Noname manuscript No.
(will be inserted by the editor)

Combining Usage and Content in an Online Recommendation System for Music in the Long-Tail

Marcos Aurélio Domingues · Fabien Gouyon · Alípio Mário Jorge · José Paulo Leal · João Vinagre · Luís Lemos · Mohamed Sordo

Received: date / Accepted: date

Abstract Nowadays, a large number of people consume music from the web. Web sites and online services now typically contain millions of music tracks, which complicates search, retrieval, and discovery of music. Music recommender systems can address these issues by recommending relevant and novel music to a user based on personal musical tastes. In this paper we propose a hybrid music recommender system, which combines usage and content data. We describe an online evaluation experiment performed in real time on a commercial web site, specialized in content from the very long tail of music content. We compare it against two stand-alone recommender systems, the first system based on usage and the second one based on content data (namely, audio and textual tags). The results show that the proposed hybrid recommender shows advantages with respect to usage-based and content-based systems, namely, higher user absolute acceptance rate, higher user activity rate and higher user loyalty.

Keywords Music recommendation · hybrid recommender system · usage data · tags · audio features

Marcos Aurélio Domingues
INESC TEC - INESC Technology and Science, Portugal
Tel.: +351-91-7457112
Fax: +351-22-2094050
E-mail: marcos.a.domingues@inescporto.pt

Fabien Gouyon
INESC TEC - INESC Technology and Science, Portugal
E-mail: fgouyon@inescporto.pt

Alípio Mário Jorge
LIAAD / INESC TEC and FCUP, U. Porto, Portugal
E-mail: amjorge@fc.up.pt

José Paulo Leal
CRACS / INESC TEC and FCUP, U. Porto, Portugal
E-mail: zp@dcc.fc.up.pt

João Vinagre
LIAAD / INESC TEC and FCUP, U. Porto, Portugal
E-mail: jnsilva@inescporto.pt

Luís Lemos
LIAAD / INESC TEC and FCUP, U. Porto, Portugal
E-mail: llemos@inescporto.pt

Mohamed Sordo
Universitat Pompeu Fabra, Spain
E-mail: mohamed.sordo@upf.edu

1 Introduction

Music discovery and consumption has changed dramatically in recent years. According to recent reports, e.g. from consultancy firms [21], the web has become an increasingly relevant source of music discovery, recently reaching the importance of traditional sources such as AM/FM radios, music TVs, or friends. Most people now consume music on their personal computers and mobile devices via Internet. However, with virtually millions of pieces of music –henceforth tracks– available from thousands of web sites or online services, avoiding overwhelming choices and finding the "right" music has become a challenge for users. Music recommender systems have emerged in response to this problem. A music recommender system is an information filtering technology which can be used to output an ordered list of music tracks that are likely to be of interest to the user [10].

Music recommendation has flourished on the Internet, and web sites as Last.fm[1], Amazon[2], Audiobaba[3],

[1] `http://www.last.fm`
[2] `http://www.amazon.com`
[3] `http://audiobaba.com`

Mog[4], Musicovery[5], Shazam[6] and Pandora[7] are successful examples of music recommenders that adapt recommendations to particular user tastes. Diverse strategies to content filtering exist: (1) demographic filtering, (2) collaborative filtering (e.g., Amazon), (3) content-based (e.g., Pandora), and (4) hybrid approaches. The first strategy is the simplest but has been proven to have severe limitations [10], hence the focus in this paper is on strategies (2) to (4).

Collaborative filtering is based on usage data (typically rating data), i.e. recommendations are made to a user depending on personal past usage and on huge amounts of usage data from other users. This technique has proven to be extremely effective, for instance in Apple's Genius music recommender (part of iTunes) and to produce better recommendations than other techniques, as judged by panels of users [3]. However, this is only true when usage data is available. Indeed, particular related problems of collaborative filtering are the "early-rater" problem (items that are seldom rated, if at all, such as new items, or the less popular items from the end of the Long Tail [2], are never recommended), and popularity bias (items with many ratings are similar to lots of other items, and are hence very often recommended) [10].

Content-based approaches are not based on usage data, but on the very content of the items themselves. This content can be described automatically [5] (e.g., Muffin[8]), based on experts annotations (e.g., Pandora), or on mining contextual data of the items (by web mining, social tagging, etc). Anchoring recommendation on the content itself is supposed to solve the "early-rater" and popularity bias problems, however these approaches are typically less successful than collaborative filtering [27] due to still relatively limited performance of automatic music content description algorithms, making item similarity prone to mistakes. Another problem of the content-based paradigm is lack of personalization (similarity does not account for any data about or from the particular user to whom the recommendation is made) [10].

In order to address the previous problems and achieve better recommendations than stand-alone techniques, usage and content-based approaches have been combined in many different ways as hybrid recommenders [7]. In this paper we propose a hybrid recommender system implemented for Palco Principal[9], a Portuguese web site of music of diverse genres: typical Portuguese music as Fado of course, but also hip-hop, jazz, etc. Most of its music tracks are underground, unknown/unpopular and rarely accessed/rated by the users. In fact, only 19.7% of its artists also exist on the Last.fm web site. This is a good example of very long tail content, for which traditional usage-based recommenders typically do not work so well [10]. The hybrid recommender is evaluated online on the Palco Principal with real time user interaction. It is compared against a usage-based recommender and a content-based recommender. We also propose performance measures to determine the impact of the recommenders in user activity and loyalty.

The paper is organized as follows. In Section 2 we present some related work. Section 3 shows how to calculate similarities among items with usage-based data, and with content-based data accounting for two types of content-based descriptions (namely, tags or audio features). In Section 4 we describe the usage-based and the content-based recommender approaches which serve as benchmarks. In that section, we also present our hybrid proposal. Section 5 shows the results obtained with a case study setup to evaluate our hybrid proposal against the usage- and content-based approaches. In Section 6 we discuss the results and present our next steps. Finally, a summary is presented in Section 7.

