CENTRO UNIVERSITARIO SENAC

Renan Valieris Bueno de Almeida
Ricardo Hiroyuki Aoyagi

Recomendação de músicas por análise musical

São Paulo
2010

RENAN VALIERIS BUENO DE ALMEIDA
RICARDO HIROYUKI AOYAGI

Recomendação de músicas por análise musical

Trabalho de Conclusão de Curso apresentado ao Centro Universitário SENAC - Campus Santo Amaro, como exigência parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador Prof. Fabrício Jailson Barth

São Paulo
2010

RENAN VALIERIS BUENO DE ALMEIDA
RICARDO HIROYUKI AOYAGI

Recomendação de músicas por análise musical.

Trabalho de Conclusão de Curso apresentado ao Centro Universitário SENAC - Campus Santo Amaro, como exigência parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador Prof. Fabrício Jailson Barth

A banca examinadora dos Trabalhos de Conclusão em sessão pública realizada em __/__/____, considerou os candidatos:

1)Examinador(a)

2)Examinador(a)

3)Presidente

Dedicamos este trabalho aos nossos neurônios queimados!

Agradecimentos

Aos nossos pais
Aos nossos professores
A nós mesmos

It is good to have an end to journey toward; but it is the journey that matters, in the end.

Ursula Le Guin.

RESUMO

A recomendação de músicas é um trabalho complexo, visto que os serviços atuais falham na recomendação de novos artistas, pelo fato da recomendação ser feita com base na opinião de outros usuários. Este trabalho tem como objetivo a criação de um software de recomendação de músicas por análise musical, que significa interpretar padrões sonoros presentes em músicas e, a partir disso, efetuar uma recomendação que o usuário sinta similaridade. A recomendação deve levar em consideração um conjunto de músicas, que pode ser de um único estilo musical, como pode ser bem diversificado. Para resolver isso, os seguintes métodos foram propostos: proximidade, a clusterização randômica e a clusterização com proximidade. Pelos resultados analisados, é possível recomendar de forma eficiente conjuntos de músicas para os diversos tipos de gostos de um usuário, sendo a proximidade o melhor método utilizado.

Palavras-Chave: recomendação; software; música; análise musical; perfil do usuário.

ABSTRACT

The music recommendation is a complex task. The current services fail in recommending new artists, because the recommendation is based on the opinion of other users. This work aims to create a music recommendation software using musical analysis, which means to interpret existing sound patterns in music, as appropriate, make a recommendation that the user feels similar. The recommendation should take into consideration a set of songs that can be a single musical style, as well as diversified. To this end, we propose the following methods: proximity, random clustering and clustering with proximity. From the results analyzed, it is possible to recommend efficiently sets of music for different kinds of tastes of a user, using the proximity as the best method.

Keywords: recommendation; software; music; musical analysis; user profile.

LISTA DE ILUSTRAÇÕES

Ilustração 1 - Fluxo Básico do Protótipo.....18
Ilustração 2 - Tela Inicial do Sistema 21
Ilustração 3 - Perfil do Usuário.....21
Ilustração 4 - Organização e Relacionamentos das Tabelas no MySQL 22
Gráfico 1 - Comparação entre os algoritmos - Playlist 1.....28
Gráfico 2 - Comparação entre os algoritmos - Playlist 2.....31
Gráfico 3 - Comparação entre os algoritmos - Playlist 3.....34
Gráfico 4 - Comparação entre os algoritmos - Playlist 4.....37
Gráfico 5 - Comparação entre os algoritmos - Playlist 5.....39
Gráfico 6 - Consolidado dois algoritmos 40

LISTA DE TABELAS

1 – Recomendações feitas a partir da Playlist 1 27
2 – Recomendações feitas a partir da Playlist 2 30
3 – Recomendações feitas a partir da Playlist 3 33
4 – Recomendações feitas a partir da Playlist 4 36
5 – Recomendações feitas a partir da Playlist 5 38

SUMÁRIO

1. INTRODUÇÃO 12
2. FUNDAMENTAÇÃO TEÓRICA 13
2.1 DISTANCIA EUCLIDIANA 14
2.2 CLUSTERIZAÇÃO 15
3. PROPOSTA E IMPLEMENTAÇÃO 16
3.1 PROTÓTIPO 16
3.2 MELHORIA DO PROTÓTIPO 19
3.3 CRIAÇÃO E MANUTENÇÃO DO PERFIL DO USUÁRIO 20
3.4 UTILIZAÇÃO DO PERFIL DO USUÁRIO 23
4. MÉTODO DE AVALIAÇÃO 25
5. RESULTADOS E DISCUSSÕES 26
5.1 TESTES COM PLAYLISTS 26
5.2 RESUMO DOS RESULTADOS 40
6. CONCLUSÕES E CONSIDERAÇÕES FINAIS 41
7. SUGESTÕES PARA TRABALHOS FUTUROS 42
8. REFERÊNCIAS 43

1. INTRODUÇÃO

O rápido crescimento da Internet e de dispositivos móveis possibilitou um avanço na área da digitalização de músicas. Muitos usuários da Internet aproveitaram para realizar *downloads* e *uploads* de músicas do seu gosto pessoal, tornando a Internet um grande depósito de arquivos com formato MP3. Esse foi o início do surgimento de *softwares* recomendadores de músicas, pois os usuários tinham dificuldades em organizar, acessar e descobrir novas músicas (MAGNO e SABLE, 2008).

Estes recomendadores, em sua maioria web, fazem a recomendação baseado em opiniões dos próprios usuários. Este tipo de recomendação não é ideal, pois as opiniões dos usuários podem ser controversas, não seguindo um padrão. Outro fator que também prova isso é a rejeição de novos artistas, que, por serem novos, podem ser rejeitados por usuários que não conhecem suas músicas. Esses recomendadores utilizam uma técnica chamada de filtro colaborativo, onde a recomendação é feita com base nas *playlists*, escolhas e informações de gosto do usuário. Como exemplo disso, temos o Last.fm(2010).

Como então recomendar uma música a uma pessoa de forma automática, sem depender de opiniões e que saiba sugerir novos artistas? Uma pessoa que ouve determinado tipo de música sabe recomendar músicas para outra pessoa que ouve o mesmo tipo de música. Isso ocorre porque ambos escutam um estilo musical, sabendo os instrumentos e ritmos do estilo que gostam.

A análise musical transforma este trabalho humano para um trabalho de máquina. Ela consiste em identificar padrões sonoros existentes no conteúdo da música, podendo assim identificar similaridades entre duas ou mais músicas. O trabalho é feito de forma que haja um processamento na música, obtendo suas faixas espectrais e fazendo a extração de suas características (ritmo ou batida, por exemplo), fazendo com que seja possível obter dados numéricos das músicas. Com os números, é possível realizar a comparação entre músicas, determinando qual se assemelha mais a outra música.

2. FUNDAMENTAÇÃO TEÓRICA

Alguns trabalhos e pesquisas já foram feitos neste tipo de recomendação por análise musical. Em Magno e Sable (2008), a idéia é utilizar a recomendação musical pela extração e análise de características em músicas, comparando os resultados obtidos com outros tipos de recomendação, que utilizam de esforço humano. Utilizando algoritmos de extração de atributos, eles mapeiam músicas e fazem as comparações para que então possam sugerir músicas similares. Esses atributos possuem relação com as caracteríticas de uma música, como por exemplo o Timbre, que é a capacidade de diferenciar o som de diferentes instrumentos tocando a mesma nota. Para verificar a proximidade de uma música com outra, verificaram alguns métodos como, por exemplo, Distância Euclidiana. Os testes indicaram que a recomendação não foi superior aos sistemas existentes hoje, mas foi satisfatória, sendo ligeiramente imprecisa. Sendo assim, a recomendação por analise de música é viável, pois ela agrega o fato de poder identificar novas músicas e novos artistas, excluindo o fator humano.

Uitdenbogerd e Schyndel (2002) buscam entender como funciona a avaliação de um *software* recomendador de músicas, apresentando diversos fatores que influenciam nisso, seja eletronicamente ou não. Isso é importante para verificar o que é relevante na recomendação feita e o que influencia o usuário a gostar ou não do sistema. Com bases em diversas referências, concluem que a satisfação do usuário de um recomendador depende de fatores do próprio usuário, como idades, etnia e personalidade. A recomendação será melhor com base em questionários feitos ao usuário para atribuir novos dados para um filtro colaborativo, mesmo sendo um recomendador por análise musical.

Barrington, Oda e Lanckriet (2009) analisaram o Genius, sistema de recomendação do iTunes da Apple, que usa de filtro colaborativo e é muito eficiente, porém carece quando falta esse tipo de informação do usuário, que são os casos de novos artistas, onde as pessoas não tem muitas informações a respeito.

Beth (2004) procedeu com a recomendação de músicas utilizando quatro métodos diferentes, analisando os resultados e concluindo sobre este tipo de recomendação automática. Os métodos utilizados foram: clusterização (método *K-means*) e três métodos envolvendo cálculo de distância: média, mediana e

mínima. Os resultados mais satisfatórios foram da distância mínima e da mediana.

Um dos componentes pesquisados utilizado foi o *framework open source* Marsyas, desenvolvido por Tzanetakis e Cook (2000), que serve justamente para o propósito do projeto, identificando informações e características de áudio, que será explicado no capítulo 3.1.

