TECHNICAL UNIVERSITY OF CRETE

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

MUSIC GENRE CLASSIFICATION BASED ON WEB DOCUMENT RESOURCES

by

Tsakogianni Georgia

SUPERVISORY COMMITTEE

POTAMIANOS ALEXANDROS (SUPERVISOR)
DIGALAKIS VASSILIOS
PETRAKIS EURIPIDES

2008

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Increasing use of digital music requires methods to organize it. In this thesis, we classify music into genres exploiting web information. Our approach is based on the assumption that documents about "similar" artists share common lexical features. We retrieve the top ranked web pages, for each artist, using the Yahoo! Search Engine. The html tags and the punctuation marks from each page are removed, keeping only the plain text, from which we extract the lexical features for the artists.

Each artist is modeled as feature vector because that is the appropriate form for applying term weighting and feature selection methods. The term weighting scheme, which is used for assigning higher values to more important data, is *tf-idf* and the applied feature selection method which selects terms that are representative for each genre, is *chi-square*.

The accuracy of this method is estimated using various classification algorithms, such as k-Nearest Neighbor, Centroid Based Algorithms and Support Vector Machines. We estimate the classification rate for a different number of retrieved pages, either employing feature selection or not. The classification results are promising, achieving accuracies up to **92.45%.**

Furthermore, we study how the fusion of lexical and audio features influences the classification results. The best classification rate for the classifier that uses audio features exclusively is **56.6%** and for the classifier that uses only lexical features is **79.25%**. Combining the web-based distances with the audio based distances from genres linearly, the classification rate is better than above methods and equal to **81.13%**. This is an important conclusion but requires further investigation for extracting general results.

ΠΕΡΙΛΗΨΗ

Η αυξανόμενη χρήση της ψηφιακής μουσικής απαιτεί μεθόδους για την οργανωσή της. Σε αυτή τη διπλωματική, κατηγοριοποιούμε τη μουσική σε είδη εκμεταλλευόμενοι την πληροφορία που παρέχεται απο το διαδίκτυο. Η προσεγγισή μας βασίζεται στην υπόθεση οτι όμοιοι καλλιτέχνες μοιράζονται κοινά λεκτικά χαρακτηριστικά. Ανακτούμε τις υψηλότερες ιεραρχικά ιστοσελίδες για κάθε καλλιτέχνη, χρησιμοποιώντας τη μηχανή αναζήτης Yahoo!. Οι html ετικέτες και τα σημεία στίξης αφαιρούνται, κρατώντας μόνο το καθαρό κείμενο απο το οποίο θα εξάγουμε τα λεκτικά χαρακτηριστικά για τους καλλιτέχνες.

Κάθε καλλιτέχνης μοντελοποιείται ως ένα διάνυσμα χαρακτηριστικών, μιας και αυτή η μορφή είναι κατάλληλη για την εφαρμογή μεθόδων που θέτουν βάρος και επιλέγουν τα χαρακτηριστικά αυτά. Η μέθοδος που χρησιμοποιείται για να θέτει υψηλότερα βάρη στους πιο σημαντικούς όρους είναι η «συχνότητα όρου × αντίστροφη συχνότητα κειμένου» (*tf-idf*) ενώ η εφαρμοζόμενη μέθοδος που επιλέγει του χαρακτηριστικότερους όρους για κάθε μουσικό είδος είναι η χ^2 (chi-square).

Η ακρίβεια της μεθόδου εκτιμάται χρησιμοποιώντας διάφορους αλγόριθμους κατηγοριοποίησης,όπως τον κανόνα του κοντινότερου γείτονα, αλγόριθμους βασισμένους σε κεντρικά σημεία και τις Μηχανές Υποστήριξης Διανυσμάτων. Εκτιμάμε το ποσοστό κατηγοριοποίησης για διαφορετικούς αριθμούς ανακτημένων σελίδων, με/ή χωρίς εφαρμογή μεθόδου για επιλογή χαρακτηριστικών. Τα αποτελέσματα της κατηγοριοποίσης είναι ελπιδοφόρα μιας και επιτυγχάνουν ακρίβεια ώς και 92.45%.

Επιπλέον, εξετάζουμε με ποιον τρόπο η μίξη λεκτικών και ακουστικών χαρακτηριστικών επηρεάζει τα αποτελέσματα της κατηγοριοποίησης. Τα καλύτερα ποσοστά για τον ταξινομητή που χρησιμοποιεί αποκλειστικά ακουστικά χαρακτηριστικά είναι 56.6% και για τον ταξινομητη που χρησιμοποιεί μόνο λεκτικά χαρακτηριστικά είναι 79.25%. Συνδυάζοντας γραμμικά τις αποστασεις που προκύπτουν απο τα λεκτικά χαρακτηριστικά με αυτές που προκύπτουν απο τα ακουστικά, το ποσοστο κατηριοποίησης είναι καλύτερο απο τις δύο προηγούμενες μεθόδους και ίσο με 81.13%. Αυτό είναι ένα σημαντικό συμπέρασμα αλλά απαιτεί περαιτέρω μελέτη ώστε να γίνει εξαγωγή γενικευμένων συμπερασμάτων.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor professor Alexandros Potamianos. His guidance, advice, help and encouragement were the basis of this thesis.

I would like to thank professors Vasilis Digalakis and Euripides Petrakis for participating to the supervisory committee.

I am also deeply grateful to Elias Iosif for his constructive comments and his important support. He was always kind and ready to offer his help generously.

I warmly thank Aggelika, Dimitris, Fey, Nikos, Peggy and Stauroula for being my surrogate family. Their love was the most valuable thing.

My apologies to my friends I have not mentioned by name. I want to thank them for being my fellow-travelers all these years.

Last but not least, I would like to thank my family, who supported me by all means. My parents -Despoina and Panayiotis- and my little sister –Lina- were the source of my strength. No words can describe my love for them.

CHAPTER 1

INTRODUCTION

The world wide web can be viewed as the largest source of information. The thematic character of this information covers a wide range of topics. These topics can be found in formal sources, as well as in informal sources. For example, a political event can be presented by a news agency reflecting the professional point of view of experienced journalists, but at the same time an individual is very likely to post his very subjective comments about the same event in his personal blog. The previous example indicates the variance of the information that is available in the web. This variance has many aspects, such as the used vocabulary, writing style, as well as other more deep linguistic parameters. In addition, the textual web information has an unstructured form. In many hyperdocuments the particular body of text for which a user is interested, is surrounded by other document's elements, e.g., menus, and many pieces of unrelated textual information like advertisements. Despite these difficulties, the web remains the richest source of information, having numerous advantages, compared to other lexical resources such as corpora that very often are built for specific domains. First, the web can be mined in order to acquire information for almost any knowledge domain. Second, the problem of data sparseness can be handled more efficiently, since the web includes many billions of documents. Note that this number of documents day by day is being increased. Moreover, this information is freely available that means zero cost, in contrast to many lexical resources like corpora for which in many cases a purchase is required. Last, but not least, the web information covers a lot of languages. This fact makes the web a multilingual source of information, which gives the ability to methods that perform web-based text mining to be more generic regarding language independence.

In this thesis, motivated by the above considerations, we attempted to exploit the web information in order to obtain textual information for the task of genre classification for the domain of music. Our approach is strongly based in the hypothesis that the lexical contents of a document that deals with an artist or a music album can be important features, which describe the deeper semantics of this particular artist or album. Of course, this lexical information can be semantically heterogeneous, including advertisements published by record companies or music stores, even short reviews written by non-professional users. In any case, more or less, these lexical features reflect the underlying semantics. Based on this assumption we expect that similar artists would share common lexical features. For example the term "orchestra" is more probable to be found in a document about classic music, rather than to a document about rock music. Thus, artists

with similar lexical features can be grouped together, under the same genre. This classification task is of great interest for many applications. For example, it can be used for recommendation systems where the system proposes to a customer some choices that are similar to his preferences or they have been made by other users with similar preferences. If one bought a music album of Mozart it is very likely that he will be interested in buying also an album of Vivaldi. Also, such classification ability can be used for exploring more efficiently a song playlist, or even for constructing automatically such a playlist.

In particular, in this thesis we use the Yahoo! search engine in order to get the URLs of documents that refer to a number of artists. The goal is to extract the lexical features that characterize these artists in order to classify them into genres. We view a web search engine as a natural solution regarding the retrieval of documents about music, since this domain is quite popular among millions of web users. To our knowledge there is not any other lexical resource about artists and music genres in the form of a typical corpus. The desired URLs are obtained as the response according to properly formed queries, sent to search engine. We download a fraction of the returned, top-ranked documents. Then, we apply some basic text processing techniques in order to normalize the raw downloaded documents. For example, we apply HTML filtering in order to remove the HTML tags, and we discard any punctuation marks. Then, we extract the lexical features of the normalized documents. We also enhance the simple feature extraction by employing a more sophisticated feature selection method. During the next step we apply several similarity metrics and classification methods in order to classify the artists of interest into genres.

The related work for classifying music by extracting information from web is cited in [18, 19]. In our thesis we strongly based on [18], where is proposed a method for artist classification. We use the same dataset which consists of 224 artists divided into 14 genres. After retrieving 50 top-ranked web pages for each artist and by applying text processing techniques, each artist is represented by a feature vector. Afterwards, they classifying artists using appropriates methods for document categorization. In [19], is reported an extension of above work, by applying this method to different datasets. One other proposed method is to classify artists into genres is by using co-occurrences on the web [21, 22, 23]. Co-occurrence analysis is based on the idea that if two artists appeared in the same context there is some kind of similarity between them.

Obviously, another useful information source for these artists is the audio features of the songs that are performed by them. In this thesis we also study how the lexical and audio features can be combined in order to achieve better classification results. We experiment with the fusion of lexical and audio features by taking the latter using a third party system. The procedure of fusion for classifying music into genres is something that has never applied according to literature and offers perspectives for deeper investigation.

The rest of the thesis is organized as follows: Chapter 2 introduces some material about term weighting and document feature selection. Chapter 3 describes several metrics and methods about document classification. A brief review of the literature about genre classification, as well as the presentation of our approach is given in Chapter 4. In Chapter 5 we describe the experimental procedure and the used datasets, along with the evaluation results. In final, in Chapter 6 we discuss the main conclusions of this work and we give interesting directions for further research.

CHAPTER 2

FEATURE WEIGHTING AND SELECTION

2.1 INTRODUCTION

Term (or feature) weighting is an important task in many areas of Information Retrieval (IR). The goal of term weighting methods is to evaluate how important a term is in a collection or corpus, so terms that characterize a given document well and discriminate the document from the others should be weighted highly. There is a wide variety of term weighting schemes reported in the literature, which are divided into two groups: statistical and linguistic. The methods in the former group are based on a statistical analysis that extracts, from a document collection, features based on word frequencies or information theoretical measures. In this approach the words can be considered as unordered and independent elements. On the other side, linguistic methods are based on linguistic dependencies and exploit the information provided by word contexts. In this chapter, we perform statistical measures only.

The above methods use all terms that exist in the document; as a result, there is much irrelevant and noisy information. So, we study also techniques for identify and keeping only relevant data. These procedures, which called feature selection, reduce the workload and increase the accuracy of a system.Feature selection can be supervised with human support in labelling the data, or unsupervised without any human involvement.

2.2 TERM WEIGHTING METHODS

In this section we study some statistical term weighting methods. We start from more simple measures and we end up to variation of more complex methods. The most simple one deals with term frequencies while the more sophisticated also considers the document frequencies of terms.

In addition, we provide a brief probabilistic interpretation of these methods. Last, we extend the study of these widely-used methods, by discussing few variations of them.

2.2.1 Term Frequency

Term frequency (tf_t) [1] is an approach that assigns to each term *t* in a document *d* a weight which is equal to the number of occurrences of *t* in document *d*.

Document *d* is represented as "bag-of-words" model, so the exact ordering of the terms and grammar are ignored but the number of occurrences of each term is important. When each term frequency has computed, each document is formalized into vector space model. Each dimension corresponds to a separate term and its value is the corresponding term frequency.

It is apparent that two documents with the same "bag-of -words" representations have the same term vectors.

The definition of term depends on the application. Typically terms are single words, or phrases. If the words are chosen to be the terms, the dimensionality of the vector is the number of words in the vocabulary (the number of distinct words occurring in the corpus).

The negative aspects of this method are the following:

i) longer documents have higher tf_t values and contain more distinct terms. These factors can conspire to raise the scores of longer documents, which (at least for some information needs) is unnatural.

ii) all terms are considered equally important but certain terms have little or no discriminating power

In the next subsection, we introduce a mechanism for attenuating the effect of terms that occur too often in the collection to be meaningful for relevance determination.

2.2.2 Inverse document frequency

Inverse document frequency (idf_t) is one of the most important and widely used concepts in information retrieval. It was first introduced by Sparck Jones [2].

The definition of the *inverse document frequency* [1] is the following:

$$idf_t = \log \frac{N}{df_t} \qquad (2.1)$$

where N is the number of documents in the collection, df_t is the number of documents that contain a term *t*, and df_t / N is an estimate of the probability p that a random document would contain a term.

The metric idf_t evaluates the discriminating power of terms within a collection of documents. The intuition is that a term which occurs in many documents is not a good discriminator, and should be given less weight than one which occurs in few documents. For instance if all documents have a term t_i, the term loses its discriminating power since N = df_t the $idf_t = \log \frac{N}{df_t} = 0$

The base of the logarithm does not matter and such log rescaling is convenient because of the additive properties of logs.

2.2.2.1 Probabilities, logarithms and additivity

Following the definitions, found in [3] we consider k the number of query terms, $t_1, t_2, \ldots, t_k$, the terms, and $p_1, p_2, \ldots, p_k$ the probability that the terms occur in a document.