2 Related Work

Recently, the music information retrieval community has focused its efforts in research and development of music recommendation services. The main music recommenders proposed in the literature are based on collaborative filtering [26,22], content-based [9,8] and hybrid [29,6] approaches.

Although the music recommendation system topic is not new, it is now inspired by new capabilities of large online services that provide not only millions of music tracks for listening to, but also radio station hosting. In [1], the authors propose a music recommender system that uses music tracks and meta-data from Internet radio streams as data source, instead of the more commonly used users playlists and feedback data. By using latent factor models and data from radio streams, a recommender system is built to predict the corresponding probability distribution of items to be played. A hybrid recommender system to recommend Internet radio stations is proposed in [30]. The system combines collaborative and user-based models. First, two ranked lists of recommended stations are generated by using the former models. Then, the lists are aggregated by the weighted sum of their ranks, generating a final list

[4] `http://www.mog.com`
[5] `http://musicovery.com`
[6] `http://www.shazam.com`
[7] `http://www.pandora.com`
[8] `http://www.mufin.com`
[9] `http://www.palcoprincipal.com`

of ranked stations that is sorted and recommended to the user.

Other fact that has inspired researches in music recommendation system is that music consumption carry on biased towards a few popular tracks. Therefore, recommender systems carry on suffering from the problem with the less popular items from the end of the Long Tail. A deep study about the Long Tail problem in music recommendation system is presented in [11]. The author not only defines and characterizes the Long Tail problem, but also proposes some solutions to address it. In [19], the authors propose a recommendation system for artists in the Long Tail by using the conventional item-based collaborative filtering technique. The system first identifies a suitable candidate pool of long tail artists to build the recommendation model. When a user requests recommendations, the model generates a candidate set with the most similar artists within the pool. Then, each candidate is scored, according to its similarity, and the highest scored candidates are recommended to the user. In a more recent paper [15], the Long Tail problem is addressed by using a method for sharing information across different items (music tracks) of the same taxonomy, which mitigates the problem of predicting music with insufficient data. In order to recommend music tracks, the recommendation system, which is based on matrix factorization, incorporates a rich bias model with terms that capture information from the taxonomy of music tracks and different temporal dynamics of music ratings.

3 Different Modalities for Item Similarities

Item-based recommender systems exploit similarity among items [22]. The system looks into the set of items that users have rated and computes the similarity between pairs of items, generating a matrix representing the similarities between all the pairs of items, according to a similarity measure. An abstract representation of a similarity matrix is shown below. Here, each item i can be, for example, a music track.

	i_1	i_2	$\cdots$	i_q
i_1	1	$sim(i_1, i_2)$	$\cdots$	$sim(i_1, i_q)$
i_2	$sim(i_2, i_1)$	1	$\cdots$	$sim(i_2, i_q)$
$\cdots$	$\cdots$	$\cdots$	1	$\cdots$
i_q	$sim(i_q, i_1)$	$sim(i_q, i_2)$	$\cdots$	1

The effectiveness of an item-based recommender system depends on the method used to calculate the similarity among the items in the matrix. Thus, in the next sections we present three different methods to calculate the similarity among music tracks. These methods tap into two different types of data: usage-based data on the one hand, and content-based data (both tags and audio features) on the other hand.

3.1 Usage-based Similarity

The simplest form of usage data is a pair $< user, item >$ meaning that *user* had a positive interaction with *item*. Examples are: “user viewed a document from a collection”, “user listened to a track”, “user liked a film”, “user bought a book”, etc. The positive nature of the interaction is often inferred from behavior. In the case of this work, we have access to playlists, which are collections of music tracks created and organized by individual users. The fact that a user adds a track to a playlist is regarded as a preference. Therefore a $< user, item >$ pair means, in our case, that a particular user added a particular track (item) to his playlist and, ergo, likes this music. Usage data such as this is a particular case of preference data where each user rates some items on a given scale (e.g., 1 to 5). In this case, we have a binary scale (i.e., likes / does not like).

To compute the similarity between pairs of music tracks from usage data, for example, m_1 and m_2, we first identify the users who have included the tracks in their playlists. Then, we compute the similarity $sim(m_1, m_2)$ between m_1 and m_2. Each track m_i can be seen as a binary vector $\overrightarrow{m_i}$ with as many positions as the number of users. In each position, there is a 1 if the track is in the playlist of the corresponding user and a 0 otherwise. In [22] the authors present three methods to measure similarity between pairs of items: cosine angle, Pearson's correlation and adjusted cosine angle. In this paper, we use the cosine angle, defined as

$$sim(m_1, m_2) = cos(\overrightarrow{m_1}, \overrightarrow{m_2}) = \frac{\overrightarrow{m_1}.\overrightarrow{m_2}}{||\overrightarrow{m_1}|| * ||\overrightarrow{m_2}||}, \qquad (1)$$

where the operator “.” denotes the dot-product of the two vectors.

The values of $sim(m_1, m_2)$ range from -1 to 1. A value closer to 1 or -1 means that the music tracks, m_1 and m_2, are very similar or dissimilar, respectively. On the other hand, if the value of $sim(m_1, m_2)$ is close to 0 it means that there is no correlation between the two music tracks.

After calculating the similarity values between any two tracks, we are able to compute the degree (score) to which any given track is recommendable to any given user (represented by the set of tracks in the respective playlist). This process will be detailed in Section 4.1.

3.2 Tag-based Similarity

Social tags are free text labels introduced by users (usually non-experts) of any system to describe the content of a web or multimedia item. In music, social tags are assigned to items such as artists, playlists or music tracks [17]. In our particular case, tags describe the content of music tracks, and are typically words or short phrases related to genre, instrument and influence. For example, music tracks in our data are typically tagged with tags like *flute*, *guitar*, *folk*, *feminine voice*, *rock* or *Daft Punk*.