Utilizaremos duas técnicas para o tratamento da informação, que resultará na recomendação da música: distância euclidiana e clusterização.

2.1 Distância Euclidiana

Na distância euclidiana, buscamos as N músicas mais próximas da música. Sendo assim, a distância D é representada pela equação 1:

$$\mathbf{D} = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}. \quad (1)$$

Onde:

P representa a Música 1, com *i* atributos.

Q representa a Música 2, com *i* atributos.

D, portanto, é a distância de Música 1 à Música 2.

Este técnica foi implementada como uma função no banco de dados, para se extrair as N músicas mais próximas de musica selecionada. É feito uma instrução do tipo SELECT contra todo o banco, efetivamente é feito um calculo da distância da musica selecionada contra todas as outras do banco, ordenado por distância e extraido as N musicas de menor distância.

Este cálculo será utilizado na recomendação por proximidade.

2.2 Clusterização

Na clusterização, as músicas são separadas em grupos, de forma que atendam determinados critérios de similaridade, baseados em seus próprios atributos.

Para isso, foi selecionado o algoritmo EM (Expectation Maximization), criado por Dempster, Laird e Rubin (1977), porque ele busca inferir variáveis ocultas a partir das variáveis disponíveis, que significa não precisar saber quais atributos são necessários para uma separação separatória.

Segundo Batzoglou e Do (2008), o algoritmo possui 2 fases. Inicialmente , o algoritmo começa com probabilidades iniciais para suas variáveis. Na fase de expectação, um modelo de probabilidade é aplicado nas variaveis utilizando as probabilidades iniciais. Na fase de maximização, as probabilidades são calculadas a partir do modelo de probabilidade. Essas fases são repetidas até as probabilidades das variáveis convergirem num valor.

O EM foi utilizado pelo *software open source* Weka, criado por Holmes, Donkin e Witten (1994).

O Weka é um *software* para classificação de dados, *data mining* e aprendizado de máquina. Como possuímos uma grande quantidade de músicas e muitos atributos para cada música, o uso do Weka auxilia o trabalho de classificação.

O resultado esperado é a divisão da base de músicas em grupos, de forma que tenham semelhança entre si e facilite uma recomendação por grupos. Por exemplo: se uma playlist possuir 5 músicas do grupo A e 5 músicas do grupo B, então uma boa recomendação serão músicas que pertencem a ambos os grupos.

O funcionamento do Weka com o EM está explicado no capítulo 3.3.

3. PROPOSTA E IMPLEMENTAÇÃO

O objetivo geral é construir um sistema de recomendação de músicas baseado em análise musical. Os objetivos específicos são:

- Analisar músicas e armazenar suas características em um banco de dados;
- Sugerir músicas baseado nas características extraídas;
- Verificar se a sugestão agradou o usuário;
- Construir um perfil de usuário;
- Aceitar *playlists* (conjunto de músicas) como entrada do sistema.

A implementação do sistema conta com duas fases: protótipo e melhoria do protótipo.

3.1 Protótipo

No início, a busca pelas músicas foi concluída com a ajuda de voluntários da faculdade. Com uma quantidade de aproximadamente 3000 de músicas, iniciou-se o processo de montagem de um protótipo. Foi criado um banco de dados para o armazenamento das informações de cada música. O banco possuía apenas a tabela data, que armazena os atributos de cada música.

Para extrair os atributos das músicas, foi criado um programa escrito em *Perl* chamado *extractor*, que utiliza o programa *bextract* do Marsyas (2000). O *bextract* é um dos principais programas da biblioteca Marsyas, pois ele consegue extrair por completo atributos de sinais de áudio e tem suporte a diversos tipos de formatos de arquivos. O principal atributo extraído é o MFCC.

O MFCC foi criado por Paul Mermelstein para identificação de características em sistemas de reconhecimento de voz, mas que foi utilizado com sucesso na área de análise musical (MAGNO e SABLE, 2008). O algoritmo extrai os coeficientes através do Cepstrum de uma janela de um trecho de áudio, utilizando a escala Mel de freqüências no lugar da comumente utilizada escala linear. O Cepstrum é o cálculo do espectro sobre um espectro, para medir a variação do próprio espectro.

Para os valores obtidos, foi feita uma amostragem de 512 bytes por janela. Com os novos valores, foram calculadas as médias e desvios padrões. Alguns testes foram feitos para verificar se o programa e o algoritmo funcionavam.

Verificou-se que também era necessário armazenar os meta-dados de cada música, para que o usuário possa saber as informações do que está sendo recomendado de forma clara. Uma nova tabela foi criada no banco de dados. A tabela metadata, armazena valores para cada música, como Artista, Título da Música, Álbum, Compositor e Gênero. Esta tabela está relacionada com a tabela data. Os meta-dados da música podem conter informações incorretas ou imprecisas. Foi implementado no *extractor* uma conexão com o site MusicBrainz (2010). Este site fornece meta-dados de forma gratuita, sendo necessário apenas enviar um trecho da música. Ele devolve as informações meta-dados da música. Caso o MusicBrainz não retorne nada, o programa utiliza a própria informação do arquivo de áudio.

Além do MFCC, o programa *extractor* também extrai os seguintes atributos:

- Chroma: é espectro de cada semitom da música, são 12 no total. O espectro é o conjunto de todas as ondas que compõem o som. Então o Chroma, mede o som das notas de uma música.
- Zero Crossings: mede quantas vezes a onda passou pelo zero. É um método usado em processamento de voz para estimar a freqüência fundamental da voz.
- Flux e Spectral Centroid: usado para determinar o timbre do áudio. O timbre é a característica sonora que nos permite distinguir uma mesma nota tocada por instrumentos diferentes.

Os atributos Zero Crossings, Centroid e Flux são divididos em canais (*Left* e *Right*) e para todos é calculado a média e desvio padrão, totalizando 62 variáveis.

Para o protótipo, foi criado uma aplicação *web* em *PHP/Perl/MySQL*. Foi desenvolvido um sistema de *upload* de músicas, onde a aplicação processa as músicas submetidas, classificando-as por similaridade em relação a outras músicas já presentes no banco de dados. As músicas processadas devem ser formato MP3. Uma vez processada, os dados da música passam a fazer parte do banco e serão utilizados nos próximos processamentos. Este protótipo tem como busca atender os seguinte objetivos mencionados:

- Analisar músicas e armazenar suas características em um banco de dados.
- Sugerir músicas baseado nas características extraídas.

A Ilustração 1 representa o processo de upload, processamento e atualização do banco de dados.

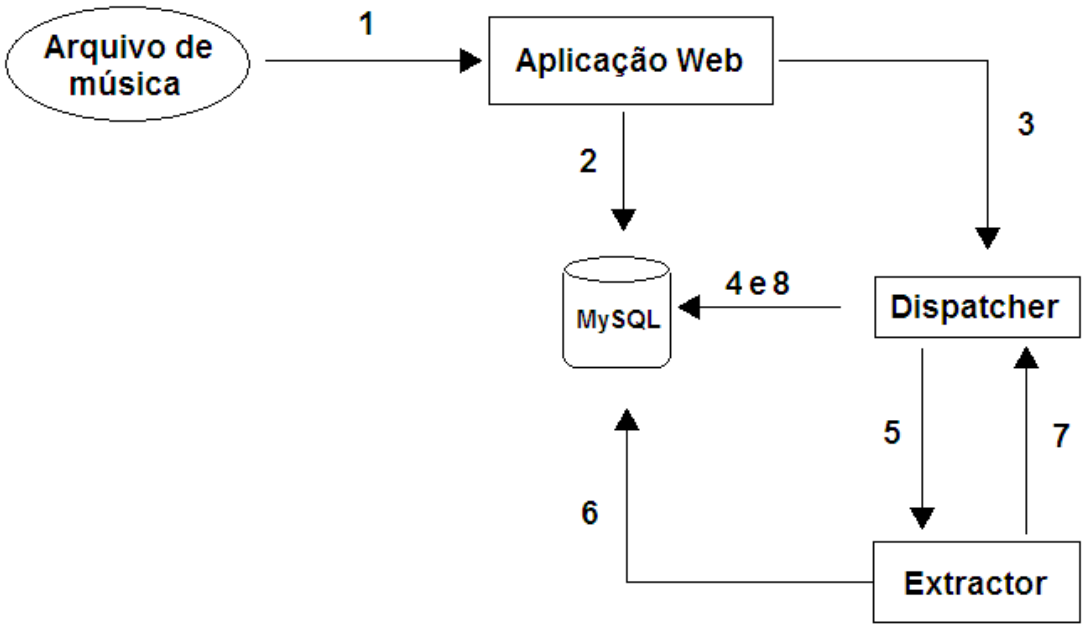

Ilustração 1- Fluxo básico do protótipo – Upload de Músicas

Quando um usuário faz o upload de uma música (1), a aplicação Web coloca ela em uma fila no banco de dados (2). Esta aplicação envia informação para um programa *Perl*, chamado de Dispatcher (3). Este programa, que executa em *background,* lê essa fila (4) e cria um processo separado para o processamento de cada música (5). Cada processo por sua vez extrai os atributos utilizando o programa *extractor* e insere as informações no banco de dados (6). Estas informações são os atributos em formato numérico, de forma a ser possível efetuar cálculos posteriormente. Quando o *extractor* termina sua execução, o processo retorna para o Dispatcher (7), que atualiza o banco de dados, informando que a música já foi processada, com êxito ou com erro (8). Na aplicação *web* tem uma página que o usuário poderá ver o status das músicas

submetidas. Nesta página, as músicas que já foram processadas com sucesso terão um link para a página de classificação, onde serão informadas as músicas mais similares em ordem crescente.

