İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

A HYBRID MUSIC RECOMMENDATION SYSTEM BASED ON DIFFERENT FEATURES OF THE MUSIC AND USERS

MSc. Thesis by

Berna ALTINEL,B.Sc.

Department : Computer Engineering

Programme: Computer Engineering

JUNE 2007

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

A HYBRID MUSIC RECOMMENDATION SYSTEM BASED ON DIFFERENT FEATURES OF THE MUSIC AND USERS

MSc. Thesis by

Berna ALTINEL

(504041538)

Date of submission : **31.07.2007**

Date of defence examination: **15.06.2007**

Supervisor (Chairman): **Assoc. Prof. Dr. Zehra ÇATALTEPE**

Members of the Examining Committee **Assoc. Prof. Dr. Şule Gündüz ÖĞÜDÜCÜ**

Assoc. Prof. Dr. Engin ERZİN (Koç Univ.)

JUNE 2007

<u>İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ</u>

MÜZİĞİN VE KULLANICILARIN FARKLI NİTELİKLERİNE GÖRE MELEZ MÜZİK TAVSİYE SİSTEMİ

Yüksek Lisans TEZİ

Müh. Berna ALTINEL

(504041538)

Teslim Tarihi : 31.07.2007

Savunma Tarihi: 15.06.2007

Danışman: Yrd. Doç. Dr. Zehra ÇATALTEPE

Juri Üyeleri Yrd. Doç. Dr. Şule Gündüz ÖĞÜDÜCÜ

Yrd. Doç. Dr. Engin ERZİN (Koç Üniv.)

HAZİRAN 2007

FOREWORD

First, I would like to thank my advisor Assistant Prof. Zehra Çataltepe for her support and guidance during my master's program. She introduced new areas of research and helped me to understand the approach to solve the research problems. Next, I would like to thank my thesis committee members for their valuable comments on my thesis.

Argela Technologies made the data used in this thesis available. George Tzanetakis's Marsyas and George Karypis's Cluto programs were used in the experiments performed. Sule Gunduz-Oguducu of Istanbul Technical University contributed with her ideas in Chapter 4, learning of the cluster contributions to recommendation part. I gratefully acknowledge these contributions.

Finally, I would like to thank my family for their great patience, support and help.

June 2007 Berna Altınel

CONTENTS

ABBREVIATIONS

WAV : Waveform audio format

CLUTO : Clustering Tool

MARSYAS : Music Analysis Retrieval and Synthesis for Audio Signals

CB : Content Based

CF : Collaborative Filtering

STA : Statistical Approach

MIR : Music Information Retrieval

MATLAB : Matrix Laboratory

GUI : Graphical User Interface

PCM : Pulse Code Modulation

MRS : Music Recommendation System

RIFF : Resource Interchange File Format

AIFF : Audio Interchange File Format

IFF : Interchange File Format

TABLE LIST

FIGURE LIST

Page No:

SYMBOL LIST

K-NN	: K-Nearest Neighbor
t	: Time
N_c	: Number of recommendations from cluster similarity metric
Ns	: Number of recommendations from singer similarity metric
Np	: Number of recommendations from popularity metric
d	: Distance
S	: Shannon Entropy
p	: Probability

MÜZİĞİN VE KULLANICILARIN FARKLI NİTELİKLERİNE GÖRE MELEZ MÜZİK TAVSİYE SİSTEMİ

ÖZET

Günümüzde müzik insanların hayatının önemli bir parçası haline gelmiştir. Müzik çalarlar giderek yaygınlaşmaktadır ve müzik tabanlı uygulamalar içeren birçok cihaz vardır. Cep telefonu bu cihazlardan birisidir. Arayan kişiye ulaşılıncaya kadar zil sesi dinlemek yerine seçilmiş bir şarkıyı dinlemek, çağrı anında telefonun zil sesi yerine müzik parçaları ile çalması, her geçen gün daha fazla kişi tarafından tercih edilen uygulamalardan sadece ikisidir. Müziğin bu kadar yaygın olduğu bir ortamda müzik tercihleri de önem kazanmaktadır. Günümüzde müzik tavsiye sistemleri kişilerin geçmiş tercihlerine bakarak ve onlara ait başka bilgileri kullanarak müzik tavsiyesinde bulunabilecek metodlar üzerinde çalışmaktadırlar. Gerek ticari, gerek akademik anlamda kullanılan birçok müzik tavsiye sistemine İnternet üzerinden de ulaşılabilmektedir.

Bu tezde, Zil-Dönüş-Tonu Sistemi ile ya da kişilerin bir miktar şarkı içinden çeşitli şarkılar seçtikleri herhangi bir system ile birlikte çalışabilecek bir müzik tavsiye sistemi üzerinde çalıştık. Bu sistem müzik parçalarını tempo, tını gibi temel özelliklerle temsil eder ve onları bu gösterimdeki uzaklık metriğine gore gruplar. Bir kullanıcıya geçmişte dinlediği şarkılara bakarak bundan sonra dinlemek isteyebileceği şarkıları tavsiye etmeye çalışır. Bunu yaparken, benzer zaman dilimleri içerisinde başka insanların dinledikleri şarkıları dikkate alır. Müzik parçaları arasındaki benzerliğe de parçaların benzerliği ve onların yorumcularının benzerliğine göre karar verir. Bunları dikkate alarak kullanıcıları geçmişteki seçimlerinin benzerliğine göre gruplar. Son olarak bu şarkı ve kullanıcı demetlerini kullanarak kişiye seçmesi muhtemel olan müzik parçalarını tavsiye etmeye çalışır. Bu çalışmada müzik parçalarını tavsiye etmek için 6 adet değişik metod kullanılmıştır.

a) İlk önce, kullanıcıların dinledikleri müzik parçaları arasındaki uzaklıklar hesaplanır. Sonra dinlenilen müzik parçalarına en küçük ortalam uzaklıkta olan müzik parçaları tavsiye edilir. (Euclid/Cosine Distance Based Music recommendation)

b) Bir kullanıcının dinlediği müzik parçalarının özellikleri, entropi ve popülarite kullanılarak müzik parçaları tavsiye edilir. (Content Based Recommendation Using Entropy and Popularity Metrics)

c) Sistemdeki bütün müzik parçaları yakın zaman diliminde dinlenilenler ve uzak zaman diliminde dinlenilenler diye 2 önemli gruba ayrılırlar ve bu gruplardan belli sayılarda şarkı seçilerek müzik parçaları tavsiye edilir. (STA)

d) Sistemdeki bütün müzik parçaları değişik niteliklerine (tını, tempo, perdesel özellikler) göre demetlenir. Her kullanıcının değişik niteliklere verdiği önem, kullanıcının daha önceden dinlediği parçalara göre belirlenir ve her niteliğe ait öbekten farklı sayıda müzik parçası tavsiye eden bir yöntem uygulanır. (Simple Adaptive Method, Adaptive Recommendation Method)

e) Kullanıcılar benzer tercihlerde bulunan diğer kullanıcılarla demetlenir ve bu duruma göre popülarite, entropi gibi metrikler de kullanılarak müzik parçası tavsiye edilir. (Learning Approach on an Adaptive Music Recommendation System with Popularity Data and Using User Grouping)

Bütün bu yöntemleri destekleyerek çalışan müzik tavsiye sistemine bir kullanıcı arayüzü de yazılmıştır. Bu çalışmanın testlerinde bir cep telefonu operatörü için çeşitli müzik içerikli uygulamalar üreten bir firmanın veri kümesi kullanılmıştır. Aynı veri kümesi üzerinde geliştirilen farklı algoritmalar denenmiş ve performansları kıyaslanmıştır. Yapılan test sonuçlarına göre, sadece müzik parçalarının benzerliğinin kullanılması ile %2-5 oranında başarılı öneriler yapılabiliyor iken, kullanıcının önem verdiği müzik özellikleri değerlendirilerek %5-%10, popülarite ve benzer müzik zevki olan kullanıcıların hesaba katılması ile %75 başarı oranı ile öneride bulunma imkanı vardır.

A HYBRID MUSIC RECOMMENDATION SYSTEM BASED ON DIFFERENT FEATURES OF THE MUSIC AND USERS

SUMMARY

Today, music has become an important part of the people's lives. Music players are widely used and there are many tools with music content integrated in some of their applications. Cellular phone is one such tool. When calling someone, hearing the Colored-Ring–Back–Tone which is a selected song, instead of the Ring-Back-Tone or hearing a song when the phone rings instead of the classical ring tone are just two of the applications which are chosen by more and more people. When music is widely used, music choices become quite important. Music recommendation systems study methods of recommending music to users based on their past music selections and other information about the users. There is academic and commercial music recommendation system available on the internet.

In this thesis, we study a music recommendation system that can be used within the Ring-Back-Tone system or any system where a user chooses some songs among a number of choices. Our system represents musical pieces with basic audio features such as beat and timbre and groups them according to a distance metric in this representation. By observing the past choices of a user, it tries to recommend songs that could be chosen by that user. While doing this, it takes into account the songs listened by other users in similar time periods. It uses the similarity among music pieces and their singers to decide on the similarity between music pieces. By using these similarities, it produces groups (clusters) of people who made similar choices in the past. Finally, by using song and user clusters, it tries to recommend audio files that are likely to be selected by a user.

We study 6 different methods to recommend music pieces:

a) First, distances between music pieces listened by users are calculated. Then the music pieces whose average distance to the songs already listened by the user are recommended. (Euclid/Cosine Distance Based Music recommendation)

b) Musical pieces are recommended by using the features of the music pieces listened by the users, entropy and popularity. (Content Based Recommendation Using Entropy and Popularity Metrics)

c) All the music pieces in the system are divided into two important groups; the ones are listened in the short period and the ones listened in the long term period. Musical pieces are recommended by selecting a specified number of music pieces from these two groups. (STA)

d) All the music pieces in the system are clustered based on different features (timbre, beat, and pitch). The importance of the features is specified based on the musical pieces

listened by the users in the past, and different number of music pieces from each cluster of each feature are recommended. (Simple Adaptive Method, Adaptive Recommendation Method)

e) Users are clustered with the other users who have similar preferences and musical pieces are recommended via using some metrics such as popularity, entropy. (Learning Approach on an Adaptive Music recommendation System with Popularity Data and Using User Grouping)

A graphical user interface is created for the music recommendation system which supports all the above mentioned methods. In this study, a user session dataset provided by a company that produces musical content applications for a cellular phone company is used. Different algorithms are used with this dataset, and their performances are compared. According to test results; while using only the similarity of music pieces it is possible to recommend with %2-5 success rate, by using the features important to a particular user, it is possible to recommend with %5-10 success rate. By using popularity and user clustering the recommendation success ratio increases to %75.

1 INTRODUCTION

Widespread use of mp3 players, cell-phones and availability of music on these devices according to user demands increased the need for more accurate music information retrieval (MIR) systems. Music recommendation is one of the subtasks of MIR systems and it involves finding music that suits a personal taste [1]. Audioscrobbler[1], iRate[2], MusicStrands[3], and inDiscover[4] are some of the music recommendation systems today [2]. Usually music recommendation systems follow a collaborative filtering or a content-based (CB) approach. Collaborative filtering (CF) is the approach used in Amazon [3], a new item is rated by some users and the item is recommended to other users based on the rating of the previous users [4, 5]. The disadvantages of the collaborative approach is that when a new item arrives, it has to be rated by someone in order to be used for the other users; recommendations tend to be usually by the same artist and may not be so interesting. In the content-based approach, based on some form of distance between the items already rated by the user and a new item, the item is recommended or not [2, 6, 7, 8]. In order to compute similarities between music pieces different approaches have been suggested. In this work, we use extraction of musical features. We are only aware of two studies [9, 10] that combine collaborative and content based methods for music recommendation. In [9] a Bayesian network is used to include both rating and content data for the recommendation and the hybrid approach is shown to produce better recommendations than using collaborative or content-based approach alone. [10] Also use a hybrid approach, where they evaluate CB, CF and STA (Statistical) methods and their combinations. Since we will compare our work to that of [10], we give more details

---

[1] www.audioscrobbler.com

[2] irate.sourceforge.net

[3] www.musicstrands.com

[4] www.indiscover.net

about their work here. In CB approach, first all the songs are clustered, then each cluster is given a weight based on whether a song the user listened before is in the cluster or not. The number of songs recommended from each cluster is chosen proportional to the weight of the cluster. The disadvantage of the CB based approach is the fact that the user is recommended songs only from the clusters s/he has listened to before. In CF approach, not only the clusters which have contributed to the songs the user listened to, but also clusters that contributed to other users are taken into account. Of course there could be clusters which contain songs not listened enough by anybody and those will be ignored. In STA approach, all the songs are divided into two groups, short term and long term. A certain number of songs are selected from the long term list and the remaining ones are selected from the short term list. STA behaves similar to the popularity in recommendation systems. Since [10] found out that CB was the least successful among the methods he experimented with, we concentrated on CF and STA. We implemented the CF approach as described in [10] and for STA, we used the time frame immediately 1, 3, 7, 15, 30 days before the time of the recommendation. We think this makes STA take better advantage of popular songs around the time of the recommendation. Although [10] recommends using 50% from among the popular songs and 50% from among the others, we also experimented with different ratios.

The rest of the thesis is organized as follows. In section 2, we review basic musical terms and existing commercial and non-commercial music recommendation systems and the algorithms and metrics that they use. Ringo, inDisvover.net, CDNow.com are some of these systems. In this section, major algorithms are also mentioned in detail such as content based approach, statistical approach and hybrid method. In Section 3, we introduce the dataset we used and the features we extracted from songs. We also give information on some clustering methods which are used for clustering of both songs and users. In Section 4, we introduce the metrics used in the recommendation systems that we consider in this thesis: Singer similarity, cluster similarity, popularity factor, entropy and user grouping. Also in this section, we introduce the recommendation methods we use: Euclid/Cosine Distance Based Recommendation, Content Based Recommendation Using Entropy and Popularity Metrics, Statistical Approach, Simple Adaptive Method, Adaptive Method, Learning Approach on an Adaptive Music recommendation System with popularity data using user grouping. Related test results are included in Section 4. In Section 5 the implementation environment and the graphical user interface of the music recommendation system is explained. In section 6, conclusion of all these studies and also the future work are included.