The combination of the annotations provided by hundreds or thousands of music users lead to the emergence of a body of domain-specific knowledge, usually referred to as "folksonomy". One way to exploit such knowledge is by looking at the correlations between tags.

In order to capture the tag correlation, an $M \times N$ matrix of tracks and tags is built, where M is the number of tracks and N the number of tags, e.g. see Figure 1. Matrix elements with values different than 0 mean that a given tag N_j has been used to annotate a given music track M_i. The rationale is that music tracks with similar tag annotations are more prone to be similar.

This technique, however, has limitations. First, the dimensions M and N can be extremely large, thus making the problem computationally expensive. And second, the matrix is usually very sparse. It is very unlikely that users will tag a music track with more than 100 tags. Moreover, many tags introduced by users are rarely used, whilst few others are very common. This phenomena, usually referred to as Long Tail distribution, is very common in social networks [2,10]

To overcome this problem, an information retrieval technique called Latent Semantic Analysis (LSA) [12] is used to analyze the inherent structure of the matrix. LSA assumes a latent semantic structure that lies underneath the randomness of word choice and spelling in noisy datasets [4]. Basically, LSA consists of two steps. In the first step, a projection of the original $M \times N$ space to a continuous space of concepts is performed, by using statistical or algebraic techniques, such as Singular Value Decomposition (SVD). Given the original sparse matrix, $\mathbf{M}$, the Singular Value Decompositon of $\mathbf{M}$ is computed as follows:

$$\mathbf{M} = U \Sigma V^*, \tag{2}$$

where U is an $M \times M$ unitary matrix of $\mathbf{M}$, Σ an $M \times N$ diagonal matrix whose diagonal entries are the singular values of M, and V^* represents the conjugate transpose of V, an $N \times N$ unitary matrix of $\mathbf{M}$ [14].

Given that the first singular values of a matrix tend to encompass most of the information from this matrix, the latter technique, additionally, allows us to reduce the dimensionality of the original matrix, by choosing a relatively small number of singular values (L), while still preserving the similarity structure among rows or columns. Finding the "right" number of dimensions, L, is not a trivial task. It depends on the applicability of the resulting vectors of "concepts". In this paper, we empirically chose a value of 50. Figure 1 depicts this process. Moreover, Information Retrieval literature [4, 20] states that, after raw data has been mapped into this latent semantic space, topic (in our case, music tracks) separability is improved.

The second step of LSA refers to the distance measure used for calculating the similarity between pairs of music tracks. The most prominent similarity distance in the literature is the cosine distance, defined as:

$$sim(t_1, t_2) = cos(\overrightarrow{t_1}, \overrightarrow{t_2}) = \frac{\overrightarrow{t_1} . \overrightarrow{t_2}}{||\overrightarrow{t_1}|| * ||\overrightarrow{t_2}||}, \tag{3}$$

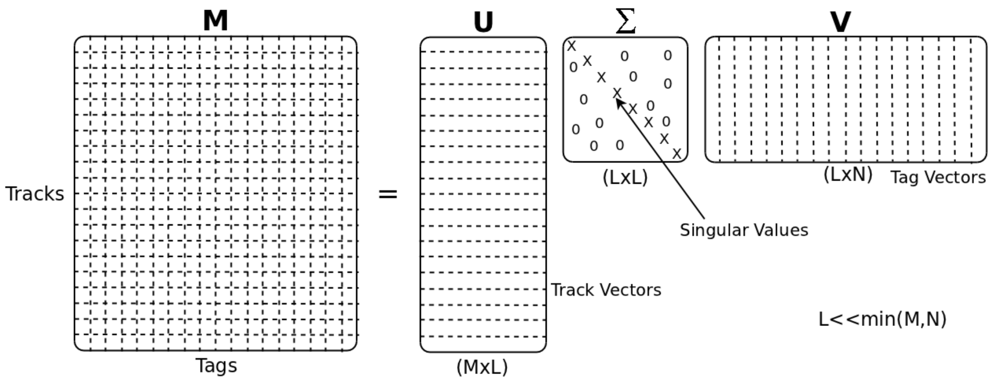
where $\overrightarrow{t_1}$ and $\overrightarrow{t_2}$ are binary vectors with all the tag concepts (i.e., they correspond to a row in matrix U in Figure 1). A value of 1 or 0 represents the presence or absence, respectively, of the tag concept for the given music track.

3.3 Audio-based Similarity

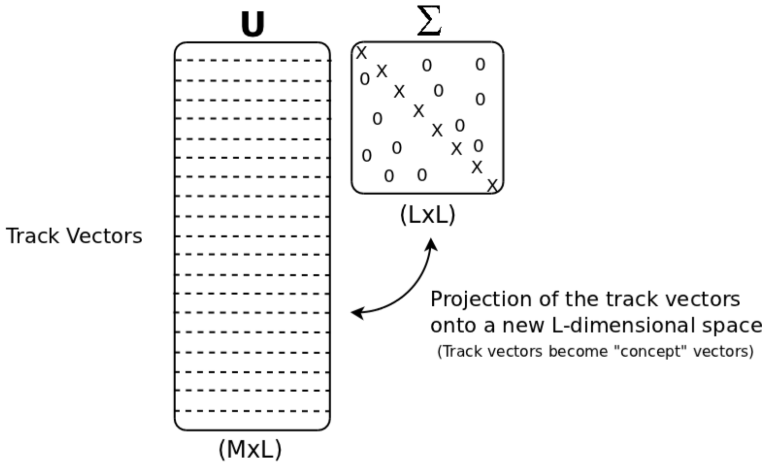
For this approach, we have used the free MARSYAS framework[10] to extract 16 audio features from 46ms frames of the audio signals with no overlap. The features are: the spectral centroid, rolloff frequency, spectral flux, and 13 MFCCs, including MFCC0 [28]. Features are aggregated in 1s texture windows, and then averaged over the whole file. Final features are the average and standard deviation. Although better audio features exist [25,24,6], we chose these features because of a relatively low dimensionality, low computational time, availability of code and the fact that they are widespread in the literature.