Esta classificação resulta do cálculo da distância Euclidiana dos atributos em relação à música selecionada. Ou seja, retorna a música mais próxima da música selecionada. Esse cálculo é uma *function* criada no *MySQL*.

O protótipo foi batizado de Armonye, uma palavra do old english que significa harmonia.

3.2 Melhoria do Protótipo

O sistema final é a melhoria contínua do protótipo, até que satisfaça todos os objetivos traçados. Sendo assim, restam alguns objetivos específicos a serem implementados:

- Verificar se a sugestão agradou o usuário.
- Construir um perfil de usuário;
- Aceitar *playlists* como entrada de dados.

Para que facilitasse futuros testes e conclusões sobre o sistema, é decidido que a base de dados deve aumentar. Desta forma, o sistema possui um grande conjunto de músicas, abrangendo diversos estilos e, conseqüentemente, diversos tipos de gostos musicais.

Após um trabalho de busca de músicas de usuários, o sistema conta com aproximadamente 14000 músicas. Algumas músicas foram separadas como playlists para simular os gostos dos usuários. A maior parte das músicas é do tipo Rock, seguido pelos gêneros de Black e Pop. A qualidade do áudio estava boa para a maior parte das músicas, mas a meta-data das mesmas estava com muitos problemas, por exemplo álbuns incorretos e gêneros que não existem ou desconhecidos.

Então o foco inicial de melhoria do protótipo é a aceitação de playlists como entrada de dados, pois desta forma é possível carregar muitas músicas em pouco tempo. Foi desenvolvido um programa PERL chamado *mkdispatch*, que reconhece um diretório com músicas e, a partir dele, monta a lista para o *dispatcher*. Este programa executa somente por linha de código, pelo shell do Linux.

Para o tratamento de músicas duplicadas, é proposta a seguinte regra: duas músicas serão iguais se, e somente se, tiverem os meta-dados Artista, Título e Álbum iguais. Caso seja encontrado duplicidade de músicas, a música será mantida no banco de dados e a música que está na fila de processamento será descartada.

3.3 Criação e Manutenção do Perfil de Usuário

A abordagem inicial do perfil do usuário foi levantar todas as músicas de um usuário e tirar uma média de todos os atributos de todas as músicas e então recomendar músicas próximas do valor obtido. Isso é ideal, caso as músicas sejam semelhantes entre si. Mas caso um usuário tenha muitos gostos musicais, como foi percebido pelas playlists de teste, sentiu-se a necessidade do sistema saber tratar esse problema: recomendar músicas de forma eficiente para pessoas com muitos gostos musicais. Dada uma playlist bem diversificada, a recomendação deve resultar numa playlist diversificada, seguindo critérios para as músicas inseridas.

O perfil do usuário é uma opção onde o usuário insere músicas no sistema e a recomendação é feita com base em todas as músicas que ele inseriu ou adicionou ao ser perfil, por meio de consulta no ambiente web.

Na tela inicial do sistema, o usuário coloca seu nome e confirma. Nesse momento, o Perfil é criado. A Ilustração 2 mostra a tela inicial.

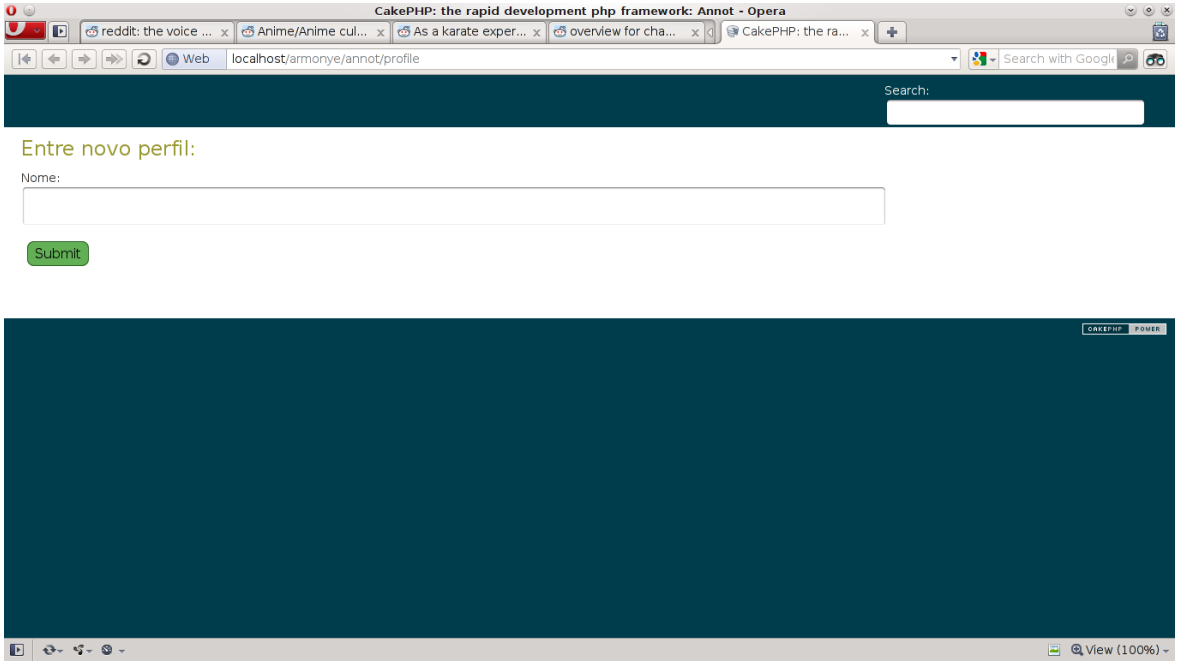

Ilustração 2- Tela Inicial do Sistema

Caso o nome já exista, ele abre o Perfil já existente. O usuário pode então adicionar ou remover músicas de seu Perfil. É certo de que toda música que ele inserir no sistema por *Upload*, já será adicionado em seu Perfil. A ilustração 3 mostra o Perfil do usuário com algumas músicas carregadas.

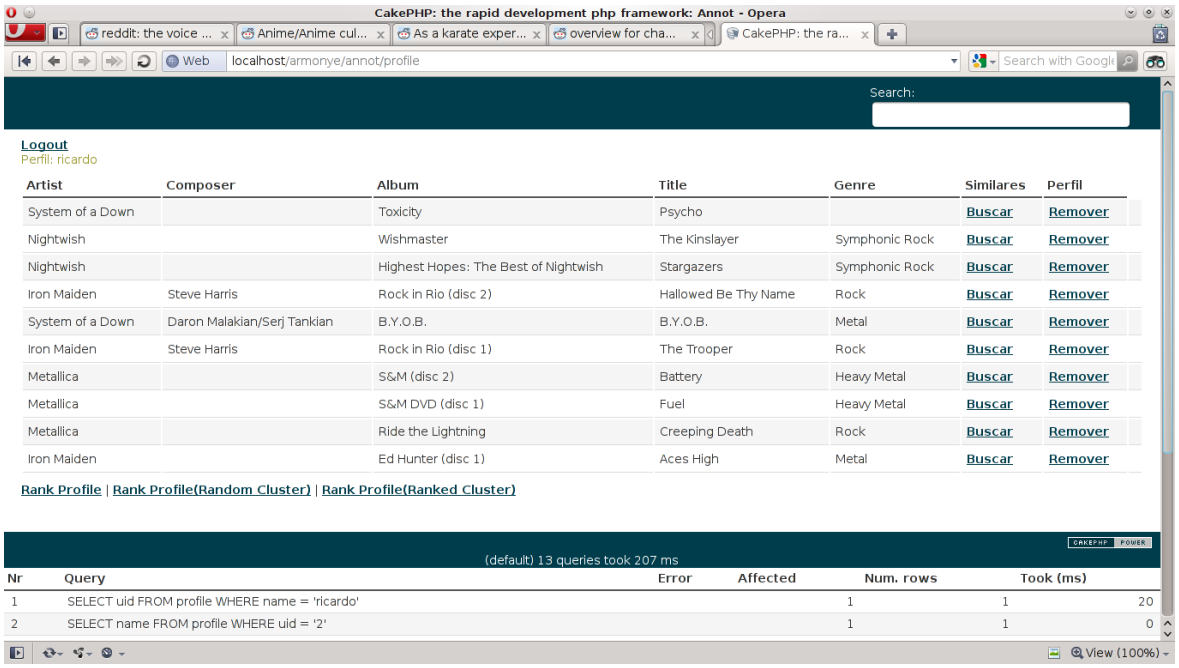

Ilustração 3- Perfil do Usuário

O sistema deve reconhecer as músicas de um determinado usuário e saber fornecer opções de adicionar ou remover músicas. Sendo assim, uma nova tabela no banco de dados foi criada para atender esse controle. É a tabela User, que esta relacionada com a tabela Data de forma N para N (N usuários possuem N músicas em seu Perfil). Sendo assim, a tabela UserXData representa esse relacionamento.

O banco de dados é definido conforme a Ilustração 2.

