Ariel Godinho
Felipe Vasconcelos

Recomendação Musical para Grupos Baseada em Modelo Híbrido

Brasil
2018, São Paulo

Ariel Godinho
Felipe Vasconcelos

Recomendação Musical para Grupos Baseada em Modelo Híbrido

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro da Computação.

Área de Concentração: Engenharia da Computação. Orientador: Prof. Dr. Jorge Luis Risco Becerra.

Brasil
2018, São Paulo

Resumo

Os Sistemas de Recomendação são um conjunto de técnicas que utilizam-se de diversas informações para recomendar itens a alguém. Atualmente, diversas plataformas digitais possuem esses sistemas para sugerir itens mais alinhados as preferências individuais de cada pessoa. Entretanto, poucos sistemas usam estas técnicas para criar recomendações que satisfaçam a grupos de pessoas.

Assim, este trabalho busca implementar diferentes técnicas de recomendação, que serão agregadas em um modelo híbrido de recomendação, para sugerir músicas a grupos de pessoas. Também será elaborada a ideia de uma plataforma digital que integraria esse sistema de recomendação híbrido e que forneceria serviços de inteligência — baseado nas informações geradas pelo sistema de recomendação — a participantes do mercado da música.

Palavras-chaves: Sistemas de Recomendação, Recomendação para Grupos, Recomendação Musical, Inteligência artificial, Machine Learning.

Lista de ilustrações

Figura 1 – Modelo Híbrido Ponderado de (BURKE, 2007). 16

Figura 2 – Representação da plataforma IDealMusic. 25
Figura 3 – Diagrama de Entidade Relacional. 27
Figura 4 – Arquitetura em três camadas. 29
Figura 5 – Diagrama da troca de informação no sistema para a recomendação de uma *playlist*. 31

Figura 6 – Distribuição de distâncias euclidianas em relação ao centroide de um *cluster*. 35
Figura 7 – Diagrama da geração de músicas candidatas a recomendação. 37
Figura 8 – Diagrama do processo de valoração das músicas candidatas. 38

Figura 9 – Página Inicial - Parte superior. 48
Figura 10 – Página Inicial - Parte inferior. 48
Figura 11 – Listagem e criação de grupos (salas). 49
Figura 12 – Página de um grupo (Churrasco da Sala) - Parte superior. 49
Figura 13 – Página de um grupo (Churrasco da Sala) - Parte inferior. 50

Lista de tabelas

Tabela 1 – Caso de Uso 1 - Autenticação. 30
Tabela 2 – Caso de Uso 2 - Criação de Playlist. 30
Tabela 3 – Caso de Uso 3 - Acesso ao Histórico de Playlists. 30

Sumário

I Introdução **8**

1 Introdução . . . **9**

1.1 Objetivo . . . 9
1.2 Motivação . . . 9
1.3 Justificativa . . . 10
1.4 Organização . . . 10
1.4.1 Capitulo 2 - Aspectos Conceituais . . . 10
1.4.2 Capitulo 3 - Metodologia . . . 10
1.4.3 Capitulo 4 - Especificação . . . 10
1.4.4 Capitulo 5 - Implementação . . . 11
1.4.5 Capitulo 6 - Testes e Avaliação . . . 11
1.4.6 Capitulo 7 - Considerações Finais . . . 11

II Desenvolvimento **12**

2 Aspectos Conceituais . . . **13**

2.1 Inteligencia Artificial . . . 13
2.2 Machine Learning . . . 13
2.2.1 Aprendizagem Não Supervisionada . . . 13
2.2.2 K-Means . . . 13
2.3 Sistema de Recomendação . . . 14
2.4 Modelo Híbrido . . . 15
2.4.1 Modelo Híbrido Ponderado . . . 15
2.5 Filtragem Baseada em Conteúdo . . . 16
2.6 Filtragem Colaborativa . . . 17
2.6.1 Problema *Cold-Start* . . . 17
2.7 Filtragem Demográfica . . . 17
2.8 Filtragem Social . . . 18
2.9 Análise e Classificação de Sinais Musicais . . . 18
2.10 Feedback Contínuo . . . 18
2.11 Geração Automática de Listas de Reprodução . . . 18
2.12 Sistemas de Recomendação para Grupos . . . 19
2.13 Demais Conceitos . . . 19
2.13.1 RM-ODP . . . 19
2.13.2 Modelo Cliente-Servidor . . . 20

2.13.3 Banco de Dados . . . 20
2.13.4 Modelo de Entidade Relacional . . . 20
2.13.5 HTTP . . . 20
2.13.6 Web API . . . 20
2.13.7 JSON . . . 21
2.13.8 REST . . . 21

3 Metodologia . . . 22
3.1 Revisão da Literatura . . . 22
3.2 Estudo sobre as Técnicas de Recomendação . . . 22
3.3 Obtenção das Bases de Dados . . . 22
3.4 Desenvolvimento dos Agentes Recomendadores . . . 22
3.5 Desenvolvimento do Mecanismo do Modelo Híbrido . . . 23
3.6 Desenvolvimento Cliente-Servidor . . . 23

4 Especificação . . . 24
4.1 Visão Empresarial . . . 24
4.2 Visão de Informação . . . 26
4.3 Visão Computacional . . . 27
4.3.1 Camada de Apresentação . . . 28
4.3.1.1 Requisitos Funcionais do Módulo de Interface do Usuário . 28
4.3.2 Camada de Lógica de Negócio . . . 28
4.3.3 Camada de Dados . . . 28
4.4 Casos de Uso . . . 30
4.5 Visão de Engenharia . . . 31
4.6 Visão de Tecnologia . . . 32
4.6.1 Servidor . . . 32
4.6.2 Cliente . . . 32