Let $p_{12\ldots k}$ be the probability that all query terms co-occur in documents and the query is of the form t_1 AND t_2 AND$\ldots t_k$.

If the occurrences of the terms in documents are statistically independent

$$p_{12\ldots k} = p_1 \cdot p_2 \ldots\ldots \cdot p_k \qquad (2.2)$$

Applying the Law of Logarithms,

$$\log(1/p_{12,..k}) = \log(1/p_1) + \log(1/p_2) + \ldots + \log(1/p_k) \qquad (2.3)$$

$$idf(t_1 \text{ AND } t_2 \text{ AND} \ldots t_k) = idf(t_1) + idf(t_2) + \ldots . idf(t_k) \qquad (2.4)$$

We can write the Equation 2.2 as:

$$\frac{df_{t_{12..k}}}{N} = \frac{df_{t_1}}{N} \cdot \frac{df_{t_2}}{N} \cdot \ldots . \frac{df_{t_k}}{N}$$

and then leads to:

$$df_{t_{12..k}} = {df_{t_1} \cdot df_{t_2} \cdot .....df_{t_k}} \Big/ {N^{(k-1)}} \qquad (2.5)$$

The $df_{t_{12..k}}$ cannot be greater than the smallest of the individual df_t given at the right side of Equation 2.5

Let df_t^* denote this smallest answer set. Obviously, df_t^* imposes a lower limit to the df_t of a query consisting of k terms, named $idf_t^* = \log \frac{N}{df_t^*}$.

Until now, we assume that terms are statistically independent, but this is not always true.

Three cases are possible:

(i) $df_{t_{12..k}} > {df_{t_1} \cdot df_{t_2} \cdot .....df_{t_k}} \Big/ {N^{(k-1)}}$ which means that terms co-occur more often than expected in the case of independence. Terms are said to be positively correlated.

(ii) $df_{t_{12..k}} = {df_{t_1} \cdot df_{t_2} \cdot .....df_{t_k}} \Big/ {N^{(k-1)}}$ which means that terms co-occur by chance. Terms are considered uncorrelated or statistically independent.

(iii) $df_{t_{12..k}} < {df_{t_1} \cdot df_{t_2} \cdot .....df_{t_k}} \Big/ {N^{(k-1)}}$ which means that terms co-occur less often than expected in the case of independence. Terms are said to be negatively correlated.

To sum up, we end up to the following conclusions:

1. When we add terms' df_t to estimate the df_t of a phrase, we prejudge terms as independent which is not always the case.

2. Computing df_t , with and without the term independence assumption can produce,

dissimilar results.

2.2.3 The tf_t-idf_t weighting

We combine the definitions of term frequency and inverse document frequency, to produce a composite weight for each term in each document, which is called $tf_t - idf_t$ weight . It is used often in information retrieval and text mining with the purpose of evaluating how important a term is to a document in a collection.

The $tf_t - idf_t$ weighting scheme assigns to term *t* a weight in document *d* given by

$$tf_t - idf_t = tf_t \times idf_t \qquad (2.6)$$

The main characteristics of this weight are the following:

(i) It has high value when *t* occurs many times within a small number of documents because this term lends high discriminating power to those documents.

(ii) It has low value when *t* occurs fewer times in a document, or occurs in many documents (its value is zero when the term occurs in all documents)

We view each document as a vector with one component corresponding to each term in the dictionary, together with a weight for each component that is given by the Equation 2.6. When a term does not occur in a document, its weight is zero.

tf_t provides an estimation of how frequent a term appears and idf_t can be interpreted as "the amount of information". tf_t can be characterized as a simple frequent measure and

idf_t as a specificity measure [5]. The difficulty which appears is the establishment of a good balance between popularity and specificity.

2.2.3.1 Probabilistic justification for tf_t - idf_t

In this section, we consider documents and queries as an ordered sequence of words or terms [6], something which is not usual in information retrieval and queries are modeled as compound events, which consists of two or more single events. In this case single events are the query terms.

Generally, the probability of a query does not depend from the order of the terms, but based on the above assumption a query of length n is concerned as an ordered sequence of *n* single terms $T_{1,}T_{2.....}T_n$.

So, for a document D_i the probability of the ordered sequence is defined as $P(T_1, T_2, ..., T_n \mid D_i)$

Assuming that there is conditional independence between query terms we end up to the following:

$$P(T_1, T_2, ..., T_n \mid D_i) = \prod_{j=1}^{n} P(T_j \mid D_i) \qquad (2.7)$$

At this point, it should be remarked that the assumption of independence between query terms does not contradict the assumption that terms in queries have a specific order. It only states that every possible order of terms has the same probability.

The probability measure that ranks documents given a query $T_{1,}T_{2.....}T_n$, applying Bayes rule is:

$$P(D \mid T_1, T_2, ..., T_n) = P(D)\frac{P(T_1, T_2, ..., T_n \mid D)}{P(T_1, T_2, ..., T_n)} \qquad (2.8)$$

And using the independence between query terms, we use Equation 2.7 and we have

$$P(D \mid T_1, T_2, ..., T_n) = P(D)\frac{\prod_{i=1}^{n} P(T_i \mid D)}{P(T_1, T_2, ..., T_n)} \qquad (2.9)$$

Furthermore, it is known that $\sum_d P(D=d \mid T_1, T_2, ..., T_n) = 1$, so we can scale the above formula using a constant *C* such that $\frac{1}{C} = \sum_d P(D = d \bigcap T_1, T_2, ..., T_n)$ and we end up to the following:

$$P(D \mid T_1, T_2, ..., T_n) = CP(D) \prod_{i=1}^{n} P(T_i \mid D) \qquad (2.10)$$

One of the common ways to compute probabilities is the Maximum Likelihood Estimation (ML). This method makes the probability of observed events as high as possible and assigns zero probability to unseen events. Using ML for the *P(T|D)* is a problem because of the frequent assignment of zero value to unseen events. This problem can be faced by combining another model which is not so sparse, like the marginal *P(T)*. The linear combination of two probabilities estimates is called linear interpolation.

$$P_{LI}(T \mid D) = a_1 P_{ML}(T) + a_2 P_{ML}(T \mid D) \quad \text{where } 0 < a_1, a_2 < 1 \text{ and } a_1 + a_2 = 1 \qquad (2.11)$$

The frequencies which are used to estimate the probabilities of the model are the following:

N is the number of documents in the collection, $tf_{t,d}$ is term frequency (the number of times the term *t* appears in the document *d*), and df_t is document frequency (the number of documents in which the term *t* appears).

The most important of above frequencies are the term frequency and the document frequency.

Given a specific document many terms have zero frequency, so this measure suffers from sparseness. On the other hand, document frequency of a term has not zero values because the terms which are used in this model are appeared at least in one document. This problem can be faced if we estimate *P(T|D)* as linear combination of a probability model based on term frequency and another based on document frequency and we have:

$$P(T_i = t_i \mid D = d) = a_1 \frac{df_{t_i}}{\sum_t df_t} + a_2 \frac{tf_{t_i,d}}{\sum_t tf_{t,d}} \qquad (2.12)$$

Also it is apparent that :

$$P(D = d) = \frac{1}{N} \qquad (2.13)$$

The combination of *tf-idf* and document length normalization give satisfactory results in various test collections. So, we will try explain this weighting scheme with probability theory using the above equations.

Combining the Equations 2.10, 2.12, and 2.13 we have the following:

$$\begin{aligned}
&P(D = d \mid T_1 = t_1, \ldots\ldots, T_n = t_n) \\
&\propto \prod_{i=1}^{n} (a_1 \frac{df_{t_i}}{\sum_t df_t} + a_2 \frac{tf_{t_i,d}}{\sum_t tf_{t,d}}) \\
&\propto \prod_{i=1}^{n} (a_1 \frac{df_{t_i}}{\sum_t df_t} + a_2 \frac{tf_{t_i,d}}{\sum_t tf_{t,d}}) \times \prod_{i=1}^{n} \frac{\sum_t df_t}{df_{t_i}} \\
&\propto \prod_{i=1}^{n} (a_1 + a_2 \frac{tf_{t_i,d} \sum_t df_t}{\sum_t tf_{t,d} \cdot df_{t_i}}) \times \prod_{i=1}^{n} \frac{1}{a_1} \\
&\propto \prod_{i=1}^{n} (1 + \frac{tf_{t_i,d}}{df_{t_i}} \cdot \frac{1}{\sum_t tf_{t,d}} \cdot \frac{a_2 \sum_t df_t}{a_1})
\end{aligned}$$

The above formula is the tf_t - idf_t weighting algorithm with document length normalization where:

$\frac{a_2 \sum_t df_t}{a_1}$ is a constant for any document and term

$\frac{tf_{t_i,d}}{df_{t_i}}$ is the tf_t - idf_t for the term t_i in the document *d*

$\frac{1}{\sum_t tf_{t,d}}$ is the inverse length of document *d*

2.2.4 Variant $tf_t - idf_t$ functions

tf_t - idf_t scheme is very popular and for this reason there are many variation of this scheme, some of them are performed at the following sections.

2.2.4.1 Sublinear tf_t scaling

In this scheme the basic difference from tf_t - idf_t is that we do not take the number of occurrences of a term but the value: $1+\log tf_t$ [1] The main reason which lead to such a modification is that the total number of occurrences of a term is not equally significant .So, the logarithmic value of tf_t, approaches better the significant times that a term appears. The definition is the following:

$$wf_t = \begin{cases} 1+\log tf_t & if\ tf_t > 0 \\ 0 & otherwise \end{cases} \qquad (2.14)$$

If we replace tf_t by wf_t we obtain: $wf_t \times idf_t$

2.2.4.2 Maximum tf_t normalization

In this technique, we normalize the tf_t values by the maximum tf_t in a document [1]. Then the normalized term frequency is given by:

$$ntf_{t,d} = a + (1-a)\frac{tf_{t,d}}{tf_{\max}(d)}, \qquad (2.15)$$

where a is a smoothing term which takes values between 0 and 1 (usually is set to 0.4 or 0.5)

This scheme is used with the purpose of eliminating the fact that we observe higher term frequencies in longer documents.

2.2.4.3 Relevance frequency rf_t

This new factor is proposed by Lan et al [4] with the purpose of improving the term discriminating power.

The definition of relevance frequency is the following:

$$rf_t = \log(2 + \frac{a}{c}) \qquad (2.16)$$

where: the constant value is 2 because the base of this logarithmic operation is 2,

a is the number of documents which belongs to a category c_i and includes term *t,* and

c is the number of documents which belongs to remaining categories ($\overline{c_i}$) and includes term *t*

The document frequency with a different formalism is given by:

$$idf_t = \log(\frac{N}{a+c}) \text{ where } N = a+b+c+d \qquad (2.17)$$

The description of *a, c* is the same as before,

b is the number of documents that belong to c_i and not contain the term *t* (–

d is the number of documents that belongs to $\overline{c_i}$ and not contain term *t*

A synopsis of above definitions

category	t_k	$\overline{t_k}$
Positive : c_i	*a*	*b*
Negative: $\overline{c_i}$	*c*	*d*

TABLE2. 1: **TWO-WAY CONTINGENCY OF A TERM T AND A CATEGORY C**

In the Table 2.1 the notations t_k and $\overline{t_k}$ indicate the existence of not of a term.

It is apparent that *d* value is much larger from the remaining and dominates the results. So the discriminative power of *a, b, c* is reduced.

In the schema rf_t is given more weight to the terms which are assigned more in positive documents. The main reason is that the positive documents belongs to one category while the negative documents spread over the remaining categories.

It is combined often with tf_t and this weighting scheme is defined as:

$$tf_t \times rf_t \qquad (2.18)$$

2.3 FEATURE SELECTION

In the following paragraphs, we describe several methods for feature selection [10, 11]. These methods are very important because: (i) they reduce the dimensionality of the initial feature set and (ii) are able to improve the quality of the selected features, leading to better classification results.

2.3.1 Information Gain (IG)

Information gain is frequently used as a term-goodness criterion in machine learning [9]. It measures the number of bits of information obtained for category prediction when a term is present or absent in a document [10, 11].

Let denote $\{c_i\}_{m=1}^{m}$ as the set of categories, the information gain (IG) is defined as follows:

$$G(t) = -\sum_{i=1}^{m} P(c_i)\log P(c_i) + P(t)\sum_{i=1}^{m} P(c_i \mid t)\log P(c_i \mid t) + P(\bar{t})\sum_{i=1}^{m} P(c_i \mid \bar{t})\log P(c_i \mid \bar{t}), \quad (2.19)$$

where: $P(c_i)$ is the probability of having the category c_i, and $P(c_i \mid t)$ is the conditional probability of having the category c_i, given that the term t is appeared in this category. The remaining probabilities have the same explanation, but they include the complementary of the term *t* (no occurrence of *t*)

The Equation 2.19 is the more general and measures the importance of a term globally with respect to all categories on average.

Given a training corpus, for each unique term we compute information gain and remove those terms whose value of information gain is lower than a predetermined threshold. This method has complexity $O(VN)$ where *N is* the total number of documents and *V* is the vocabulary size.

2.3.2 Mutual Information (MI)

Mutual information is a criterion which is frequently used in statistical language modeling of word association. For the definition of this method, we recall Table 2.1 which represented the two way contingency of a term *t* and a category c_i.