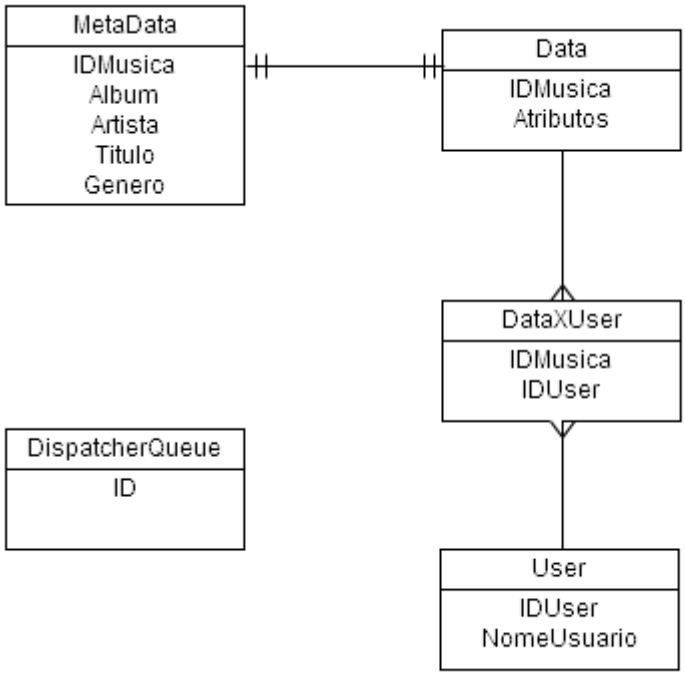

Ilustração 4- Organização e relacionamentos das tabelas no MySQL

Nota-se que a tabela DispatcherQueue não possui relacionamento com nenhuma tabela, visto que apenas guarda as músicas que serão processadas pelo sistema.

Na tabela Data, os Atributos representam a lista de aproximadamente 60 atributos que o *bextract* extrai de cada música, conforme mencionados em 3.1.

Para a criação do perfil do usuário foi necessário buscar músicas e simular playlists, pois os testes são feitos localmente. Com uma grande quantidade de músicas a disposição, foi possível separar alguns grupos de estilos semelhantes e também playlists diversificadas. Com isso, será possível receber sugestões para cada música de uma *playlist*.

Por fim, as músicas na tabela Data tiveram seus atributos normalizados, para que o sistema saiba efetuar os cálculos sem perder muita informação, devido a grande variabilidade dos atributos das músicas (Distance Tutorial, 2010). Isso ocorre, pois há uma grande quantidade de atributos e uma grande quantidade de estilos de musicas. Após a normalização, o banco de dados passou pela clusterização feita pelo Weka, sendo proposta a divisão das músicas em 50 clusters. Estes dois processos demoraram cerca de 40 horas.

No Weka para esta classificação é o EM (Expectation Maximization). De acordo com Witten e Frank (2000 apud LENZ, 2009, p.36, a aplicação deste conceito, com relação ao algoritmo EM implementado no Weka, é obtida através da validação cruzada, a qual é utilizada para determinar o número de cluster da seguinte maneira:

1. Inicialmente, o número de clusters K passa a ser igual a 1;
2. Os elementos da amostra são distribuídos randomicamente entre as décimas partes da amostra;
3. Executa-se o algoritmo EM 10 vezes, variando-se as partes selecionadas para treinamento de acordo com a maneira usual da 10-fold cross validation;
4. Uma média das probabilidades é estabelecida para todos os resultados obtidos;
5. Caso o valor das médias das probabilidades tenha aumentado com relação ao obtido na iteração anterior, então K = K + 1 e retorna-se ao passo 2.
6. Caso contrário o algoritmo termina, separando assim em K clusters.

3.4 Utilização do Perfil de Usuário

Analisando o perfil de um usuário, o sistema pode efetuar a sugestão de três formas (de acordo com a escolha do usuário):

- Proximidade;
- Clusterização com sugestão randômica;
- Clusterização com proximidade.

Por proximidade, será analisada a distância euclidiana para cada música do perfil do usuário. Então será selecionada a música mais próxima, caso esta já não tenha sido sugerida ou não esteja no perfil do usuário. Nestes casos, será sugerida a segunda música mais próxima e assim por diante. Supondo que o Perfil do usuário possui 10 músicas, a sugestão retornará uma playlist com 10 músicas, cada música similar com a playlist original.

Usando a clusterização, as músicas foram agrupadas seguindo determinados critérios e a recomendação será feita pelos próprios grupos (conforme explicado em 2.2). Sendo assim, dada uma música, podemos sugerir uma música que pertença ao mesmo grupo de forma randômica ou por proximidade. Para cada música, é feita a análise proposta.

Caso seja randômica, a recomendação é mais rápida, porém depende de um agrupamento “ideal”, onde todos os tipos de músicas estejam bem definidos em cada agrupamento.

Pela proximidade, é verificada dentro do próprio grupo qual música é mais próxima da música analisada. Esse método é mais demorado comparado ao rândomico, porém tem mais chances da recomendação ser melhor, visto que leva em conta não somente o grupo, mas a análise de todos os atributos de todas as músicas contidas nele.

Após feita a recomendação, é obtida a avaliação do usuário para a recomendação feita pelo sistema. Para cada música sugerida, o usuário tem a opção de adicionar a música ao seu Perfil.

4. MÉTODO DE AVALIAÇÃO

Para cada algoritmo de recomendação, utilizaremos um conjunto de músicas que simulam uma playlist de usuário. As músicas dessas playlists já estão no sistema. Para iniciar os testes, deve-se:

- Buscar todas suas músicas de uma playlist e atribuir a um perfil de usuário;
- Para cada música de cada playlist, verificar qual a melhor recomendação feita, desde que não recomende músicas de sua própria playlist ou que já foi recomendado. Nessa situação, proceder com a segunda melhor recomendação.

No algoritmo de proximidade e no de clusterização com proximidade, a avaliação do teste será a comparação do que o sistema considera bom com o nível de aceitação do usuário. O sistema leva em conta a classificação de músicas baseado na distância e classifica da seguinte forma:

- Caso o valor da distância da música sugerida seja menor que 1, então a recomendação foi boa;
- Caso o valor da distância estiver entre 1 e 2, a recomendação não foi a ideal, significando que poderia ser melhor. Neste caso, a base de dados não possui uma música similar, lembrando que pode ser melhorada com o tempo e uso do sistema por mais usuários;
- Caso o valor da distância seja maior que 2, significa que a recomendação foi ruim e não há nada parecido com a música no banco de dados. Também vale lembrar que pode melhorar com o tempo de uso do sistema por mais usuários.

Na aceitação do usuário, ele vai classificar cada recomendação como boa ou ruim, de acordo com o gosto dele. O nível de aceitação será a porcentagem de recomendações que ele considerou como Boa. Alguns voluntários foram selecionados para efetuar esta avaliação.

Para a avaliação da clusterização com sugestão randômica, não há a comparação com a distância, então será considerado apenas a avaliação do usuário para dizer se a recomendação foi boa ou ruim.

5. RESULTADOS E DISCUSSÕES

Todas as playlists teste serão avaliadas para verificar se o sistema está efetuando a sugestão de forma eficiente. Os critérios para avaliação foram apresentados no capítulo 4. São 5 playlists no total.

5.1 Testes com Playlists

A playlist 1 é composta basicamente de músicas de metal, que se caracteriza por ser um estilo com ênfase na guitarra e na bateria. A Tabela 1 mostra os resultados de recomendação para os três algoritmos de sugestão para a playlist 1.

Tabela 1- Recomendações feitas a partir da Playlist 1

Playlist 1	Playlist Sugerida - Proximidade			Playlist Sugerida - Clusterização		Playlist Sugerida - Clusterização com Proximidade		
	Música	Distância	Usuário	Música	Usuário	Música	Distância	Usuário
System of a Down - Psycho	Ill Niño - All the Right Words	1.059019998	Boa	Enya - Only Time	Ruim	Ill Niño - Re-Birth	1.15791555	Boa
Nightwish - The Kinslayer	Nightwish - End of All Hope	0.798481024	Boa	Fresno - Evaporar	Ruim	myGRAIN - Walk Puppet Walk	0.8475055	Boa
Nightwish - Stargazers	Nightwish - The Pharaoh Sails to Orion	0.903336666	Boa	Liquid Spill - Insanity	Ruim	Nightwish - Beauty of the Beast	0.9669924	Boa
Iron Maiden - Hallowed Be Thy Name	Iron Maiden - The Evil That Men Do	0.605405658	Boa	Liquid Spill - Insanity	Ruim	Iron Maiden - The Mercenary	0.62281914	Boa
System of a Down - B.Y.O.B.	System of a Down - Cigaro	0.864947841	Boa	HIM - Beautiful	Ruim	Infernal - Burning Up	1.1007072	Ruim
Iron Maiden - The Trooper	Iron Maiden - 2 Minutes to Midnight	0.472262196	Boa	Os SemiNovos - Foi o cão	Ruim	Iron Maiden - Iron Maiden	0.48352938	Boa
Metallica - Battery	Metallica - Enter Sandman	0.537427108	Boa	Madonna - Ray of Light (radio edit)	Ruim	Metallica - Enter Sandman	0.53742711	Boa
Metallica - Fuel	Metallica - For Whom the Bell Tolls	0.963608055	Boa	Timbaland - Talk That Shit	Ruim	Metallica - Fuel	1.11430818	Boa
Metallica - Creeping Death	Metallica - Escape	0.531605839	Boa	Era - Mother	Ruim	Metallica - Escape	0.53160584	Boa
Iron Maiden - Aces High	Dark Tranquillity - Insanity's Crescendo	1.012911165	Boa	Fresno - Evaporar	Ruim	Iron Maiden - Bring Your Daughter... to the Slaughter	1.18574768	Boa

Primeiramente, a música Liquid Spill – Insanity aparece duplicada na recomendação por clusterização. Isso ocorreu pelo fato da recomendação ser aleatória dentro do cluster, podendo trazer a mesma música dentro da playlist recomendada. Outro ponto é a música Metallica – Fuel que foi recomendada no algoritmo de Clusterização com Proximidade para o mesmo nome que foi dado. Apesar de terem o mesmo nome, as músicas são diferentes. A música recomendada é ao vivo e conta com uma orquestra sinfônica.