2. LITERATURE SURVEY

This section contains basic musical terms, the detailed survey of both commercial and non-commercial music recommendation systems and related algorithms.

2.1 Musical Terms

Rhythm, melody, harmony, timbre, instruments, dynamics, tempo and meter, which are often called the basic elements of music, are the essential aspects of a musical piece. While music theory describes various pieces of music in terms of their similarities and differences in these musical terms, music is also usually grouped into genres based on similarities in all or most elements [20]. The musical term definitions here are mostly gathered from [20], [21], [22], and [23].

Rhythm: The placement of the sounds in time is the rhythm of a music piece. Most rhythm terms concern more familiar types of music with a steady beat.

Melody: Melody of a music piece is the string of notes that sounds most important.

Harmony: Harmony refers to the procedure by which chords of music are constructed and the system by which one chord follows another chord in time. A chord may be defined as a combination of three or more different tones conceived as a related unit and sounding at the same moment in time.

Timbre: is a common synonym for tone Color which should be defined as "the characteristics of an instrument's sound, or a combination of instrumental sounds"". Instruments: The musical instrument used could give an idea on the genre of the music, for example, piano or violin is often used in classical music.

Dynamics: The term for gradations of amplitude (louds and softs) in music is dynamics. Dynamic levels are a natural indicator for emotional mood.

Meter: Meter is counted with Arabic numbers. Count one is known as the downbeat. Two patterns of two-beat meter (duple meter) are counted 1-2 | 1-2 (the "|" mark separates one group of two and the "_" mark represents an accent of loudness or length). Three patterns of three-beat meter (triple meter) are counted 1-2-3 | 1-2-3 | 1-2-3 | 1-2-3.

Four patterns of four-beat meter (quadruple meter) are counted 1-2-3 4 | 1-2-3-4 | 1-2-3-4 | 1-2-3-4. Five patterns of five-beat meter (quintuple meter) are counted 1-2-3-4-5 | 1-2-3-4-5 | 1-2-3-4-5 | 1-2-3-4-5 | 1-2-3-4-5. Patterns may be created in this manner with any number of numbers limited only by practical considerations.

Tempo: Tempo (an Italian word) identifies the rate of speed of the beat of music and is measured by the number of beats per minute. There is a machine known by the term metronome which emits a steady short "click" or flash that may be adjusted to various rates of speed (tempi), thereby indicating at what speed (how fast or slow) a composition should proceed. A beat may be slow or fast. "Romantic" songs tend to have a medium tempo, while dance music may range from slow to fast tempo. March music reflects a comfortable marching pace -- about 120 beats per minute. Faster tempi (plural of tempo) are more energizing while slower tempi are more soothing.

2.2 Music recommendation Systems

2.2.1 Ringo

The following information and sample screen views about Ringo are mostly gathered from [13].

Ringo uses Social Information Filtering to recommend files to people. It is different from content based filtering from aspect of needing to have users to rate music files on which they will be recommended in the future.

After having an account in Ringo (one can join by e-mailing Ringo@media.mit.edu),the system requires the person to fill up a music list by rating each song. After rating these items Ringo gets to know about the person. The more rating is done, the better the system knows and makes better recommendations to this person. Any person can add albums to the Ringo database.

This system is created by Upendra Shardanand and his team at MIT. Originally, RİNGO had only 575 artists in its database. Then it increased to more than 3000 artists and 9000 albums.

Some sample views from Ringo are as follows:

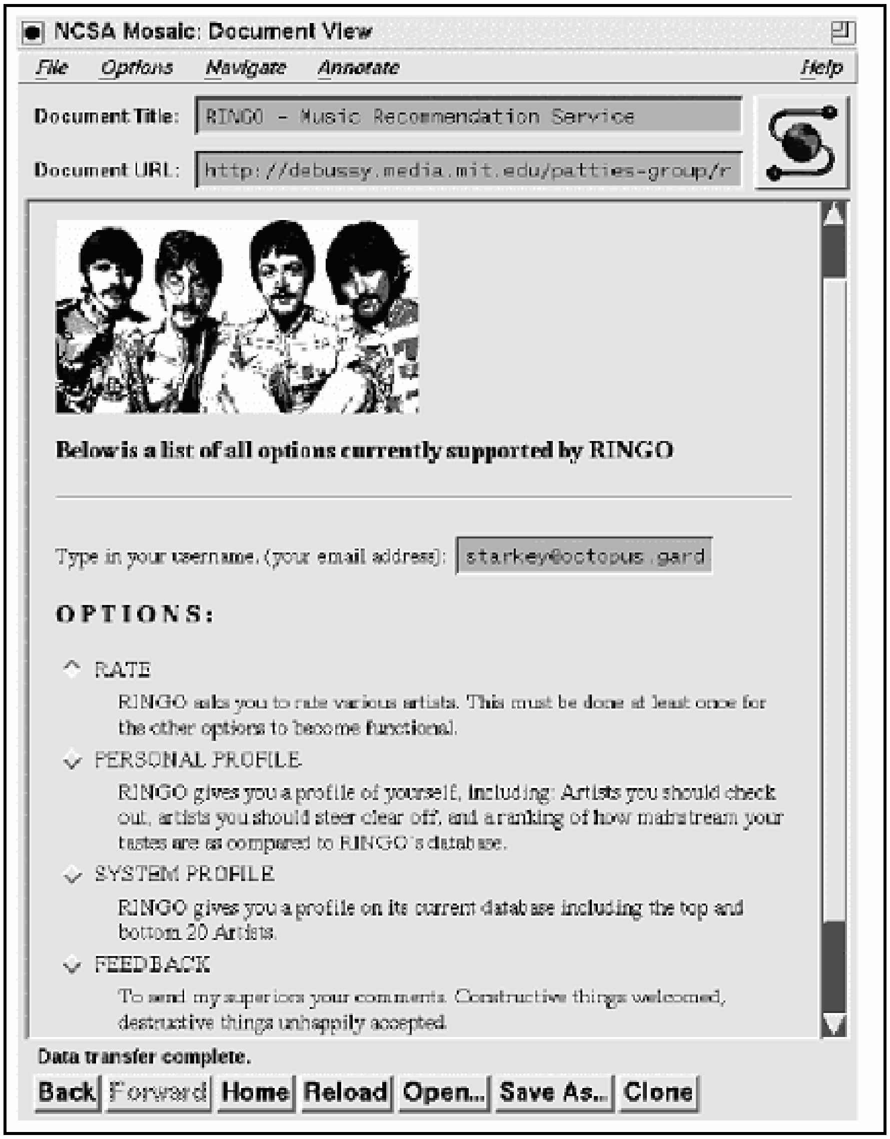


Figure 2.1: A Page From Ringo`s World Wide Web Interface [13]

6	"10,000 Maniacs"
3	"AC/DC"
3	"Abdul, Paula"
2	"Ace of Base"
1	"Adams, Bryan"
	"Aerosmith"
	"Alpha Blondy"
6	"Anderson, Laurie"
5	"Arrested Development"
	"Autechre"
3	"B-52s"
	"Babes in Toyland"
	"Be Bop Deluxe"
5	"Beach Boys, The"
	"Beastie Boys"
4	"Beat Happening"
7	"Beatles, The"
1	"Bee Gees"
	"Bigod 20"
	"Biz Markie"
5	"Bjork"
4	"Blondie"
	"Blues Traveler"
1	"Bolton, Michael"
1	"Bon Jovi, Jon"
	"Bowie, David"
	"Brown, Ray"
6	"Bush, Kate"

Figure 2.2: Part of One Person's Survey [13]

7 : BOOM! One of my FAVORITE few! Can't live without it.
6 : Solid. They are up there.
5 : Good Stuff.
4 : Doesn't turn me on, doesn't bother me.
3 : Eh. Not really my thing.
2 : Barely tolerable.
1 : Pass the earplugs.

Figure 2.3: Ringo`s Scale For Rating Music [13]

Artist	Rating	Confidence
"Orb, The"	6.9	fair
"Negativland"	6.5	high
Reviews for "Negativland" They make you laugh at the fact that nothing is funny any more — user@place.edu		
"New Order"	6.5	fair
Reviews for "New Order" Their albums until 'Brotherhood' were excellent. Since then, they have become a tad too tame and predictable. And sadly, they have been overplayed. — lost@elsewhere.com		
"Sonic Youth"	6.5	fair
Reviews for "Sonic Youth" Confusion is Sex: come closer and I'll tell you.		
"Grifters"	6.4	fair
"Dinosaur Jr."	6.4	fair
"Velvet Underground, The"	6.3	low
Reviews for "Velvet Underground, The" The most amazing band ever.		
"Mudhoney"	6.3	fair

Figure 2.4: One of Ringo's Suggestions [13]

2.2.2 CDNOW.com

The following information and sample screen views of CDNow.com system are mostly gathered from [14].

CDNOW.com is one of the music recommendation systems created by Amazon and it gives recommendations based on users' previous ratings.

When a new user, becomes a member of this site, the system requires that s/he rates some songs. With these ratings, the system stores every shopping record in its database. The system also has shopping records of other people. Using all of these data, the system gives some recommendations from the 'new release' or the 'coming soon' or the current files. If the user wants, s/he has the opportunity to improve his/her recommendations by rating more and more items.

Some sample screen views from this system are as follows:

Figure 2.5: A Rating Page from CDNow.com [14]

The rating categories are as follows:

- Not rated
- I hate it
- I do not like it
- It is OK
- I like it
- I love it

After these ratings the user can get his/her recommendations. Again and again s/he has the opportunity to improve his/her recommendations.

2.2.3 InDiscover

The following information and sample screen views about inDiscover system are mostly taken from [15].

InDiscover aims to provide high quality context-sensitive sets of recommendations based on explicit rating-based Collaborative filtering. InDiscover is database-driven and leverages techniques from multidimensional databases (OLAP).

InDiscover uses Collaborative filtering techniques and a rule engine to generate a list of recommended songs in the form of a play list. By taking into account the way a user has rated other songs, and how others have rated songs, inDiscover is able to predict how much the user would like songs the user has not rated. By applying rules to these predictions, the system outputs a list of recommendations that it thinks the user will like.
The following scenario is described for the new user:

- Once s/he registers, s/he will be able have songs recommended to her/him based on her/his mood, location, and basic tastes in music.
- By rating songs in the multiple categories, the system will be able to determine what user likes and recommend him/her songs and compose them into a play list which the user can download.
- The more songs the user rates, the better the system will be able to determine his/her tastes and recommendations will become more accurate.

Some sample screen views are as follows:

Figure 2.6: A Rating Page from inDiscover's system [15]

Figure 2.7: Some Sample Recommendations from the System [15]

2.3 Algorithms Used For Recommendation Systems

2.3.1 Content Based Method

Based on content based filtering approach, the purpose of the CB method is to recommend the music objects that belong to the music groups the user is recently interested in. Here, the music group candidates for future recommendation are based only on the history of that user. The CB method is applied in [10] as follows: The whole history is kept in a database. This information consists of which user chooses which audio file and when. In order to decide on recommendations for the user, that user's past audio groups are extracted. For instance:

Audio file -1: music group -2,

Audio file -2: music group -5

…

In order to compute the weight of a music group, the number of audio files listened in that group divided by the total number of audio files is used. The following formula (2.1) taken from [10] which calculates weight values of music groups:

$$GW_i = \sum_{j=1}^{n} TW_j * MO_{ji} \quad \textbf{(2.1)}$$

where TWj is the weight of transaction Tj

n is the number of latest transactions used for analysis

MOji is the number of music objects that belong to music group Gi in transaction Tj

Just multiplying the calculated GWi value by the number audio files to be recommended, result shows that the number of recommendations from that group. In other words:

$$R_i = \left\lfloor N * \frac{GW_i}{\sum_{k=1}^{M} GW_k} \right\rceil \quad \textbf{(2.2)}$$

where N is the number of music objects in the recommendation list

GWi is the weight of the target group

M is the total number of music groups in MRS

The recommendation system [16] also uses Content Based method, but only partially.

2.3.2 Collaborative Filtering

(CF) is the method of making predictions about the interests of a user. While doing this it uses two kind of information:

a) The information about that user,

b) The information about the other users.

CF method claims that users who have similar past choices will probably have similar future choices,too. For this reason,the systems with CF method stores both two above mentioned information. By using them, it tries to build an artificial logic in order to decide the future behavior of a user who will have predictions about his/her choices..

To get information about people on their tastes can be done in many ways:The easist one is just to trace the people and store their choices. Another way can be just send them a simple rating list and ask them to rate those items. By looking at those ratings , an artificial logic behind the system can produce some predictions about the future behaviour.

There are commercial sites that implement collaborative filtering systems. For example:

- Amazon [3]
- Barnes and Noble [1]
- Findory.com [2]
- half.ebay.com [3]
- Hollywood Video [4]
- Last.fm – music [5]
- Loomia - web service [6]
- Musicmatch [7]

---

[1] http://www.barnesandnoble.com/

[2] http://findory.com/

[3] http://www.half.ebay.com/

[4] www.hollywoodvideo.com

[5] http://www.last.fm/

[6] http://loomia.com/

[7] http://www.musicmatch.com/

- Netflix [1]
- StoryCode - books [2]

2.3.3 STA

STA is one of the methods used in [10]. In [10] two different hot music groups are defined: ***the long-term hot music group:*** the music group containing the most music objects in the access histories of all users; the ***short-term hot music group:*** the music group containing the most music objects in the latest five transactions in the access histories of all users. These lists are, in some sense popular song lists that show what audio files are listened by others in which frequency.

2.3.4 Hybrid Recommendation Systems

Hybrid recommendation systems use a combination of the three mentioned recommendation methods. For example in [9] rating and cluster similarity are used. [9] also uses Content based and Collaborative filtering as recommendation algorithms.

