriminate between classes. The 58 chosen features were the ones that best differentiated classes. The classification was made in a hierarchical way: instead of a direct classification into one of the 17 classes, there were successive classifications, one for each split in the tree. For each split in the tree there was an identification of the 20 features that best distinguished the sub classes that composed the split (from the initial group of 58 features). These groups of “best” 20 features (one group of 20 features for each split) were used in the classification process. All classification stages used their own training dataset – only composed of the classes being evaluated at each stage. For each of the 17 audio classes, 50 pieces of 30 seconds long were used. The classification was made using Gaussian Mixture Models. The classification results varied from 40 % to 60 % (of correctly classified samples) when classifying sub genres, and reached higher values when distinguishing between classical and non-classical (90 %) or between chamber music and orchestral music (75 %).

The study of the artist filter and its influence in classification is done in [51], [48], [52] and [53].

In [51] experiments are carried out using classification with Gaussian Mixture Models. Two experiments are done: the first experiment assesses the influence of the artist filter in an experimental comparison between classifications based on the models of songs versus based on the models of genres. In the classification based on the models of songs, GMM are computed for each song in the training set and in the testing set. The songs in the testing set are classified according to the position of their closest neighbour in the training set. In the classification based on the models of genres, GMM are computed for each genre (from the training dataset), and for each song in the testing set. The test songs are classified according to their closest genre model.

The second experiment assesses the influence of the artist filter in an experimental comparison of the classification achieved when using different sets of features (MFCC *versus* Fluctuation Patterns). The classification was done using the song GMM approach, referred above (when the MFCC features were used) and using a computation of the Euclidian distance between the test songs to all train songs.

The dataset used was composed of 729 songs, from 128 artists and 6 different genres. In the first experiment the features extracted were composed of MFCC, and in the second experiment the features extracted were MFCC and Fluctuation Patterns.

There is an influence of the artist filter in the results obtained: in the first experiment, the average accuracy percentages were 75.72 % and 69.00 % (for the classifications based on song-GMM and genre-GMM, respectively) if no artist filter was used. If the artist filter was used the average accuracy percentages were 58.50 % and 61.22% (for the classifications based on song-GMM and genre-GMM, respectively).

In the second experiment, the average accuracy percentages were 75.72 % and 63.10 % (for the classifications based on MFCC and Fluctuation Patterns, respectively), if no artist filter was used. If the artist filter was used, the average accuracy percentages were 58.50 % and 55.63 % (for the classifications based on MFCC and Fluctuation Patterns, respectively).

Studies of the influence of the artist filter are done in [48], [52], [53], which confirm that the use of the artist filter can considerably lower and modify the classification results, and therefore should be used, in order to make possible a proper analysis of the performed classification experiments, and their true significance.

Classification experiments using the Latin Music Database are done in [54] and in [55]. In [54], experiments are carried out in the following way: features are extracted from three segments 30 s long, from each musical file: these segments are from the beginning, from the middle and from the end of the file. The training dataset is composed of 150 samples from each musical genre, and the testing dataset is composed of 90 samples from each genre. A validation dataset was also used, composed of 60 samples from each genre. For each segment a single feature vector is calculated. The extracted feature vector characterizes timbre, pitch and rhythm. These features vectors were used to train classifiers. The used classifiers are Naive Bayes, Support Vector Machines, MLP Neural Networks, KNN ($K$=3) and J48. The testing is done feeding test feature vectors to the classifiers, which will classify them separately. The output of the classifiers for each segment is then used in combination procedures (such as the majority vote or the product rule). The results of these combinations will assign a single genre to the three musical segments from the song. It is found that this ensemble method improves the accuracy obtained in 1 % to 7 %, compared to the accuracy obtained with the classification using a single classifier. It is also found that the segment from the middle of the songs is the one that provides the best classification accuracy. The obtained classification results vary from 46 % to 65 %, the best results being achieved with the SVM classifier.

## Chapter 5

# 5. Conclusions

The experiments described in this dissertation had the goal to assess the classification performance of several classifiers and to assess the influence of several factors in that performance. These factors are: the use of an artist filter in the definition of the testing and training datasets; the nature of the datasets being classified; the size of the testing and training datasets; and the mode of the testing.

The performance of the classifiers used varied according to the experiences, but considering only the results obtained when the artist filter was used, the classifiers KNN and SVM provided the best classification results, achieving values of percentage of correctly classified instances in the range between 42 % and 47 %.

It was confirmed the great influence of the artist filter in the classification performances – without an artist filter the classification experiments produced much higher values (some of them completely unrealistic - 99 % of correctly classified instances using KNN, for example). This happened in part due to the nature of the instances being classified: consecutive instances extracted from the same song were very similar in their values, and in the classification experiments without the artist filter there was no way to prevent the existence of these similar instances in the testing and training datasets. Naturally, the classification performance without the artist filter achieved high values. Conceptually, the use of the artist filter should be recommended in all experiments of musical genre classification, so that the results obtained are as representative as possible of the genre identification ability of the classifiers – the task here is to identify genres, not particular artists or songs.