5 Implementação . . . 33
5.1 Busca de Bases de Dados . . . 33
5.1.1 Consumo da API do Spotify . . . 33
5.2 Desenvolvimento Aplicação-Servidor . . . 34
5.2.1 Integração com o Spotify . . . 34
5.2.2 Agente Recomendador Baseado em Clusterização por Features . . . 34
5.2.3 Agente Recomendador Baseado em Conteúdo . . . 35
5.2.4 Moderador . . . 36
5.2.4.1 Fase de Treino . . . 36
5.2.4.2 Geração de Candidatos . . . 36
5.2.4.3 Pontuação dos Candidatos . . . 37

5.2.5 Desenvolvimento da API 37
5.3 Desenvolvimento Aplicação-Cliente 38

6 Testes e Avaliação 39
6.1 Agentes de Recomendação 39
6.2 Moderador 39
6.3 Testes Retroativos 39

III Considerações Finais 41

7 Considerações Finais 42
7.1 Cumprimento dos Objetivos 42
7.2 Contribuições 42
7.3 Trabalhos Futuros 42
7.4 Conclusões 44

Referências 45

Apêndices 47

APÊNDICE A Interface do Usuário 48

Parte I

Introdução

1 Introdução

1.1 Objetivo

O objetivo desse trabalho é criar um sistema para recomendar músicas a um conjunto de pessoas de modo a agradá-las. Para atingir esse objetivo, serão usadas diferentes técnicas de recomendação — das áreas de inteligência artificial e *machine learning* — para gerar uma lista de músicas que satisfaça as preferências musicais dessas pessoas. Essas diferentes técnicas de recomendação serão agregadas por um modelo de sistemas de recomendação híbrido.

O outro objetivo desse trabalho é utilizar esse sistema de recomendação como uma fonte de informação para planejar uma plataforma digital. A plataforma digital permitiria a representantes do mercado da música o acesso a serviços de inteligência musical.

1.2 Motivação

O primeiro fator da motivação, para os integrantes deste trabalho, foi de aprimorar a formação e de obter mais conhecimento nas áreas de inteligência artificial e *machine learning*. Atualmente essas áreas possuem muita relevância no mercado em geral, mas ainda estão sendo pouco cobertas pelo curso de engenharia da computação.

O segundo fator de motivação é a vontade de elaborar a ideia de um modelo de negócio viável para uma empresa de tecnologia. Pois, dado o desejo dos integrantes de empreender na carreira profissional, a elaboração dessa ideia permite explorar o processo de idealização de um novo negócio e de fortalecer os conhecimento mínimos para uma eventual experiência profissional.

Outro fator de grande desejo para os integrantes foi a possibilidade resolver um problema real em que tenham presenciado. No caso, o problema da dificuldade que um grupo de pessoas têm em decidir as músicas que querem ouvir juntas e, quando essa decisão é feita, do problema de insatisfação gerado em alguns indivíduos do grupo.

Por fim, os integrantes também possuem a motivação acadêmica de contribuir com área de sistemas de recomendação. Nos últimos anos essa área começou a ganhar muita relevância, e assim, ainda há muito a ser explorado. Desse modo, este trabalho pode auxiliar novos pesquisadores a estudar sobre a implementação de um sistema híbrido de recomendação.

1.3 Justificativa

Este trabalho possui grande importância pelo avanço que a área de sistemas de recomendação ganhou nos últimos anos. Esse avanço foi, em grande parte, motivado pelo evolução da internet e do surgimento de diversas empresas de tecnologia que começaram a utilizar sistemas de recomendação como uma forma de aumentar o valor dos seus negócios (FRANCESCO; LIOR; BRACHA, 2011). Dentre essas novas empresas, os negócios voltados para o mercado do entretenimento — como o serviço de *streaming* de músicas Spotify, o serviços de *streaming* de filmes Netflix e o a plataforma de compartilhamento de vídeos Youtube — se destacaram pela experiência gerada aos seus usuários pelos sistemas de recomendação integrados aos seus serviços.

Apesar da área de sistemas de recomendação estar em alta, sistemas capazes de recomendar para grupos — e não apenas indivíduos — são escassos na atualidade. No meio acadêmico, já existem pesquisas voltadas à recomendação para grupos, mas ainda são poucos os projetos de implementação desses sistemas de recomendação e nenhum se assemelha ao proposto por este trabalho (LU et al., 2015), o que ressalta a importância deste projeto de criar uma implementação satisfatória para um sistema híbrido de recomendação para grupos.

1.4 Organização

Abaixo está a organização seguida pelo documento.

1.4.1 Capitulo 2 - Aspectos Conceituais

Neste capítulo, serão apresentados os principais os conceitos discutidos neste trabalho que foram estudados nas literatura referenciada ou aprendidos no curso e demais outros temas necessários para a conclusão do projeto.

1.4.2 Capitulo 3 - Metodologia

Neste capítulo, será feita a descrição do processo seguido para criação deste trabalho.

1.4.3 Capitulo 4 - Especificação

Neste capítulo, será feita a descrição da especificação do sistema.

1.4.4 Capitulo 5 - Implementação

Neste capítulo, serão apresentados todos os passos para a implementação do sistema.

1.4.5 Capitulo 6 - Testes e Avaliação

Neste capítulo, serão apresentados os testes realizadas, a técnica de avaliação proposta e os resultados obtidos.

1.4.6 Capitulo 7 - Considerações Finais

Neste capítulo, serão apresentados o que foi atingido com o trabalho, os possíveis trabalhos futuros e as conclusões finais do projeto.

Parte II

Desenvolvimento

2 Aspectos Conceituais

2.1 Inteligencia Artificial

Apesar de falta de consenso sobre uma definição única para Inteligência Artificial, como exposto em (WANG, 2008), será adotada a seguinte definição: "*O projeto e a construção de agentes inteligentes que recebem percepções do ambiente e realizam ações que afetam esse ambiente*" (RUSSELL; NORVIG, 2010).

2.2 Machine Learning

Machine learning, ou aprendizado de máquina, é o campo da Ciência da Computação que estuda algoritmos e modelos matemáticos que computadores usam para progressivamente aumentar seu desempenho em uma tarefa específica.