Lets, remember again the definition of parameters: *a* is the number of times *t* and c_i co-occur, *b* is the number of times *t* occurs without c_i, *c* is the number of times c_i occurs without *t,* and *N* is the total number of documents

The mutual information between *t* and c_i is defined as:

$$I(t,c_i) = \log \frac{P(t \cap c_i)}{P(t)P(c_i)} \qquad (2.20)$$

and can be estimated using the definitions for *a, b, c* as:

$$I(t,c_i) \approx \log \frac{a \times N}{(a+c) \times (a+b)} \qquad (2.21)$$

The above equations compute the goodness of a term into a specific category, in the case that we want to estimate the significance of a term in a global feature selection, there are the two following alternatives:

$$I_{avg}(t) = \sum_{i=1}^{m} P(c_i) I(t,c_i) \qquad (2.22)$$

or

$$I_{\max}(t) = \max_{i=1}^{m} \{I(t,c_i)\} \qquad (2.23)$$

The weakness of mutual information is that the estimated values is influenced by marginal probabilities of terms. We re-write Equation 2.20 with the following formalization:

$$I(t, c_i) = \log P(t \mid c_i) - \log P(t) \qquad (2.24)$$

It is apparent that terms with equal conditional probabilities $\log P(t \mid c_i)$ have higher score if they are rare terms because $\log P(t)$ is lower.

Furthermore, we should note that mutual information computation have the same complexity as information gain $O(VN)$.

2.3.3 Chi-square (χ^2)

The χ^2 measures the lack of independence of a term *t* and a category c_i. For the definition of this method, we use the definitions for *a, b, c* from the previous section and also define a new parameter *d* which expresses the number of times neither c_i not *t* occurs. So, this measure is defined as follows:

$$\chi^2(t, c_i) = \frac{N \times (ad - cb)^2}{(a+c) \times (b+d) \times (a+b) \times (c+d)} \qquad (2.25)$$

The way for combining all categories scores for each term is the following:

$$\chi^2_{avg}(t) = \sum_{i=1}^{m} P(c_i) \chi^2(t, c_i) \qquad (2.26)$$

or

$$\chi^2_{\max}(t) = \max_{i=1}^{m}\{\chi^2(t, c_i)\} \qquad (2.27)$$

The computation of Chi-square has a quadratic complexity as the previous methods.

2.3.3.1 Assessing Chi-square as a feature selection method

The Chi-square can be compared to the χ^2 distribution [1] with one degree of freedom, so the Yates' correction should be used, which makes it harder to reach statistical significance. Whenever a statistical test is used multiple times the probability of getting at least an error increases. For instance, if 1000 hypotheses are rejected, each with 0.05 error probability then $0.05 \times 1000 = 50$ calls of the test will be wrong on average. We should note that in text classification ussually it doesn't matter, if a few terms are added or removed from feature set. Chi-square ranks features with respect to their usefulness and does not make statements about statistical dependence or independence of terms. So, it is not such important the fact that Chi-square is not strictly adhere to statistical theory.

2.3.3.2 Comparison between Mutual Information and Chi-square

Occurrence of a term in a large collection only once means that it is statistical significant, but it is not so informative. The Chi-square tends to select more rare terms than Mutual Information. On the other hand, Mutual Information does not necessarily select terms that maximize classification accuracy, but in general, both methods success comparative accuracy, which is good. Mutual Information reaches its peak value of accuracy selecting fewer terms than Chi-Square. The last method selects more significant features but rear terms, so it is essential to collect more terms with the purpose of covering all documents in the class. Even though Chi-square needs more terms are of better quality than those selected by Mutual Information.

These methods are characterized as *greedy* because they tend to select features that contribute no incremental information and this fact impacts accuracy.

2.3.4 Term Strength (TS)

Wilbur and Sirotkin [12] propose term strength for vocabulary reduction in text retrieval. The criterion of this method based on how common a term is in related documents. This method is applied to documents whose similarity is above a threshold. The document similarity is measured using cosine similarity (it will be explained in the Chapter 3). The definition of this measure is:

$$s(t) = P(t \in y \mid t \in x), \qquad (2.28)$$

where: *x*, *y* is pair of related documents .

So, $P(t \in y \mid t \in x)$ expresses the conditional probability that a term occurs in the second half of a pair of documents given that it occurs in the first half.

Term Strength based on the assumption that documents with many shared terms are related, so terms that are in the overlapping area of those documents are informative. An important parameter in this technique is the threshold on document similarity values. One solution is the using of a reasonable value of AREL, i.e., the average number of related documents per document. This can be achieved experimentally, by applying different thresholds.

The complexity of Term Strength is quadratic to the number of training documents.

2.3.5 Lagus and Kaski (LK) method

On the contrary to Chi-square which uses *df* , LK [13] uses only *tf.* This ranking formula is defined as:

$$f_{tc_i} = \left(\frac{tf_{tc_i}}{\sum_{t'} tf_{t'c_i}}\right) \cdot \frac{\left(tf_{tc_i} \Big/ \sum_{t'} tf_{t'c_i}\right)}{\sum_{c_i'} \left(tf_{tc_i'} \Big/ \sum_{t'} tf_{t'c_i'}\right)}, \qquad (2.29)$$

where: tf_{tc_i} is the average term frequency in class c_i, $\frac{tf_{tc_i}}{\sum_{t'} tf_{t'c_i}}$ is the importance of t in c_i relative to the importance of other terms, and $\frac{(tf_{tc_i} / \sum_{t'} tf_{t'c_i})}{\sum_{c_i'} (tf_{tc_i'} / \sum_{t'} tf_{t'c_i'})}$ is the importance of t in c_i relative to importance of t in all other classes.

There is a variation of above definition, which demonstrates the effects of extreme discrimination. In this method tf_{tc_i} is normalized over the whole collection in such way that a word which appeared 100 times in only one class c_i is equally important to a word that appears only once in a class c_i and never otherwise. This variation is defined as:

$$f_{tc_i,\mathrm{variation}} = \left(\frac{tf_{tc_i}}{\sum_{c_i'} tf_{tc_i}}\right) \cdot \frac{(tf_{tc_i} / \sum_{c_i'} tf_{tc_i'})}{\sum_{c_i''} (tf_{tc_i''} / \sum_{c_i'} tf_{tc_i'})}, \qquad (2.30)$$

The effects of this approach produce meaningful results when used with a specialized dictionary relevant to the purposes of classification.

2.4 SUMMARY

Term weighting methods have as main purpose to discriminate the most important terms and give them a higher value while feature selection techniques select features, which retain original physical meaning, providing a better understanding for the data. In many applications these methods are combined, improving the accuracy for text classification. In the following chapter we describe the main similarity metrics and two algorithms which are applied often in text categorization.

CHAPTER 3

SIMILARITY METRICS AND DOCUMENT CLASSIFICATION

3.1 INTRODUCTION

In the previous chapter we studied various term weighting and feature selection techniques for the discrimination of the most important terms of a document (or a set of documents) and assign them higher values. We modeled each document as a vector of terms weighted according to one of several terms. In this chapter we discuss metrics that attempt to provide a numerical estimation of semantic similarity between term vectors. These metrics are appealed either to binary vectors or to real-valued vectors. Real-valued vectors most times have the weights which are computed by term weighting methods.

Furthermore, we study one very important task of data mining, document classification. It is used to find valuable information from a huge collection of text documents available for instance in digital libraries, knowledge databases and web. A classifier has to be trained, in order to predict the class that an object belongs to. According to literature, machine learning algorithms are a proper choice for document classification. So, we describe two representative algorithms of machine learning, k-Nearest Neighbor (k-NN) and Support Vectors Machines (SVMs).

3.2 DEFINITION OF DISTANCE METRIC

The similarity functions are inverse formulations of distance functions, which compute dissimalarities between documents.

Let D be a set of elements called documents. A function $d : D \times D \rightarrow [0, +\infty)$ is called distance function and has the following properties:

i) $d(a,b) \geq 0$ and $d(a,b) = 0 \Leftrightarrow a = b$

ii) $d(a,b) = d(b,a)$

iii) $d(a,b) + d(b,c) \geq d(a,c)$

The last property is known as “triangle inequality”

A more formal mathematical justification [8] for the inverse relation between distance function and similarity ones, is presented below:

The Minkowski distances $L_p(a,b) = (\sum_{i=1}^{d} | a_i - b_i |^p)^{1/p}$ are the standard metrics for geometrical problems. For $p = 2$ we obtain Euclidean distance, which take values in the space $[0, +\infty)$ (with 0 closest). Using a monotonic decreasing function, it is possible for a distance metric to be converted into a similarity measure, which takes values in the space [0,1] (with 1 closest) .

For Euclidean space, we chose to relate distances *d* and similarities *s* using $s = e^{-d^2}$.

Consequently, Euclidean normalized similarity is defined as:

$$s^{(E)}(a,b) = e^{-\|a-b\|_2^2} \qquad (3.1)$$

3.3 MAIN SIMILARITY METRICS

In this section, we present briefly the notion of similarity metrics, as well as some useful properties of them. Next, we outline several metrics that are extensively applied for a wide range of classification tasks.

3.3.1 Definition and Properties

Similarity metrics [7] express a way to calculate the likeness between documents. In the following definitions $\overrightarrow{v_j}, \overrightarrow{v_k}$ are vectors, which contain binary values (0 if a term does not exist in the document, 1 if the term exists in the document) or real values which have been computed by a term weighting method. The notation $w_{i,j}$, $w_{i,k}$ is used to represent the values of the vectors $\overrightarrow{v_j}, \overrightarrow{v_k}$.

3.3.1.1 Simple matching coefficient (dot product)

Simple matching coefficient is the simplest similarity measure and is given by:

$$s_{j,k} = (\overrightarrow{v_j}, \overrightarrow{v_k}) = \sum_{i=1}^{n} w_{ij} w_{ik} \qquad (3.2)$$

If D_j and D_k are conceived as sets of terms, the set theoretic counterpart of the simple matching coefficient is:

$$S_{j,k} = \mid D_j \cap D_k \mid \qquad (3.3)$$

3.3.1.2 Dice coefficient

Dice coefficient is given by:

$$s_{j,k} = \frac{2 \cdot (\overrightarrow{v_j}, \overrightarrow{v_k})}{\sum_{i=1}^{n} (w_{ij} + w_{ik})} \qquad (3.4)$$

If D_j and D_k are conceived as sets of terms, the set theoretic counterpart of Dice's coefficient is:

$$D_{j,k} = \frac{2 \cdot |D_j \cap D_k|}{(|D_j| + |D_k|)} \qquad (3.5)$$

3.3.1.3 Jaccard coefficient

Jaccard coefficient is given by:

$$s_{j,k} = \frac{(\vec{v_j}, \vec{v_k})}{\sum_{i=1}^{n} (w_{ij} + w_{ik}) \Big/ 2^{w_{ij} \cdot w_{ik}}} \qquad (3.6)$$

If D_j and D_k are conceived as sets of terms, the set theoretic counterpart of Jaccard's coefficient is:

$$J_{j,k} = \frac{|D_j \cap D_k|}{(D_j \cup D_k)} \qquad (3.7)$$

3.3.1.4 Overlap coefficient

Overlap coefficient is given by:

$$s_{j,k=} \frac{(\vec{v_j}, \vec{v_k})}{\min(\sum_{i=1}^{n} w_{ij}, \sum_{i=1}^{n} w_{ik})} \qquad (3.8)$$

If D_j and D_k are conceived as sets of terms, the set theoretic counterpart of Overlap coefficient is:

$$O_{jk} = \frac{|D_j \cap D_k|}{\min(|D_j|, |D_k|)} \qquad (3.9)$$

3.3.1.5 Cosine Similarity

Cosine similarity is given by:

$$s_{j,k=} \frac{(\overrightarrow{v_j}, \overrightarrow{v_k})}{(\| \overrightarrow{v_j} \| \cdot \| \overrightarrow{v_k} \|)} \qquad (3.10)$$

If D_j and D_k are conceived as sets of terms, the set theoretic counterpart of cosine similarity is:

$$C_{j,k} = \frac{| D_j \cap D_k |}{(| D_j | \cdot | D_k |)^{1/2}} \qquad (3.11)$$

3.3.2 Main characteristics of similarity metrics

The main properties of these metrics are the following:

i) They are usually normalized (except simple matching coefficient) and take on values between 0 and 1 (property of normalization)

ii) Their values does not depend on the order, so, they are interchangeable in formulae (property of symmetry or commutativity)

iii) A value equal to 1, stand for absolute similarity (property of reflexivity)

We note that these metrics are different normalized form of simple matching coefficient.

All above can be mathematically formalized as follows:

Let D be a set of elements called documents. A function $\sigma : D \times D \rightarrow [0,1]$ is called similarity measure if the following three properties hold:

i) $0 \leq \sigma(a,b) \leq 1, \ \forall a,b \in D,$ normalization

ii) $\sigma(a,b) = \sigma(b,a), \ \forall a,b \in D,$ symmetry or commutativity

iii) $a = b \Rightarrow \sigma(a,b) = 1, a,b \in D,$ reflexivity

3.4 ALGORITHMS FOR DOCUMENT CLASSIFICATION

In the following paragraphs we describe some methods for classification: (i) k-nearest neighbor, (ii) centroid based algorithm and (iii) support vector machines. The effectiveness of these methods regarding document classification has been proved in many works [15, 16]. The goal of document classification is strongly related with the approach proposed by this thesis.

3.4.1 k-Nearest Neighbor Classification

A k-NN classifier decides the class of an object by analyzing its k nearest neighbors [1] within the training class. Particularly, the criterion is which class is most common amongst its k nearest (majority voting), so this algorithm is sensitive to the local structure of data. If k=1 (1-NN), then the classification decision relies on the class of a single training, so it is not very robust, for this reason it is preferable k to be greater than one. It is obvious that the choice of k is very significant. It is used k to be odd as this avoid ties votes. The choice of parameter k is often based on experience, most common choices are k=3 and k=5, but they also used much larger values between 50 and 100, accordingly to application. Larger values of k reduce the effect of noise on the classification, but the boundaries between classes are less distinct.