When the artist filter was used there was no significant influence in the classification results of the size of the training and testing datasets.

The experiments that did not use the artist filter should be looked at with a skeptical spirit, as the classification results achieved suspiciously high values. Still, in the experiments when the artist filter was not used, it was found that the classification performance increased when the proportion between training/testing dataset sizes went in the direction of a larger training set and a smaller testing set (when the training and testing instances were not the same and were defined splitting randomly a larger instance dataset). When

the same instances were used to train and test the classifiers, the performance achieved was the highest. This result is expected: the classification results should be much better if the instances used to test the classifier are the same instances that were used to train it. The classification performance also improved with the size of the dataset – this can be explained with the fact that a growth in the size of the dataset used will improve the odds that very similar instances (from the same artists and from the same songs) are found in the training and testing sets.

With the George Tzanetakis collection an experience was made that compared two ways to test the classifiers: using a percentage split (90 % used for training and 10 % used for testing), and using a 10 fold cross validation. The results obtained with the percentage split and the 10 fold validation are very similar and it can be concluded that the 10 fold cross validation procedure might not be justifiable if the dataset used is large enough.

The classification experiments with the George Tzanetakis collection produced better results than the experiments with the Latin Music Database (in these experiments, no artist filter was used). This may be due to the fact the musical genres in the George Tzanetakis collection are much more different between them, than the genres of the LMD (that after all, belong to a more general musical genre – Latin).

An estimation of the timbral similarities between the genres of the Latin Music Database was made in a relatively detailed degree.

It was found that the average timbral distance between a genre and all the other genres varies a lot: Bachata, Tango and Merengue seem to be well defined in their timbre – they are found to be far away from the majority of the LMD genres. On the other hand, genres like Forró, Pagode and Salsa are found to be close to a great number of genres. Experiments of this kind are valuable in a way that they can give insights on the feature characteristics that differentiate musical genres (and that surely change from genre to genre).

# References


[1] R. Dannenberg, J. Foote, G. Tzanetakis, and C. Weare, "Panel: new directions in music information retrieval," in *Proc. International Computer Music Conference*, Habana, 2001.

[2] F. Pachet and D. Cazaly, "A taxonomy of musical genres," in *Proc. Content-Based Multimedia Information Access (RIAO)*, Paris, 2000.

[3] J.-J. Aucouturier and F. Pachet, "Representing musical genre: a state of the art," *Journal of New Music Research*, vol. 32, pp. 83-93, 2003.

[4] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic genre classification of music content," *IEEE Signal Processing Magazine*, vol. 23, pp. 133-141, 2006.

[5] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," CUIDADO I.S.T. Project Report, 2004.

[6] E. Gomez, A. Klapuri, and B. Meudic, "Melody description and extraction in the context of music content processing," *Journal of New Music Research*, vol. 32, 2003.

[7] A. P. Klapuri, "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness," *IEEE Transactions On Speech And Audio Processing*, vol. 11, pp. 804-816, 2003.

[8] G. Zoia, D. Mlynek, and R. Zhou, "A multi-timbre chord/harmony analyzer based on signal processing and neural networks," in *Proc. 6th Workshop on Multimedia Signal Processing*, Siena, 2004, pp. 219-222.

[9] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, pp. 293-302, July 2002.

[10] F. Gouyon and S. Dixon, "A review of automatic rhythm description systems," *Computer Music Journal*, vol. 29, pp. 34-54, 2005.

[11] F. Gouyon, S. Dixon, E. Pampalk, and G. Widmer, "Evaluating rhythmic descriptors for musical genre classification," in *Proc. AES 25th International Conference*, London, 2004, pp. 196-204.

[12] J.-J. Aucouturier, B. Defreville, and F. Pachet, "The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music," *The Journal of the Acoustical Society of America*, vol. 122, pp. 881-891, 2007.

[13] J.-J. Aucouturier and F. Pachet, "Improving timbre similarity: how high is the sky," *Journal of Negative Results in Speech and Audio Sciences*, vol. 1, 2004.

[14] J.-J. Aucouturier and F. Pachet, "The influence of polyphony on the dynamical modelling of musical timbre," *Pattern Recognition Letters*, vol. 28, pp. 654-661, 2007.

[15] J.-J. Aucouturier and F. Pachet, "A scale-free distribution of false positives for a large class of audio similarity measures," *The Journal of The Pattern Recognition Society*, vol. 41, pp. 272-284, 2008.

[16] J.-J. Aucouturier, "Dix expériences sur la modélisation du timbre polyphonique," These de Doctorat, Universite Paris 6 2006.

[17] K. West and S. Cox, "Finding an optimal segmentation for audio genre classification " in *Proc. 6th International Conference on Music Information Retrieval*, London, 2005, pp. 680-685.

[18] N. Scaringella and G. Zoia, "On the modelling of time information for automatic genre recognition systems in audio signals," in *Proc. 6th International Conference on Music Information Retrieval*, London, 2005, pp. 666-671.

[19] A. Meng, P. Ahrendt, and J. Larsen, "Improving music genre classification by short-time feature integration," in *Proc. 6th International Conference on Music Information Retrieval*, London, 2005, pp. 604-609.