[1] http://www.netflix.com/

[2] http://www.storycode.com/

3 MUSIC RECOMMENDATION DATA AND CLUSTERING

3.1 Dataset

3.1.1 Dataset Overview

The dataset we use in this study is obtained from Argela Technologies [24]. It is a real dataset obtained using Colored-ring-back-tone (CRBT) product of this company. The CRBT is a service which makes it possible to listen to the music before connecting to the other party [25]. The dataset contains which users requested which songs for their CRBT services. There is really no user rating in the dataset. If a user selects a song, we assume that s/he rates that song favorably.

The dataset consists of music categories shown in Table 3.1.

Table 3.1: Category List in the Dataset

Category Id	Category Name
4	Popular songs
5	Unforgettable
7	Requested
104	Foreign
105	Fantasy/Arabesque
106	Sports Team March
108	Series – Movie
109	Turkish Art Music/Turkish Folk Music
110	Classical Music
111	March

222	Turkish Pop
223	Rock / Rap
230	Free
330	Tarkan
337	Fun
373	Name Specialized
412	Love Songs

Under these categories related singers and their songs are available.

Number of distinct records in the dataset is 1 356 456, which means in about 2 years, this system is used 1 356 456 times. The number of distinct users who used this system is 760 345.

The dataset contains answers to the following questions:

- What are the categories?
- What are the songs below these categories?
- Which songs are bought by a specific user? When this user bought these songs?
- How many numbers of melodies bought?
- How many numbers of melodies bought today?
- How many numbers of melodies bought last week, per day?
- What are the top 10 melodies bought?
- What are the top 10 melodies bought today?
- What are the top 10 melodies bought yesterday?
- What are the top 10 melodies bought last week?
- Who are the active users?
- Who are the inactive users?

3.1.2 Feature Extraction

We obtain features of each of the songs listened by the users at this step. Later we use the distances/similarities between these features to produce groups or user groups.

3.1.2.1 Dataset Format Conversion

In this part, these files, in MP3 format, are converted into WAV format by [26]. This conversion is done since wav is a format which stores uncompressed digital sound while MP3 stores compressed sound.

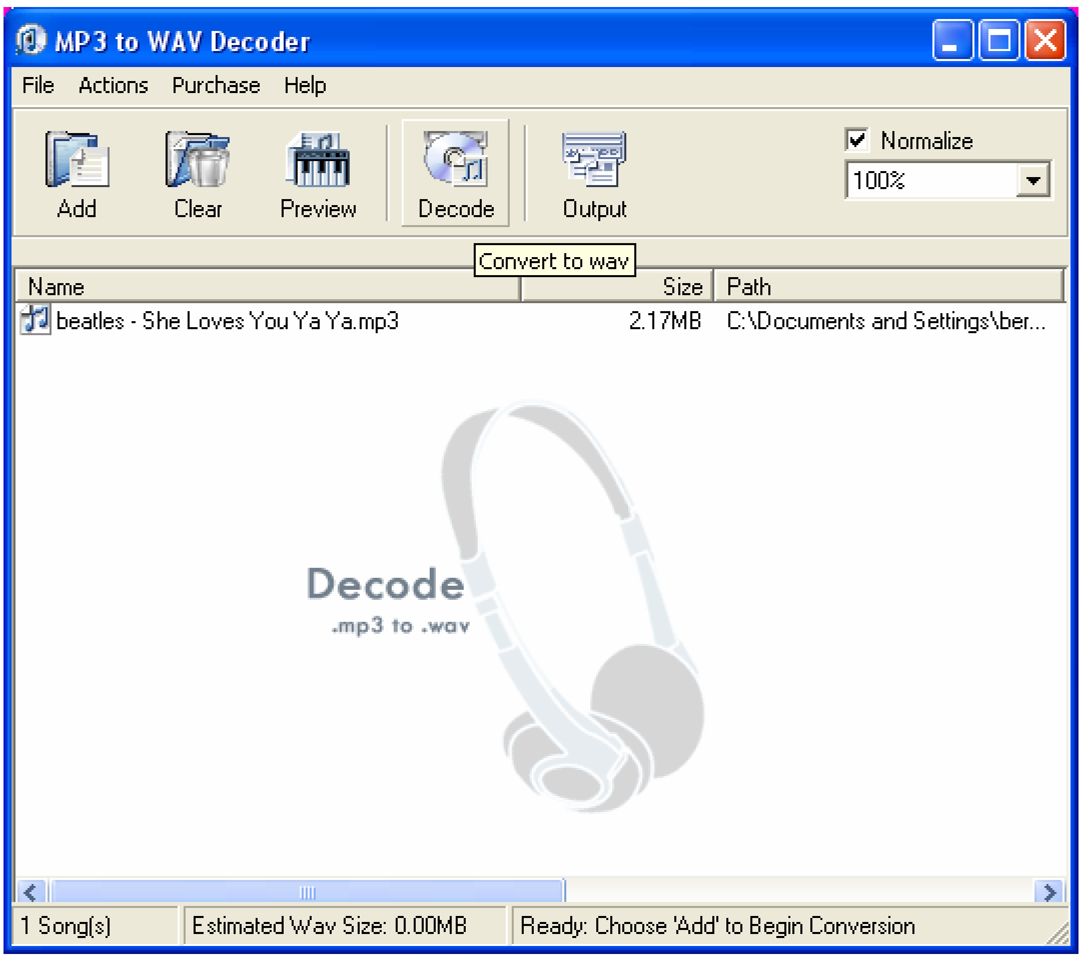


Figure 3.1: Figure of the User Interface of MP3-Wav Decoder [26]

3.1.2.2 Marsyas Feature Extraction:

By using Marsyas (Music Analysis Retrieval and Synthesis for Audio Signals) program [30] which is written and made freely available be George Tzanetakis, the audio features of the files in WAV format are easily extracted. The following command is used for feature extraction:

./extract GENRE [fileName1] [fileName2]

fileName1: The name of the file which contains the list of the audio files whose features will be extracted

fileName2: The name of the file that will contain the extracted features for all the audio files

Table 3.2 shows features of a sample file:

Table 3.2: Feature List of an Audio File

File Name	AliyeDiziMuzigi.wav
Feature-1(Beat)	0.0434309
Feature-2(Beat)	0.0352177
Feature-3(Beat)	0.81089
Feature-4(Beat)	224
Feature-5(Beat)	42
Feature-6(Beat)	52.7025
Feature-7(Stft)	42.2281
Feature-8(Stft)	48.4173
Feature-9(Stft)	289.867
Feature-10(Stft)	26.1476
Feature-11(Stft)	41.5248
Feature-12(Stft)	205.9

Feature-13(Stft)	64.4612
Feature-14(Stft)	0.0131437
Feature-15(Stft)	-42.7093
Feature-16(Mfcc)	5.40386
Feature-17(Mfcc)	-1.27607
Feature-18(Mfcc)	1.41948
Feature-19(Mfcc)	-0.690552
Feature-20(Mfcc)	5.65494
Feature-21(Mfcc)	0.493887
Feature-22(Mfcc)	0.315312
Feature-23(Mfcc)	0.235454
Feature-24(Mfcc)	0.140227
Feature-25(Mfcc)	131.932
Feature-26(Mpitch)	49
Feature-27(Mpitch)	4
Feature-28(Mpitch)	87288
Feature-29(Mpitch)	7
Feature-30(Mpitch)	-1

In Table 3.2,

- the first 6 ones the BEAT features,
- the next 9 ones STFT features,
- the next 10 ones MFCC features,
- the next 5 ones MPITCH FEATURES.

A total of 30 features are extracted from each file.

Before the features are used in subsequent steps, they are normalized using z-score normalization, i.e. from each feature the sample mean for that feature is subtracted and the result is divided by the sample standard deviation for the feature.

3.1.2.3 Last Form of Dataset User Profile File

After the feature extraction, music pieces can now be used in the music recommendation system. By matching the file names and the features the user profile files are prepared. A user profile file contains the following:

- User id,
- Audio file name ,
- Start date of the usage of that file(Number of days since1/1/1970) ,
- End date of the usage of that file(Number of days since 1/1/1970) ,
- Time elapsed(# OF DAYS),
- Extracted features[1-30]

Contents of an example user-profile file are shown below:

USER ID : 905054101180,

FILE NAME : Tarkan-Shhh,

START DATE (# OF DAYS SINCE 1/1/1970): 13066,

END DATE (# OF DAYS SINCE 1/1/1970) : 13248,

TIME ELAPSED (# OF DAYS) : 182,

FEATURES [1-30] : 0.0581167, 0.0426348, 0.733607, 50, 145, 148.305, 77.3342, 195.177, 247.831, 90.8758, 144.773, 1583.94, 22390.9, 1437.48, 0.0697602, -43.2702, 4.73296, -1.07003, 1.83471, 0.0712359, 5.96749, 0.482293, 0.477396, 0.213573, 0.14816, 255.645, 20, 11.6125, 11, 3

Each user listens to a certain number of songs during the dataset collection timeframe. We thought that the length of a session known for a user could make a difference on the recommendation success on the next song, the more songs a user has listened to, the more we know about him and hence can make a good recommendation for him. For this reason, we grouped the users according to the number of songs that they have listened to. This resulted in the following user profile files:

- User-profile file-3 (the users who listen 3 music files throughout the test period)
- User-profile file-4 (the users who listen 4 music files throughout the test period)
- User-profile file-5 (the users who listen 5 music files throughout the test period)

….

- User-profile file-135 (the users who listen 135 music files throughout the test period)

The following user profile files are also prepared:

- User-profile file-more_than_3 (the users who listen at least 3 music files throughout the test period)
- User-profile file-more_than_4 (the users who listen at least 4 music files throughout the test period)
- User-profile file-more_than_5 (the users who listen at least 5 music files throughout the test period)

…

- User-profile file-more_than_135 (the users who listen at least 135 music files throughout the test period)

3.2 Clustering and Related Algorithms

The following information about clustering and related algorithms is mostly gathered from [29].

3.2.1 Clustering

The simplest definition of clustering could be "making groups of objects based on what they have in common from the aspect of a specific point".

3.2.2 CLUTO Clustering Software

In order to perform grouping of songs and users we used the freely available Cluto software by George Karypis [12]. The CLUTO software is distributed as a single file that contains binary distributions for Linux, Sun, OSX, and MS Windows platforms.

Cluto allows a number of clustering methods (input using the –clmethod option). Please see the CLUTO manual for more details:

Rb :(repeated bisections): In this method, the desired k-way clustering solution is computed by performing a sequence of k − 1 repeated bisections.

Rib: In this method the desired k-way clustering solution is computed in a fashion similar to the repeated-bisecting method but at the end, the overall solution is globally optimized.

Direct: In this method, the desired k-way clustering solution is computed by simultaneously finding all k clusters.

Agglo: In this method, the desired k-way clustering solution is computed using the agglomerative paradigm whose goal is to locally optimize (minimize or maximize) a particular clustering criterion function (which is selected using the -crfun parameter).

Graph: In this method, the desired k-way clustering solution is computed by first modeling the objects using a nearest-neighbor graph (each object becomes a vertex, and each object is connected to its most similar other objects), and then splitting the graph into k-clusters using a min-cut graph partitioning algorithm.

Bagglo: In this method, the desired k-way clustering solution is computed in a fashion similar to the agglo method; however, the agglomeration process is biased by a partitional clustering solution that is initially computed on the dataset.

Using –sim option, it is possible to use different similarity measures between the points to be clustered. There are three different readily available similarity metrics:

Cos: (default) The similarity between objects is computed using the cosine function.

Corr: The similarity between objects is computed using the correlation coefficient.

Dist: The similarity between objects is computed to be inversely proportional to the Euclidean distance between the objects.

3.2.3 Clustering Music Pieces in the Dataset

The following CLUTO commands are used to cluster the music files:

```
vcluster.exe
        -clmethod=graph
        -sim=corr
        -clustfile=clustersGraphCorr_mpitch_stft_beat_10.txt
        features_mpitch_stft_beat.txt 10
```

The last number shows the number of clusters. We experimented with 10, 20 and 30 clusters in general.

Different sets of MARSYAS features are used as inputs to the clustering algorithm:

BEAT

STFT

MFCC

MPITCH

BEAT & STFT

BEAT & MFCC

…

BEAT & STFT & MFCC

BEAT & STFT & MPITCH

….

An example clustering output using all features is shown in Table 3.3:

Table 3.3: Example Clustering Output of an Audio File

FileId	**Filename**	**Cluster Id**
1	SadikKaran-BakGidersemDonmem.wav	20
2	AnneSarkilari.AjdaPekkan-AglamaAnne.wav	12
3	AnneSarkilari.BEN_ANNEMI_ISTERIM.wav	18
4	AnneSarkilari.Kibariye-Annem.wav	12
5	AskSarkilari.KenanDogulu-AskimAskim.wav	16
6	AskSarkilari.Kirac-OlurYa.wav	11
7	AskSarkilari.SezenAksu_HERSEYI_YAK.wav	1
8	AskSarkilari.SezenAksu-IkiliDelilik.wav	10

9	AskSarkilari.Tarkan-AyrilikZor.wav	2
10	AskSarkilari.Yalin-Kucucugum.wav	4
11	diziFilm.erkinkoray-hababamsinifi.wav	11
12	diziFilm.KiracAliyeDiziMuzigi-BirGunBeniOzlersenEger.wav	10
13	diziFilm.Kirac-AliyeDiziMuzigi.wav	11
14	diziFilm.Kirac-BirIstanbulmasali.wav	11
15	EnBegenilenler.GeceYolculari-SeninleBirDakika.wav	19
16	EnBegenilenler.handeyener-askinatesi.wav	6
17	EnBegenilenler.ismailYkBombabomba.com.wav	18
18	EnBegenilenler.KenanDogulu-BasHarfiBen.wav	9
19	EnBegenilenler.MFO-Sarilaleler.wav	16
20	EnBegenilenler.Pink-WhoKnew.wav	16
21	FanteziArabesk.Alisan.Alisan-KalbimEllerinde.wav	10
22	FanteziArabesk.Alisan.Alisan-OlayBitmistir.wav	2
23	FanteziArabesk.Alisan.Alisan-YalanOldu.wav	19
24	FanteziArabesk.EbruGundes.EbruGundes-BenSecilmemSecerim.wav	7
25	FanteziArabesk.EbruGundes.EbruGundes-Cingenem.wav	14
26	FanteziArabesk.EbruGundes.EbruGundes-	20

	DonNeOlur.wav	
27	FanteziArabesk.EbruGundes.EbruGundes-HatalarimdanBirisin.wav	10

4 METRICS AND METHODS USED IN THE PROPOSED SYSTEM

4.1 Metrics Used In the Proposed System

4.1.1 Song Clustering

All of the audio files based on all possible feature combinations are given to CLUTO as an input file and all the related output files are gathered. So for an audio file all possible feature combination clustering ids become available for the recommendation system studies below.