After extracting the audio features for each track, we calculate the similarity among the tracks. The similarity is calculated by the Euclidian distance through the 16 audio features. Here, we define the Euclidian distance between 2 tracks, a_1 and a_2, as follows

[10] `http://marsyas.info`



(a) Singular Value Decomposition (SVD) of the matrix.


U
Σ
Track Vectors
(LxL)
Projection of the track vectors
onto a new L-dimensional space
(Track vectors become "concept" vectors)
(MxL)

(b) Projection of the track vectors onto vectors of “concepts”.

Fig. 1 Illustration of the Singular Value Decomposition + Dimension Reduction used in LSA.

$$sim(a_1, a_2) = euclidian(\overrightarrow{a_1}, \overrightarrow{a_2}) = \sqrt{\sum_{f=1}^{16} (\overrightarrow{a_{1_f}} - \overrightarrow{a_{2_f}})^2}, \tag{4}$$

where $\overrightarrow{a_1}$ and $\overrightarrow{a_2}$ are vectors with the 16 audio features.

Note that contrarily to the cosine, where the similarity is directly proportional to the measure, with the Euclidian distance the similarity is inversely proportional to the measure, i.e., the lower the measure the higher the similarity.

4 Music Recommendation Based on Diverse Modalities

In this section, we show how the similarity methods presented in Section 3 can be used to recommend music tracks. We start by describing a usage-based and a content-based recommender system, which are used as benchmark systems in this paper. Then, we propose a hybrid recommender system that combines both usage and content.

Notice that we are only dealing with recommendation of individual music items, which a given user can then add in a playlist he/she edits manually. We do not address the problem of recommending playlists of music items [13].

4.1 Usage-based Recommendation

Usage-based recommendation is made on the basis of the similarity matrix between tracks described in Section 3.1. Given a user, his playlists are merged and the music tracks in it are used as seeds ($\mathcal{S}$) for the recommendations. The general procedure follows the Item-based Collaborative Filtering algorithm [22]. For each recommendable music track m (typically any track that is not included in the user's playlists) we fetch its k closest neighbors $N(m)$. These are the k tracks with maximum similarity to m. We then calculate the activation weight $ActWeight$ of each track m which is not already in the playlists of the user [18]. For that, we first identify the intersection $N(m) \cap \mathcal{S}$ between the neighbors of the track m and the seeds that characterize the user. Then, we sum the similarity values between each track s in this intersection and m. To normalize the activation weight, this sum is divided by the sum of similarities of m with each of its neighbors. Note that we exclude for recommendation tracks that are already in the playlist.

$$ActWeight(m) = \frac{\sum\limits_{s \in N(m) \cap \mathcal{S}} sim(m,s)}{\sum\limits_{n \in N(m)} sim(m,n)}. \quad (5)$$

Finally, we can recommend to a user the tracks with highest activation weight.

4.2 Content-based Recommendation

The content-based recommender system that we describe in this section combines tags and audio features to recommend music tracks. Arguably, there is no clear consensus in the literature about the definition of "content" vs. "context" of music items. In this paper, tags are considered descriptors of musical items' content rather than context, hence the combination of tags and audio features in the same recommendation engine. As for the usage data at hand, we choose to consider it as contextual data, as is usually done in the literature [23]. As proposed in [10], audio features should be good for low-level similarities (e.g., the main timbre of music tracks), while tags should be good supplements as they account for higher-level information that could not be reliably computed from audio (e.g., *feminine voice*).

The system starts by computing two item-item similarity matrices (Section 3). One matrix is computed using tags (Section 3.2) and the other one using audio features (Section 3.3). Once we have the two matrices, we can generate the recommendations. Given a seed music track, $s \in \mathcal{S}$, the system first fetches its k closest neighbors on each matrix, generating two lists of recommendable music tracks, i.e., one based on tags and the other based on audio features. Then, the system ranks each list separately, taking into account the similarities, and computes a final rank where the position is the sum of the two scores in every independent ranking. Finally, the k best ranked music tracks, according to the final ranking, are recommended.

4.3 Recommendation Combining Usage and Content

The recommendation strategy that combines Usage and Content data, referred to as **Mix**, is described in this section. Given a user playlist, we produce three lists of k recommendations. One obtained from usage data (R_u), one from tags (R_t) and the third from audio data (R_a). These three lists are sorted by inverse order of relevance of the recommendations. For each list, the recommended tracks are assigned ranks from k (top recommendation) to 1. The combined rank for each track is the average of the three ranks. For example, if a track m is the first recommendation in R_u, second in R_t and does not occur in R_a, and assuming $k = 100$, the combined rank is $(100 + 99 + 0)/3 = 66.33$.

4.4 Blacklisting the Recommendations

In our music recommendation application, we also have a source of negative information, called blacklist (B). When recommendations are shown to the user, he has the option of blacklisting a particular recommendation. This way, the blacklisted track is not shown again. Here, we exclude from each similarity matrix the tracks in the blacklist B of the seed user. Moreover, the blacklist information is used to calculate a global acceptance index $AccI$ of each track. This index captures the tendency of a track for being blacklisted and is calculated from the number of times a track is blacklisted $B(m)$ and the number of times it is included in a playlist $P(m)$. The value 1 means that the track is not included in any blacklist.

$$AccI(m) = 1 - \frac{B(m)}{B(m) + P(m) + 1}. \quad (6)$$

After calculating $AccI(m)$ it is multiplied by the final rank to obtain the score of the track. This will penalize tracks that are blacklisted by a large number of users.

5 Case Study

The recommendation strategies described in the previous section have been deployed on Palco Principal, a start-up company that holds a web site of Portuguese music since 2007. Besides music recommendations, the site also provides services like news, advertisements, social networking and an application for users to access the services of the site through their mobile phone.