A good k can be selected by various heuristic techniques, for example cross-validation [14]

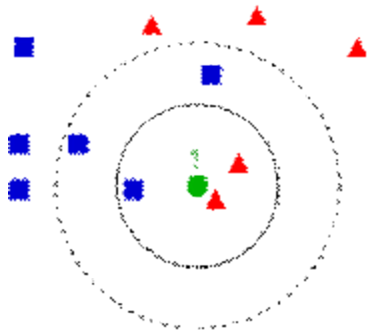

FIGURE3. 1**: EXAMPLE OF K-NN CLASSIFICATION**

This figure illustrates that the classification of an object depends strongly to the parameter k. If we choose k=3 is classified to the class of triangles (2 triangles, 1 square) , on the other hand if we choose k=5 is classified to the class of squares (3 squares, 2 triangles) .

In this algorithm, it is not required an explicit training step. Training set is a set of objects for which the classification is known. In order to identify neighbors the distance metric, which used mostly, is the Euclidean.

A more accurate way for computing a class score is to weight the votes of k-NN by using cosine similarity metric as follows:

$$score(c,d) = \sum_{d' \in S_k(d)} I_c(d') \cos(\vec{v}(d'), \vec{v}(d)), \qquad (3.12)$$

where $S_k(d)$ is the set of k nearest neighbors of d and $I_c(d')$ indicates if d' belongs to class c by taking value 1 and 0 respectively.

Then, we assign the object of test set to the class with highest value. This weighting scheme solves the problem of tied votes because we measure similarity and not the exact number of neighbors.

The naïve version of algorithm is easy to be implemented but it has same major drawbacks:

1. The class with the most frequent samples tends to dominate the prediction of an object, as they tend to come up in the k nearest neighbors due to their large number.
2. The efficiency of classification is decreasing with the number of training objects
3. Classification time strongly depends on the number of training objects

For above reasons much research effort have been put into selecting or scaling features to improve classification. For instance in [15], it is proposed a new method, which called density-based-instances-reduction (DBIR), to eliminate the training instances.

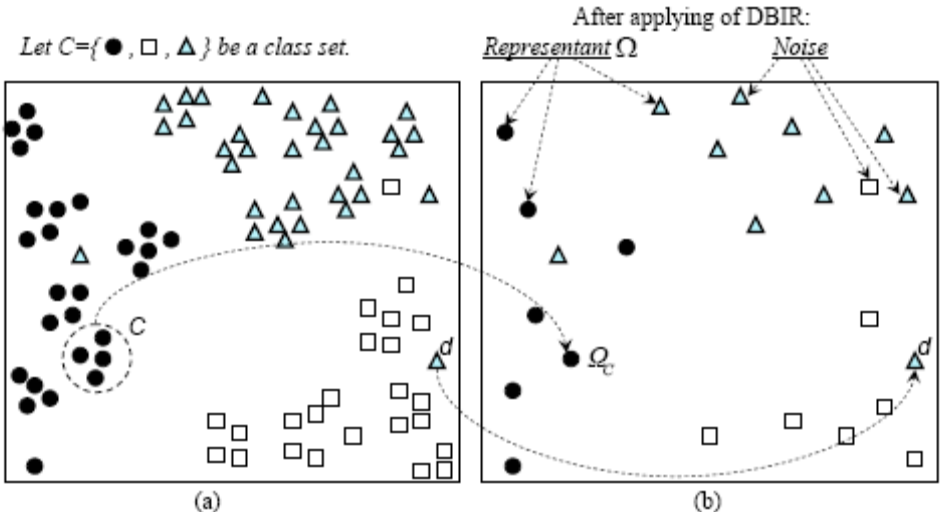

FIGURE3. 2**: OBJECTS BEFORE AND AFTER DATA REDUCTION USING DBIR**

3.4.3 Centroid based Algorithm

Instead of considering similarity of an unclassified document to all docs in a category, a natural alternative is to somehow take a single representative document per category, called a prototype, and to compare the unclassified document to each of the category prototypes. At least intuitively,this is bound to save computation if compared to the kNN approach in the previous section.

For a given category the prototype vector is the category representative vector of the documents assigned to this particular category. There are a number of different approaches to what is considered the optimal prototype vector. In this algorithm the prototype vector is computed by averaging the co-occurring weights from documents in a train set of a given category [35, 36]. The result is an efficient method that is relatively easy to implement.

The prototype for each category is defined as follows:

$$\overrightarrow{c_k} = \frac{1}{|C_k|} \cdot \sum_{i \in C_k} w_{i,j} \qquad (3.13)$$

where $|C_k|$ is the number of documents in the training data of a category amd $w_{i,j}$ is the weight for each element of the feature vector.

Once the prototype for each category has been calculated, the unclassified document is compared with the prototype for each category and thus assigned to the category which has the best similarity score. The definition below is used for computing this similarity:

$$\cos(\vec{d}_{i,j}, \vec{c}_{i,j}) = \frac{\vec{d}_{i,j} \bullet \vec{c}_{i,j}}{\left\|\vec{d}_{i,j}\right\| \times \left\|\vec{c}_{i,j}\right\|} = \frac{\sum_{i=1}^{n} \vec{d}_{i,j} \times \vec{c}_{i,j}}{\sqrt{\sum_{i=1}^{n} \left(\vec{d}_{i,j}\right)^2 \times \sum_{i=1}^{n} \left(\vec{c}_{i,j}\right)^2}} \quad (3.14)$$

where $\vec{d}_{i,j}$ is the document vector and $\vec{c}_{i,j}$ is the prototype vector.

3.4.3 Support Vectors Machines

Support Vectors machines (SVM's) is a supervised algorithm [16]. Considering data as two set of vectors in a n-dimensional space, an SVM constructs a separating hyperplane in that space, which maximizes the margin between two data sets.

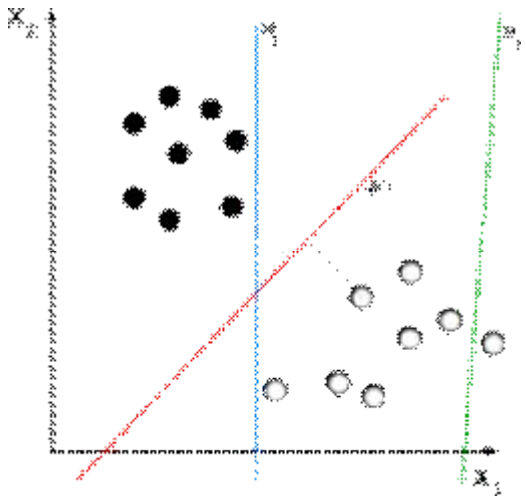

FIGURE3. 3: **THREE DIFFERENT HYPERPLANES H1, H2,H2**

As we can see from Figure 3.3, H3 doesn't separate the 2 classes. H1 does, with a small margin and H2 with the maximum margin.

At this point, they are performed some mathematical formalization about how classification works in Support Vectors Machines.

Let consider the training data as set points of the form:

$$D = \{(x_i, c_i) \mid x_i \in \mathbb{R}^p, c_i \in \{-1,1\}\}_{i=1}^{n} \qquad (3.15)$$

where is either 1,or-1, indicating the class to which the point x_i belongs. Each x_i is a *p*-dimensional real vector. As we referred previously, the points of the classes should be divided by an hyperplane which maximizes the distance from the neighbor datapoints of both classes. Based on Equation 3.13 we realize that the hyperplane should divide the points having $c_i = -1$ from those having $c_i = 1$. Any hyperplane is a set of symbols $\vec{x}$ which satisfying:

$$\vec{w} \cdot \vec{x} - b = 0 \qquad (3.16)$$

where $\vec{w}$ is a normal vector which perpendicular to hyperplane and the parameter $\frac{b}{\|\vec{w}\|}$ determinates the offset of the origin along normal vector $\vec{w}$.
We want to choose $\vec{w}$ and *b* such that the margin to be maximized. If the training data is linear separable, we select the margin hyperplanes in a way that there are no points between them.

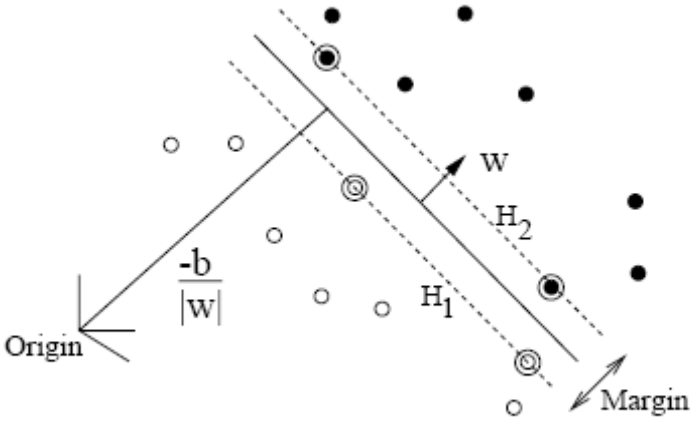

FIGURE3. 4: **LINEAR SEPARATING HYPERPLANES FOR THE SEPARABLE CASE. THE SUPPORT VECTORS ARE CICLED.**

SVMs belong to a family of generalized linear classifiers and they are also known as maximum margin classifiers due to their property to maximize the geometric margin.

In the case that does not exist a hyperlane which can split the training data, is used a method which is called *soft margin*. This choose a hyperplane that spilts the training data as distinctly as possible. This modification has as a result the popularization of SVMs. This method is described as follows:

$$c_i(\vec{w}\cdot\vec{x_i}) \geq 1-\xi_i \qquad 1 \leq i \leq n \qquad (3.17)$$

In Equation 3.15 is introduced a new variable ξ_i which measures the degree of misclassification of the element x_i (error penalty).

The optimazition of the hyperplane becomes a trade off between a large margin and a small error penalty.

The below figures depict the non separable case.

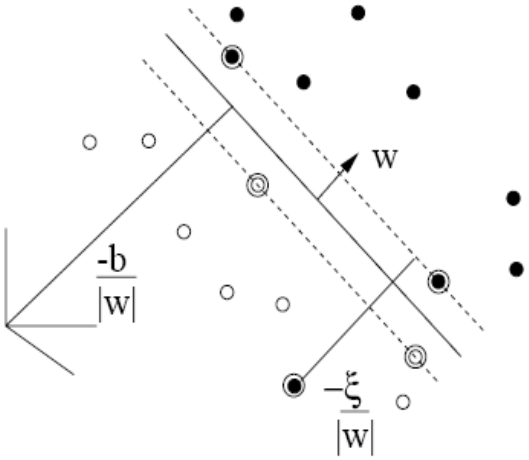

FIGURE3. 5: **LINEAR HYPERPLANES FOR THE NON-SEPARABLE CASE.**

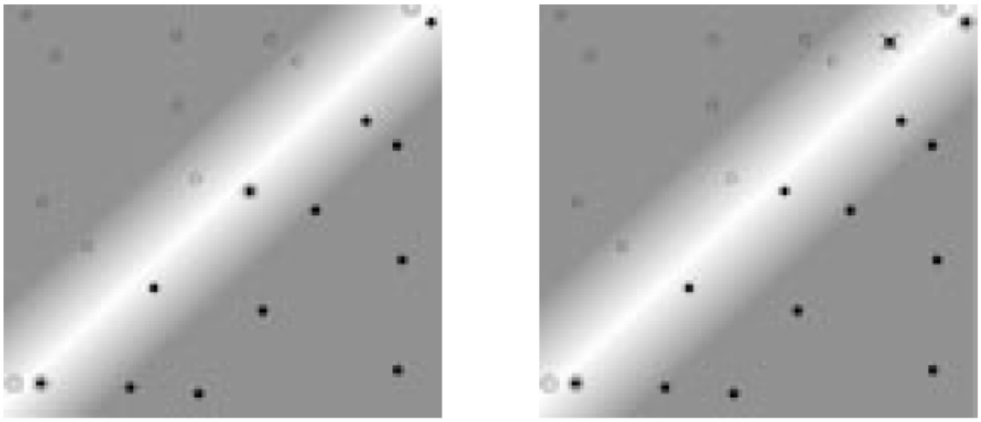

FIGURE3. 6: **THE LINEAR CASE, SEPARABLE (LEFT) AND NOT (RIGHT)**

Furthermore, we should note that in 1992 Bernard Roser et al. [17] suggested a way to create non-linear classifiers using kernel trick. Kernel trick is a method for using linear classifier observations into a higher-dimensional space, where the linear classifier is subsequently used. The classifier is a hyperplane in the high dimensional feature space, but it is not linear in the original input space.

Some kernels functions are:

1. Polynomial (homogeneous) : $k(\vec{x},\vec{x}') = (x \cdot x')^d$
2. Polynomial (inhomogeneous) : $k(\vec{x},\vec{x}') = (x \cdot x' + 1)^d$
3. Radial Basis Function: $k(\vec{x},\vec{x}') = \exp(-\gamma \| \vec{x} - \vec{x}' \|^2)$ for $\gamma > 0$
4. Gaussian Radial Function: $k(\vec{x},\vec{x}') = \exp(-\frac{\| \vec{x} - \vec{x}' \|^2}{2\sigma^2})$
5. Sigmoid: $k(\vec{x},\vec{x}') = \tanh(k\vec{x} \cdot \vec{x}' + c)$, for some (not every) k>0 and c<0

3.5 SUMMARY

In this chapter we studied some basic similarity metrics and their main properties. Their numerical estimation can be used for text classification. The documents can be classified using the maximal similarity to a class. Thereafter, we presented some more sophisticated classification methods which score significant results in text classification. We apply these algorithms to our experiments, as we see in the following chapters, and the results were satisfactory.