Na recomendação feita pelo algoritmo de proximidade, podemos ver que duas músicas não resultaram em recomendações ideais. Ainda assim, estas músicas atendem ao estilo da playlist 1 e agradaram ao usuário.

Na recomendação feita pelo algoritmo de clusterização com sugestão randômica, a recomendação foi ruim. Nenhuma música recomendada se assemelha com a playlist 1.

Por fim, a recomendação feita pelo algoritmo de clusterização com proximidade foi boa. Quatro músicas não foram ideais, mas três delas ainda agradaram o usuário. Apenas uma música não se encaixou no gênero.

Pelo Gráfico 1, é comparado a eficiência de cada algortimo com a aceitação do usuário.

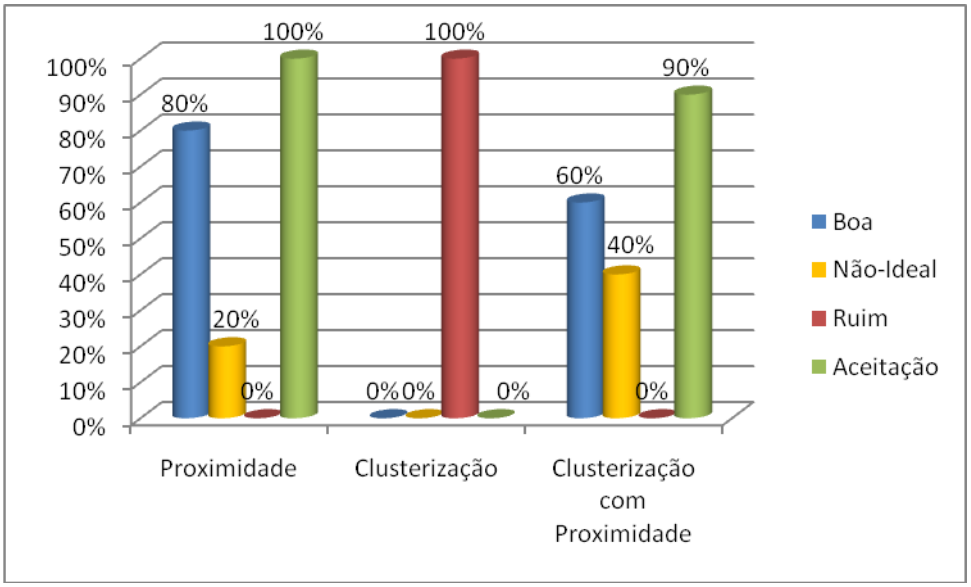

Gráfico 1 – Comparação entre os algoritmos – Playlist 1

Para a Playlist 1, a melhor recomendação é feita pela proximidade, visto que a aceitação pelo usuário foi de 100%. Apesar de haver diferença entre o algoritmo proximidade e algoritmo clusterização com proximidade, este último ainda consegue ser uma boa recomendação, visto que recomendou muitas músicas do mesmo gênero e a diferença foi baixa.

A playlist 2 é diversificada, contendo músicas de muitos gêneros. Ela é composta de 10 músicas. A Tabela 2 mostra os resultados de recomendação para os três algoritmos de sugestão para a playlist 2.

Tabela 2- Recomendações feitas a partir da Playlist 2

Playlist 2	Playlist Sugerida - Proximidade			Playlist Sugerida - Clusterização		Playlist Sugerida - Clusterização com Proximidade		
	Música	Distância	Usuário	Música	Usuário	Música	Distância	Usuário
島みやえい子 - Higurashi no Naku Koro ni	Future Loop Foundation - Monika's Summer	0.894833361	Ruim	Dark Tranquillity - Sacred Reich	Ruim	Future Loop Foundation - Monika's Summer	0.89483336	Ruim
Annagrace - Let the Feelings Go	Lady GaGa - I Like It Rough	0.922533424	Boa	Maria Gadú - Baba	Boa	Cassie - Ditto	1.08973278	Ruim
Dargaard - Caverna Obscura	Dargaard - Ev Χειμωνι - Dark Horizons	0.947808755	Boa	Citizen Cope - Theresa	Ruim	Dargaard - Ev Χειμωνι - Dark Horizons	0.94780876	Boa
Era - Sentence	Era - Looking for Something	1.048803077	Boa	Love Transistor - Wherever	Ruim	Blank & Jones - Revealed (Radio Mix) with Steve Kilbey	1.34666441	Ruim
Maria Gadú - Escudos	Legião Urbana - Mais Uma Vez	1.321868396	Boa	Arch Enemy - Silverwing	Boa	Legião Urbana - Mais Uma Vez	1.3218684	Boa
Tribalistas - Velha infância	Hoobastank - What Happened to Us?	1.190348851	Boa	Avril Lavigne - When You're Gone	Ruim	Maria Gadú - Bela Flor	1.22053949	Boa
Il Divo - Ave Maria	Il Divo - Somewhere	0.934679328	Boa	Michael Jackson - Got to Be There	Boa	Il Divo - Somewhere	0.93467933	Boa
Loreena McKennitt - All Souls Night	Eduardo Mata & The Dallas Symphony Orchestra - Alborada Del Gracioso	1.221807259	Ruim	John Tesh - I Will Always Love You	Ruim	浜崎あゆみ - CAROLS	1.35737451	Boa
Jeito Moleque - Cara	Jeito Moleque - Sua Casa Vai Cair	0.600425258	Boa	Steintryggur - rasinagain	Ruim	Molejo - Voltei	1.16018709	Boa
Whitney Houston - I Wanna Dance With Somebody (Who Loves Me)	Taylor Dayne - Tell It to My Heart	0.769330509	Boa	Daytona - We're Not Gonna Take It	Boa	Dexys Midnight Runners - Come On Eileen	1.17949601	Ruim

Pela recomendação feita pelo algoritmo de proximidade, podemos ver que três músicas não resultaram em recomendações ideais. Mesmo assim, agradaram ao usuário.

Na recomendação feita pelo algoritmo de clusterização com sugestão randômica, a recomendação foi fraca. Apenas 40% das músicas agradou o usuário.

A recomendação feita pelo algoritmo de clusterização com proximidade não foi ideal. Apenas três músicas foram de boa recomendação. Mas para o usuário, 60% das músicas foram aceitas.

Pelo Gráfico 2, é comparado a eficiência de cada algortimo com a aceitação do usuário.

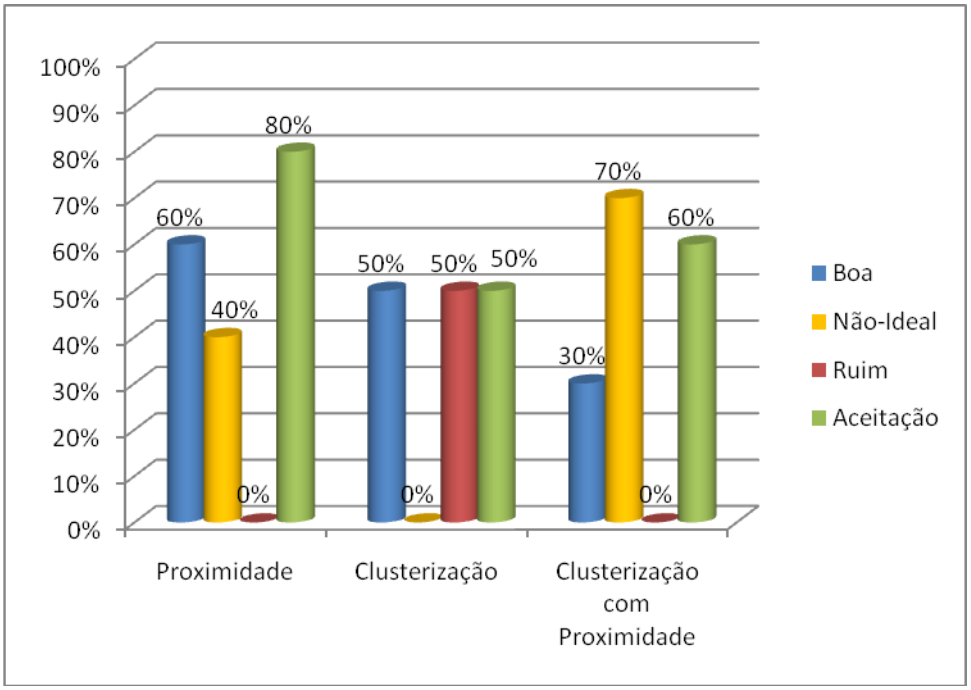

Gráfico 2 – Comparação entre os algoritmos – Playlist 2

Para a Playlist 2, a melhor recomendação é feita pela proximidade, com uma aceitação de 80%. A recomendação por clusterização com proximidade pode ser considerada razoavelmente boa, com uma aceitação de 60%. Na recomendação por clusterização randômica, a aceitação foi de 40%, significando uma recomendação fraca. Comparado com a Playlist 1, esse resultado foi

interessante, visto que a playlist 2 é totalmente diversificada, enquanto a playlist 1 é focada em um gênero.