During the period of our study, the site had about 76000 registered users (61223 listeners and 14777 artists/bands who uploaded music) and 61000 music tracks. From the tags available in the site, we used 373 tags which can be categorized into three classes: genre (e.g., hip hop), instrument (e.g., clarinet) and influence (e.g., Daft Punk). There is a minimum of 1, a mean of 3.52 and a maximum of 36 tags per track. Minimum of 1 is due to the fact that the web site imposes users to provide at least 1 tag for each track uploaded.

As already stated, most of the music tracks in the Palco Principal are underground, in other words, they are unknown/unpopular and rarely accessed/rated by the users. In fact, 79% of the music tracks have from 1 to 10 accesses, 19% have from 11 to 100 accesses, and only 2% of the music tracks have more than 100 accesses. Furthermore, only 19.7% of the artists on the Palco Principal web site also exist on the Last.fm web site. This means that we are in front of a *very long tail* problem [2].

In the site, each of the recommenders are used separately. When a user opens the page for managing playlists, the recommender is invoked in real time and the results are shown to the user (Figure 2). The user can then listen to recommended tracks, select tracks to add to his playlist (by clicking on the heart) or to his blacklist (by clicking on the cross). Notice that the recommender systems assume that users already have their own playlist beforehand. For a newly signed-up user, who does not have a playlist, the system recommends the top listened tracks on the Web site.

5.1 Evaluation Methodology

To compare the merits of the three recommenders (**Usage**, **Content** and **Mix**) we have performed an online evaluation [16] and followed the reactions of users during 22 weeks, between 10/20/2010 and 03/22/2011. These were real users with no knowledge of the evaluation in course. Each new user was assigned one of the three recommenders during this period. The assignment was decided by the remainder of the division of the user ID by 3. This way, we had a random assignment of users to each of the recommenders, and the same user would always get recommendations from the same source.

User activity has been recorded in two different ways. One was Google Analytics (GA) and the other was the site's internal data base (DB). In the case of GA, we have associated events to user actions of adding to playlist and adding to blacklist. In the case of DB, we have the playlist and blacklist tables in the data base. To be able to identify whether each track added to the playlists had been automatically recommended, we added a source field indicating which recommender had done the job. In the end, we have observed some non-significant differences in the values obtained from GA and DB, which comforted us in the quality of the data to be analyzed.

To measure the variation of the recommenders effects, we have divided the 22 weeks into 11 periods of 2 weeks. For each period we have measured the number of sessions (S), the number of additions to playlists (P) and the number of additions to blacklists (B) for each recommender.

From these three basic measures we have defined the following derived measures:

$$\mathit{Activity\ rate} = (P + B)/S, \quad (7)$$

$$\mathit{Absolute\ acceptance\ rate} = P/S, \quad (8)$$

$$\mathit{Relative\ acceptance\ rate} = P/(P + B). \quad (9)$$

Google Analytics also provides information about the number and frequency of users who return to the site. For a given period, $L(x)$ is the number of users who return x times to the site. Loyalty can then be measured in many different ways. We have tried to capture loyalty by counting users returning 3 times or more and using as reference the number of users who return less than three times. We call this measure Loyalty3 rate.

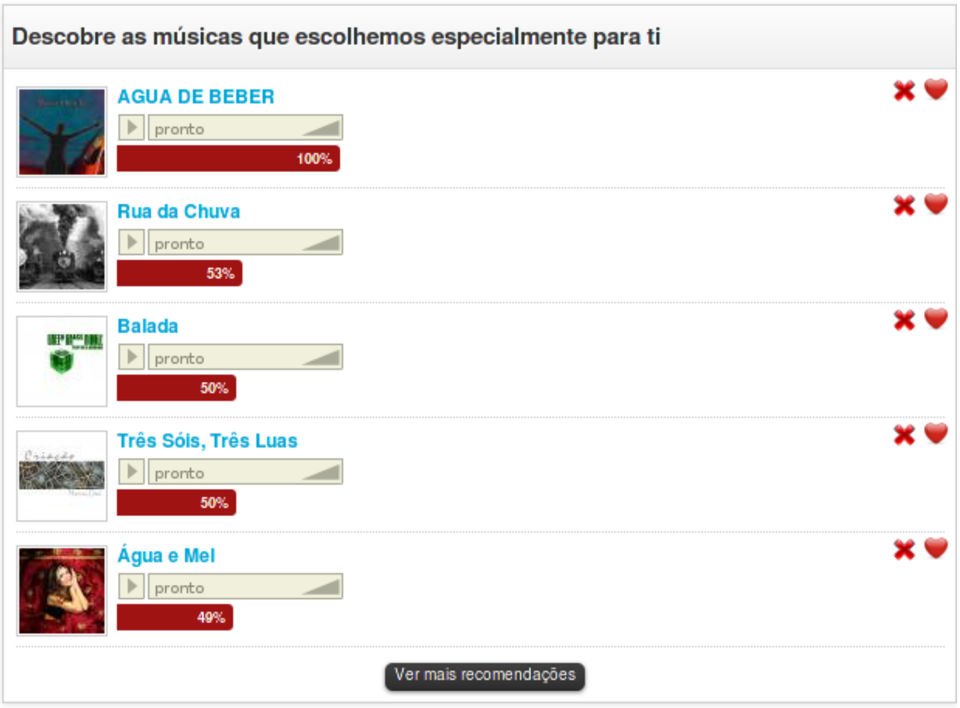
$$\mathit{Loyalty}3\ \mathit{rate} = \frac{\sum_{x \geq 3} L(x)}{L(1) + L(2)}. \quad (10)$$

For each measure, and each recommender, we have collected samples with values from the 11 periods. We then compare averages and standard deviations of the measures and perform two-tailed t-tests ($\alpha = 0.05$) to determine the significance of the differences. We also show graphically the evolution of the measures during the evaluation period.