2.2.1 Aprendizagem Não Supervisionada

No contexto de machine learning, aprendizagem não supervisionada é um ramo que estuda algoritmos que são capazes de aprender a partir de dados de treino não categorizados. Em contraste com aprendizagem supervisionada, os dados de entrada não precisam ser previamente avaliados, seja por que o processo é custoso ou porque simplesmente não se sabe como classificar tais dados.

2.2.2 K-Means

Algoritmos não supervisionados são extremamente competentes em clusterização, ou agrupamento de dados, possibilitando inferências sobre uma grande quantidade de dados sem ser necessária interação humana. K-Means é um método de clusterização muito popular em análise e dados e data mining, principalmente por sua precisão e facilidade de implementação.

Este algoritmo recebe como entrada os dados a serem clusterizados (*dataset*) e a quantidade de *clusters* desejados (denominado k), iniciando com uma atribuição aleatória de centroides (centro de cada *cluster*), geralmente estes sendo pontos quaisquer dos *dataset*. Após a atribuição inicial, o algoritmo faz iterações onde atribui cada ponto do *dataset* ao centroide mais próximo e, em seguida recalcula o centroide de cada *cluster* usando distância euclidiana.

O algoritmo se finaliza quando não ocorrem mais mudanças ou quando a diferença em relação à iteração anterior se tornar menor que um certo valor pré determinado. O resultado final é um agrupamento cuja performance depende diretamente da qualidade dos dados e do k definido. Nota-se que o processo de escolha do k pode não ser trivial, neste trabalho utilizamos o método do cotovelo (*Elbow Method*) para determiná-lo.

2.3 Sistema de Recomendação

Os sistemas de recomendação se baseiam na coleta de informações da preferência de seus usuários para poder sugerir recomendações que satisfaçam o gosto desses usuários. A coleta pode ser realizada de modo explícito, como a nota dada pelo usuário sobre um item, ou implícito, como a obtenção do histórico de utilização desse usuário. Para criar estas recomendações, existem diversas técnicas de filtragem. Entre as principais técnicas, temos: a filtragem baseada em conteúdo, a filtragem colaborativa, a filtragem demográfica, a filtragem baseada em conhecimento e a filtragem social.

No começo da utilização de sistemas de recomendação, esses sistemas utilizavam apenas uma técnica de filtragem para criar suas recomendações. Com os avanços dos estudos na área, novas abordagens misturando diversas técnicas obtiveram melhores resultados se comparadas a sistemas que utilizavam apenas uma técnica. O motivo, em grande parte, é devido aos problemas inerentes a cada abordagem. Os sistemas de filtragem baseada em conteúdo sofrem por analise em conteúdo limitado e por alta especialização. Já os sistemas de filtragem colaborativa sofrem com o problema de *Cold-Start* e de escassez. Assim, ao se misturar diferentes técnicas, ocorre uma suavização desses problemas. O que, por sua vez, eleva a qualidade da recomendação criada. Essa nova abordagem, de se misturar diversas técnicas em um único sistema, é chamada de modelo híbrido.

Para conseguir avaliar um sistema de recomendação criado, diferentes técnicas e métricas existem que podem auxiliar na comparação e refinamento desses sistemas. Para ajudar nessa avaliação, aspectos relacionados com a acurácia, a novidade, a dispersabilidade e a estabilidade dessas predições podem ser analisados.

Do ponto de vista da criação de um sistema de recomendação, os aspectos gerais levados em conta ao criar o sistema passam pelo entendimento do tipo de dados que se possui, os algoritmos de filtragem que serão utilizados, as diferentes técnicas para cada tipo de algoritmo de filtragem, os objetivos buscados pelo sistema, a qualidade esperada da recomendação, assim como outros fatores como a performance geral do sistema.

2.4 Modelo Híbrido

Um modelo utilizado para gerar recomendações é denominado híbrido por fazer uso de diversas técnicas de recomendação para obter seu resultado. Os sistemas híbridos de recomendação costumam ter uma melhor avaliação por resolver problemas que ocorrem quando se usa apenas um método de recomendação. Por exemplo, o problema de Cold Start ocorre quando se usa a técnica de Filtro Colaborativo e pode ser corrigido ao mesclar com a técnica de recomendação baseada em conteúdo.

Na literatura estudada (BURKE, 2007), são exploradas 7 diferentes estratégias de recomendação híbrida, abaixo são pontuados os tipos de sistema:

- **Ponderado:** A pontuação de diferentes recomendações é combinada numericamente.
- **Alternante:** O sistema escolhe entre as recomendações de seus componentes e aplica a selecionada.
- **Misturado:** Recomendações de diferentes recomendadores são apresentadas juntas.
- **Combinação de Características:** Características geradas por diferentes fontes de conhecimento são combinadas e passadas a um único algoritmo de recomendação.
- **Ampliação de Características:** Uma técnica de recomendação é usada para computar características, que são então passadas como entrada para a próxima técnica.
- **Cascata:** Recomendadores são dados prioridades estritas, com os de menor prioridade sendo responsáveis por resolver empates.
- **Meta-nível:** Uma técnica é aplicada e produz algum tipo de modelo, que então é usado como entrada pela próxima técnica

2.4.1 Modelo Híbrido Ponderado

O Modelo Híbrido Ponderado consiste na cooperação de múltiplos agentes de recomendação, de forma a combinar suas valorações, produzindo um valor único para cada item a ser recomendado. O beneficio de usar este modelo está na cobertura dos pontos fracos de um agente pelos pontos fortes de outro, minimizando problemas como Cold Start, e no geral provendo uma recomendação mais inteligente.

Neste trabalho o modelo tem o objetivo adicional de unir as recomendações feitas para diferentes usuários, garantindo que todos os usuários tenham uma experiência agradável com as recomendações.