A playlist 3 possui músicas alternadas entre os gêneros Rap, Black e Pop, onde a característica principal são as batidas e ritmos constantes. Quanto a canção, o foco está nas rimas. A Tabela 3 mostra os resultados de recomendação para os três algoritmos de sugestão para a playlist 3.

Tabela 3- Recomendações feitas a partir da Playlist 3

Playlist 3	Playlist Sugerida - Proximidade			Playlist Sugerida - Clusterização		Playlist Sugerida - Clusterização com Proximidade		
	Música	Distância	Usuário	Música	Usuário	Música	Distância	Usuário
The All-American Rejects - Move Along	Bullet for My Valentine - Spit You Out	0.93055276	Ruim	Fresno - Uma Musica	Ruim	Ill Niño - Te Amo... I Hate You	1.03111425	Ruim
The All-American Rejects - It Ends Tonight	The All-American Rejects - Change Your Mind	1.116360712	Boa	YELLOW ZEBRA - 紅い月	Ruim	The All-American Rejects - Straightjacket Feeling	1.2365673	Boa
Ludacris - How Low (Pitbull & Ciara Remix)	Ludacris - How Low (Chris Brown & Ciara Remix)	0.302773729	Boa	Haitian Fresh - International Boss	Boa	Ludacris - How Low (Chris Brown & Ciara Remix)	0.30277373	Boa
Kardinal Offishall - Love My City	Jo Shine - Get It In	0.861188505	Boa	Q-Minus - Watch It Hit The Floor	Boa	Shawty Lo - 911	0.96848114	Boa
Kardinal Offishall - Burnt	Kelly Rowland - Come Back	0.876022181	Boa	Spyzer - I Feel So Free (DJ Joe K Radio Edit)	Boa	Kelly Rowland - Come Back	0.87602218	Boa
David Guetta - Love Don't Let Me Go	Notorious B.I.G. - Tonight	1.136551991	Boa	Jeito Moleque - Avidaé Engraçada	Boa	Notorious B.I.G. - Tonight	1.13655199	Boa
Black Eyed Peas - Boom Boom Pow (Fatman Scoop Remix)	Timbaland - Tomorrow In The Bottle	0.970812971	Boa	Dire Straits - On Every Street	Ruim	Timbaland - Tomorrow In The Bottle	0.97081297	Boa
OneRepublic - Stop And Stare	Ivete Sangalo - Ilumina	0.977747273	Boa	Gardenian - Chaos in Flesh	Ruim	Ivete Sangalo - Ilumina	1.06395099	Boa
OneRepublic - Come Home	Paolo & Isabella - What Dreams Are Made Of (Ballad version)	1.044408859	Boa	Ray Cash - Bumpin' My Music	Boa	Paolo & Isabella - What Dreams Are Made Of (Ballad version)	1.04440886	Boa
Black Eyed Peas - I Gotta Feeling	The Black Eyed Peas - Boom Boom Pow	1.015615735	Boa	Fragma - Tocas Miracle 2008	Boa	Fort Minor - Red to Black (feat. Kenna, Jonah Matranga and Styles of Beyond)	1.02850678	Ruim

Muitas músicas recomendadas pela Proximidade foram também recomendadas pela Clusterização por Proximidade. Isso significa que a clusterização para esses casos foi boa.

Na recomendação feita pelo algoritmo de proximidade, podemos ver que quatro músicas resultaram em recomendações não-ideais. Ainda assim, apenas uma música não agradou o usuário.

Na recomendação feita pelo algoritmo de clusterização com sugestão randômica, a recomendação foi razoável. Metade das músicas agradou o usuário.

A recomendação feita pelo algoritmo de clusterização com proximidade não foi ideal. Quatro músicas foram de boa recomendação. Entretanto, para o usuário a aceitação foi de 80%.

Pelo Gráfico 3, é comparado a eficiência de cada algortimo com a aceitação do usuário.

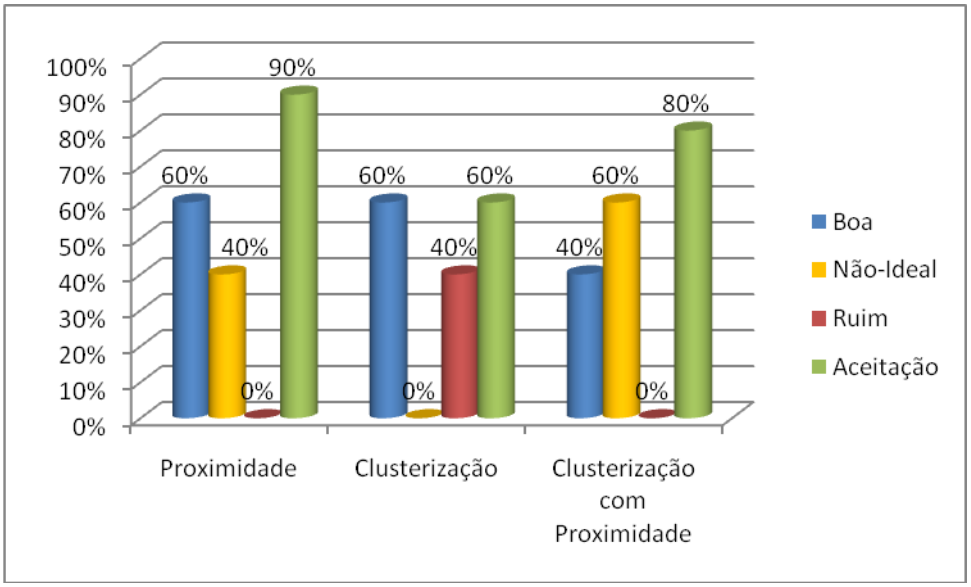

Gráfico 3 – Comparação entre os algoritmos – Playlist 3

Para a Playlist 3, a melhor recomendação é feita pela proximidade, com uma aceitação de 90%. A recomendação por clusterização com proximidade é boa, com uma aceitação de 80%. Na recomendação por clusterização randômica, a aceitação foi de 60%, significando uma recomendação razoavelmente boa.

A playlist 4 é composta de 10 músicas, em sua maioria de Metal. A Tabela 4 mostra os resultados de recomendação para os três algoritmos de sugestão para a playlist 4.

Tabela 4- Recomendações feitas a partir da Playlist 4

Playlist 4	Playlist Sugerida - Proximidade			Playlist Sugerida - Clusterização		Playlist Sugerida - Clusterização com Proximidade		
	Música	Distância	Usuário	Música	Usuário	Música	Distância	Usuário
Before the Dawn - Fear Me	Before the Dawn - Reign of Fire	0.748287278	**Boa**	Soilwork - Your Beloved Scapegoat	**Boa**	CROW'SCLAW - Extend Sky	1.0404187	**Boa**
Kalmah - The Black Waltz	Kalmah - The Groan of Wind	0.501690141	**Boa**	Soilwork - Your Beloved Scapegoat	**Boa**	Insomnium - Dying Chant	0.75977839	**Boa**
Lacuna Coil - Entwined	Lacuna Coil - Humane	1.07417873	**Boa**	Disarmonia Mundi - Mouth for War	**Ruim**	Lacuna Coil - Halflife	1.22253418	**Boa**
Tystnaden - The Joke	Tystnaden - Metaphora	0.551267487	**Boa**	Fergie - Big Girls Don't Cry (Live)	**Ruim**	Tystnaden - Metaphora	0.55126749	**Boa**
Lacuna Coil - No Need to Explain	Lacuna Coil - Soul Into Hades	0.845448514	**Boa**	Röyksopp - Only This Moment (Radio Edit)	**Ruim**	Demetori - Mystic Oriental Dream ~ Ancient Temple	0.91685959	**Boa**
Engenheiros do Hawaii - Alucinação	Babyface - Every Time I Close My Eyes	1.05359813	**Ruim**	Dark Age - Minus Exitus	**Ruim**	Babyface - Every Time I Close My Eyes	1.05359813	**Ruim**
Nickelback - Flat on the Floor	Paramore - Misery Business	1.123587784	**Boa**	Los Hermanos - O Velho e o Moço	**Ruim**	Paramore - Misery Business	1.12358778	**Boa**
Arch Enemy - The Last Enemy	Lostprophets - Start Something	1.345588363	**Boa**	Wintersun - Starchild	**Boa**	Blackmore's Night - Memmingen	1.51881614	**Ruim**
Shadows Fall - Mark of the Squealer	Shadows Fall - Will to Rebuild	0.535167361	**Boa**	Foo Fighters - Gimme Stitches	**Ruim**	Shadows Fall - Will to Rebuild	0.53516736	**Boa**
All That Remains - Become the Catalyst	All That Remains - Empty Inside	0.680588626	**Boa**	Titãs - Enquanto houver sol	**Ruim**	All That Remains - Empty Inside	0.68058863	**Boa**

Na recomendação feita pelo algoritmo de proximidade, podemos ver que quatro músicas resultaram em recomendações não-ideais. Ainda assim, apenas uma música não agradou o usuário, atingindo um nível de aceitação de 90%.