CHAPTER 4

MUSIC GENRE CLASSIFICATION

4.1 INTRODUCTION

In this chapter we describe the method we follow for music genre classification based on web document resources. Classification of music is a challenging task. One of the most common approaches is to classify music into genres. In the following sections we analyze in what way we retrieve and process the web pages for artists. The documents are modeled as feature vectors. This model is appropriate for applying the document categorization techniques, which are described in the following paragraphs. Furthermore, we describe how we combine web with audio distances.

4.2 RELATED WORK

The methods for measuring artist similarity and music classification can be divided into three categories according to the type of features that they use. In the following subsections are performed the related work for web-based information, audio features and their combination.

4.2.1 Music Classification based on lexical features

There are many ways for extracting information from web. One of them is by using information which is available in common web sites. The concept is to retrieve top ranked pages from a search engine and extract the desirable features by using data mining techniques [18, 25]. In [18], is proposed a method for artist classification. They use a dataset which consists of 224 artists divided into 14 genres. After retrieving 50 top-ranked web pages for each artist and by applying text processing techniques, each artist is represented by a feature vector. Afterwards, they classifying artists using appropriates methods for document categorization, such as SVM's and k-NN and they achieve high accuracy. In [19], is reported an extension of above work, by applying this method to different datasets. They focus on techniques for removing the irrelevant pages and they

also proposed a simplified method for extracting features for artists, which achieve similar results to more complex ones which are proposed in [18]. They achieve classification accuracy which is slightly better which means that the already good results cannot be easily improved. The above methods are supervised, which means that the artist should be labeled, on the contrary, in [20], is proposed an hierarchical description of music collections. The artists are grouped into overlapping clusters according to their similarity.

One other efficient way is to find artist similarity is using co-occurrences on the web. Co-occurrence analysis is based on the idea that if two artists appeared in the same context there is some kind of similarity between them. In [21], using only the top ranked pages they estimate the number of the web pages containing each artist and each pair of artists. The remaining content of the web page is ignored. Furthermore, artist similarities are computed and the classification accuracy of the system is estimated by using k-NN. A similar method is studied in [22], but it is applied an extra technique which is said pattern-based co-occurrence count. Using of patterns which express the desired relations can provide access to relevant data. Furthermore, by exploiting co-occurrences on the web can be used for defining a genre or music style [23]. One method which is stated in [24] focus on structure data exploiting the co-occurrences of samplers and radio stations playlist. Using this information, the calculate similarities between artists and songs.

Additionally, useful information can be obtained using based on user preferences [26] and rating [27]. In [27] is described a method for estimating similarity between songs retrieving rating data for them.

4.2.2 Music Classification based on audio features

In this paragraph we are referred to audio information retrieval briefly. Proposes of a set of signal for direct modeling of music signal are cited in [28]. They use these features for music genre classification applying k-NN and Gaussian mixtures models. In [29], it is reported a method for music classification into rock, piano and jazz based on timbral features. A classifier of many types of audio features is proposed in [30] and model classes of music with frame-level features. There are many publications for music classification based on signal analysis of music but it is out of scope of this thesis to analyze them further.

4.2.3 Music Classification by combining lexical and audio features

Some works are based on both methods described in Sections 4.2.1 and 4.2.2. Important research have been on combining features derived from audio recordings and community metadata that was derived from data mining text from the web [31]. Many useful applications [32, 33], have developed due to combination of audio and web data features. In [32] is proposed a method for building a search engine that is capable of finding music that satisfies natural language queries. The retrieved web based information is complemented by audio based similarity which improves the results of the retrieval. In [33], is presented an automatic recorder of reviews, the function of which is established on audio-to-terms relation.

4.3 FEATURE EXTRACTION & CLASSIFICATION METHODS

In this section, we sketch how we built our system for music genre classification. Our method relies on the steps that described in [18].

Given a list of artist names, we use Yahoo! to retrieve the top ranked web pages for each artist. The appropriate choice of the query is very important because eliminates the number of irrelevant pages. In this method we used the query which is proposed in [18], "artist"+music+review. After retrieving web pages, we remove all HTML tags, keeping only the plain text. Furthermore, we use the stop word list from Cornel University for removing common words, like a, I, and, my etc. because they do not offer any information.

We do the above procedure for different number of retrieving pages (50, 100 and 200) with the purpose of observing how this parameter influences our results.

The scheme we use for weighting terms is tf - idf . So, for each artist *a*, and each term *t* appeared in the retrieved documents for *a,* we count the number of occurrences $tf_{t,a}$ and also, the df_t , which is the number of documents, where the term *t* is appeared.

If we denote *N* the total number of retrieved pages this weighting schema is given by:

$$w_{t,a} = \begin{cases} (1+\log tf_{t,a}) \log_2 \frac{N}{df_t}, & \text{if } tf_{t,a} > 0 \\ 0, & \text{otherwise} \end{cases}$$

As we referred in Chapter2 $w_{t,a}$ has high value when *t* occurs many times within a small number of documents and it has low value when it occurs fewer times in document or occur in many documents.

The Tables 4.1, 4.2, 4.3, 4.5 present the terms which have the higher *tf-idf* value for the artists/bands: Tchaikovsky, Elvis Presley, Madonna, Deep Purple and Eminem, respectively. We refer some information for certain terms in order to understand how discriminative these terms are.

Tchaikovsky

TERMS	***SHORT DESCRIPTION***
Balakirev	Russian composer
Tchaikovsky	The name of artist
Swan	Composition of Tchaikovsky "Swan of Lake"
Russian	Origin of Tchaikovsky
Moscow	The capital of Russia

TABLE 4. 1: **TERMS WITH HIGH TF-IDF VALUE FOR TCHAIKOVSKY**

Elvis Presley

TERMS	***SHORT DESCRIPTION***
Priscilla	Wife of Elvis Presley
Guralnick	Music critic who wrote his biography
Graceland	Presley's villa located in Memphis
Colonel	Colonel Thomas Parker was his manager
Jailhouse	"Jailhouse Rock" –Presley's song

TABLE 4.2 **TERMS WITH HIGH TF-IDF VALUE FOR ELVIS PRESLEY**

Madonna

TERMS	***SHORT DESCRIPTION***
Madge	Madonna's nickname
Kabbalah	Her religion
Immaculate	Her album: "The immaculate collection"
Shep	Shep Pettibone: remixer
Ciccone	Her surname

TABLE 4.3: **TERMS WITH HIGH TF-IDF VALUE FOR MADONNA**

Deep Purple

TERMS	*SHORT DESCRIPTION*
Blackmore	Ritchie Blackmore-former guitarist of band
Morse	Steve Morse-current guitarist of band
Gillan	Ian Gillan- former singer of band
Glover	Roger Glover-bassist of band
Paice	Ian Paice- drummer of Deep Purple

TABLE 4.4: **TERMS WITH HIGH TF-IDF VALUE FOR DEEP PURPLE**

Eminem

TERMS	*SHORT DESCRIPTION*
Mathers	His surname
Shady	His album: "The Slim Shaddy"
Mockingbird	His song
Curtain	His album: "Curtain Call"
Puke	His song

TABLE 4. 5:**TERMS WITH HIGH TF-IDF VALUE FOR EMINEM**

Terms such as song/album titles, artist names or specific characteristics of their personal life are weighted highly because has discriminative value.

We use two approaches for reduce the dimensionality for each vector:

(i) We keep the term weights for terms which are appeared in at least 5 documents

(ii) We use a standard feature selection approach, which is called Chi-square.

The Chi-square measures the independence of a term *t*, from a category *c* by selecting terms which are important for each category and have discriminative power between category *c* and all others. This method is defined as follows:

$$\chi^2_{t,c} = \frac{N(AD - BC)^2}{(A+B)(A+C)(B+D)(C+D)},$$

where *A* is the number of documents in c which contain *t, B* is the number of documents not in *c* which contain *t*, *C* is the number of documents in *c* without *t* and *D* is the number of documents not in *c* that not contain *t.*

Before we apply this feature selection method each artist consists of all terms which are appeared to artists' documents, so the dimensionality of the feature vector of a genre was approximately 5000-15000, depending on the number of retrieved pages.

Now the feature vector of each artist is defined in a different way. We keep the terms with *n* highest chi-values for each category and join them into a global list. For instance, if we keep 100 highest values with no overlap between them and we have 10 genres of music, each artist is described by a vector with 1000 dimensions. For these terms we assign tf - idf values, which are computed in the previous step.

The table below shows the terms with 30 highest chi values for 5 illustrative music genres, that we use in this implementation.

classic	rap/hip-hop	heavy-metal	reggae	electro
symphony	rap	metal	reggae	techno
composers	eminem*	heavy	dancehall	electronic
von*	rapper	judas*	marley*	electronica
orchestra	dre*	sabbath*	shaggy*	dj
beethoven*	rhymes	iron*	ub*	daft*
composer	rappers	skid*	eddy*	moby*
symphonies	hip	megadeth*	ziggy*	fatboy*
violin	gangsta	purple*	maxi*	cox*
opera	coast	leppard*	jah	jaxx*
ludwig*	notorious	ozzy	capleton*	basement*
concerto	ll*	maiden*	grant*	twin*
vienna	hop	rob	jamaica	aphex*
conductor	enemy*	mustaine	alpha*	moloko*
composed	busta*	metallica*	blondie*	house*
german	pac*	priest*	circle*	duo
mozart*	grandmaster*	row*	makers	ambient
works	dmx*	riffs	bob*	prodigy*
haydn*	ya	ian	morgan	norman
orchestral	shady	slave	jamaican	discovery
friday	cent*	hysteria	conscious	kish
der	snoop*	dave*	bonafide	bangalter
classical	furious	dickinson	dem	felix
chamber	chronic	beast	labour	kash
philharmonic	tupac	kiss*	dub	rooty
sonatas	mathers	thrash	shabba*	roisin
franz*	flash*	powerslave	dragonfly	drukqs
herbert*	shakur	halford	avenue	rave
joseph*	flavor	countdown	reign	homework
mahler*	dogg*	harris	boombastic	buxton
chopin*	dj	gillan	crossover	pair

TABLE 4. 6: **TERMS FOR 5 GENRES WITH HIGHEST CHI-VALUES**

Observing the Table 4.1 we realize that the artists' names have high discriminative power for each genre(artist' names which are included in our dataset have next of their name the symbol *). They also appeared and other artists who are not part of the query terms and belong to the appropriate category. We also realize that common words which appeared too frequent to the retrieved documents, like music, artist and review have been rejected because they offer no amount of information.

For the classification of artists, we use three different methods either using feature selection or not:

(1) k-Nearest Neighbor

We use 7-NN with two different criteria for assigning an artist to a genre.

(a) Euclidean distance for the computation of the neighbors and then applying simple majority voting for deciding the genre of an artist.

(b) Using the mathematical formula for computing cosine similarity between artists

$score(c,d) = \sum_{d' \in S_k(d)} I_c(d') \cos(\vec{v}(d'), \vec{v}(d))$, where $S_k(d)$ is the set of k nearest neighbors of d and $I_c(d')$ indicates if d' belongs to class c by taking value 1 and 0 respectively. An artist is assigned into a genre using maximum similarity.

(2) Centroid based classifier- Cosine Similarity

To reduce the demand for processing cost we also implement a centroid classifier. This approach makes a centroid document for each category and then ranks the categories. The genre of each artist is decided according to maximum cosine similarity between category's centroid document and artist's document.

We define each genre selecting some of the artists and the remaining is used for test.
The cenroid prototype for each genre is computed as follows:

$$\overrightarrow{g_k} = \frac{1}{|G_k|} \cdot \sum_{i \in G_k} w_{i,j}$$

The similarity artist-genre is computed as:

$$\cos(\vec{a}_{i,j}, \vec{g}_{i,j}) = \frac{\vec{a}_{i,j} \bullet \vec{g}_{i,j}}{\left\|\vec{a}_{i,j}\right\| \times \left\|\vec{g}_{i,j}\right\|} = \frac{\sum_{i=1}^{n} \vec{a}_{i,j} \times \vec{g}_{i,j}}{\sqrt{\sum_{i=1}^{n} \left(\vec{a}_{i,j}\right)^2} \times \sqrt{\sum_{i=1}^{n} \left(\vec{g}_{i,j}\right)^2}}$$

where a, g are the feature vectors which describe an artist and a genre, respectively.

(3) SVMs

For SVMs, we use a linear kernel as implemented in Matlab OSU toolobox. SVMs is an appropriate method for high dimensional and sparse data. SVM's are used widely for text classification. They belong to Machine learning algorithms and are known for their good performance. We prove that claim with the experimental results which will be presented in the next Chapter.

4.4 FUSION

Furthermore, we want to examine how the classification accuracy is affected when we combine completely irrelevant features. In this method, we apply fusion of audio based and web based distances from genres.

Extraction of lexical features

The lexical features are extracted as is described in the previous section. We download 100 pages for each artist, we compute tf_idf ,chi-values and we apply centroid based algorithm for classifying the artists.

Extraction of audio features

The audio based features, which are used [34], are the mel-frequency ceptral coefficients (MFCC's) which consist a compact representation of the spectrum of an audio signal. They are short time spectral features. The mel-frequency cepstral coefficients algorithm is based on transformation from time domain to frequency domain and filtration with perceptual

filterbank. Firstly, for each frame, is calculated the fast fourier transform and the result is stored in a vector F. The result of this operation is being filtered with each filter from Mel filterbank and the result is aggregated and stored in a vector S. After that, the logarithm of the vector S is being calculated. Finally, this vector is being transformed by discrete cosine transform.