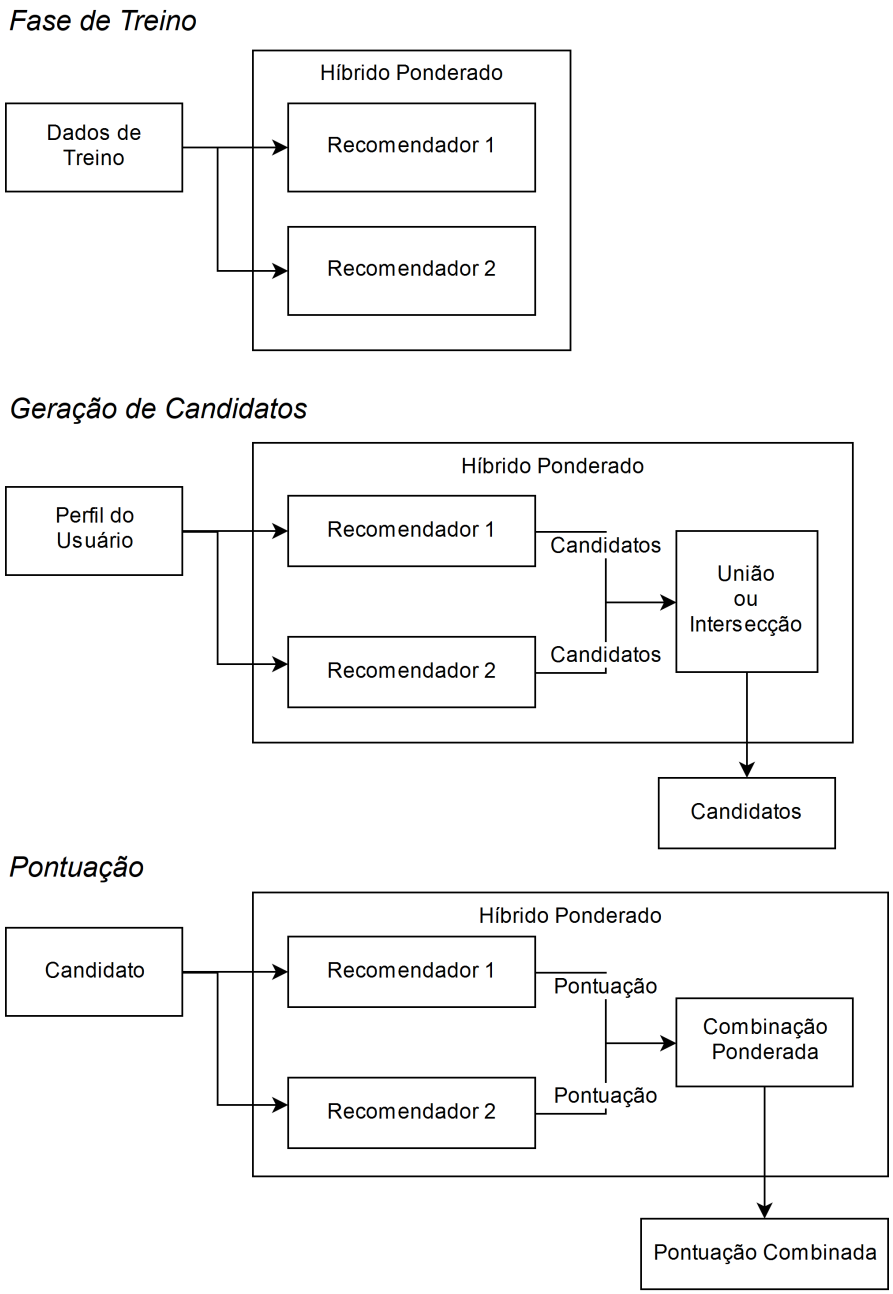

Figura 1: Modelo Híbrido Ponderado de (BURKE, 2007).

2.5 Filtragem Baseada em Conteúdo

A filtragem baseada em conteúdo utiliza informações passadas dos usuários para entender as suas preferências e recomendar algum item que satisfaça esses usuários. Como exemplo, um usuário que faz uma compra de um livro em um e-commerce, pode começar a receber recomendações de livros que sejam similares a esse livro comprado.

Essa similaridade pode ser encontrada pelo sistema por diferentes fontes, como o gênero do livro, o autor, palavras-chave sobre o livro, entre outro. Assim, é possível utilizar diferentes formas de interação do usuário — como comprar um livro, avaliar o livro, olhar livros de um gênero específico — para captar a preferência dele por um item

e, em seguida, buscar itens que sejam similares para recomendar a esse usuário.

No escopo deste projeto, chamaremos de Metadados as informações associadas a uma música, como nome do autor, nome da música e estilo musical. Estes dados podem ser usados, por exemplo, para encontrar relações entre músicas que possuam o mesmo estilo musical, ou artistas que possuam proximidade musical.

2.6 Filtragem Colaborativa

O filtro colaborativo busca criar recomendações cruzadas de usuários considerados similares. O filtro é baseado na ideia de que se usuários possuem preferências em comum, um determinado item de um usuário pode ser recomendado a outro usuário similar que desconhece esse item. Essa similaridade entre os usuários é encontrada pela existência de preferências em comum. Essas preferências podem ser tanto explicitas, como a avaliação direta do usuário, ou implícitas, como histórico de interação com um item.

Assim, um exemplo de filtro colaborativo poderia ser desenvolvido para analisar a similaridade entre usuários a partir dos seus históricos e assim propor musicas que possam agradar esse arquétipo de usuário.

2.6.1 Problema *Cold-Start*

O problema Cold-Start é um dos principais problemas estudado em sistemas de recomendação. Esse problema ocorre dado a baixa quantidade de informação existente no sistema, que dificulta a predição de uma recomendação satisfatória.

Quando o sistema é novo e não há informações suficientes para obter usuários similares, a filtragem colaborativa não consegue produzir predições relevantes. Da mesma forma, a inserção de um novo item ou novo usuário no sistema também dificulta a técnica de filtragem colaborativa a recomendar esse novo item a alguém e a gerar uma recomendação que satisfaça esse novo usuário com preferências desconhecidas.

Uma das principais formas de mitigar esse problema é criando sistemas de recomendação híbridos. Pois assim, pode-se utilizar uma filtragem baseada em conteúdo ou uma filtragem demográfica para criar recomendações nesse estágios em que a filtragem colaborativa não consegue ter bons resultados (BOBADILLA et al., 2013).