For the following audio file :(Total number of clusters at each time is 20)

Table 4.1: Clustering Results of an Audio File

File Name	…/Destiny'sChild-LoseMyBreath.wav
Clustering id STFT features based	2
Clustering id BEAT features based	3
Clustering id MFCC features based	4
Clustering id MPITCH features based	6
Clustering id ALL features based	11
Clustering id STFT & MFCC features based	12
Clustering id STFT & MPITCH features based	19
Other possible feature combinations…	…

4.1.2 Singer Similarity

The dataset, explained in the Section 3, contains 17 main categories. Under these categories, there are songs, and their related singers. For instance, these are the three audio files from this study's dataset.

[1]…Foreign/ElvisPresley-It'sNowOrNever.wav

[2]…Foreign/ElvisPresley-LoveMeTender.wav

[3]…Foreign/Eminem-LikeToySoldiers.wav

To find the similarity score between two audio files is very easy:

For instance:

File [1] and [2] has 2 scores, 1 from the similarity of the category of Foreign and the other is the similarity of the singer,

While file [2] and [3] has 1 score from the similarity of the category of Foreign.

4.1.3 Popularity

Popularity means "what do others listen, what do they prefer?". Popularity is a very important metric, which increases the success ratio of the results. The reason for the increase in success ratio is, if an item is popular it means most people listen to it, which means if the system recommends it it will be a successful recommendation.
Based on days the user session data are available, a matrix which shows the number of times a song is requested on a day is created as follows:

Table 4.2: Number of Times Each Song is Listened on a Day (Popularity Matrix)

Date	**Total Count**	**File-1 Count**	**File-2 Count**	**File-3 Count**	**…**	**File 730**
01.01.2006	100	3	5	1		6
02.01.2006	120	80	23	1		1
03.01.2006	80	7	34	34		2
04.01.2006	180	56	45	4		3
05.01.2006	167	1	2	11		14
06.01.2006	200	14	2	3		15

The matrix has the following information:

<u>On a specific day:</u> How many times a file is preferred?

<u>On that specific day:</u> How many times all files are preferred in totally?

So, when it comes to calculate a rating ratio, popularity factor of an audio file on a specific day:
For instance, for the file-1, on 01.01.2006;

$$\frac{3}{100} \equiv 0.03$$

For instance, for the file-2, on 01.01.2006;

$$\frac{5}{100} \equiv 0.05$$

4.1.4 User Grouping

User grouping is mainly used in Learning–Recommendation system Method, which is explained further in this section.

User grouping factor attempts to find similar users who preferred similar audio files. In order to do this, a distance metric between sessions is defined as follows:

$$d(sessioni, sessionj) = \frac{1}{Ni * Nj} \sum_{ii=1}^{Ni} \sum_{jj=1}^{Nj} d(sessioni(ii), sessionj(jj)) \qquad \textbf{(4.1)}$$

where N_i is the number of the songs in the first session,

N_j is the number of the songs in the second session,

sessioni (ii):the ii^{th} audio file in the first session,

sessionj (jj):the jj^{th} th audio file in the first session.

And the distance between two songs x and y are computed as the distance between their MARSYAS features:

$$d(x,y)=\sqrt{\frac{1}{|x|}\sum_{i=1}^{|x|}\left(x[i]-y[i]\right)^2} \quad \textbf{(4.2)}$$

where x is the first audio file,

y is the second audio file,

x[i]: i^{th} MARSYAS feature of the first audio file,

y[i]: i^{th} MARSYAS feature of the second audio file.

After these calculations, the following matrix which shows distances between users (actually user sessions) is produced:

Table 4.3: Matrix of Distances between Users

	User-1	**User-2**	**User-3**	**…**	**User-n**
User-1	**0**	0.012	0.0001	..	0.0078
User-2	0.012	**0**	0.008	..	0.00001
User-3	0.0001	0.008	**0**	..	0.00002
…	..	..	..	**0**	..
User-n	0.0078	0.00001	0.00002	…	**0**

When this matrix is input to CLUTO, similar to grouping of songs, a grouping of users is produced. These groups will be called user clusters.

4.2 Methods Used In the Proposed System

In this section, we present the recommendation methods that we experiment with in the following section.

The main framework of our recommendation system is shown in the following figure:

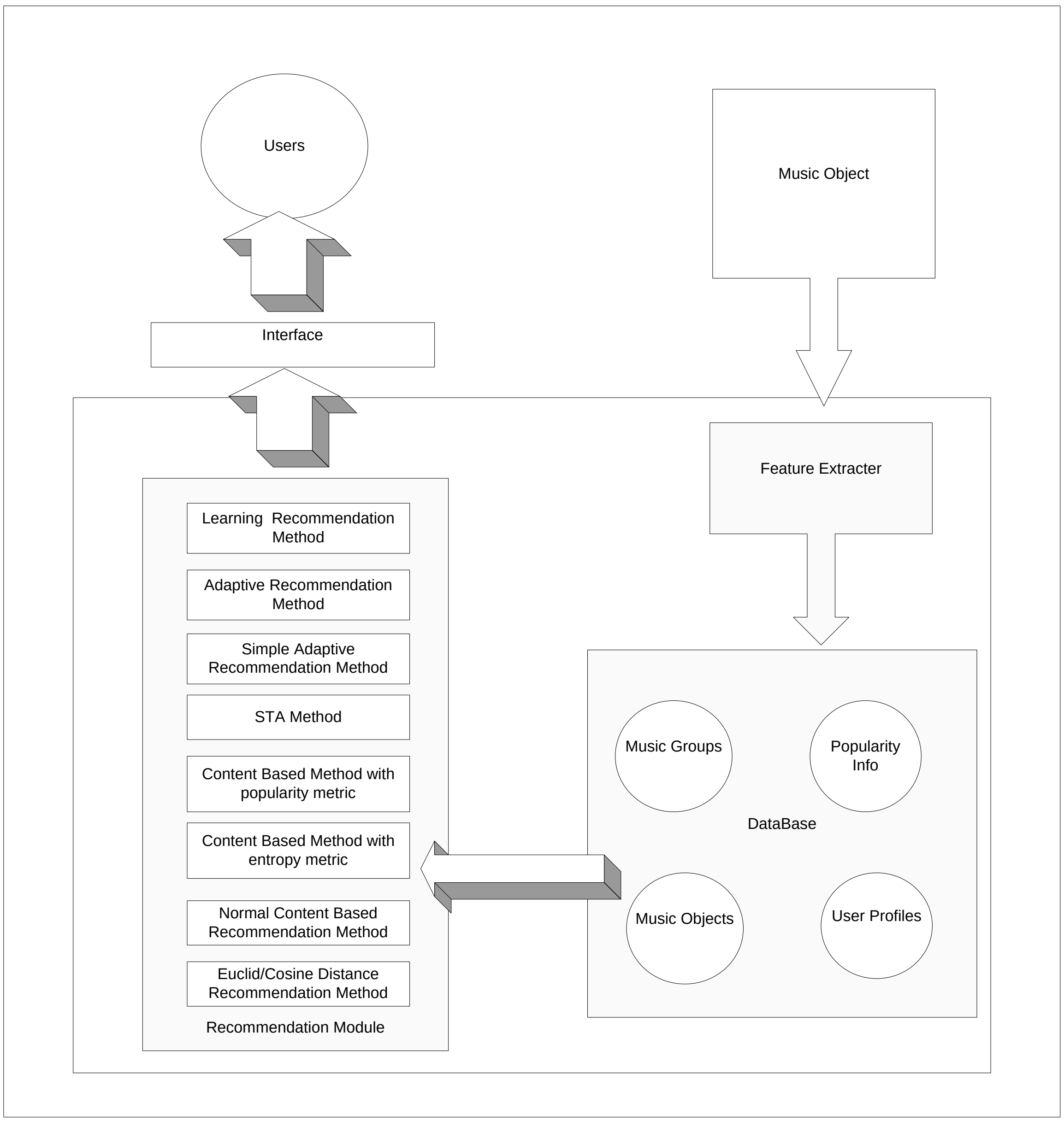


Figure 4.1: General Form of our Music Recommendation System

4.2.1 Euclidean/Cosine Distance Based Recommendation:

The first recommendation system we study is a very simple one and it works like a nearest neighbor classifier [31].

After all the audio files in the dataset, which is mentioned in section 3, is converted into wav format; their BEAT, STFT, MFCC, MPITCH features are extracted via MARSYAS [28]. After these operations have been performed, every audio file has its own 30 features. The following table shows 2 different audio files and their corresponding Marsyas features.

Table 4.4: Marsyas Features of Two Different Audio Files

File Name	Classical-Beethoven-9thsymphony.wav	Classical-Piano-Concerto.wav
Feature -1	0.0315373	0.113804
Feature -2	0.0291096	0.0573399
Feature -3	0.923022	0.503847
Feature -4	258	234
Feature -5	246	156
Feature -6	491.438	537.107
Feature -7	117.435	107.125
Feature -8	249.581	249.4
Feature -9	235.037	213.197
Feature -10	217.245	196.161
Feature -11	291.317	341.968
Feature -12	27.7762	238.225
Feature -13	15279.1	18557.3
Feature -14	4259.77	4123.06
Feature -15	0.0220842	0.0216745
Feature -16	-53.7041	-57.6657
Feature -17	5.48251	4.69975
Feature -18	1.01455	1.39223

Feature -19	0.770429	1.10165
Feature -20	0.617137	0.986999
Feature -21	2.53691	2.61319
Feature -22	0.407667	0.391915
Feature -23	0.163391	0.0990582
Feature -24	0.0667999	0.0419988
Feature -25	0.0454699	0.0400316
Feature -26	79.7333	60.1302
Feature -27	20	20
Feature -28	4.66491	2.12872
Feature -29	10	10
Feature -30	-1	-1

In the dataset, there are a total of 11398, 1215 and 518 user sessions of length 5, 10 and 15 respectively. Due to time limitations, 2000 (session length=5), 1000 (session length=10) and 500 (session length=15) users are used in the experiments. Every user in the session length of 5 file has 5 audio songs listened in a specific time period. The following is a general form of a user's session file:

UserSession1 = [piece1, t1], [piece2, t2], [piece3, t3], [piece4, t4], [piece5, t5]

UserSession2 = [piece6, t6], [piece7, t7], [piece8, t8], [piece9, t9], [piece10, t10]

UserSession3 = [piece11, t11], [piece12, t12], [piece13, t13], [piece14, t14], [piece15, t15]

We separate our data randomly into 90% train and 10% test set.

Inputs:	outputs
Train:	
[piece1, t1], [piece2, t2], [piece3, t3], [piece4, t4], t5	piece5
[piece6, t6], [piece7, t7], [piece8, t8], [piece9, t9], t10	piece10

Every user in this session info file has this general formula on their past audio file choices. In order to guess what the user listened at time of t(5) or t(10),the following calculations are done: First ,the Euclid/Cosine distance between the first user's first song and the second user's first song, second song, third song, fourth song and the fifth song. The same is done for the first user's second and third and the fourth songs. After that, an average value is obtained by simply taking the average of these calculated values.

User-1	User-2
T (p): audio file-1	T (a): audio file-1
T (q): audio file-2	T (b): audio file-2
T (t): audio file-3	T(c): audio file-3
T(x): audio file-4	T (d): audio file-4

T(y): audio file-5	T (e): audio file-5

We compute distance between two lists of songs as follows in the Equation 4.1:

Distance between two songs x and y are computed as the distance between their MARSYAS features as in the Equation 4.2:

If the song predicted is within the first k (1, 2, 5, 10…etc.) returned from the recommendation system, then we assume a successful recommendation.

We partition the training data again into 90% train and 10% validation set. We choose the value of parameters that result in the minimum error on the validation set.

We report errors based on the existence of the output song within the top 1, 2, 5, 10 of the songs recommended by the system. The system recommends the songs those have minimum distance errors.

After performing these calculations on all of the users in the session legth-5, 10, 15 files, the following results are obtained for an example test scenario:

Table 4.5: Error Distances between the Correct Song and the Recommended One

Session's last audio file	Session Distance to the recommended audio file	Recommended audio file	Error between the correct audio file and the recommended one
Ibrahim Tatlises-Bileydim	173855.3	Hadise-Stir Me Up	555.9671
Ozlem Tekin-Cinayet	175573.4	Hadise-Stir Me Up	359.2185
Sebnem Ferah-Can Kiriklari	180033.7	Hadise-Stir Me Up	638.1646
Seksendort-Affet	180223.8	Hadise-Stir Me Up	1464.851
Yildiz Tilbe-Ummadigin Anda	181397	Hadise-Stir Me Up	1974.331
Kenan Dogulu-Askim Askim	186240.7	Hadise-Stir Me Up	748.0977
Edip Akbayram-Hasretinle Yandi	190108.3	Hadise-Stir Me Up	448.2439
Gokhan Ozen-Kalbim Seninle	190156.4	Hadise-Stir Me Up	523.581
Metin Arolat-Ruhum Seninle	191724	Hadise-Stir Me Up	540.8978
Hadise-Stir Me Up	193144.2	Hadise-Stir Me Up	0
--			
Ibrahim Tatlises-Bir Kulunu Cok	143234.4	Kibariye-Yak Butun Fotograflari	2194.842
Gokhan Ozen-Kalbim Seninle	145132.3	Kibariye-Yak Butun Fotograflari	320.9787
Sibel Can-Yalnizlar Treni	147106.4	Kibariye-Yak Butun Fotograflari	517.8491
Seksendort-Olurum Hasretinle	148399.3	Kibariye-Yak Butun Fotograflari	1383.517
Yildiz Tilbe-Ummadigin Anda	149157	Kibariye-Yak Butun Fotograflari	1234.01
Yalin-Yagmur	149597.3	Kibariye-Yak Butun Fotograflari	1562.437
Irem-Hayal Et Sevgilim	149680.7	Kibariye-Yak Butun Fotograflari	1880.389
Ferhat Gocer-Don Diyemedim	150785.3	Kibariye-Yak Butun Fotograflari	385.0924
Kargo-Sonbahar	151180.5	Kibariye-Yak Butun Fotograflari	521.8391
Irem-Beyaz yalan	153790.2	Kibariye-Yak Butun Fotograflari	756.6326
--			

4.2.2 Content Based Recommendation Using Entropy and Popularity Metrics:

This method based on the [10] content based recommendation algorithm, which is mentioned in section 2. [10] used MIDI files, whereas our recommendation system is based on audio files. In addition, we consider the fact that every user may give different importance to certain aspects of songs, such as melody, tempo etc. We try to find the most important aspect for a certain user based on an entropy measure and recommend to him based on that aspect.