The whole procedure is described by the follow equations:

(1) $F(i) = [real(F(i))]^2 + [imag(F(i))]^2$

(2) $S(k) = \sum_{i=0}^{N/2} (F(i) \cdot M(i))$

(3) $L(m) = \log(S(k))$

(4) $C(n) = \sum_{i=0}^{L-1} L(i) \cdot \cos\left(\frac{\pi n}{2L}(2i+1)\right)$

Where F is the result of the FFT, k is every filter, C is the Mel frequency cepstral coefficients and n the coefficient number of M coefficients. The m MFCC features are organized in a fxm matrix, where each row consists of the m MFCC values for a frame and there are f rows, the number of frames into which the signal has been segmented.

The reason that is used the mel scale is because many experiments has shown that the ear's perception to the frequency components in the speech does not follow the linear scale but the mel-frequency scale, which should be understood as a linear frequency spacing between 1KHz and logarithmic spacing above 1KHz. The common used formula that reflects the relation between the mel frequency and the physical frequency is given by:

$M(f) = 1125 \cdot \log\left(1 + \frac{f}{700}\right)$, where f is the frequency in hertz.

The classification algorithm, which is used, is k-means. This algorithm classifies objects based on attributes and features into k number of groups. K is a positive integer number. This is one of the simplest unsupervised learning algorithms that solve the clustering problem. The main idea is to define k centroids, one for each cluster. These centorids should be placed in an appropriate way because different location could cause different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. The first grouping has completed. Now, the k new centorids that have

produced must be recalculated. This loop is repeated until the k centorids do not change their location. This means that they do not move any more.

However there are some difficulties with this algorithm. Firstly, although the algorithm will produce the desired number of clusters, maybe the centorids will not be representative of the data. Secondly, the method is computationally inefficient. Each step of the procedure requires calculation of the distance between every possible pair of data points and comparison of all the distances. This requires a lot of time. The main drawback of the k-means algorithm is that the cluster result is sensitive to the selection of the initial cluster centroids. The initial choice is of great importance for the whole procedure. If it could be ensured good initial clustering centorids using other techniques, then the k-means algorithm would work properly to find the optimal clustering centers. It is clear that in order to use this algorithm, the number k of clusters need to be specified. However, for continuous distributions, there exists a set of k principal points for all positive integers k. There is no right or wrong values for k. Instead, the appropriate choice depends on the particular application and must be determined by the investigator.

For the fusion, we use the distances from artists to music genre. The exact methodology which is followed is described below:

All songs are given as mp3-files and they were converted to .wav files since the build-in routines that Matlab contains can process .wav files. This was achieved using the lame decoder, an open source tool. Since these mp3 files have been created by extracting from audio CD's, the sampling frequency is 44100 Hertz (Hz), which means that the amplitude value of the audio signal is scanned and stored 44100 times per second. However, the processing of wave files and the feature extraction can be done with a lower quality of sound. So, the sampling frequency is reduced at 16000 and 11025Hz. Now the wave files are ready to be processed. Every wave file is processed with the frontend and the MFCC's features extracted from the wave files.

After extracting MFCC's from the songs of each music genre, they are separated into training and test and the k-means algorithm is applied to training data. The output of k-means algorithm is a table of centroids for each category. Then, it is computed the Euclidean distances for each song from them. Each song is assigned to the category, from which the Euclidean distance is minimum.

Next, the songs of each artist are sorted out and it is computed the mean value of above distances. These values represent the distance of each artist from the categories.

Steps for fusion

We recall that in web based data is used cosine similarity, so the criterion for assigning an artist to a category is the maximum similarity. So, for combining web based with audio based distances we should normalize the distances such that they have the same range of values. Furthermore, we take the inverse values of similarity and add them with audio based ones:

$$dist_{fusion} = w \cdot dist_{audio} + (1-w) \cdot \frac{1}{sim_{web}},$$

where *w,* (1-*w*) take values 0-1 and are the weight which is given to each distance, $dist_{audio}$ is the distance for audio data and $\frac{1}{sim_{web}}$ the distance for web data.

The genre of each artist is decided by the minimum value of $dist_{fusion}$.

4.5 SUMMARY

In this chapter, we referred the basic steps that we follow at this thesis. We presented the schemes for term weighting and selection that we used, the algorithms for document classification, as well as the procedure for doing the fusion of web based with audio based distances. In Chapter 5, we present the results that come up from this approach and evaluate them.

CHAPTER 5

EXPERIMENTS AND RESULTS

5.1 INTRODUCTION

In this chapter we describe the experiments and the results of our thesis. Firstly, we describe our datasets and we refer in which experiments they used. We use three different classification methods, different number of retrieved pages as well as feature selection methods. We try to explain how each of above parameters affect our results, observing the classification rates. The last experiments which are carried up are concerned fusion method and we evaluate it.

5.2 DATASETS

For our experiments we use two different datasets. To evaluate our method we use mainly the first dataset which consists of 224 artists who divided equally into 14 genres, this dataset is taken form [18]. The genres are appeared in the Table 5.1. The second dataset is used for the experiments which are for the fusion of data. Our purpose is to have the same dataset with Anthi Markaki, who classified music based on audio data in her diploma thesis. It contains 113 artists who belong to 9 genres. Each genre has different amount of artists. In Table 5.2 is shown the genres for the second dataset.

GENRES
country
folk
jazz
blues
rnbsoul
heavy-metal
altindie
punk
Rap/hip-hop
electro
reggae
classic
Rock'n'Roll
Pop

TABLE5. 1**: MUSIC GENRES OF DATASET 1**

GENRES
Classical
Dance-Electro
Hard rock-Heavy
Jazz-Blues
Latin
Punk-alternative
Rap
Rock-pop-classic
Rock-pop-alternative
Classical
Dance-Electro

TABLE5. 2: **MUSIC GENRES OF DATASET 2**

5.3 EXPERIMENTS AND RESULTS FOR CLASSIFICATION

Each artist is represented as a feature vector where each dimension corresponds to a separate term and its value is the tf - idf estimation. For the training data, which define a genre we select randomly the half artists. For these experiments, we use the dataset 1 so, it consists of 8 artists. The remaining 8, are our testing set. The classification accuracies are estimated via 50 experiments, each time training and test set are defined randomly.

Furthermore, every experiment is repeated for different number of retrieved web pages with and without employing a feature selection method.

For each algorithm, the overall results are presented in one table (Tables 5.3, 5.4, 5.5, 5.6) for comparative reasons. We also consider essential to represent a table (Table 5.7) with analytical results, so that we can realize how accurate the classification for each genre is.

The classification algorithms which are applied as it referred in previous chapter are: 7-NN using Euclidean distance as evaluation method (Table 5.3), 7-Nearest Neighbor using cosine similarity as evaluation method (Table 5.4), Centroid based algorithm (Table 5.5) and Support Vector Machines (Table 5.6)

We present the classification rate for each algorithm in the below tables:

	for 50 docs	*for 100 docs*	*for 200 docs*
classification rate (no fs)	45.91%	47.50%	51.84%
classification rate (with fs)	46.04%	67.54%	78.43%

TABLE5. 3**: CLASSIFICATION RATE FOR 7-NN (EVALUATION USING EUCLIDEAN DISTANCE)**

	50 docs	*100 docs*	*200 docs*
classification rate (no fs)	75.12%	80.95%	82.48%
classification rate (with fs)	85.16%	87.96%	87.95%

TABLE5. 4**: CLASSIFICATION RATE FOR 7-NN(EVALUATION USING COSINE SIMILARITY METRIC)**

	50 docs	*100 docs*	*200 docs*
classification rate (no fs)	82.82%	88.57%	89.66%
classification rate (with fs)	88.21%	90.46%	**92.45%**

TABLE5. 5**: CENTROID BASED ALGORITHM**

	50 docs	*100 docs*	*200 docs*
classification rate (no fs)	78.46%	86.32%	88.5%
classification rate (with fs)	83.27%	87.68%	91.41%

TABLE5. 6**: CLASSIFICATION RATE FOR SVM**

Observing the Tables 5.3, 5.4, 5.5, 5.6 we end up to the following valuable conclusions:

1) In most cases the classification is more accurate for bigger number of retrieved documents. It is obvious that the information amount is greater than the noise that the further number of documents introduces.

The Figure 5.1 depicts this observation:

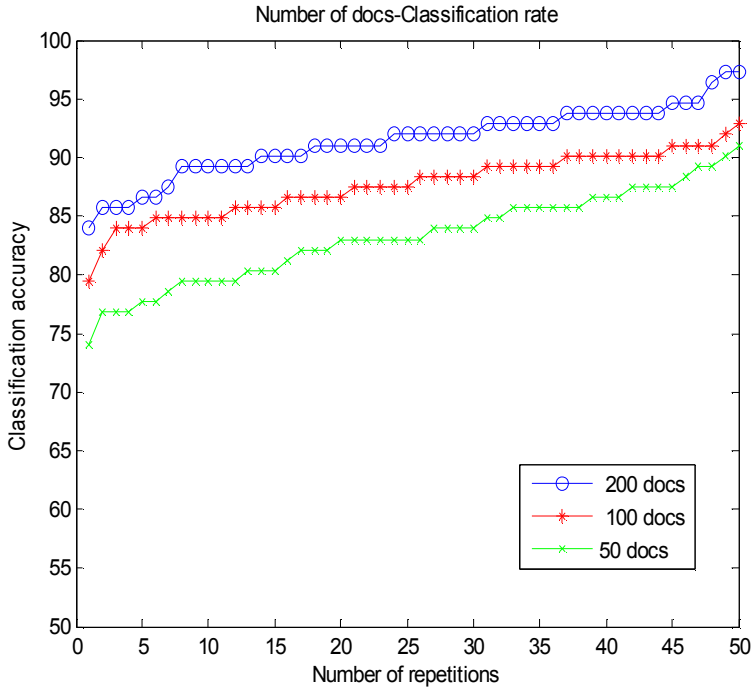

FIGURE5. 1: **THE CLASSIFICATION RATE VS NUMBER OF DOCUMENTS**

2) The feature selection improves significantly the classification accuracy. It is useful for reducing computational complexity and can improve both performance and accuracy. It appears that approximately 1200-1400 terms are not only sufficient to characterize a text but also improves the classification results. The problem of document classification is a high dimensional problem, and selection is a very efficient way to solve it, because it eliminates noise and redundancy. In few word, we observe that this method minimizes the number of features and keeps them, which maximize the discriminative power of a genre or an artist respectively.Figure 5.2 shows the classification accuracy of SVMs with and without feature selection.

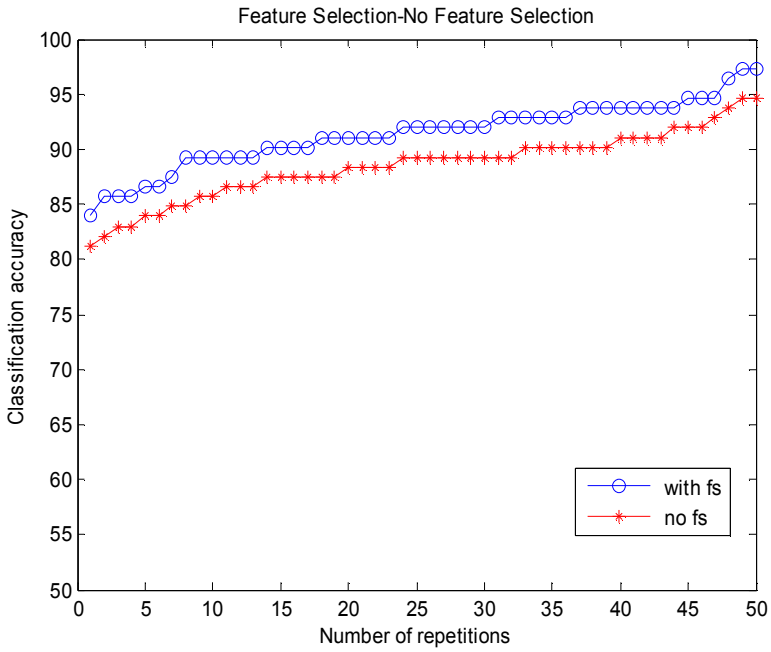

FIGURE5. 2: **CLASSIFICATION RATE VS FEATURE SELECTION**

3) The best classification algorithms for our approach are SVMs and central based algorithm. Exactly, we should say that central based algorithm outperforms SVMs slightly. The more poor results are appeared for the classification method k-NN which uses Euclidean distance for deciding the neighbors of a document. It is very simple algorithm and is sensitive to the noise, especially when the algorithm is applied without feature selection. K-NN using cosine similarity as an evaluation method achieves better overall results.

At this point, we try to explain the behavior of these classification algorithms.

As it referred in [16] SVMs are appropriate for document classification for the following reasons:

i. SVM does not depend on the number of features so it is able to handle high dimensional feature spaces.

ii. Because of efficient handling of very large feature spaces feature selection is optional.

iii. SVM is well suited for problems with sparse instances, like document vectors, which have only a few entries with no zero values.

iv. Most document classification problems are linearly separable, so SVMs is appropriate for these tasks.

Furthermore, we should note that text categorization approaches based on prototypes are computationally more attractive than k-NN algorithm, mainly when working with large set of documents. Rather than comparing each new document to every document that is already categorized, one only has to compare the new documents to the median document of each category. A database of documents would normally contain more than 10000 documents, so this leads to an extremely expensive computation, when using the k-NN.
One of the advantages of the centroid-based scheme is that it summarizes the characteristics of each class, in the form of the centroid vector. The advantage of the summarization performed by the centroid vectors is that it combines multiple prevalent features together, even if these features are not simultaneously present in a single document. For this reason, the centroid-based classification algorithm tends to perform better than the k-nearest neighbor classification algorithm.
Observing the Tables 5.3, 5.4 we conclude that K-NN classifiers rely on the distances between each point and its neighbors, improving the distance function used for selecting the nearest neighbors can significantly improve the classifier's performance. Using cosine similarity of *tf-idf* weighted vectors is more effective than Euclidean distance for text categorization.