Na recomendação feita pelo algoritmo de clusterização com sugestão randômica, a recomendação foi fraca: apenas três músicas agradaram o usuário.

A recomendação feita pelo algoritmo de clusterização com proximidade foi regular, metade das músicas classificadas como uma boa recomendação e a outra metade como não-ideal. Ainda assim, a aceitação do usuário foi de 80%

Pelo Gráfico 4, é comparado a eficiência de cada algortimo com a aceitação do usuário.

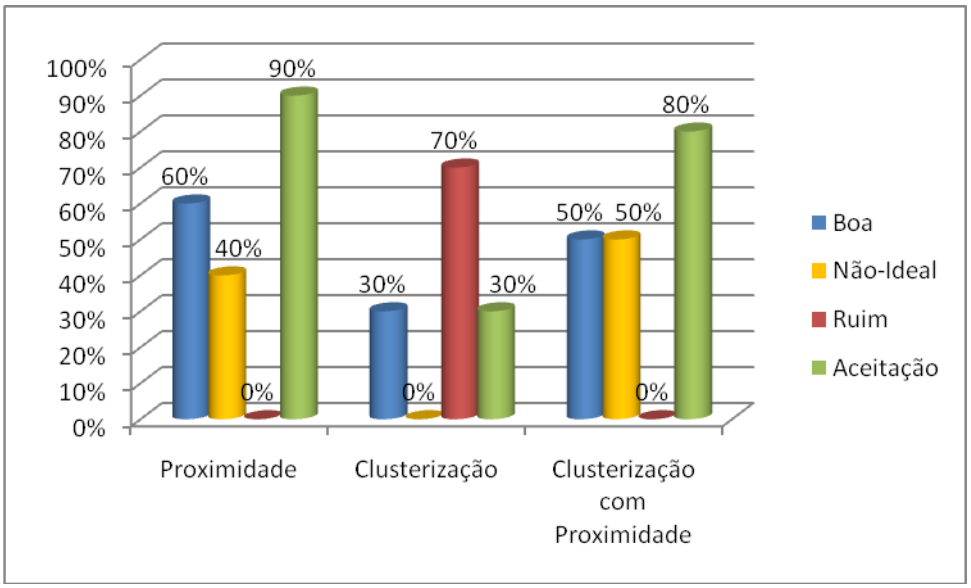

Gráfico 4 – Comparação entre os algoritmos – Playlist 4

Para a Playlist 4, a melhor recomendação é feita pela proximidade, com uma aceitação de 90%. A recomendação por clusterização com proximidade é boa, com uma aceitação de 80%. Na recomendação por clusterização randômica, a aceitação foi de 30%, significando uma recomendação ruim.

A Playlist 5 é composta de 10 músicas de gêneros variados. A Tabela 5 mostra os resultados de recomendação para os três algoritmos de sugestão.

Tabela 5- Recomendações feitas a partir da Playlist 5

Playlist 5	Playlist Sugerida - Proximidade			Playlist Sugerida - Clusterização		Playlist Sugerida - Clusterização com Proximidade		
	Música	Distância	Usuário	Música	Usuário	Música	Distância	Usuário
Apocalyptica - Toreador	Apocalyptica - Refuse/Resist	0.808440195	Boa	Beastie Boys - Freaky Hijiki	Ruim	Apocalyptica - Refuse/Resist	0.8084402	Boa
Pink Floyd - Outside the Wall	John Lennon - Imagine	0.86203916	Ruim	Britney Spears - Circus	Boa	Eduardo Mata & The Dallas Symphony Orchestra - Alborada	1.11148739	Boa
B.B. King - Tomorrow Night	Mickey Gilley - Stand by Me	1.325971938	Ruim	Pearl Jam - Gods' Dice	Ruim	Morphine - All Wrong	1.36169721	Boa
Chico Buarque e Caetano Veloso - Ana de	Chico Buarque e Caetano Veloso - Janelas Abertas no.	1.058969344	Boa	Pink Floyd - Don't Leave Me Now	Ruim	Chico Buarque e Caetano Veloso - Janelas Abertas no. 2	1.05896934	Boa
Foo Fighters - Cold Day in the Sun	Silverchair - Do You Feel the Same	1.140063559	Boa	Cabocla - Sem Palavras	Ruim	NxZero - Espero a minha vez	1.14836436	Ruim
Orbital - Adnan's	Lasgo - Alone (extended mix)	1.10578258	Ruim	Lil Wayne - Earthquake	Ruim	Karnak - Hymboraewqueyra	1.22646248	Ruim
Karnak - Espinho Na Roseira / Drumonda	Karnak - Lee-o-Lua	1.132795166	Boa	Xandria - Isis/Osiris	Ruim	Karnak - Lee-o-Lua	1.13279517	Boa
Karnak - Cala a Boca Menina	Christina Aguilera - Without You	1.105981841	Ruim	Nightwish - Creek Mary's Blood (orchestral instrumental score)	Ruim	Christina Aguilera - Without You	1.10598184	Ruim
Legião Urbana - Eduardo e Mônica	Legião Urbana - Faroeste caboclo	1.034959462	Boa	Kylie Minogue - I Should Be So Lucky	Ruim	Laura Pausini - Dispárame, dispara	1.19105826	Ruim
Adriana Calcanhotto - Teu Nome Mais Secreto	Adriana Calcanhotto - Três	1.096836569	Boa	Pink Floyd - Don't Leave Me Now	Ruim	Adriana Calcanhotto - Três	1.09683657	Boa

Na recomendação feita pelo algoritmo de proximidade, podemos ver que apenas 2 músicas resultaram em recomendações ideais. Ainda assim, a aceitação foi de 60%.

Na recomendação feita pelo algoritmo de clusterização com sugestão randômica, a recomendação foi ruim: apenas uma música agradou o usuário.

A recomendação feita pelo algoritmo de clusterização com proximidade foi fraca. Apenas uma música tida como boa recomendação e as demais como não-ideal. Ainda assim, a aceitação do usuário foi de 60%

Pelo Gráfico 5, é comparado a eficiência de cada algortimo com a aceitação do usuário.

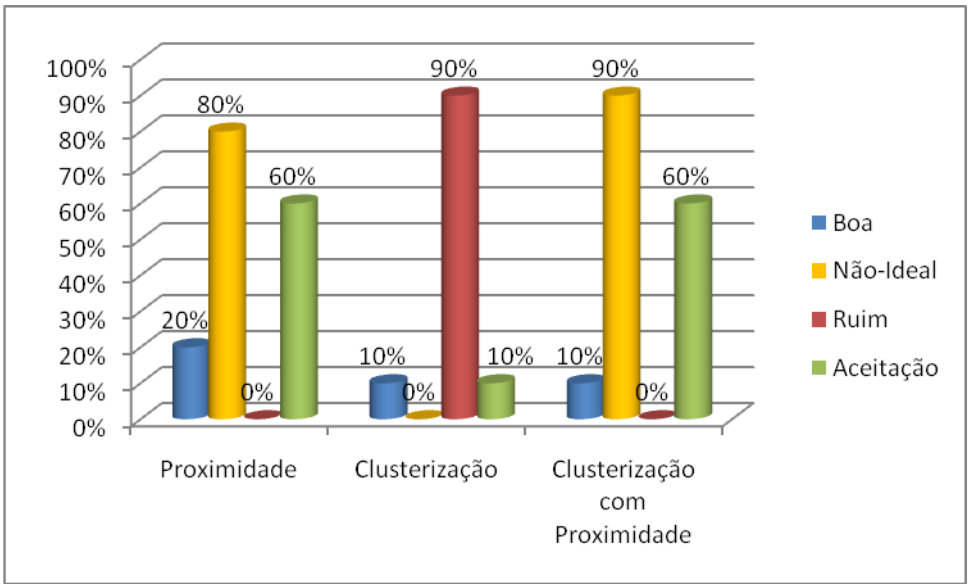

Gráfico 5 – Comparação entre os algoritmos – Playlist 5

Para a Playlist 5, houve um empate entre a melhor recomendação. Proximidade e clusterização com proximidade atingiram 60% de aceitação. Na recomendação por clusterização randômica, a aceitação foi de 10%, significando uma recomendação ruim.

5.2 Resumo dos resultados

Com base nos testes feitos no capítulo 5, o gráfico 6 mostra um consolidado dos resultados.