First of all, all user sessions (we used length of 5, 10, 15 user sessions in our tests) are clustered in CLUTO based on

BEAT (6 features) only,

STFT (9 features) only,

MFCC (10 features) only,

MPITCH (5 features) only,

All features,

BEAT & STFT features,

BEAT & MFCC features,

BEAT & MPITCH features,

STFT & MFCC features,

STFT & MPITCH features,

…

BEAT &STFT & MFCC features,

BEAT &STFT & MPITCH features,

…

and all other possible feature combinations.

A session file with these above possible features are prepared. For each feature file the user session is given to CLUTO program to be clustered. An example result could be:

Table 4.6: Clustering Results

	song-1	song-2	song-3	song-4
Cluster no based on BEAT features only	1	2	3	6
Cluster no based on STFT features only	3	4	6	1
Cluster no based on MFCC features only	4	5	2	1
Cluster no based on MPITCH features only	8	9	10	5
Cluster no based on BEAT & STFT features	2	4	6	7
Cluster no based on BEAT & MFCC features	1	4	6	8
Cluster no based on all features	5	6	8	10
Cluster no based on BEAT & MFCC & MPITCH features	2	5	7	9
Other feature combinations…				

Then for every feature combination, entropy values are calculated for that user session:

For entropy calculation the following formula is used:

$$S = -\sum_{i=1}^{C} p_i \log p_i \qquad \textbf{(4.3)}$$

In this formula, C=20 is the number of song clusters for a certain MARSYAS feature combination (which corresponds to a row in table 4.6), pi shows the number of songs that fell in cluster i in a certain session divided by the session length (total number of songs in the session). If the entropy is high for a feature set, it means the songs of the session are distributes all around the place and hence user's songs can not be grouped successfully based on that metric. We should choose the feature set that results in the minimum entropy for each specific user.

Table 4.7: Clustering Results with Entropy Values

	song-1	song-2	song-3	Song--4	Entropy value
Cluster no based on BEAT features only	1	2	3	6	A
Cluster no based on STFT features only	3	4	6	1	B
Cluster no based on MFCC features only	4	5	2	1	C
Cluster no based on MPITCH features only	8	9	10	5	D
Cluster no based on BEAT & STFT features	2	4	6	7	E

Cluster no based on BEAT &MFCC features only	1	4	6	8	F
Cluster no based on all features	5	6	8	10	G
Cluster no based on BEAT & MFCC & MPITCH features only	2	5	7	9	H
Other feature combinations…					.

Then the feature combination for the user is selected whose entropy value is min among the others. This means that, a user's only the features are used in the following CB recommendation algorithm that has the min entropy value.

In order to get advantage of the popularity metric, we recommend a certain portion of the songs using this method and we fill up the remaining songs based on the popular songs at the time of the recommendation.

Table 4.8 shows the success of recommendation for varying ratio of recommendations from the popular songs. A recommendation is successful if the N_i'th song is among the recommended songs. As expected, as the percentage of popular songs increase, recommendation success increases.

Table 4.8: Success Results for Content Based Recommendation

Session Length	#Songs	#Users	%Popular	%Classical CB	%Success
5	20	2000	20	80	21
5	20	2000	40	60	30
5	20	2000	60	40	40
5	20	2000	80	20	44
10	20	1000	20	80	22
10	20	1000	40	60	32
10	20	1000	60	40	41
10	20	1000	80	20	46
15	20	500	20	80	22
15	20	500	40	60	33
15	20	500	60	40	44
15	20	500	80	20	50

4.2.3 STA

We perform the STA [10] method (it is mentioned in section 2), similar to [10]:
Short Term Recommended: The songs which are preferred in the last 3 months. (3 month is an example value; it depends on the dataset distribution.).In the tests;

<u>Short term rate:</u> shows the ratio how many songs are selected from short term songs list

Long term rate: shows the ratio how many songs are selected from long term songs list

The followings are the test results:

Table 4.9: Test Results of STA Method

Experiment ID	Total Recommendation Number	Short Term Recommended Rate	Long Term Recommended Rate	Number of users	Length Of Session	Number Of Correct Recommended Users
1	20	0.5	0.5	100	10	27
2	20	0.75	0.25	100	10	29
3	20	0.8	0.2	100	10	29
4	20	0.9	0.1	100	10	28
5	20	1	0	100	10	20
6	20	0	1	100	10	22
7	20	0.5	0.5	338	10	128
8	20	0.75	0.25	338	10	95
9	20	0.8	0.2	338	10	94
10	20	0.9	0.1	338	10	90
11	20	1	0	338	10	67
12	20	0	1	338	10	35
13	25	0.75	0.25	338	10	144
14	30	0.75	0.25	338	10	145
15	10	0.75	0.25	338	10	44
16	5	0.75	0.25	338	10	25

17	35	0.75	0.25	338	10	135
18	40	0.75	0.25	338	10	133
19	50	0.75	0.25	338	10	153
20	75	0.75	0.25	338	10	189
21	100	0.75	0.25	338	10	220
22	150	0.75	0.25	338	10	269
23	200	0.75	0.25	338	10	312
24	20	0.75	0.25	37	10	9

4.2.4 Simple Adaptive Recommendation:

In this method we use all three components (cluster similarity, singer similarity and the popularity metrics, mentioned in section 4.1) and learn the percentage values (percentage of songs to recommend from each of the three clusterings) for each component. We do the learning as follows:

For instance the user has 10 songs in his/her session; we skip the last song (because we want to find it at the end of this recommendation) and produce possible permutations with the remaining 9 songs as follows:

Song-1,song-2,song-3,song-4,song-5,song-6,song-7,song-8 ,?

Song-2, song-3, song-4, song-5, song-6, song-7, song-8,?

Song-3, song-4, song-5, song-6, song-7, song-8, ?

Song-4, song-5, song-6, song-7, song-8, ?

Song-1, song-3, song-4, song-5, song-6, song-7, song-8, ?

Song-2, song-4, song-6, song-7, song-8,?

….

Then, in order to find the last missing song, every time only the following methods are used:

- Only Content based with entropy factor,
- Only singer similarity,
- Only popularity factor.

While the algorithm is running the method that finds the correct result gets a point. Simply the method which has the maximum points is used in order to find the last song (10^{th} song) and the other methods are given 0 percentage.

The results of this recommendation scheme are shown in Table 4.10. As seen in the table; the percentage of success for Simple Fair Recommendation is a lot higher than the Content Based Recommendation. Simple Fair recommendation test results are in the Table 4.10.

Table 4.10: Test Results of Simple Adaptive Recommendation Method

Experiment Id	Session Length	Cluster Number	Recommendation Number	Number of users	#correct File	#correct singer	#correct cluster	#Correct Singer&Cluster
1	15	20	5	338	79(%23.3)	28	73	0
2	15	20	10	338	106	50	114	1
3	15	20	20	338	150	85	167	6
4	15	20	30	338	185	100	188	7
5	15	20	40	338	206	106	197	14
6	15	20	50	338	221	111	203	16
7	15	20	60	338	234	116	205	16
8	15	20	70	338	242	118	206	16
9	15	20	80	338	249	118	207	16
10	15	20	90	338	263	119	207	16
11	15	20	100	338	267	121	208	16
12	15	20	150	338	290	126	213	18
13	15	20	200	338	307	131	214	18

14	15	20	250	338	318	136	215	19
15	15	20	300	338	323	136	218	19
16	15	20	350	338	327	137	219	19
17	15	20	400	338	330	137	219	19
18	15	20	450	338	331	137	219	19
19	15	20	550	338	335	137	219	20
20	15	20	650	338	336	137	219	20
21	15	20	700	338	337	137	219	20
22	15	10	5	37	10	3	11	0
23	15	10	10	37	10	6	18	0
24	15	10	20	37	12	10	22	0
25	15	10	40	37	14	11	27	3
26	15	10	80	37	18	14	28	4
27	15	10	150	37	24	17	28	4
28	15	10	300	37				
29	15	10	5	610	164	68	97	1

30	15	10	10	610	220	106	177	1
31	15	10	20	610	299	168	243	3
32	15	10	40	610	398	210	299	7
33	15	10	80	610	498	225	318	12
34	15	10	150	610	560	243	325	14
35	15	10	300	610	597	253	326	15
36	15	10	20	337	140	85	127	7
37	15	30	20	337	146	90	82	7
38	15	40	20	337	144	90	48	5
39	15	60	20	337	142	89	29	1
40	15	80	20	337	148	91	27	1
41	15	100	20	337	147	91	24	4
42	15	20	20	338	150	79	160	2
43	15	20	20	338	136	79	63	6
44	15	20	20	338	135	77	160	2
45	15	20	20	338	134	74	156	1

46	15	20	20	338	121	94	169	2
47	15	20	20	338	117	91	163	4

4.2.5 Adaptive Recommendation:

In this recommendation scheme, we choose N_c, N_s, N_p from among a certain number (1000) of different possible values. These values are calculated by an auto-generated program in the computer. As we did in the previous recommendation algorithm, for each user, we evaluate each N_c, N_s, N_p combination's score based on how well they can predict each remaining song permutation. We choose the combination that gives the best success rate.

For instance, if the user has 10 songs in his/her session, we skip the last song and produce all possible recommendation combinations as follows:

Song-1,song-2,song-3,song-4,song-5,song-6,song-7,song-8 ,?

Song-2, song-3, song-4, song-5, song-6, song-7, song-8 ,?

Song-3, song-4, song-5, song-6, song-7, song-8, ?

Song-4, song-5, song-6, song-7, song-8, ?

Song-1, song-3, song-4, song-5, song-6, song-7, song-8, ?

Song-2, song-4, song-6, song-7, song-8,?

….

Then, in order to find the last missing song, in every time the following methods are used:

- Content based with entropy factor,
- Singer similarity,
- Popularity factor.

Every time the following auto-generated weight numbers are used:

{0, 0.1, 0.99}

{0 ,0.2, 0.98}

{0, 0.3, 0.97}

…

{0.50, 0.25, 0.25}

…

{0.75, 0.1, 0.15}

…

…

…

The first auto-generated number is: cluster similarity weight ratio

The second auto-generated number is: cluster similarity weight ratio

The first auto-generated number is: cluster similarity weight ratio

In total, the weight ratio combination is used which gets more accurate recommendations. The results for the adaptive recommendation method are shown in Table 4.11.

Table 4.12 contains a comparison between simple adaptive recommendation method and adaptive recommendation method. According to this table, the success ratio of Adaptive recommendation seems to be smaller than that of simple Adaptive Recommendation. We think that this is due to the fact that Simple Adaptive Recommendation uses a component (like singer for example) and ignores the other two (like content and popularity for example) when it makes its decision. Whereas, Adaptive Recommendation is able to evaluate contributions from all components at the same time. Another reason may be that there are too many possibilities in adaptive recommendation and the recommendation system may be overfitting the training data. [31].

Table 4.11: Test Results of Adaptive Recommendation Method

Experiment Id	**Session Length**	**Cluster Number**	**Recommendation Number**	**Number of users**	**#correct File**	**#correct singer**	**#Correct Singer& Cluster**
1	5(4)	20	20	608	395	**395**	**15**
2	10(9)	20	20	303	190	**106**	**90**
3	15(14)	20	20	518	362	**362**	**4**

Table 4.12: A Comparison between Simple Adaptive Recommendation and Adaptive Recommendation

Session Length	%RecomSuccess Simple Adaptive Recommendation	%RecomSuccess Adaptive Recommendation
5	70	65
10	71	63
15	73	70

4.2.6 Learning Approach on an Adaptive Music recommendation System with Popularity Data and Using User Grouping

This recommendation method follows the learning of the popularity, singer and content cluster weights, however in addition the user groups are also produced and taken into consideration. The recommendation method, recommends songs adaptively to each user based on the following criteria:

- Popularity metric,
- Singer similarity,
- Content Based Method with entropy metric,
- User grouping factor.

The percentage values are calculated adaptively. The following part explains how the user grouping mechanism works:

This method divides the time period that covers all the user session data into the following parts:

t-cluster,

t-train,

t-recommendation.

The songs that were listened to by a certain user in each of these time frames are processed separately as explained below:

t-cluster:

In this time-scope users in the system are clustered based on what they listened all through this time period. Clustering is done via CLUTO. (ClMethod: GRAPH, similarity: CORR).

Every song in the dataset (we have approximately 730 songs) has its own

Beat (6)

Stft (9)

Mfcc (10)

Mpitch (5)

…

All (30) features after MARSYAS feature extraction operation, which is mentioned in section 3. Based on these features every user session in this time-scope is sent to our clustering mechanism. This mechanism observes all possible feature (BEAT, STFT, MFCC, MPITCH, MPITCH&STFT, etc) combinations and extracts their related clustering results. Based on these results a simple Shannon entropy calculation is performed based on each clustering results. The minimum entropy leads us to the features we need to use for this user. This is the same scenario that we used in Content Based approach with entropy metric, mentioned in section 2.