In the figure below we observe the classification results for these classification approaches.

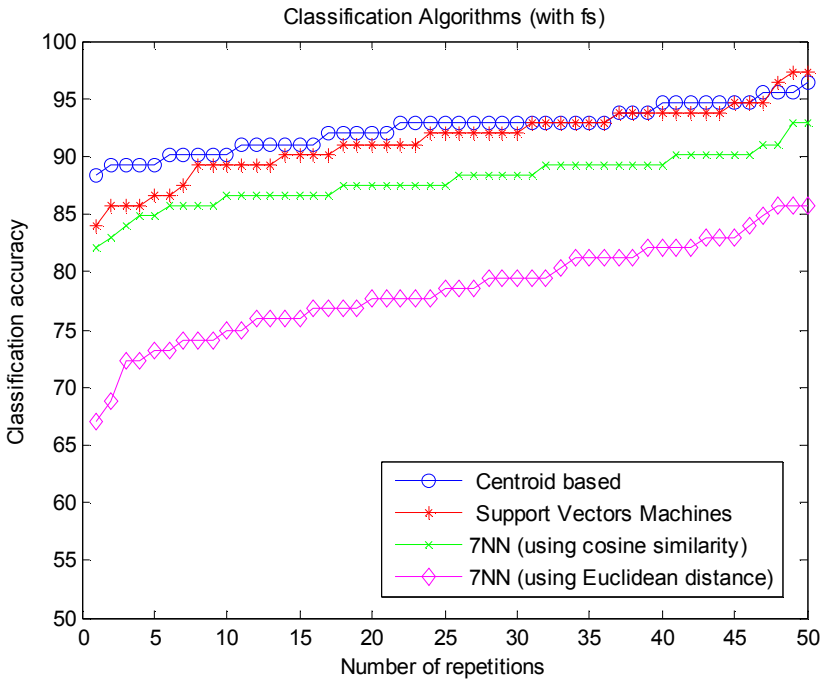

FIGURE5. 3: **CLASSIFICATION RATE VS CLASSIFICATION ALGORITHM**

At this point, we want to study the classification accuracy for each genre. In the Table 5.7 we present the classification results analytically. These scores are for SVM algorithm without feature selection and for 50 documents. We chose to show this case (i.e., with no further processing for improving results) because it depicts clearly how each genre behaves.

Genre	Classification Rate for 50 docs (no fs)
Country	77.25%
Folk	82.5%
Jazz	80.5%
Blues	83.5%
Rnbsoul	62.5%
heavy-metal	70.5%
Altindie	51.5%
Punk	74%
Rap/hip-hop	91.25%
Electro	85.75%
Reggae	84%
Classic	94.75%
Rock'n'Roll	70.25%
Pop	90.25%
Total	**78.46%**

TABLE5. 7: **CLASSIFICATION RATE FOR EACH GENRE**

Observing the classification rates which are appeared in the Table5.7 we end up to the following conclusions:

1) The genres classical, jazz, rap/hip-hop and electro are not easily confused with others genres. These genres are very different from the others and are clearly defined.

2) The remaining genres have lower classification rate which is probably happen by virtue of classification into genres considered to be arbitrary.

The second conclusion leads us to apply methods for achieving better accuracy either by downloading more web pages or by applying feature selection methods. We note for 200 retrieved documents per artist, using feature selection and the centroid based algorithm we score results of up to **92.45%.**

5.4 EXPERIMENTS AND RESULTS FOR FUSION

In this experiment we classify 113 artists in 9 genres, using the second dataset which described in Section 5.2. We point out that we do this experiment for a different dataset with the purpose of having the same dataset with Anthi Markaki. We approximately use the half artists for training data and the remaining for test. The amount of artists is not the same for each genre. So, the number of artists selected for training and testing set is appeared in Table 5.8:

Genre	Training Data	Test Data
Classical	8	7
Dance-Electro	7	6
Hard rock-Heavy metal	3	2
Jazz-Blues	8	7
Latin	7	6
Punk-alternative	8	8
Rap	3	2
Rock-pop-classic	8	7
Rock-pop-alternative	8	8

TABLE5. 8: **TRAINING AND TESTING DATA FOR EACH GENRE**

We downloaded 100 pages for each artist and we followed the same procedure as before to compute tf_idf and chi-values. We classify data using centroid based algorithm.

After we sorted out the songs of artists that are included in testing set, for each artist we took the mean value of distances. The classification results for web-based and audio-based methods are **79.25%** and **56.6%** respectively:

Before we combine the distances which come out from these methods, we normalize the distances based on audio and web data such that they have the same range of values.

Using the relation $dist_{fusion} = w \cdot dist_{audio} + (1-w) \cdot \frac{1}{sim_{web}}$ which is referred to the previous chapter, we decide in which genre an artist belongs to, keeping the minimum distance.

We try to put different weights for the audio based and web based distances. The results are shown in the table below:

Weight for web based distances	Weight for audio based distances	Classification rate
0.1	0.9	64.15%
0.2	0.8	67.92%
0.3	0.7	71.7%
0.4	0.6	73.58%
0.5	0.5	77.36%
0.6	0.4	81.13%
0.7	0.3	81.13%
0.8	0.2	79.25%
0.9	0.1	81.13%

TABLE5. 9: **CLASSIFICATION RATE FOR FUSION OF DISTANCIES**

We take the higher classification rate (**81.13%**) if we use 0.6, 0.7, 0.9 weight for web-based distances and 0.4, 0.3, 0.1 for audio-based distances. Any combination of above distances gives better results from the baseline of audio data (56.6%). We note that web based distances should be weighted higher than audio-based ones, in order to achieve better classification results. We suppose that if the baselines were closer we would be able to observe better if this fusion leads to significant better results. Furthermore, we should note that this dataset is small for extracting general conclusions.

5.5 SUMMARY

In this chapter we describe the experimental procedure, as well as, the results for each method. We classify artists into genres using cosine similarity metric, k-NN and SVMs. As we observe the best results have achieved for 200 retrieved documents, using feature selection and centroid based algorithm, which are up to 92.45%. In Chapter 6, we evaluate overall our results; ending up to valuable conclusions and we also discuss some ideas about some future steps which will probably improve the performance of our system.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this chapter, we discuss the evaluation results of this thesis with respect to the several approaches that were used. This can help us to understand in more details the nature of the experimental data and their exploitation by the applied metrics for genre classification. Also, we compare our results, to the results reported in [18], which was considered by us as a general guideline in order to build the main part of the system presented by this work. In addition, we compare the results of the combination of lexical and audio features with the classification results of the baseline system that uses only lexical or audio features. Next, we outline several directions for future work that take into account more sophisticated techniques for document processing and feature selection in order to achieve higher classification results.

6.1 CONCLUSIONS

The main part of our approach was partially based on the work proposed in [18]. We used the Yahoo! search engine in order to get the URLs of documents about artists. We chose the web as information source because we viewed this source as the most appropriate for obtaining music information covering a wide range of artists and genres, without requiring any cost. The goal was to classify these artists into genres and calculate the classification rate. We downloaded different numbers of the top-ranked documents, returned by the search engine. Then, we extracted the lexical features contained in these documents making the assumption that similar artists share common lexical features. The very general conclusion is about the capability of web for providing a huge amount of textual information about music. In practice every submitted query had a truly vast amount of hits, equal to many hundreds of thousands. Thus, the web was shown to be the ideal information source for obtaining information about music. Recall that we used two different datasets of artists: (i) the same dataset used in [18], and (ii) a dataset compiled by the Speech Group of Information Processing and Computers Networks Laboratory, Technical University of Crete, Greece. For the first dataset we used only lexical features as it was done in [18], while for the second dataset we used a combination of lexical and audio features.

Regarding the first dataset, the artists were classified into 14 different genres. In general we observed that the category of "classic music" had the best classification rates. This seems to be reasonable because this kind of music has lexical features that are appeared almost only in this category, such as "orchestra" that is unlikely to appear in a document about "reggae music". We experimented with three different sets of downloaded documents: 50, 100 and 200 documents. We observed that as the number of documents increases we achieve higher classification rates. The difference in accuracy was more notable especially in the case of 50 and 100 documents. This is an indication that more documents can provide better features and better estimation of their counts, compared to fewer documents. Moreover, we used three different types of methods for artist classification: (i) nearest neighbor rule, (ii) centroid based algorithm, and (iii) support vector machines. It was observed that centroid based algorithm outperforms the other two methods, but support vectors machines slightly. The lowest results were given by k-NN algorithm when used Euclidean distance for computing the neighbors. The poor performance of this method suggests that K-NN classifiers rely on the distance function used for selecting the nearest neighbors. Using cosine similarity the accuracy of the system is improved significantly. In addition we used the chi square method in order to perform better feature selection. The improvement of this feature selection was very satisfactory for every used metric. All in all, the best classification rate is equal to **89.66%** for 200 documents, using centroid algorithm, without any feature selection. This rate was further improved by the use of feature selection and was **92.45**%. The published results in [18] are referred to Support Vector Machines and the achieved correlation rate is reported to be 87% for a set of more than 50 downloaded documents. In our case our best results for Support Vector Machines is 91%, using 200 documents and employing feature selection. Despite that the authors of [18] do not give an exact number of the downloaded documents, we can see that our results are similar to their results.

Regarding the second dataset, the artists were classified into 9 different genres. The main purpose of using this dataset was to investigate if the combination of lexical and audio features can achieve better results compared to classifiers that use only one type of these features. The best classification rate for a classifier that considered only audio features was equal to 56.6%. The best classification rate for a classifier using only lexical features was equal to 79.25%. We can see that the lexical features as individual features are able to obtain better results than the audio features. The combination of lexical and audio features achieves higher classification rate that is equal to 81.13%. This is an important conclusion but we should note that this dataset is small for extracting general statements.

6.2 FUTURE WORK

The nature of the presented work has many issues for further research. First of all, the web by itself provides a huge field for additional considerations. The variety of documents that are available is so large that we can apply several criteria in order to retrieve documents that are more relevant to the submitted queries. This can be done by using more than one query and the combination of multiple queries. Also, it is interesting to evaluate the proposed methods for different web search engines. Moreover, we can study the structure of the downloaded documents in greater details. A web document consists of several parts such as title, body, and also other components are embedded in it, like tables and enumerated lists. Of course each of the above components has a particular role regarding the information that it contains. So, it is important to investigate how the extracted lexical features can be weighted according to their position within the document.

Regarding the downloaded documents we can experiment with more pre-processing methods like stemming, in order to reduce the dimensionality of the feature space. This space can be further improved if we also try to find techniques that are able to distinguish the part of text that contains the information of interest. This step requires the removal of advertisements and other non-relevant pieces of text that very often appear in web documents. After the above text processing stages, we can proceed to the employment of more schemes for feature weighting. These schemes can take into account more sophisticated variations of the tf-idf scheme. In this direction we can also use more methods for feature selection, such as Mutual Information and Informaton Gain. It is important to identify which lexical features characterize efficiently the domain of music, and compare their role with respect to other domains, such as movie domain. This will help us in order to have better understanding of the music domain in order to develop more accurate classifiers.

Clearly, the fusion of lexical and audio features opens a new field of research that is huge, because two distinct research areas are combined. We believe that the close corporation between experts from both fields will lead to systems of higher accuracy. This will improve the classification power of the individual features. For example, a weak audio feature can become more important if it is combined with the appropriate lexical feature. In final, it is interesting to test the proposed methods in real applications and calculate their usefulness by getting evaluation feedback from real users.

APPENDIX

ANALYTICAL PRESENTATION OF RESULTS

In this Appendix, we present the results which achieved for each genre, applying various classification algorithms either using feature selection methods or not.

A.1 RESULTS FOR 7-NN –USING EUCLIDEAN DISTANCE FOR EVALUATION (DATASET 1)

Without feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	70.75%	69.5%	81.25%
folk	10.25%	9.25%	10%
jazz	49.75%	57.25%	48%
blues	97.5%	96.5%	96.75%
rnbsoul	20.25%	18.75%	19.25%
heavy-metal	38%	36.25%	54.5%
altindie	32%	31.5%	12.75%
punk	38.75%	43.5%	61.25%
Rap/hip-hop	67.25%	62.25%	61.75%
electro	52.75%	55%	58.5%
reggae	47.5%	48.75%	62%
classic	50.25%	56.5%	76.25%
Rock'n'Roll	17%	27.75%	31.5%
Pop	50.5%	52.25%	52%
Total	**45.91%**	**47.50%**	**51.84%**

TABLE A.1 : **CLASSIFICATION RESULTS FOR 7-NN WITHOUT FEATURE SELECTION (EVALUATION USING EUCLIDEAN DISTANCE)**

With feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	39.75%	81.25%	94%
folk	51.25%	38.25%	67%
jazz	50.50%	87.75%	93.25%
blues	73%	95.5%	95.5%
rnbsoul	15.5%	53%	75%
heavy-metal	44.25%	73%	86.75%
altindie	45.50%	53.5%	47.25%
punk	18.25%	43.25%	71%
Rap/hip-hop	69.25%	93.25%	94.5%
electro	53%	67.25%	81%
reggae	37.5%	67.75%	84.25%
classic	86.25%	96.75%	100%
Rock'n'Roll	20%	35.75%	35.25%
Pop	40.5%	59.25%	73.25%
Total	**46.04%**	**67.54%**	**78.43%**

TABLE A.2: **CLASSIFICATION RESULTS FOR 7-NN WITH FEATURE SELECTION (EVALUATION USING EUCLIDEAN DISTANCE)**

Observing Tables A.1 and A.2, we can conclude that the algorithm 7-NN has low accuracy using Euclidean distance as evaluation method. The genres which classified well comparing to others are Classic, Rap/Hip-hop, Blues, Jazz and Country. This means that this method classifies well only the genres which are difficult to be confused with others. Furthermore, feature selection improves the results, mainly for the case of 100 and 200 retrieved documents.