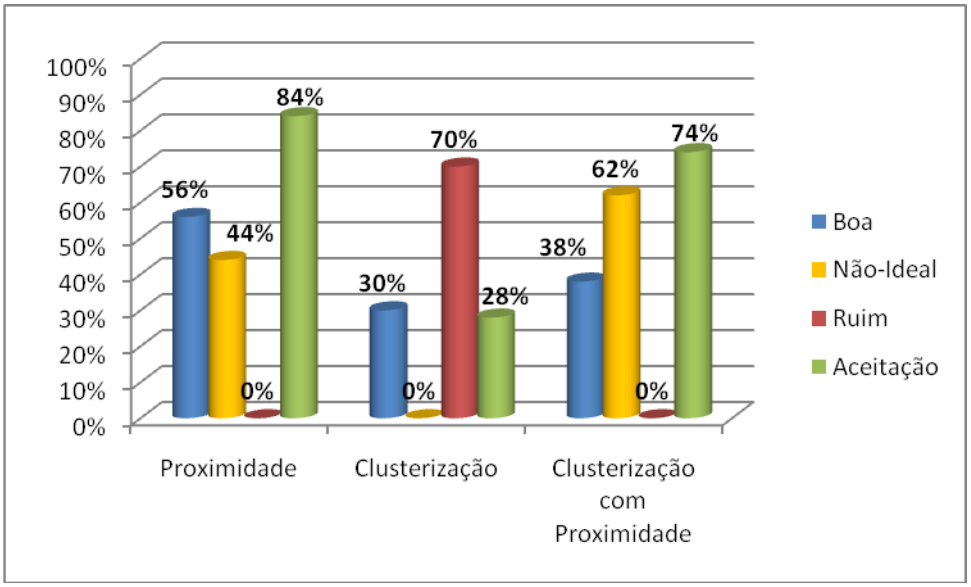

Gráfico 6 – Consolidado dos algoritmos

A recomendação por proximidade é a melhor recomendação de todas com uma taxa de aceitação de 84%. Em seguida, a clusterização com proximidade atinge 74% de aceitação, podendo ser considerado uma boa recomendação também. A clusterização com sugestão randômica é ruim, pois atingiu 28% de aceitação.

Quanto aos resultados do sistema em relação a distância, nenhuma música foi classificada como ruim. Na proximidade, 56% das músicas foram classificadas como Boa e 44% como Não-Ideal. Na clusterização por proximidade, 38% das músicas foram classificadas como Boa e 62% como Não-Ideal.

Em termos de performance, o algortimo Clusterização foi o mais rápido, retornando em 250 milisegundos, seguido do algoritmo de Clusterização com Proximidade em 1 segundo e por último o algoritmo de Proximidade com 40 segundos.

6. CONCLUSÕES E CONSIDERAÇÕES FINAIS

Analisando os resultados, é possível efetuar a recomendação de forma eficiente utilizando o método de Proximidade e atingir uma boa aceitação (84%). Porém, o tempo de processamento é alto, em torno de 40 segundos. Uma alternativa para o método de Proximidade, visto que seu tempo de processamento é alto, é a Clusterização com Proximidade, que obteve uma aceitação de 74% e possui um tempo de processamento de 1 segundo.

Do ponto de vista dos Meta-Dados, para a recomendação de músicas ser eficiente, o sistema depende da precisão dos meta-dados extraídos da música. O MusicBrainz é um projeto Open Source, por isso seus resultados nem sempre são precisos, pelo fato de qualquer pessoa ser capaz de adicionar informação ao site. O sistema, portanto, pode não fornecer a informação correta da música.

Sobre os objetivos traçados, todos foram cumpridos resultando assim no sistema final. A implementação do processamento de uma playlist inteira possibilitou a facilidade nos testes, pois foi possível subir uma larga quantidade de músicas. Com isso, o perfil de usuário pode ser construído e testado, gerando as recomendações pelo método proposto: por análise musical.

Portanto, a recomendação de músicas por análise musical pode ser feita de maneira eficiente, recomendando músicas apenas pelo som produzido, evitando assim problemas de Meta-Dados e recomendações de artistas pouco conhecidos. O melhor método para esta recomendação é a proximidade.

7. SUGESTÕES PARA TRABALHOS FUTUROS

Para trabalhos futuros, o ideal é a melhoria do sistema, que pode ser feita de diversas formas. Abaixo alguns itens que podem ser implementados:

- Levar em consideração feedbacks negativos;
- Opção de não sugerir do mesmo artista, para que o usuário tenha uma chance maior de conhecer novos artistas.
- Normalizar e Clusterizar as músicas em tempo real, nem que seja algo aproximado, mas que possibilite o usuário carregar as músicas e já obter uma sugestão;
- Descrever qual o motivo da sugestão da música;
- Utilizar sistemas de meta-dados pagos;
- A partir das playlists dos usuários armazenadas no sistema, estes dados poderiam ser utilizados para aprendizagem de máquina. Acaba sendo uma nova forma de recomendação.
- Disponibilização na Internet, para que possa ser alimentado simultaneamente por diversos usuários. Assim terá uma quantidade maior de músicas e atingirá mais gostos musicais.

REFERÊNCIAS

Barrington, Luke, Oda, Reid, Lanckriet, Gert. **SMARTER THAN GENIUS? HUMAN EVALUATION OF MUSIC RECOMMENDER SYSTEMS,** in *10th International Conference on Music Information Retrieval*, Kobe, Japan, 2009, disponível em <http://ismir2009.ismir.net/proceedings/OS4-4.pdf> Acesso em: 28 setembro 2010.

BATZOGLOU, Serafim, DO, Chuong B., **WHAT IS THE EXPECTATION MAXIMIZATION ALGORITHM?**, in *Nature Biotechnology*, Volume 26, Number 8, August,2008, disponível em <http://ai.stanford.edu/~chuongdo/papers/em_tutorial.pdf> Acesso: 13 fevereiro 2011.

BETH, Logan. **MUSIC RECOMMENDATION FROM SONG SETS,** in *5th International Conference on Music Information Retrieval*, Barcelona, Spain, 2004, disponível em < http://ismir2004.ismir.net/proceedings/p077-page-425-paper141.pdf> Acesso em: 23 setembro 2010.

bextract – Marsyas User Manual , disponível em < http://marsyas.info/docs/manual/marsyas-user/bextract.html >. Acesso em: 02 dezembro 2010.

Dempster, A. P., Laird, N. M., Rubin, D. B., **Maximum Likelihood from Incomplete Data via the EM Algorithm**, Harvard University and Educational Testing Service, Massachusetts, United States, 1977

Distance Tutorial: Normalized Rank, disponível em < http://people.revoledu.com/kardi/tutorial/Similarity/Normalized-Rank.html>. Acesso em: 08 agosto 2010.

Last.Fm - Sobre a Last.fm, disponível em <http://www.lastfm.com.br/about>. Acesso em: 26 agosto 2010.

LEE, Tan, YEUNG, Y.T., SUMAN, S., LEE, Y.S.W., MAK, C.M., XIA, Lam W.K Yunqing, ZHENG, Nengheng, CHAN, Alex, **DJX:A CONTENT-BASED RECOMMENDATION SYSTEM FOR CHINESE POP SONGS**, in *10th International Conference on Music Information Retrieval*, Kobe, Japan, 2009, disponível em < http://ismir2009.ismir.net/proceedings/LBD-12.pdf> Acesso em: 23 setembro 2010.

LENZ, Alexandre Rafael, **UTILIZANDO TÉNICAS DE APRENDIZADO DE MÁQUINA PARA APOIAR O TESTE DE REGRESSÃO**, 2009. Dissertação (Mestrado em Informática) - Setor de Ciências Exatas, Universidade Federal do Paraná.

MAGNO, Terence, SABLE, Carl, **A COMPARISON OF SIGNAL-BASED MUSIC RECOMMENDATION TO GENRE LABELS, COLLABORATIVE FILTERING, MUSICOLOGICAL ANALYSIS, HUMAN RECOMMENDATION, AND RANDOM**

BASELINE, in *9th International Conference on Music Information Retrieval*, Pennsylvania, USA, 2008, disponível em < http://ismir2008.ismir.net/papers/ISMIR2008_157.pdf> Acesso em: 23 setembro 2010

Marsyas – Music Analisys, Retrieval and Synthesis for Audio Signals, disponível em < http://marsyas.info/ >. Acesso em: 16 novembro 2010.

Mel-frequency cepstrum – Wikipedia, the free encyclopedia, disponível em < http://en.wikipedia.org/wiki/Mel-frequency_cepstrum>. Acesso em: 13 fevereiro 2011.

MusicBrainz – Welcome to MusicBrainz!, disponível em <http://www.musicbrainz.org>. Acesso em: 21 outubro 2010.

TZANETAKIS, George, COOK, Perryl, **MARSYAS: A framework for audio analysis**, Department of Computer Science and Department of Music, Princeton University, New Jersey, United States, 2000, disponível em <http://marsyas.info/pdfs/0000/0005/organised00gtzan.pdf> Acesso em: 09 dezembro 2010.

UITDENBOGERD, Alexandra, SCHYNDEL, Ron, **A Review of Factors Affecting Music Recommender Success** , in *3rd International Conference on Music Information Retrieval*, Paris, France, 2008, disponível em <http://ismir2002.ismir.net/proceedings/02-FP07-1.pdf> Acesso em: 23 setembro 2010

Weka – Weka 3 – Data Mining with Open Source Machine Learning Software in Java < http://www.cs.waikato.ac.nz/ml/weka/>. Acesso em: 16 novembro 2010.

WITTEN,Ian H. , FRANK, Eibe, **DATA MINING: PRACTICAL MACHINE LEARNING TOOLS AND TECHNIQUES WITH JAVA IMPLEMENTATIONS**. Morgan Kaufmann Publishers Inc., 2000.

WITTER, Ian H., DONKIN, Andrew, HOLMES, Geoffrey, **WEKA: A MACHINE LEARNING WORKBENCH**, in *Proc Second Australia and New Zealand Conference on Intelligent Information Systems*, Brisbane, Australia, 1994, disponível em <http://www.cs.waikato.ac.nz/~ml/publications/1994/Holmes-ANZIIS-WEKA.pdf> Acesso: 08 dezembro 2010.