After this clustering, users with their history (history lengths are like: 2-song-history, 3-song-history.4-song-history) are assigned to one of the following user-feature-specific-clusters:

User-group based on BEAT features (Approximately 20 clusters)

User-group based on STFT features (Approximately 20 clusters)

User-group based on MFCC features (Approximately 20 clusters)

User-group based on MPITCH features (Approximately 20 clusters)

User-group based on ALL features (Approximately 20 clusters)

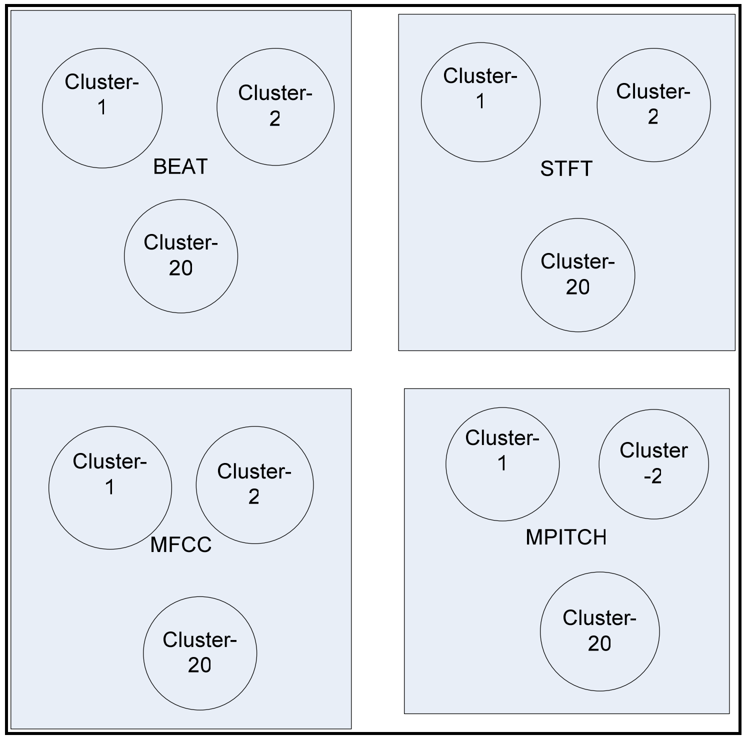


Figure 4.2: User Grouping Based On Marsyas Features

t-train:

In this time period the centroids of the above mentioned groups are calculated (based on taking the averages of the related features). Throughout this time period, any user (who reaches the system before t-cluster time or who is the new arrival) is attempted to be inserted into a group. For instance;

a) if the user is not new for the system

S/he is inserted into his/her own group, but the centroid of the user group is re-calculated.

b) If the user is one of the new arrivals

S/he is attempted to be inserted into a group based on the Euclidean distance calculation. The user will be send into the group with the minimum distance value between the centroid of the group and his/her song features.

The system applies the same operation for any coming user in this time period. So through this period the centroid values of these groups are re-calculated.

t-recommendation:

Here the system performs its recommendation operations based on the following logic:

Any coming user to the system will be recommended with the recommendation lists which are reported by the user groups.

Every user group (BEAT, STSFT, MFCC, and MPITCH) prepares a recommendation list based on the coming user's features and its own inner group's song-features. The important question to be anwered is which group will send how many songs. The answer is actually based on the following idea: At the beginning every group will send equal number of songs (for instance 5 songs). In the first step groups will send their recommendation lists. After that the system will compare the lists and looks for the actual song and based on the correctness it gives a success point to each group. Succesfull groups will send the more songs at the next steps while unsuccessful groups will send the same number of songs with the previous step. The algorithm goes on in the following logic.[1]

This procedure mentioned here, is used only for a portion of recommendation songs. The others will be recommended with again based on popularity, singer similarity.

The weight values of the factors (grouping users mentioned in this section, content based approach with entropy factor mentioned in the above part of this section, popularity mentioned in section 4.1,singer similarity mentioned in section 4.1) will be decided as follows:

Grouping users and content based approach with entropy factor: %20

Singer similarity: %5

Popularity: %75

Which gives more correct results?

---

[1] This specific cluster learning idea is based on a discussion with Sule Gunduz Oguducu, whom we thank for her contribution.

The success ratio is shown in the following table:

Table 4.13: Test Results of Learning Approach

Session Length	Number of recommendations	Number of total files	Success Found%
5	20	2000	78
10	20	1000	80
15	20	500	81

4.2.7. Summary of Experimental Results

The following table shows a summary of the above mentioned music recommendation methods:

Table 4.14: Comparison of all Methods Proposed in this Hybrid System

Music Recommendation Method	**Success Ratio**
Euclid/Cosine Distance Based Recommendation	%1-%5
Content Based Recommendation Using Entropy and Popularity Metrics	%21-%50
STA	%7-%43
Adaptive Recommendation Method	%65-%70
Simple Adaptive Recommendation Method	%70-%73
Learning Approach	%78-%81

According to Table 4-14, Euclid/Cosine Distance Based Recommendation method produces results with %1-%5 success ratios. This method works based on only calculated distance values and tries to find the minimum one. Since these results are not sufficient, it is a good idea to look for what other people listen to in short-term and long-term time periods. This mechanism is embedded in STA method and it gives %7-%43 success ratios. In STA, there is no special effort to trace what that user listened at past who actually will have recommendations from the system. So using popularity and entropy metrics which means combining CB method with STA method and using entropy factor increase the recommendation success ratio to %21-%50. But, still something which is very important is missing: making all these things adaptively. Adaptive recommendation and simple recommendation methods recommend based on an artificial logic which is produced by only that user specifically. For every user the system produces new rules dynamically. Table 4.14 shows that adaptivity increases recommendation success ratio to %65-73. And contributing user grouping factor with learning mechanism also increases success ratio to %78-%81.

5 IMPLEMENTATION OF THE SYSTEM

We implemented all the recommendation systems mentioned above. In this section, we give the implementation details.

5.1 Implementation Environment:

After extracting marsyas features of the audio files, data needs to be arranged. Also since the dataset contains more than 50 user session files with the total of approximately 1,300,000 distinct users. For these purposes it is hard to use this data without some helper functions .To extract needed information the following helper functions are implemented:

- Create user-session info with the session lengths of …(5,10,15,20,25,30,…135)
- Create user-session info with the session lengths of more than…(5,10,15,20,25,30,…135)
- Add these fields to the user-session info files :

  The user-id,

  The audio file name is selected,

  The time that file is selected,

  Maryas features of that audio file.
- Extract all marsyas features of the file with the id … (1, 2, 6, 8, 12, 78,123,600…)
- Extract stft marsyas features of the file with the id … (1, 2, 6, 8, 12, 78,123,600…)

- Extract mfcc marsyas features of the file with the id … (1, 2, 6, 8, 12, 78,123,600…)
- Extract beat marsyas features of the file with the id … (1, 2, 6, 8, 12, 78,123,600…)
- Extract mpitch marsyas features of the file with the id (1, 2, 6, 8, 12, 78,123,600…)
- Extract stft&beat marsyas features of the file with the id (1, 2, 6, 8, 12, 78,123,600…)
- Extract … marsyas features of the file with the id … (1, 2, 6, 8, 12, 78,123,600…)
- Normalize feature values.
- Create an input matrix for clustering for the user session…
- Arrange the output matrix after clustering as an input file for mat lab source code.
- Extract short term song list.
- Extract long term song list.
- Create a matrix which shows on which date how many times which song is chosen.
- Create a matrix as an input file for cluto to cluster users.
- Arrange user groups file.

These are the main helper functions which are implemented as separate classes in C# (Visual Studio .Net).

Then the main algorithms for each method are implemented in Matlab (Matlab Version 6.5 Release 13).In Matlab the following functions are created:

- Function for calculate Euclid distance between user sessions.
- Function for calculate Cosine distance between user sessions.
- Function for calculate the distance between songs based on their marsyas features.
- Function for recommending songs based on the minimum distance.
- Function for recommending based on content based approach.
- Function for recommending based on statistical approach.
- Function for creating auto-generated weight values for adaptive methods
- Function for recommending based on simple adaptive approach.
- Function for recommending based on adaptive approach.
- Function for recommending based on learning approach.
- Functions for finding error rate/success rate for each implemented method.

5.2 Graphical User Interface of the proposed Music recommendation System

The following graphical interface is prepared in order to use the recommendation system methods created&improved in this thesis. The following fields exist:

- Session file name
- Method for recommendation algorithm
- Number of clusters
- Number of recommendation songs

These parametric values, of course can be improved.

The followings are some sample screen views from the interface:

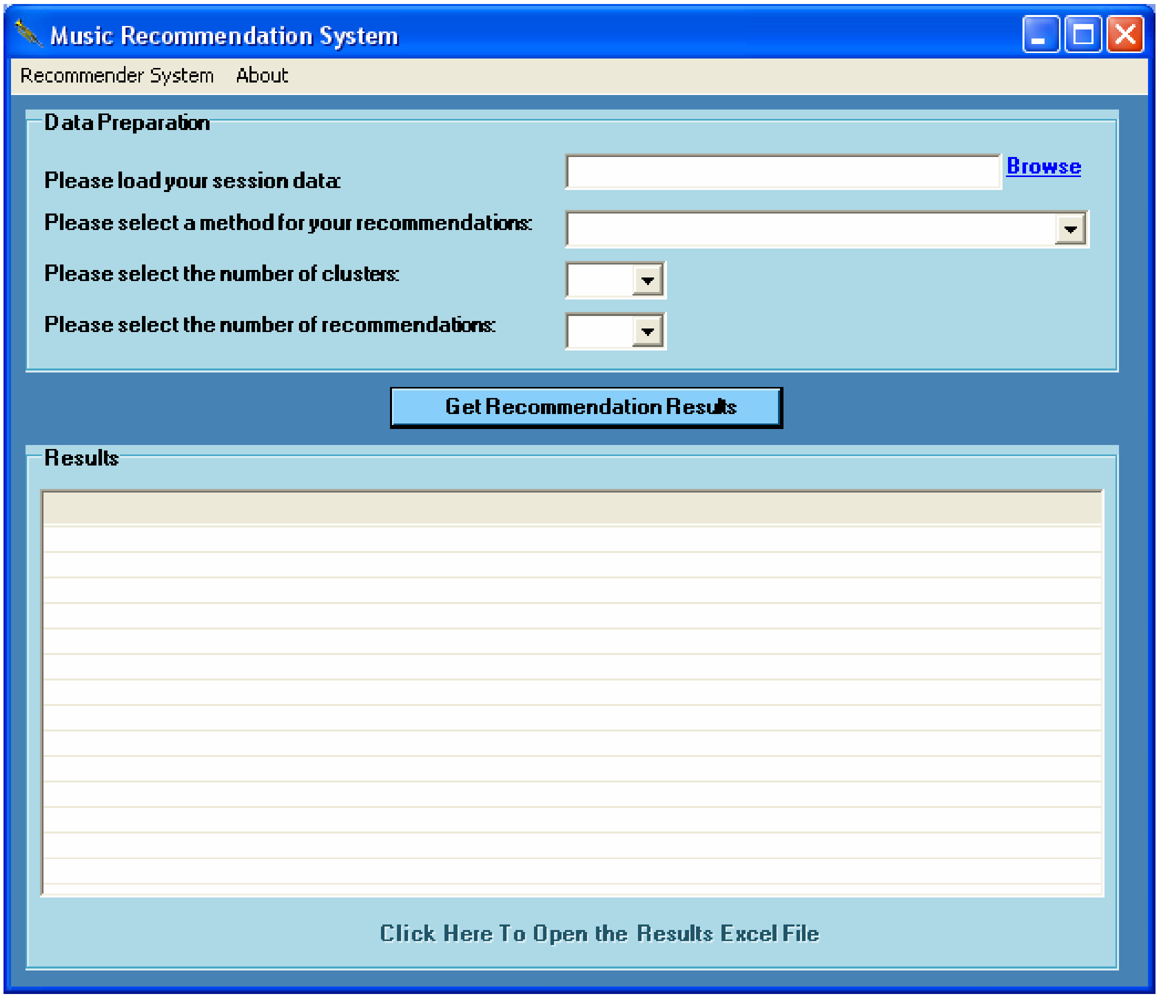


Figure 5.1: Music recommendation System-GUI-1

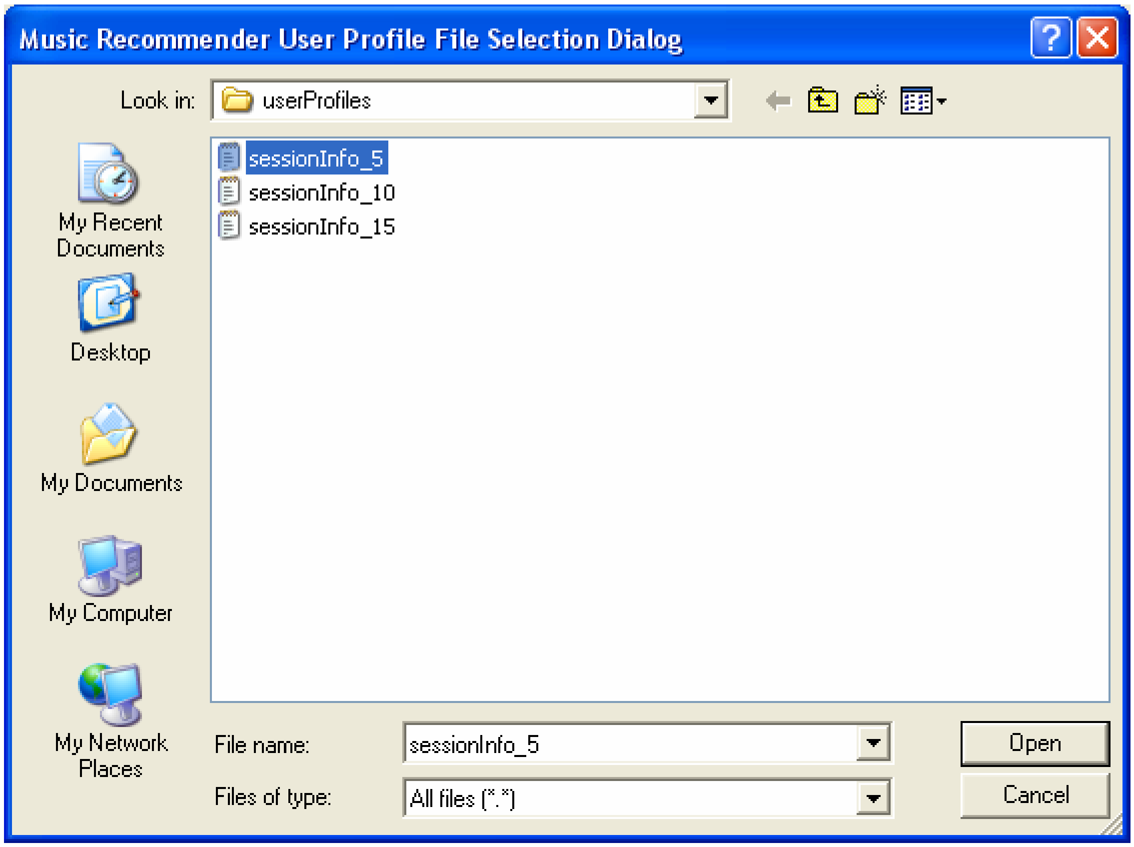


Figure 5.2: Music recommendation System-GUI-2

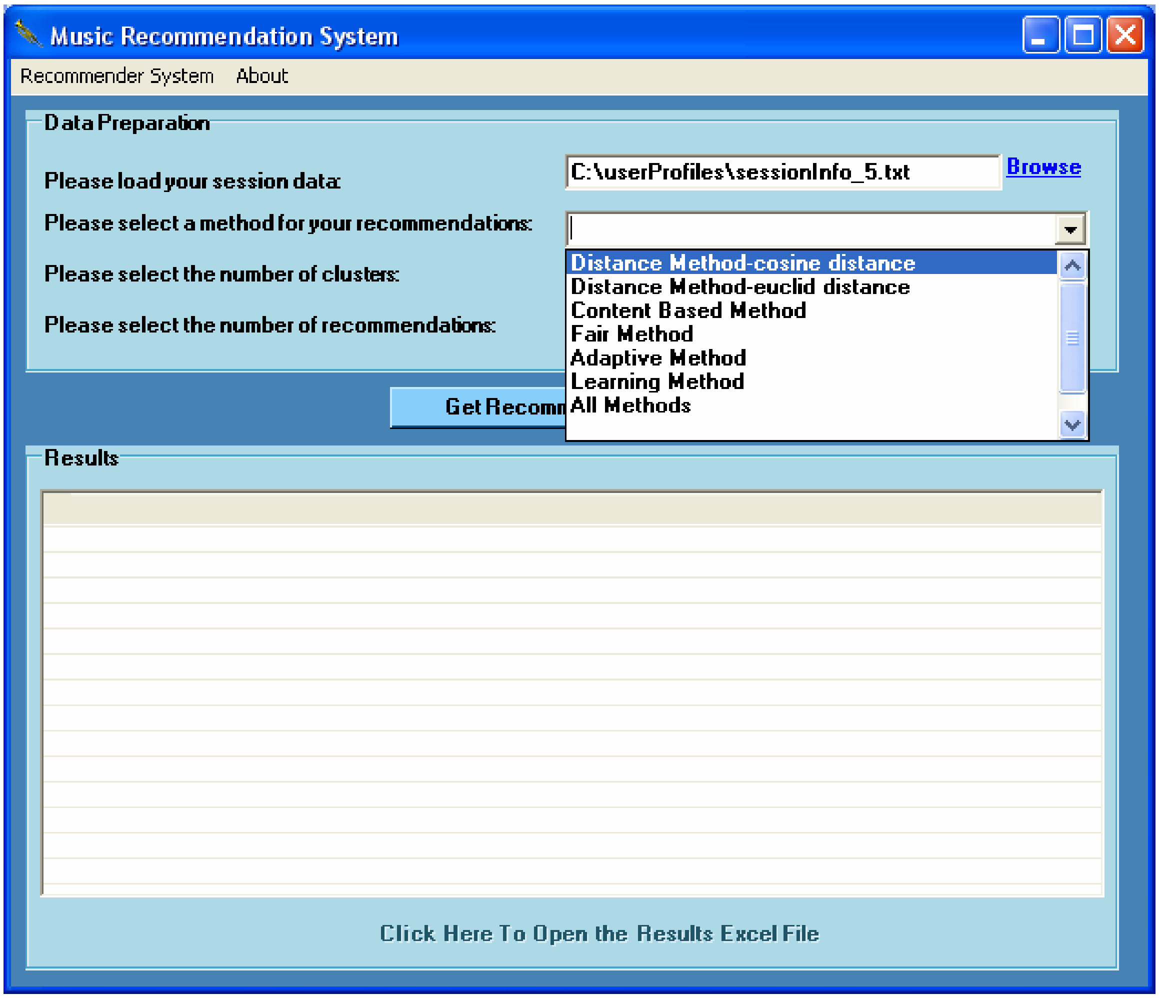


Figure 5.3: Music recommendation System-GUI-3

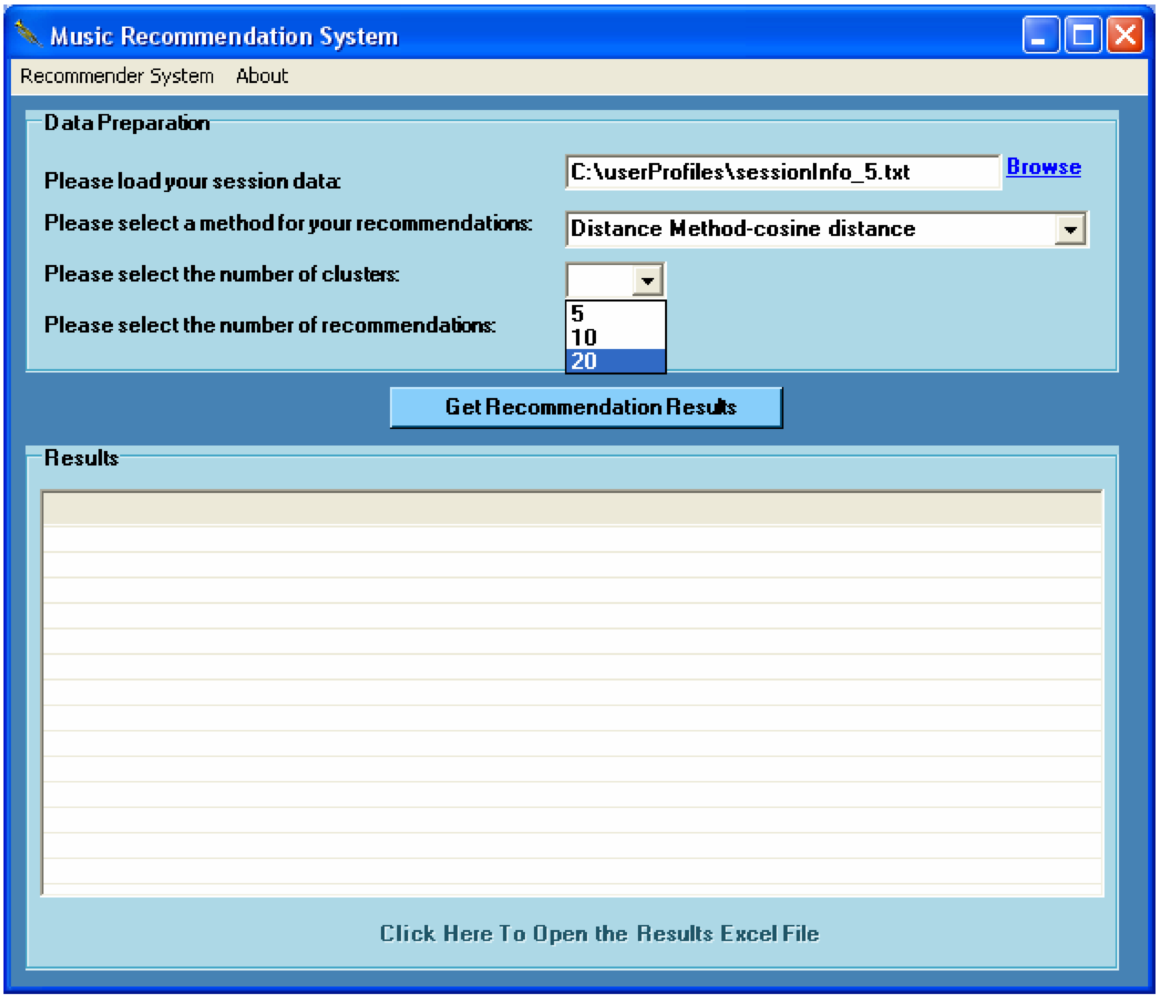


Figure 5.4: Music recommendation System-GUI-4

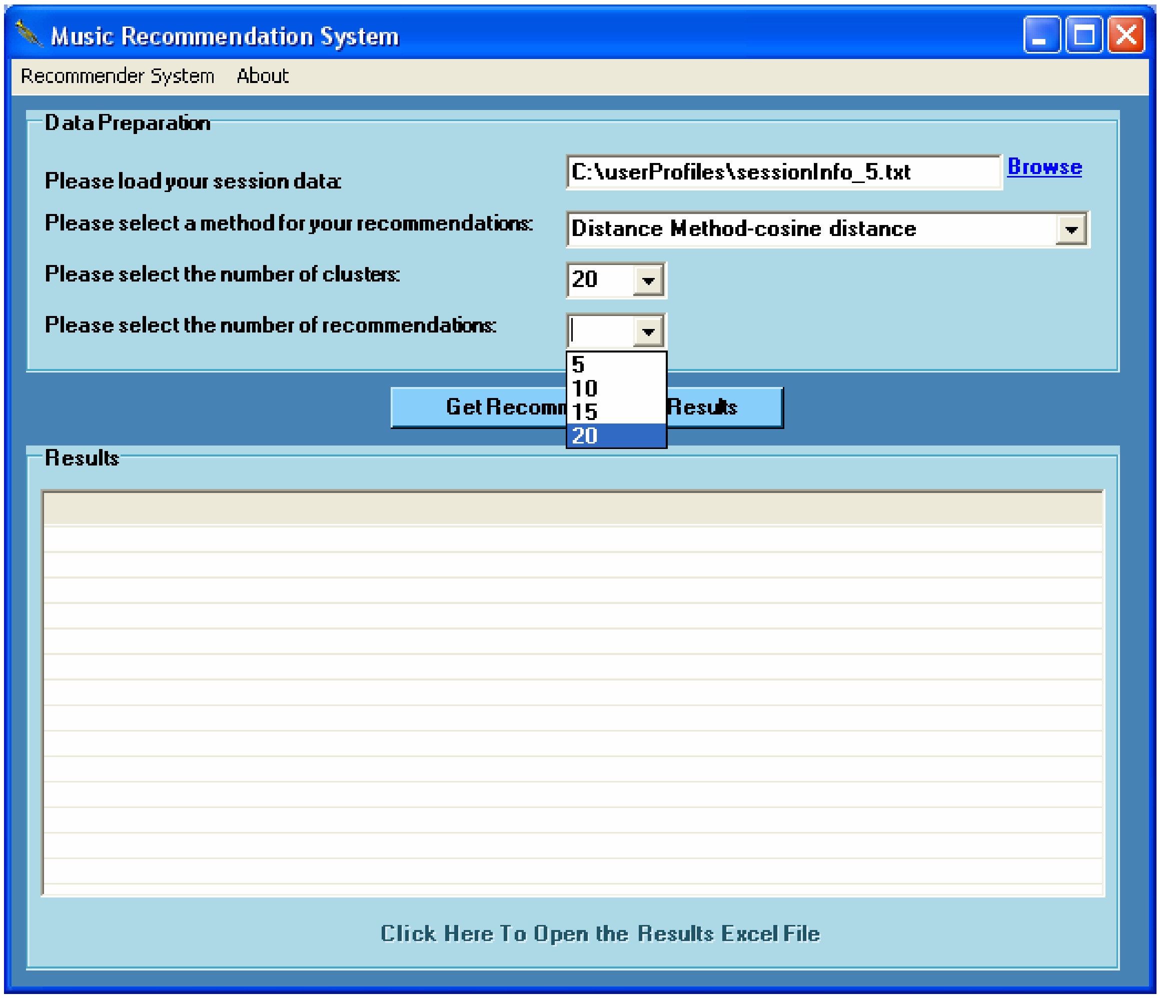


Figure 5.5: Music recommendation System-GUI-5

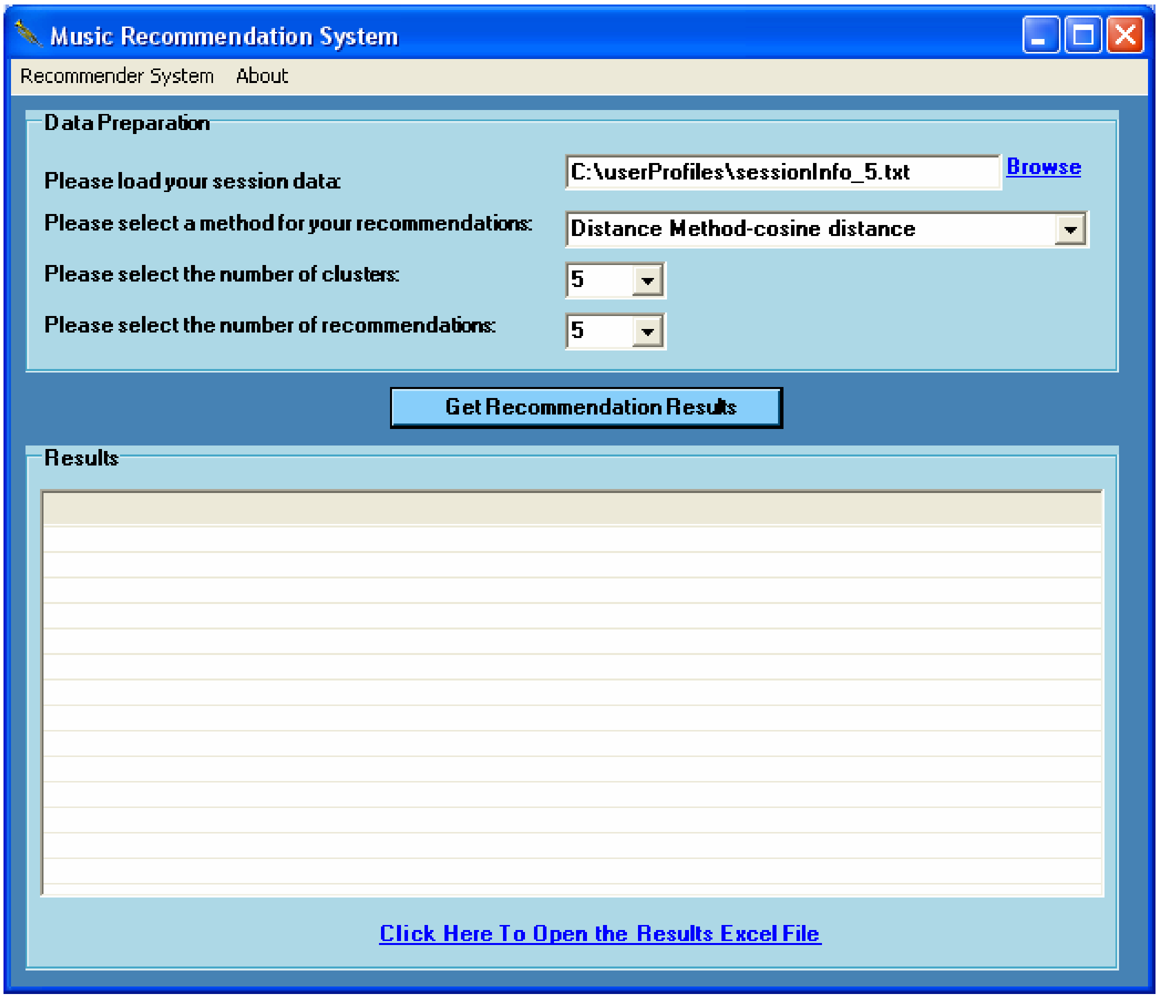
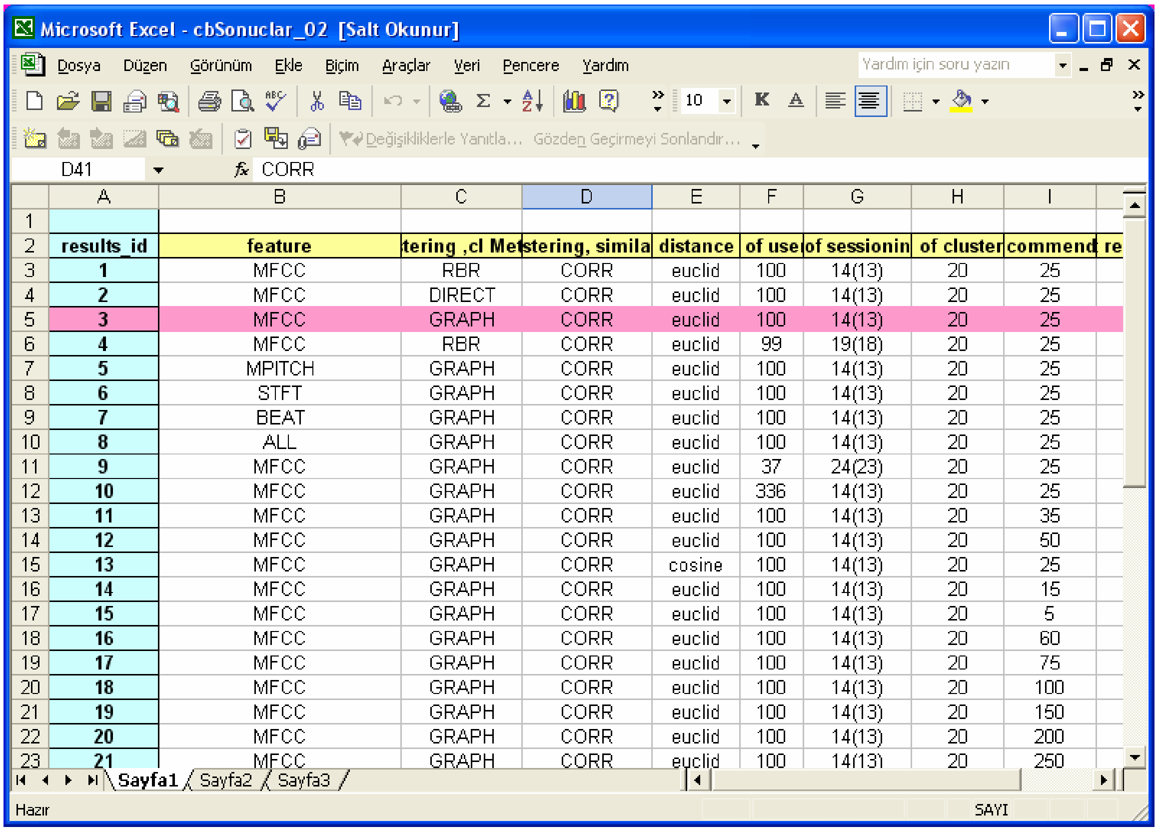


Figure 5.6: Music recommendation System-GUI-6

After entering all the parametric values, and just entering the 'Get Recommendation Results' button ,the algorithm runs, and whenever it finishes the 'Click Here To Open the Results Excel File' link is activated. Just clicking the link, the corresponding excel file is opened, or it can be loaded into the list which is located just above the link in the screen.



	A	B	C	D	E	F	G	H	I	
1										
2	results_id	feature	tering ,cl Met	stering, simila	distance	of use	of sessionin	of cluster	commend	re
3	1	MFCC	RBR	CORR	euclid	100	14(13)	20	25	
4	2	MFCC	DIRECT	CORR	euclid	100	14(13)	20	25	
5	3	MFCC	GRAPH	CORR	euclid	100	14(13)	20	25	
6	4	MFCC	RBR	CORR	euclid	99	19(18)	20	25	
7	5	MPITCH	GRAPH	CORR	euclid	100	14(13)	20	25	
8	6	STFT	GRAPH	CORR	euclid	100	14(13)	20	25	
9	7	BEAT	GRAPH	CORR	euclid	100	14(13)	20	25	
10	8	ALL	GRAPH	CORR	euclid	100	14(13)	20	25	
11	9	MFCC	GRAPH	CORR	euclid	37	24(23)	20	25	
12	10	MFCC	GRAPH	CORR	euclid	336	14(13)	20	25	
13	11	MFCC	GRAPH	CORR	euclid	100	14(13)	20	35	
14	12	MFCC	GRAPH	CORR	euclid	100	14(13)	20	50	
15	13	MFCC	GRAPH	CORR	cosine	100	14(13)	20	25	
16	14	MFCC	GRAPH	CORR	euclid	100	14(13)	20	15	
17	15	MFCC	GRAPH	CORR	euclid	100	14(13)	20	5	
18	16	MFCC	GRAPH	CORR	euclid	100	14(13)	20	60	
19	17	MFCC	GRAPH	CORR	euclid	100	14(13)	20	75	
20	18	MFCC	GRAPH	CORR	euclid	100	14(13)	20	100	
21	19	MFCC	GRAPH	CORR	euclid	100	14(13)	20	150	
22	20	MFCC	GRAPH	CORR	euclid	100	14(13)	20	200	
23	21	MFCC	GRAPH	CORR	euclid	100	14(13)	20	250	

Figure 5.7: Example Output File from Music recommendation System-GUI

6. CONCLUSION AND FUTURE WORK

In this thesis, different music recommendation systems are implemented and tested.

The effect of varying the degree of recommendation from each of the different groupings (song clustering, popularity, singer similarity, user grouping) are examined. First Content Based method is studied with the entropy approach which contains only song clustering approach. This adaptive content based approach resulted in better results compared to normal content based method. Then STA (Statistical Approach) which is another formulation of popularity is considered. STA considers audio files from short-term and long-term time period. Then new methods of the combination of these metrics are created. One of them is called Simple Adaptive Recommendation method, which contains singer similarity, popularity and song clustering metrics. The Adaptive Recommendation Method contains these metrics, too. But the calculation criterion is slightly different from the one in Simple Adaptive Recommendation. The test results show that the percentage of success for Simple Adaptive Recommendation is a lot higher than the Content Based Recommendation and also slightly more than Adaptive Recommendation method.

Then user grouping factor is introduced and with the history of the users, what they listened in the past, users are grouped. Then the same tests are repeated. The results show that user grouping factor increases the success ratio to about %75 which is a very good result for a music recommendation system.

Experimentation with other data sets and improvement of the learning mechanism in the user grouping method are the possible future study directions.

REFERENCES

[1] **Typken ,R. and Wiering , F. and Veltkamp,R. C.,** 2005.A Survey of Music Information Retreival Systems, in Proc. of the International Conference on Music Information Retrieval (ISMIR) .

[2] **Celma ,O. and Ramirez, M. and Herrera, P.,** 2005.Foafing the Music: A Music recommendation System Based on RSS Feeds and User Preferences, in Proc. of ISMIR

[3] **Linden ,G. and Smith,B. and York**,**J.** 2003. Amazon.com recommendations: Item-to-item Collaborative filtering," IEEE Internet Computing, 4(1).