2.7 Filtragem Demográfica

O filtro demográfico se utiliza em informações demográficas para sugerir itens aos usuários. Assim, é possível usar o sexo, a idade, a localidade, a classe social, entre outros aspectos para poder recomendar itens que satisfaçam essas segmentações populacionais.

2.8 Filtragem Social

O filtro social vem sendo cada vez mais empregado nos sistemas de recomendação dado o aumento de redes sociais existente. Essas redes sociais possibilitam a obtenção de informações sociais que são geradas nesses sistemas e podem servir de insumo para os sistemas de recomendação. O filtro colaborativo também se beneficia ao possuir uma nova fonte de informação a ser usada.

2.9 Análise e Classificação de Sinais Musicais

Essa técnica propõe a criação de agrupamentos musicais usando Machine Learning a partir de características do sinais musicais. Grupos musicais podem ser criados com os quais pode-se comparar músicas a partir de diversas características, avaliando a distância relativa entre elas.

A partir desta análise, pode-se sugerir músicas a partir de modelos de classificação previamente definidos. Por exemplo, pode-se criar uma classificação que sugere músicas "dançantes", em que essa característica seria traduzida em um aspecto do sinal musical e assim ao utilizar esse algoritmo de classificação obteríamos músicas com o mesmo aspecto definido.

2.10 Feedback Contínuo

Com o feedback contínuo, pode-se fazer uso das ações do usuário para refinar sua playlist. Desse modo, se o usuário pular uma música bem no seu início, isso pode representar que ele não quer essa música e assim pode-se melhorar o resto da lista de reprodução, retirando músicas similares das opções de recomendação.

2.11 Geração Automática de Listas de Reprodução

Em (BONNIN; JANNACH, 2015) é definido o termo Lista de Reprodução como: "*Uma lista de reprodução é uma sequência de faixas de músicas (registros de áudio)*", também se define o problema da geração de listas de reprodução como: "*Dado (1) um conjunto de faixas de música, (2) um banco de dados de conhecimento prévio e (3) algumas características-alvo para a lista de reprodução, criar uma sequência de faixas de músicas que satisfaçam as características-alvo da melhor forma possível*".

É importante notar, então, que no processo de geração automático é preciso que se escolha as características-alvo de modo consciente e, ao se fazer essa escolha, entender os diferentes problemas que possam ser enfrentados. Por exemplo, ao se escolher um

mecanismo de geração baseado em algoritmos de similaridade (artistas relacionados, gêneros parecidos, etc), a lista de reprodução gerada deve trazer baixa diversidade entre as faixas, o que pode ser visto como uma medida de baixa qualidade da lista de reprodução. Da mesma forma, um algoritmo baseado em filtro colaborativo, que poderia trazer uma maior qualidade percebida de diversidade, pode trazer uma baixa qualidade de homogeneidade e de harmonia da lista de reprodução. Assim, é de extrema importância que as caraterísticas-alvo da lista de reprodução sejam coerentes com os algoritmos escolhidos para sua geração, pois uma mesma característica pode ser vista como positiva ou negativa dado o contexto em que se é analisada. Também vale ressaltar, a importância que a técnica híbrida possui para sistemas de recomendação ao poder unir essas diferentes abordagens buscando um resultado intermediário se comparado a tomada de uma única abordagem de recomendação.

Também em (BONNIN; JANNACH, 2015) é possível mergulhar mais a fundo na área de sistemas de recomendação no caso específico para lista de reprodução. Principalmente, nas informações que podem ser utilizadas para criar algum algoritmo de recomendação. Detre elas, destacamos a utilização de características do sinal de áudio como ritmo, altura e timbre como uma forma de buscar uma similaridade entre diferentes faixas sonoras. Outro tipo de informação importante para ser usada em algoritmos de recomendação, é a popularidade de uma determinada faixa de música, que não só é uma boa abordagem para sistemas de recomendação em fase inicial — que não possui muitas informações sobre seus usuários (problema de Cold-Start) — como também pode ser superior a algoritmos mais complexos (MCFEE et al., 2012 apud BONNIN; JANNACH, 2015).

2.12 Sistemas de Recomendação para Grupos

Dentro da literatura, os trabalhos de PolyLens e MusicFX são os mais conhecidos sistemas de recomendação para grupos. O primeiro é um dos mais antigos e buscou criar um sistema que alterasse as música ambiente de uma academia dado as preferências dos frequentadores naquele momento (ALI; KIM, 2015). Já o segundo buscou criar uma sistema de recomendação para recomendar filmes a grupos de pessoas utilizado um algoritmo de filtragem colaborativa, para isso usou a base do sistema MovieLens — sistema de recomendação individual de filmes (O'CONNOR et al., 2001).

2.13 Demais Conceitos

2.13.1 RM-ODP

O Modelo de Referência para Processamento Distribuído Aberto (RM-ODP) é um modelo de referência que permite especificar o sistema baseado em 5 visões: empresarial,

informação, computação, engenharia e tecnologia.

2.13.2 Modelo Cliente-Servidor

O Modelo Cliente-Servidor é uma estrutura de aplicação distribuída baseada na segregação de tarefas e cargas de trabalho entre dois lados de uma aplicação, de um lado o servidor que provê um serviço e, do outro lado, o cliente que requisita o serviço. Este modelo é a base da maioria da internet, onde servidores tem seus recursos distribuídos para diversos usuários, enquanto os clientes são locais ao usuário e não dividem seus recursos.

2.13.3 Banco de Dados

Banco de dados, ou database, é uma coleção organizada de dados, geralmente armazenada e acessada por meio de um sistema computacional. Em computação, databases tem o objetivo de prover armazenamento eficiente, organização e acesso rápido a grandes quantidades de dados, se tornando praticamente necessárias em Ciência de Dados e Aprendizado de Máquina.