[20] S. Smith, "The scientist and engineer's guide to digital signal processing," www.dspguide.com.

[21] T. Kamm, H. Hermansky, and A. Andreou, "Learning the Mel-scale and optimal VTN mapping," in *WS97 Workshop*, John Hopkins University, Center for Language and Speech Processing, 1997.

[22] K. Vedala, http://en.wikipedia.org/wiki/Mel_scale, 2005.

[23] "Marsyas user manual," http://marsyas.sness.net/docs/manual/marsyas-user/index.html.

[24] G. Tzanetakis, "Manipulation, analysis and retrieval systems for audio signals ": Doctoral Dissertation, Princeton University, Princeton, USA, 2002.

[25] C. M. v. d. Walt and E. Barnard, "Data characteristics that determine classifier performance," in *17th Annual Symposium of the Pattern Recognition Association of South Africa*, Parys, South Africa, 2006.

[26] "Weka 3: Data Mining Software in Java," http://www.cs.waikato.ac.nz/~ml/weka/index.html.

[27] H. Zhang, "The optimality of Naive Bayes," in *Proc. 17th International FLAIRS Conference*, Miami Beach, 2004.

[28] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[29] K. Bennett and C. Campbell, "Support vector machines: hype or hallelujah?," *SIGKDD Explorations*, vol. 2, pp. 1-13, 2000.

[30] K. Teknomo, "K-Nearest neighbours tutorial," http:\\people.revoledu.com\kardi\tutorial\KNN\.

[31] "Decision trees for supervised learning," http://grb.mnsu.edu/grbts/doc/manual/J48_Decision_Trees.html.

[32] J.-J. Aucouturier and F. Pachet, "Music similarity measures: What's the use," in *Proc. 3rd International Conference on Music Information Retrieval*, Paris, 2002.

[33] E. Pampalk, "MA Toolbox for Matlab - implementing similarity measures for audio ": http://www.ofai.at/~elias.pampalk/ma/.

[34] I. Nabney and C. Bishop, "Netlab neural network software," http://www.ncrg.aston.ac.uk/netlab/index.php.

[35] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symposium on Mathematics, Statistics and Probability*, Berkeley, 1967, pp. 281-298.

[36] J. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," International Computer Science Institute, Berkeley, 1998.

[37] B. Logan and A. Salomon, "A content-based music similarity function," Compaq Computer Corporation Cambridge Research Laboratory: Technical Report Series, 2001.
[38] "Yossi Rubner Homepage," http://vision.stanford.edu/~rubner/.
[39] C. Silla, A. Koerich, and C. Kaestner, "The Latin music database," in *Proc. 9th International Conference on Music Information Retrieval*, Philadelphia, 2008, pp. 451-456.
[40] "Axé (gênero musical)," http://pt.wikipedia.org/wiki/Axé_(gênero_musical).
[41] "Bolero," http://en.wikipedia.org/wiki/Bolero.
[42] "Forró," http://pt.wikipedia.org/wiki/Forró.
[43] "Música nativista," http://pt.wikipedia.org/wiki/Música_nativista.
[44] S. Hutchinson, "Merengue: popular music of the Dominican Republic," http://www.iasorecords.com/merengue.cfm.
[45] "Salsa music," http://en.wikipedia.org/wiki/Salsa_music.
[46] "Música sertaneja," http://pt.wikipedia.org/wiki/Música_sertaneja.
[47] "History of Tango," http://en.wikipedia.org/wiki/History_of_Tango.
[48] E. Pampalk, A. Flexer, and G. Widmer, "Improvements of audio-based music similarity and genre classification," in *Proc. 6th International Conference on Music Information Retrieval*, London, 2005, pp. 628-633.
[49] M. Mandell and D. Ellis, "Song-level features and support vector machines for music classification," in *Proc. 6th Int. Symp. Music Information Retrieval*, London, 2005, pp. 594-599.
[50] J. J. Burred and A. Lerch, "A hierarchical approach to automatic musical genre classification," in *Proc. 6th International Conference on Digital Audio Effects* London, 2003.
[51] A. Flexer, "A closer look on artist filters for musical genre classification," in *Proc. 8th International Conference on Music Information Retrieval (ISMIR'07)*, Vienna, 2007, pp. 341-344.
[52] A. Flexer, "Statistical evaluation of music information retrieval experiments," *Journal of New Music Research*, vol. 35, pp. 113-120, 2006.
[53] E. Pampalk, "Computational models of music similarity and their application to music information retrieval," Doctoral Dissertation, Vienna University of Technology, Vienna, Austria, 2006.
[54] C. Silla, A. Koerich, and C. Kaestner, "Automatic music genre classification using ensemble of classifiers," in *Proc. IEEE International Conference on Systems, Man, and Cybernetics*, Montréal, October 2007, pp. 1687-1692.
[55] C. Silla, A. Koerich, and C. Kaestner, "Feature selection in automatic music genre classification," in *Proc. 10th IEEE International Symposium on Multimedia*, Berkeley, December 2008, pp. 39-44.