[4] **Shardanand, U. and Maes, P.** 1995. Social information filtering: Algorithms for automating"Word of Mouth", ACM CHI'95 Conference on Human Factors in Computing Systems, pp. 210–217.

[5] **Cohen, W. and Fan**, **W.** 2000.Web-Collaborative filtering: Recommending music by crawling the Web, Computer Networks, vol. 33, no. 1–6, pp.685–698.

[6] **Hoashi, K., Matsumoto, K. and Inoue, N.** 2003. Personalization of user profiles for content-based music retrieval based on relevance feedback, ACM Multimedia, pp.110–119.

[7] **Logan, B.** 2004.Music recommendation from song sets, in Proc. of the International Conference on Music Information Retrieval (ISMIR).

[8] **Cano, P.K.** 2005. An Industrial-Strength Content-based Music recommendation System, in Proceedings of 28th Annual International ACM SIGIR Conference, Salvador, Brazil.

[9] **Yoshii**,**K.** 2006. Hybrid Collaborative and Content-based Music recommendation Using Probabilistic Model with Latent User Preferences, in Proc. of the International Conference on Music Information Retrieval (ISMIR).

[10] **Chen ,H-C. and Chen ,A.L.P.**,2005. A music recommendation system based on music and user grouping, Journal of Intelligent Information Systems, Volume 24, Numbers 2-3, 113-132.

[11] **Tzanetakis, G. and Cook, P. 2002.** Musical genre classification of audio signals, IEEE Transactions on Speech and Audio Processing, vol. 10, no. 5, pp. 293–302.

[12] **Karypis, G. "**Cluto: A Clustering Toolkit Manual", University of Minnesota, Department of Computer Science Technical Report.

[13] **Shardanand, U.** 1994. Social Information Filtering for Music recommendation, Msc Thesis September, MIT,

[14] Retrieved from the Internet web page, 2007. CDNow.com , http://www.amazon.com/exec/obidos/tg/browse/

[15] Retrieved from the Internet web page, 2007. Helping artist get heard and others to discover independent music, www.indiscover.get

[16] Retrieved from the Internet web page, 2007.Music Surfer , http://musicsurfer.iua.upf.edu/

[17] Retrieved from the Internet web page, 2007. C++ Library for Audio and Music, http://clam.iua.upf.edu/

[18] Retrieved from the Internet web page, 2007. Labrosa, http://labrosa.ee.CFumbia.edu/projects/musicsim/

[19] Retrieved from the Internet web page,2007 . Music-ir, http://www.music-ir.org/evaluation/

[20] Retrieved from the Internet web page, 2007. Music Elements, http://www.gprep.org/~music/musikbok/chap2.html

[21] Retrieved from the Internet web page, 2007. Music Elements, http://cnx.org/content/m11646/latest/

[22] Retrieved from the Internet web page, 2007. Music Elements, http://bridgman.tripod.com/theory/mhr.htm

[23] Retrieved from the Internet web page,2007. Music Elements, http://trumpet.sdsu.edu/M151/Elements_of_Music5.html

[24] Argela Technologies, Maslak, Istanbul, Turkey, +90 (0)212 328 12 48

[25] Ring Back Tone Product Specification Document, 2006, provided by Argela Yazilim ve Bilisim Teknolojileri Sanayi AS.,

[26] Retrieved from the Internet web page, 2007. Mthreedev, http://www.mthreedev.com/

[27] **Aucouturier, J-J. and Pachet, F.,**2003, Representing Musical Genre:A State of the Art ,Journal of new Music Research , Vol. 32, No. 1, pp. 83–93

[28] Retrieved from the Internet web page, 2007. Marsyas, http://marsyas.sf.net/

[29] Retrieved from the Internet web page, 2007. Clustering Tutorial, http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/

[30] **Tzanetakis, G. and Cook, P**. ,1999, MARSYAS: a framework for audio analysis, Cambridge University Press 4: 169-175

[31] **Duda, R. O., Hart, P.E., and Stork, D.G.,** 2000.Pattern Classification, 2nd Edition, Wiley.

AUTOBIOGRAPHY

Berna Altınel has born in Istanbul in 1980. She was graduated from Haydarpasa High School in 1999. Then she received the degree of computer engineer from Yeditepe University with high honor in 2004. For her graduation project she implemented a Turkish Morphological Analyzer. After that she started the master of computer science program in Istanbul Technical University in 2004. Within the same year she also started to work for a software company. Now she is working at TAGSOFT Software Company in Istanbul as a software engineer.