5.2 Results

In this section we discuss the results obtained with our case study. During the evaluation period there were about 57000 sessions involving recommendations, where 1327 users made 3267 additions to playlists and 3123 additions to blacklists.

We start by analyzing the relative acceptance rate. In Table 1, **Mix** shows a slightly lower relative acceptance rate than **Content** and **Usage**. However, the



Descobre as músicas que escolhemos especialmente para ti

AGUA DE BEBER
pronto
100%

Rua da Chuva
pronto
53%

Balada
pronto
50%

Três Sóis, Três Luas
pronto
50%

Água e Mel
pronto
49%

Ver mais recomendações

Fig. 2 Recommendations as shown to the user.

differences are not significant (this is due to the high variability of all three recommenders with respect to the 11 periods of 2 weeks, Figure 3, as shown in the relatively high standard deviations), and all three recommenders have an average relative acceptance around 0.5. This can be understood as follows: in response to a given recommendation, the user is as likely to react with an addition to playlist (i.e., a positive reaction) than an addition to blacklist (i.e., a negative reaction). This appears to be true for all three recommenders.

Table 1 Relative acceptance rate. Differences between methods are not statistically significant (p-value > 0.05).

Systems	Mean	Std. Deviation	p-value
Mix	0.499	0.157	-
Content	0.512	0.164	0.848
Usage	0.600	0.125	0.162

This does not however mean that the three recommenders have a similar performance. Indeed, given a recommendation, a user can not only react by an addition to playlist or to blacklist, but also not react at all –which in our opinion is another negative reaction. As can be seen in Table 2, activity rate measure, our data shows that for the same number of recommendations, the **Mix** recommender results in more user activity than the other two. The system **Mix** has gains of 123% and 87% when compared to **Content** and **Usage**, respectively. In other words, it appears that users are more likely to react to recommendations when confronted with recommendations of **Mix** than those of the other two. This means that users will generate more additions to playlist, and more additions to blacklist, with **Mix** than with **Content** and **Usage**. This increased activity is very visible in Figure 4. For the period 6 (from 12/29/2010 to 01/11/2011), **Mix** is worse than **Content** and **Usage**. In all remaining periods, **Mix** always outperforms the other two systems.

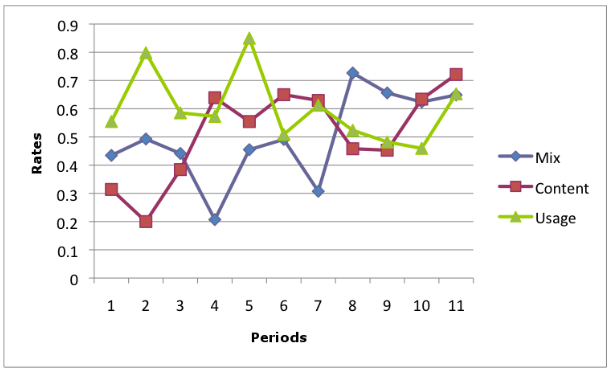


Fig. 3 Relative acceptance rate per period.

In Table 3, absolute acceptance rate, we can see that **Mix** is significantly better than **Content** and **Usage** systems. When compared to **Content**, **Mix** presents a gain of 119%. With respect to the **Usage**, it shows a gain of 50%. This means that users getting the **Mix**

Table 2 Activity rate. Values with (*) represent recommendation methods whose differences with Mix are statistically significant (p-value < 0.05).

Systems	Mean	Std. Deviation	p-value
Mix	**0.165**	0.061	-
Content	0.074 (*)	0.025	0.001
Usage	0.088 (*)	0.021	0.002

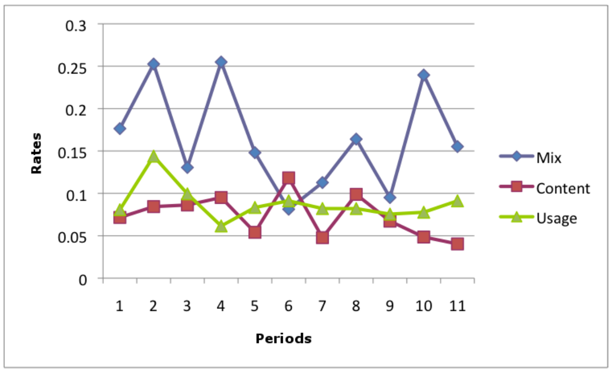


Fig. 4 Activity rate per period.

suggestions had a significant tendency for reacting more positively to recommendations. In Figure 5 we can see that the behavior of the **Mix** tends to be much better than the competitors with time. This may be due to a higher variety in recommendations motivating users to listen to more tracks and interact more.

Table 3 Absolute acceptance rate. Values with (*) represent recommendation methods whose differences with Mix are statistically significant (p-value < 0.05).

Systems	Mean	Std. Deviation	p-value
Mix	**0.081**	0.038	-
Content	0.037 (*)	0.018	0.013
Usage	0.054 (*)	0.023	0.049

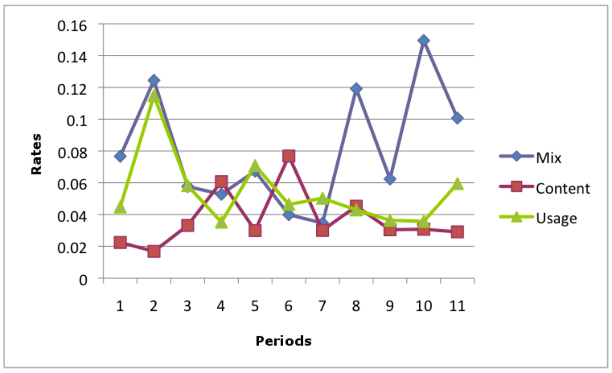


Fig. 5 Absolute acceptance rate per period.