2.13.4 Modelo de Entidade Relacional

O modelo de entidade relacional é uma forma de representar a relação de informação de um sistema. Esses modelos são muito utilizados na construção de banco de dados ao permitir que se visualize o relacionamento entre as entidades e o acesso aos atributos.

2.13.5 HTTP

HTTP (*Hypertext Transfer Protocol*) é um protocolo de comunicação, da camada de aplicação do modelo OSI, para sistemas de informação de hipermídia, distribuídos e colaborativos (FIELDING et al., 1999).
O protocolo foi fundamental para a existência da internet e seu funcionamento é baseado no método de requisição-resposta dentro de um modelo cliente-servidor.

2.13.6 Web API

Web API (*Application Programming Interface*) é um interface que possibilita a comunicação entre dois sistemas na web ao expor os serviços de um sistema por meio dessa interface. Essa troca é feita por um padrão de troca de informação, como o JSON. Dentre os diferentes estilos de arquiteturas para serviços web, esse projeto irá utilizar o estilo REST.

2.13.7 JSON

JSON (*JavaScript Object Notation*) é um padrão para troca de informação amplamente utilizado na computação. O padrão define um formato de texto de fácil leitura e escrita por humanos, assim como de fácil criação e analise por máquinas.
Como é representada em texto, não possui nenhuma dependência com qualquer linguagem de programação, sendo assim, uma das melhores formas de se trocar informações entre sistemas diferentes. O JSON utiliza duas estruturas para representar um dado, um coleção de pares atributo-valor e uma lista ordenada de valores (JSON, 2018).
Abaixo um exemplo no formato JSON.

```
1 {"formato_json": {
2   "atributo": "valor",
3   "lista_ordenada": ["valor", "valor", "valor"]
4 }}
```

2.13.8 REST

O estilo de arquitetura REST (*Representational State Transfer*) define um conjunto de restrições para a criação de serviços web. Esses serviços permitem que outros sistemas possam consumir os recursos, então disponibilizado pelo servidor, através de operações previamente definidas e sem estados. As principais operações da arquitetura REST são *POST*, *GET*, *UPDATE* e *DELETE*.

3 Metodologia

A metodologia seguida por este trabalho é descrita abaixo.

3.1 Revisão da Literatura

Na primeira parte do projeto, foram lidas as referências da literatura sobre os principais temas a serem abordados. No inicio foram buscados *surveys* sobre sistemas de recomendação para entender o estado em que a pesquisa se encontra. Depois, foram buscadas referências mais aprofundadas sobre temas relevantes como recomendação para grupos, modelos híbridos para sistemas de recomendação e formas de avaliar sistemas de recomendação.

3.2 Estudo sobre as Técnicas de Recomendação

Esta etapa abrange as pesquisas feitas sobre sistemas de recomendação, sistemas de recomendação híbridos e sistemas de recomendação para grupos. Nesta parte do projeto definimos quais técnicas seriam utilizadas e como integrá-las.

3.3 Obtenção das Bases de Dados

Para realização dos treinos e testes, faz-se necessário uma base de dados extensa, portanto, esta etapa consistiu na coleta dos dados que seriam utilizados no desenvolvimento dos agentes e na estruturação de uma base de dados que atendesse às necessidades do projeto.

3.4 Desenvolvimento dos Agentes Recomendadores

Para desenvolvimento dos agentes de recomendação, partimos de técnicas bem estabelecidas de recomendação e utilizamos a base de dados criada na etapa anterior para executar os treinos e testes necessários ao seu desenvolvimento. Como o propósito do trabalho é ser uma plataforma horizontalmente escalável, implementamos os agentes separadamente de forma modular.

3.5 Desenvolvimento do Mecanismo do Modelo Híbrido

Esta etapa consistiu em criar um algoritmo capaz de misturar diversas recomendações de diferentes agentes e de diferentes usuários para criar uma recomendação unificada para um grupo.

3.6 Desenvolvimento Cliente-Servidor

Buscando criar uma arquitetura viável para a plataforma digital. O sistema foi pensado usando o modelo cliente-servidor. Assim, todo o processo de recomendação ficou dentro do servidor. Nele também estaria o acesso as fontes de dados (interna e externa). Por fim, também seria criado uma API para que o cliente pudesse se comunicar com o servidor.

O cliente seria o sistema de interação com o usuário, por ele seria requisitado as funcionalidades desejadas (logar no sistema, criar uma sala, pedir uma recomendação, etc) que então seriam requisitas na API criada no servidor.

4 Especificação

Neste capítulo será apresentada a proposta deste trabalho e sua especificação. Para o desenvolvimento dessa especificação, optamos por usar o Modelo de Referência para Processamento Distribuído Aberto (RM-ODP).

4.1 Visão Empresarial

Atualmente existem diversas serviços de musica no mercado, a grande maioria deles investem alto em sistemas de recomendação individual para gerar valor ao seus clientes. Entretanto esses serviços não disponibilizam nenhum sistema de recomendação para grupos de pessoas, uma dor muito visível em momentos de convívio social quando se quer ouvir músicas, mas não se consegue chegar a um consenso — o que muitas vezes leva a uma insatisfação de boa parte das pessoas presentes. Desse modo, a plataforma seria capaz de melhorar o nível de satisfação das pessoas nessas situações ao facilitar a decisão de escolha e a possibilitar a descoberta de novas músicas entre os participantes na recomendação de músicas.

Do ponto de vista comercial, a plataforma digital, busca se estabelecer como uma fonte de informação sobre o uso, costumes e preferências de grupos ao ouvirem músicas juntos. Essas informações coletadas são de muito valor e diversas finalidades poderiam ser dadas a elas.

Os serviços de musica — como Spotify, Deezer, Google Música, Youtube — desejaria estar integrados na plataforma, para que as recomendações geradas para grupos fossem tocadas e salvas em suas próprias plataformas, podendo aumentar o uso de seus serviços e de usuários em seus sistemas. Por exemplo, pode-se integrar as recomendações criadas com o serviço do Spotify para que essas recomendações possam ser acessada internamente pelo serviço. Mas nada impede que essa mesma lista de reprodução pudesse ser executada em outro serviço como o Youtube. Outro motivo de interesse é a possibilidade de usarem a plataforma para obter informações exclusivas sobre os grupos, como por exemplo, o relacionamento de diferentes usuários — que pode ser utilizado para aprimorar o seus sistemas de recomendação.