A.2 RESULTS FOR 7-NN –USING COSINE SIMILARITY FOR EVALUATION (DATASET 1)

Without feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	76%	83.25%	86.25%
folk	49.5%	42.5%	44.25%
jazz	91.75%	94.25%	88.25%
blues	86.5%	81.5%	89.5%
rnbsoul	49.25%	67%	67.25%
heavy-metal	70.25%	78%	90.5%
altindie	52.25%	79.25%	71.25%
punk	72.75%	69%	87.5%
Rap/hip-hop	93.75%	96.25%	98.25%
electro	64.25%	83.75%	85.5%
reggae	79.75%	87.75%	85.25%
classic	100%	100%	100%
Rock'n'Roll	73.75%	76.5%	72.5%
Pop	92%	94.25%	88.5%
Total	**75.12%**	**80.95%**	**82.48%**

TABLE A.3: **CLASSIFICATION RESULTS FOR 7-NN WITHOUT FEATURE SELECTION (EVALUATION USING COSINE SIMILARITY)**

With feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	84.5%	89%	93.25%
folk	65.25%	71%	62.5%
jazz	94.25%	97.5%	97.25%
blues	97%	94%	94.5%
rnbsoul	65%	74.5%	76.5%
heavy-metal	89.25%	91.25%	93.25%
altindie	64.25%	66.75%	66.25%
punk	86.5%	88.5%	91.5%
Rap/hip-hop	99.75%	99.75%	100%
electro	94.25%	94.25%	90.25%
reggae	93%	92%	93%
classic	100%	100%	100%
Rock'n'Roll	79.5%	83%	81.25%
Pop	79.5%	90%	91.75%
Total	**85.16%**	**87.96%**	**87.95%**

TABLE A.4: **CLASSIFICATION RESULTS FOR 7-NN WITH FEATURE SELECTION (EVALUATION USING COSINE SIMILARITY)**

The tables A.3, A.4 depict that the evaluation method influence intensively the classification results. Comparing to the Tables A.1, A.2 we observe that the classification results have improved significantly. This method is very accurate generally, the only which perform lower classification results are: Folk, Rnbsoul and Altindie. This implies that the concept of genre is not well defined for them.

A.3 RESULTS FOR COSINE SIMILARITY (DATASET 1)

Without feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	81%	90.5%	96%
folk	70.25%	77.25%	84%
jazz	86.5%	91%	92.25%
blues	81.5%	84.25%	87.25%
rnbsoul	64.5%	81%	77.50%
heavy-metal	78.75%	90%	94.25%
altindie	71.75%	81.5%	89.25%
punk	80.75%	87.5%	88.5%
Rap/hip-hop	94.25%	100%	96.5%
electro	88.75%	93.5%	91.25%
reggae	87.25%	92.25%	87.25%
classic	100%	100%	100%
Rock'n'Roll	81.5%	82.5%	83.75%
Pop	92.75%	88.75%	87.5%
Total	**82.82%**	**88.57%**	**89.66%**

TABLE A.5: **CLASSIFICATION RESULTS FOR COSINE SIMILARITY WITHOUT FEATURE SELECTION**

With feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	88.75%	90.25%	92%
folk	83.75%	83.5%	90.25%
jazz	91.5%	93.5%	94.5%
blues	92.75%	92%	94.25%
rnbsoul	73.5%	83.25%	79.75%
heavy-metal	95%	92.5%	93.25%
altindie	67.5%	75.75%	87.75%
punk	89%	90.5%	93.75%
Rap/hip-hop	99%	99.25%	99%
electro	97.25%	97%	92.75%
reggae	93.25%	93%	95%
classic	100%	100%	100%
Rock'n'Roll	80%	86.5%	90.25%
Pop	83.75%	89.5%	91.75%
Total	**88.21%**	**90.46%**	**92.45%**

TABLE A.6: **CLASSIFICATION RESULTS FOR COSINE SIMILARITY WITH FEATURE SELECTION**

The tables A.5, A,6 show the classification rate using cosine similarity metric. These results are the higher that achieved in this thesis. The genres Rnbsoul , Altindie and Folk have satisfactory classification accuracy comparing to the method k-Nearest Neighbors.

A.4 RESULTS FOR SUPPORT VECTOR MACHINES (DATASET 1)

Without feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	77.25%	93.25%	95.75%
folk	82.5%	62.75%	79%
jazz	80.5%	92.5%	93.5%
blues	83.5%	89.5%	88.5%
rnbsoul	62.5%	75.5%	74.25%
heavy-metal	70.5%	90%	91.25%
altindie	51.5%	74.5%	85.5%
punk	74%	92.75%	90.75%
Rap/hip-hop	91.25%	99%	96.25%
electro	85.75%	94.25%	92.75%
reggae	84%	90.75%	87.75%
classic	94.75%	93.5%	100%
Rock'n'Roll	70.25%	78.5%	78%
Pop	90.25%	81.75%	86.5%
Total	**78.46%**	**86.32%**	**88.5%**

TABLE A. 7: **CLASSIFICATION RESULTS FOR SUPPORT VECTOR MACHINES WITHOUT FEATURE SELECTION**

With feature selection

Genre	Classification Rate (50 docs)	Classification Rate (100 docs)	Classification Rate (200 docs)
country	84%	91.25%	93.25%
folk	86.5%	83.5%	92.5%
jazz	83.25%	92.5%	92.75%
blues	85.50%	90.75%	92.75%
rnbsoul	74.5%	84.25%	84.5%
heavy-metal	82.25%	85.5%	93.25%
altindie	57.25%	74.25%	88%
punk	89%	91.75%	91.25%
Rap/hip-hop	95%	95.25%	95%
electro	91.5%	91.5%	92%
reggae	91.5%	88.25%	86.5%
classic	92.25%	97.75%	100%
Rock'n'Roll	70%	76.75%	88%
Pop	83.25%	84.25%	90%
Total	**83.27%**	**87.68%**	**91.41%**

TABLE A.8: **CLASSIFICATION RESULTS FOR SUPPORT VECTOR MACHINES WITH FEATURE SELECTION**

The Tables A.7, and A.8 show the classification rate using SVMs. It is a common method which is used for text classification providing good results. We observe that the genres Rnbsoul , Altindie and Folk classifies with higher accuracy with SVMs.

A.5 RESULTS FOR FUSION (DATASET 2)

Classification rate for each genre using audio features (using k-means algorithm)

Genre	# artists classified	Classification rate
Classical	7/7	100%
Dance-Electro	5/6	83.3%
Hard rock-Heavy metal	1/2	50%
Jazz-Blues	4/7	57%
Latin	3/6	50%
Punk-alternative	4/8	50%
Rap	1/2	50%
Rock-pop-classic	2/7	28.5%
Rock-pop-alternative	3/8	37.5%
Total	**30/53**	**56.6%**

TABLE A.9: **CLASSIFICATION RATE USING AUDIO FEATURES**

Classification rate for each genre using audio features (using cosine similarity with feature selection)

Genre	# artists classified	Classification rate
Classical	7/7	100%
Dance-Electro	6/6	100%
Hard rock-Heavy metal	0/2	0%
Jazz-Blues	5/7	71%
Latin	6/6	100%
Punk-alternative	5/8	62.5%
Rap	1/2	50%
Rock-pop-classic	4/7	57.14%
Rock-pop-alternative	7/8	87.5%
Total	**42/53**	**79.25%**

TABLE A.10 : **CLASSIFICATION RATE USING LEXICAL FEATURES**

We observe from Tables A.10, A.11 that for the same set of artists the method based on lexical features have better classification results than the one based on audio features.

REFERENCES

[1] Manning , C., Raghavan, P. and Schutze, H., "Introduction to information retrieval",2008 (freely available in http://nlp.stanford.edu/IR-book/html/htmledition/irbook.html)

[2] Spärck Jone,K., "A statistical interpretation of term specificity and its application in retrieval", *Journal of documentation* 28 no.1,pp 11-21,1972

[3] Robertson, S. "Understanding inverse document frequency: on theoretical arguments for idf", *Journal of documentation* 60 no. 5, pp 503–520,2004

[4] Lan, M., Tan, C., and Low, H., "Proposing a new term weighting scheme for text categorization", in *AAAI*, Boston, pp.763-768, 2006

[5]Aizawa, A. , ''An information-theoretic perspective of tf-idf measures", *InformationProcessing and management*, vol. 39, pp. 45-65,2003

[6] Hiemstra, D., "A probabilistic justification for using tf.idf term weighting in information retrieval", *International journal on digital libraries* 3(2), pp.131-139, 2000

[7] Rijsbergen,C.J. 1979, "Information Retrieval" second edition (freely available in http://www.dcs.gla.ac.uk/keith/preface.html)

[8] Strehl. A., "Relationship-based clustering and cluster ensembles for high-dimensional data mining". Phd Thesis, the University of Texas at Austin, May 2002.

[9] Quinlan, I.R, "Induction of decision trees". *Machine learning 1,* pp.81-106,1986

[10] Yang,, Y., Pederson, J., "A comparative study on feature selection in text categorization", *In Proc. Of the 14th International Conference on Machine Learning, pp.412-420,1997*

[11] Liu, T. *et al.* 'An evaluation on feature selection for text clustering', *Proceedings of ICML 2003*, pp. 488-495, 2003

[12] Wilbur, J., and Sirotkin, K., "The automatic identification of stop words.", *Journal of information science:,* 18:45-55,1992

[13] Lagus,K, kaski, S.: "Keyword selection method for characterizing text document maps." In: proc intl conf artificial neural networks (1999)

[14] Kohavi, R., "A study of cross-validation and bootstramp for accuract estimation and model selection." *Proceeding of the fourteen international joint conference on artificial intelligence 2 (12):* 1137-1143, 1995.

[15] Kriegel, H. P., Pryakhin, A., Schubert, M., "Multi-represented k-NN classification for large class sets" *Proc. 10th int. Dasfaa.* 2005

[16] Burges, C. J. C., "Geometry and invariance in kernel based methods." *Advances in kernel methods—support vector learning*, pages 89–116, 1999

[17] Boser, B.E., Guyon, I.M., and Vapnik N.," A training algorithm for optimal margin classifiers". In d. Haussler, editor, 5th annual acm workshop on colt, pp 144-152, 1992.

[18] Knees, P., Pampalk, E., Widmer, G. "Artist Classification with web-based data", *In Proceedings of 5th International Conference on Music Information Retrieval (ISMIR'04),* pages 517–524, Barcelona,Spain, October 2004

[19] Knees,P., Schedl, M., Pohle, T. "A Deeper look into web-based classification of music artist", *Proceedings of the 2nd Workshop on Learning the Semantics of Audio Signals*,2008

[20] Pampalk, E., Flexer, A. , Widmer, G. "Hierarchical Organization amd description of music collections at artist level", In *Proc. of the 9th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'05), Vienna, Austria,* 2005.

[21]Scheld, M., Knees, P., Widmer, G. " A-web based approach to assessing artist similarity using co-occurencies", *In: Proceedings of the Fourth International Workshop on Content-Based Multimedia Indexing,2005*

[22] Geleijnse, G., Korst, J., "Web-based artist categorization", In *Proceedings of ISMIR'06*, pages 266 . 271, 2006.

[23] Scheld, M., Knees, P., Widmer, G. "Discovering and visualizng protypical artists by web-based co-occurrence analysis" ,*ISMIR 2006*

[24] Pachet, F., Westerman, G., Laigre. D "Musical data mining for electronical music distribution"

[25] Whitman, B., Lawrence, C. , "Inferring description s and similarity for music from community metadata", *In: Proceedings of First International Conference of Web Delivering of* Music ,2002

[26] Celma O. Ramirez m., Herrera P. "Foafing the music:a music recommendation system based on RSS feeds and user preferences.", roc. Of the 6th Int. Conference on Music Information Retrieval (ISMIR), pp. 464-457.

[27] Sianey M., White W., "Similarity based on rating data" Proc. of the International Symposium on Music Information Retrieval, Vienna, Austria, 2007.

[28] Tzanetakis G., Cook P., " Music genre classification of audio signals", Proc. of the International Symposium on Music InformationRetrieval, Vienna, Austria, 2007
[29]Deshpande H., Singth R., Nam U., "Classification of music signals in the visual domain", *In Proceedings of the COST-G6 Conference on Digital Audio Effects*, 2001.
[30] West k., Cox S., "Features and classifiers for the automatic classification of musical audio signals", in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR'04)*, Barcelona, Spain, October 2004.
[31] Whitman B., Smaragdakis P. , " Combining musical and cultural features for intelligent style detection", *Proc. of the 3rd Int. Conf. on Music Information Retrieval*, 2002
[32] Knees P. Pohle T. Schedl M. "A musical search engine built upon audio based and web based similarity measures", In *ACM SIGIR*, 2007
[33] Whitman B., Ellis D., "Automatic record reviews", In: Proc Intl Conf Music. Information Retrieval (2004)
[34] Markaki A. "Music genre Classification using Spectral and Temporal features" Graduate Thesis 2008
[35] Han, E.H., Karypis, G.: "Centroid-Based Document Classification: Analysis and Experimental Results",*Proc. 4th PKDD'00*, Lyon, France, 2000, 424–431
[36] Shankar S. and Karypis, G. , " Weight Adjustment Schemes for a Centroid Based Classifier Text Mining Workshop", *Proc Knowledge Discovery and Data Mining (KDD '00)*, 2000.