We also computed the loyalty3 rate. This indicator shows the proportion of the number of users visiting the site three or more times with respect to the ones who return at most twice. In Table 4 we see that the **Mix** recommender is similar to **Content** but significantly better than **Usage**. There, the system **Mix** presents a gain of 16% when compared to **Usage**. In Figure 6 we can see that there is a higher difference in the beginning, but afterwards the three solutions tend to have similar results.

Table 4 Loyalty3 rate. Values with (*) represent recommendation methods whose differences with Mix are statistically significant (p-value < 0.05).

Systems	Mean	Std. Deviation	p-value
Mix	**1.880**	0.376	-
Content	1.870	0.171	0.867
Usage	1.620 (*)	0.196	0.044

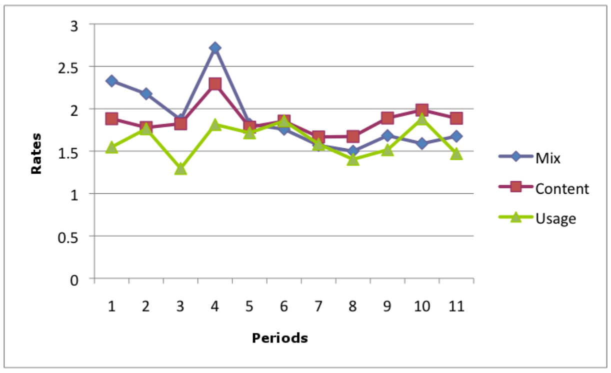


Fig. 6 Loyalty3 rate per period.

5.3 Relating Activity and Loyalty

One interesting question is how does the level of response to recommendations affect loyalty. To try to answer that question we have looked at the relation between each of the activity/acceptance measures with the loyalty3 measure. In Figure 7, we can see that the rise in activity tends to increase the loyalty of the users in the case of the **Mix** recommender (with a Pearson correlation of 0.56) and to a lesser extent in the case of the **Content** recommender (0.12 correlation). The **Usage** recommender shows a practically zero correlation between activity rate and loyalty. We can see that **Mix** shows a wider dispersion of values. One tentative explanation for these observations is that recommenders bring more activity and involvement and generate more loyalty. However, the relation between activity rate and loyalty is not directly observable here, since we are not considering the activity of loyal users only, but comparing activity and loyalty for all users and for each two weeks period. The wider spread of loyalty3 for the **Mix** recommender suggests that its recommendations may be more controversial.

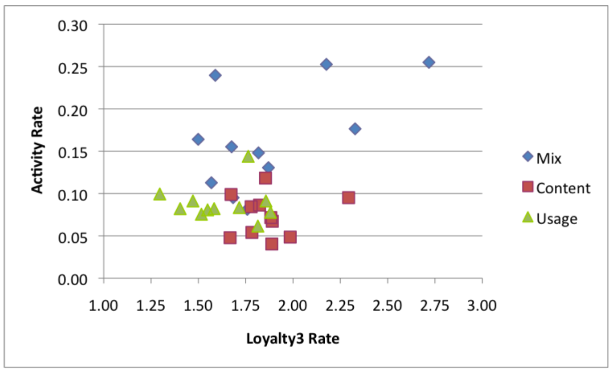


Fig. 7 Activity rate versus loyalty3 rate.

Whereas activity seems to be positively related with loyalty, acceptance (absolute and relative) does not, in the case of the **Mix** recommender (Figures 8 and 9). This may suggest that returning users, despite being more active, tend to reject more **Mix** recommendations. The **Content** recommender shows positive correlation between acceptance and loyalty (0.36 and 0.30 for absolute acceptance rate and relative acceptance rate, respectively). This recommender has the lowest average activity and acceptance rates but it is the one that shows a better relation with the loyalty measure. This may indicate that it is able to generate catchy recommendations for returning users.

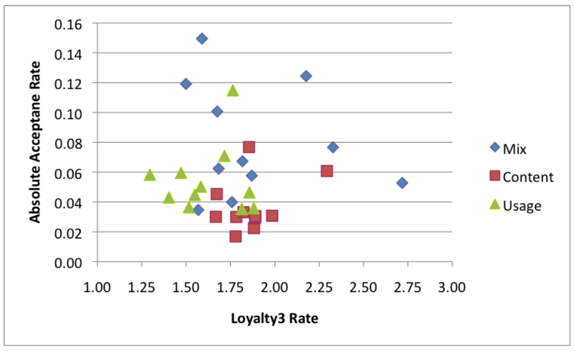


Fig. 8 Absolute acceptance rate versus loyalty3 rate.

6 Discussion

In general, our case study shows that **Mix** generates more activity and at least the same amount of positive

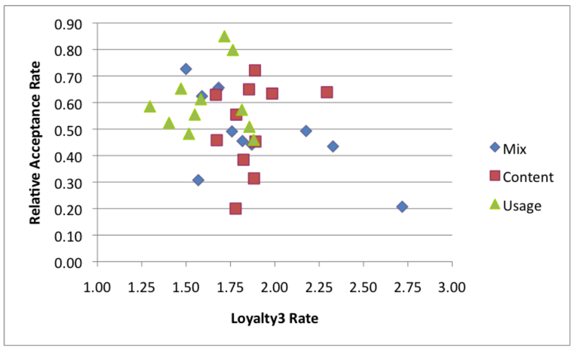


Fig. 9 Relative acceptance rate versus loyalty3 rate.

responses of **Content** and **Usage** (or more, depending on the evaluation measure). This may be due to a higher variety in recommendations generated by **Mix**, motivating users to listen to more tracks and interact more with them. We see in Figures 6 and 7 that **Mix** has good results in terms of promoting user loyalty. All in all, this makes us argue that **Mix** is a better option to music recommendation than the other two recommenders.