A plataforma também seria de muito interesse para outros atores da industria musical. A parte da indústria focada no mercado de música ao vivo (Show Business), como as danceterias, boates e similares poderiam usar a plataforma para devolver uma experiência mais direcionada ao seu público. Por exemplo, uma danceteria poderia criar um grupo na plataforma e incentivar os seus clientes a entrarem nesse grupo. Assim, essa danceteria

obteria informações muito relevantes sobre o gosto musical de seus cliente adequando a musica que tocaria em seu estabelecimento. Da mesma forma, os cliente frequentes também poderiam acessar as músicas a qualquer momento. Até mesmo uma rádio, poderia se beneficiar ao usar a plataforma agregando seus ouvintes em um grupo e podendo entender as preferências de seus ouvintes. As rádios e parte da indústria fonográfica, também se interessariam na plataforma por informações que ajudassem a tomar decisões estratégicas, como por exemplo, ao entender a infiltração de determinado artista ou gênero musical por segmentação demográfica.

A figura 2 mostra o papel desempenhado pela plataforma proposta. Na horizontal vemos que ela disponibilizaria a grupos de usuários listas de reprodução musical (*playlists*). Na vertical, a ideia da plataforma disponiblizar inteligência a potenciais clientes da industria musical. Também foi proposto um nome e uma identidade visual para a plataforma, brincando com as palavras identidade (*id*) e ideal (*ideal*) para trazer a ideia de recomendações personalizadas a cada pessoa e com a melhor precisão.

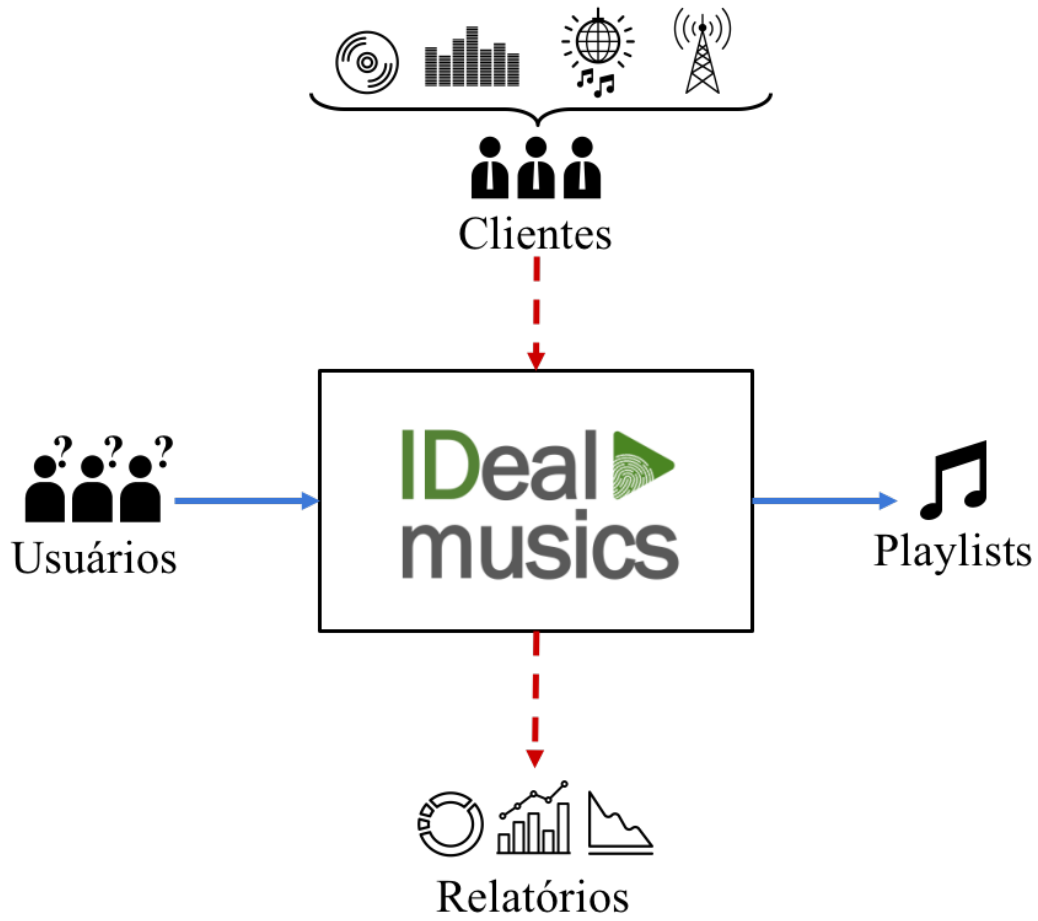

Figura 2: Representação da plataforma IDealMusic.

4.2 Visão de Informação

Na Figura 3 é possível visualizar o Diagrama de Entidade Relacionamento mínimo para o banco de dados interno. Nesse diagrama, são contempladas as principais informações necessárias para o funcionamento da plataforma. Assim, um usuário possui avaliações (implícitas ou explicitas) sobre músicas e pode participar de diversos grupos. Cada grupo pode gerar uma *playlist* — a partir do relacionamento dos usuários com suas as músicas e artistas preferidos.

Ao dizer que esse diagrama é mínimo, significa que dependendo da fonte de dados externa a ser acoplada na plataforma, novos atributos possam ser agregados nas entidades e novas funcionalidades de recomendação sejam possíveis de serem feitas. Assim, esse diagrama pretende expor as funcionalidade que serão contempladas independentemente das as fontes externas acopladas.