We should note, however, that **Mix** presents a poor performance in the mid periods. This is very visible in Figures 3, 4 and 5. This may be due to the conditions under which this study has been conducted. During the evaluation, for operational reasons, there were no updates in the recommender models. This may have caused some saturation in the recommendations to the users, which might have lead to a general lower response rate during the mid periods by the **Mix** recommender. Consequently, we believe that the most important point of future work is related to an adaptation over time of the recommender models.

We are currently developing a monitoring tool for continuously collecting and analyzing the activity of the recommenders of the site. This will allow the owners of the site to keep an eye on the impact of the recommenders. On the other hand, it will give us more reliable data and will enable us to look into other facets of the recommendations, such as variety and sensitivity to the order. With that information we will be able to better understand what makes users more active, as well as to design recommenders that may have different mixes, depending on the profile of the user.

7 Summary

In this paper we have proposed and evaluated a music recommender system that combines usage and content data. Evaluation was conducted online, with real users, on a commercial music web site, during 22 weeks. Our work is end-to-end and included the development of the recommenders, their deployment and maintenance and all the evaluation setup. The users were dynamically divided in three groups and we have collected data, using Google Analytics and the site's internal data base, on how users responded to the recommendations shown. We have proposed some measures for comparing the performances of the solutions. We concluded that **Mix** is, overall, a better option to provide music recommendation than the other two systems. **Mix** is currently the core recommendation engine on `http://www.palcoprincipal.com`. Future work relates to adaptation over time of recommendation models and more detailed monitoring of user data.

Acknowledgements This work is supported by the QREN-AdI Palco3.0/3121 PONORTE; the European Commission, FP7 (Seventh Framework Programme), ICT-2011.1.5 Networked Media and Search Systems, grant agreement No 2877 11; and the ERDF through the Programme COMPETE and by the Portuguese Government through FCT - Foundation for Science and Technology, project ref. PTDC/EAT-MMU/1122 55/2009 - (FCOM-01-0124-FEDER-014732).

References

1. N. Aizenberg, Y. Koren, and O. Somekh. Build your own music recommender by modeling internet radio streams. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1–10, New York, NY, USA, 2012. ACM.
2. C. Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion, 2006.
3. L. Barrington, R. Oda, and G. Lanckriet. Smarter than genius? human evaluation of music recommender systems. In *International Symposium on Music Information Retrieval (ISMIR 2009)*, Japan, 2009.
4. J. Bellegarda. Latent semantic mapping [information retrieval]. *Signal Processing Magazine, IEEE*, 22(5):70–80, 2005.
5. D. Bogdanov, M. Haro, F. Fuhrmann, E. Gomez, and P. Herrera. Content-based music recommendation based on user preference examples. In *RecSys'10 Workshop on Music Recommendation and Discovery (Womrad 2010)*, Spain, 2010.
6. D. Bogdanov, J. Serra, N. Wack, P. Herrera, and X. Serra. Unifying low-level and high-level music similarity measures. *IEEE Transactions on Multimedia*, 13(4):687–701, 2011.
7. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370, 2002.
8. P. Cano, M. Koppenberger, and N. Wack. Content-based music audio recommendation. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*, 2005.
9. P. Cano, M. Koppenberger, and N. Wack. An industrial strength content-based music recommendation system. In *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'05)*, 2005.
10. O. Celma. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.
11. O. Celma. *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.
12. S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41:391–407, 1990.
13. B. Fields and P. Lamere. ISMIR 2010: Finding a path through the jukebox - the playlist tutorial, 2010. `http://ismir2010.ismir.net/proceedings/tutorial_4_Lamere-Field.pdf`.
14. G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, pages 205–224, 1965.
15. N. Koenigstein, G. Dror, and Y. Koren. Yahoo! music recommendations: Modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, pages 165–172, New York, NY, USA, 2011. ACM.
16. R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1):140–181, 2009.
17. P. Lamere. Social tagging and music information retrieval. *Journal of New Music Research*, 37(2):101 – 114, 2008.
18. L. Lemos, A. M. Jorge, and J. P. Leal. Deployment and evaluation of a usage based collaborative filtering recommendation system with blacklists. In *Third International Workshop on Web and Text Intelligence (WTI 2010)*, Brazil, 2010.
19. M. Levy and K. Bosteels. Music recommendation and the long tail. In *Proceedings of the Workshop on Music Recommendation and Discovery (WOMRAD)*, pages 55–58, 2010.
20. C. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent semantic indexing: A probabilistic analysis. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168. ACM, 1998.
21. P. Ruppert, R. Hart, and S. Evans. The 2007 digital music survey, Entertainment Media Research, 2007. `http://www.slideshare.net/patsch/emr-digital-music-survey-2007`.
22. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Tenth International Conference on World Wide Web (WWW 2001)*, pages 285–295, Hong Kong, 2001.
23. M. Schedl and P. Knees. Context-based music similarity estimation. In *Proceedings of the 3rd International Workshop on Learning the Semantics of Audio Signals (LSAS 2009)*, 2009.
24. D. Schnitzer, A. Flexer, M. Schedl, and G. Widmer. Using mutual proximity to improve content-based audio similarity. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, 2011.
25. K. Seyerlehner, M. Schedl, T. Pohle, and P. Knees. Using block-level features for genre classification, tag classification and music similarity estimation. In *PProceedings of the 6th Annual Music Information Retrieval Evaluation eXchange (MIREX-10)*, 2010.

26. U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of SIGCHI Conference on Human Factors in Computing Systems*, pages 210–217, 1995.
27. M. Slaney. Web-scale multimedia analysis: Does content matter? *IEEE Multimedia*, 18:12–15, 2011.
28. G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293 – 302, 2002.
29. K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In *Proceedings of the 7th International Conference on Music Information Retrieval*, page 296301, 2006.
30. V. Zaharchuk, D. I. Ignatov, A. Konstantinov, and S. Nikolenko. A new recommender system for the interactive radio network fmhost. In *Proceedings of the International Workshop on Experimental Economics and Machine Learning (EEML)*, pages 72–85, 2012.