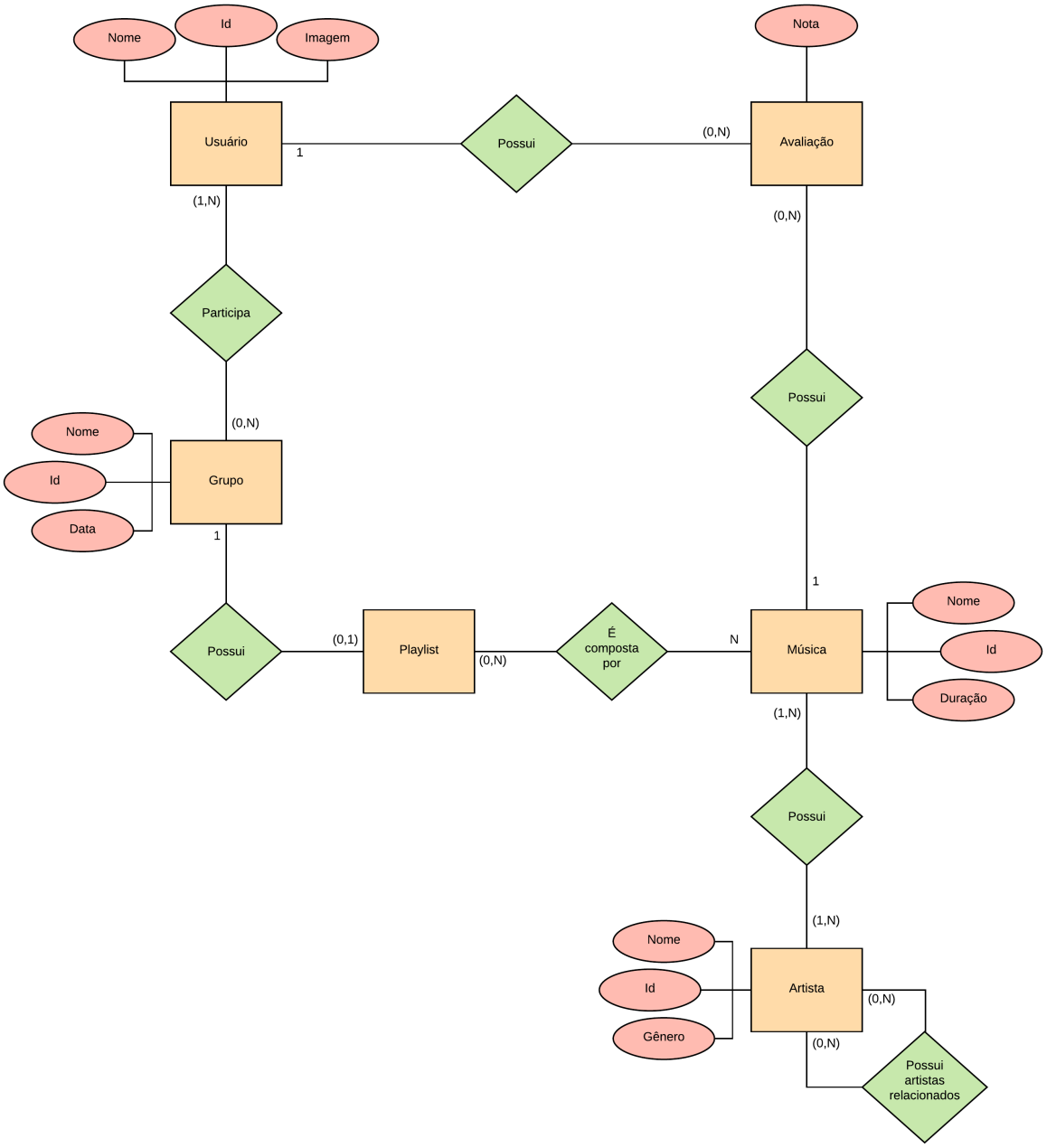

Figura 3: Diagrama de Entidade Relacional.

4.3 Visão Computacional

Na Figura 4 é possível visualizar a arquitetura em três camadas do sistema. Também é possível ver as relações de dependências entre as camadas, fontes de dados e usuários. A seguir será descrito os módulos de cada camada.

4.3.1 Camada de Apresentação

Na Camada de Apresentação temos os módulos de Interface do Usuário e Lógica de Apresentação. O primeiro módulo é responsável por permitir a interação do usuário com todo o sistema. Já o segundo, é responsável por controlar as funcionalidades e as transições das interfaces durante a interação do usuário. Dentro do ponto vista do modelo Cliente-Servidor, essa camada é a representação da Aplicação-Cliente.

4.3.1.1 Requisitos Funcionais do Módulo de Interface do Usuário

RF-INT-1: Permitir o usuário a realizar a autenticação na plataforma (*Log in*).

RF-INT-2: Permitir o usuário a sair da plataforma (*Log out*).

RF-INT-3: Permitir o usuário a acessar grupos criados na plataforma.

RF-INT-4: Permitir o usuário a criar grupos.

RF-INT-5: Permitir o usuário com perfil de anfitrião a iniciar a recomendação dentro do grupo.

RF-INT-6: Permitir a visualização dos participantes do grupo.

RF-INT-7: Permitir a visualização da *playlist* gerada no grupo.

RF-INT-8: Permitir a a execução da *playlist* em algum sistema externo.

4.3.2 Camada de Lógica de Negócio

Na Camada de Lógica de Negócio temos os módulos de Funções de Recomendação, Funções sobre Grupos e Funções Administrativas. O primeiro módulo dessa camada é responsável por gerar as recomendações solicitadas por grupos de usuários. O segundo módulo é responsável por executar funcionalidades de controle sobre os grupos de usuários, como a criação de grupos, busca dos participantes de um grupos e busca da *playlist* de um grupo. O último módulo é responsável por funções de autenticação dos usuários enquanto utilizam a sistema.

4.3.3 Camada de Dados

Na Camada de Dados temos o módulo de Acesso as Fontes de Dados, que permite requisitar informações das fontes internas e externas do sistema e desacoplando essas fontes de dados da Camada de Lógica de Negócio.

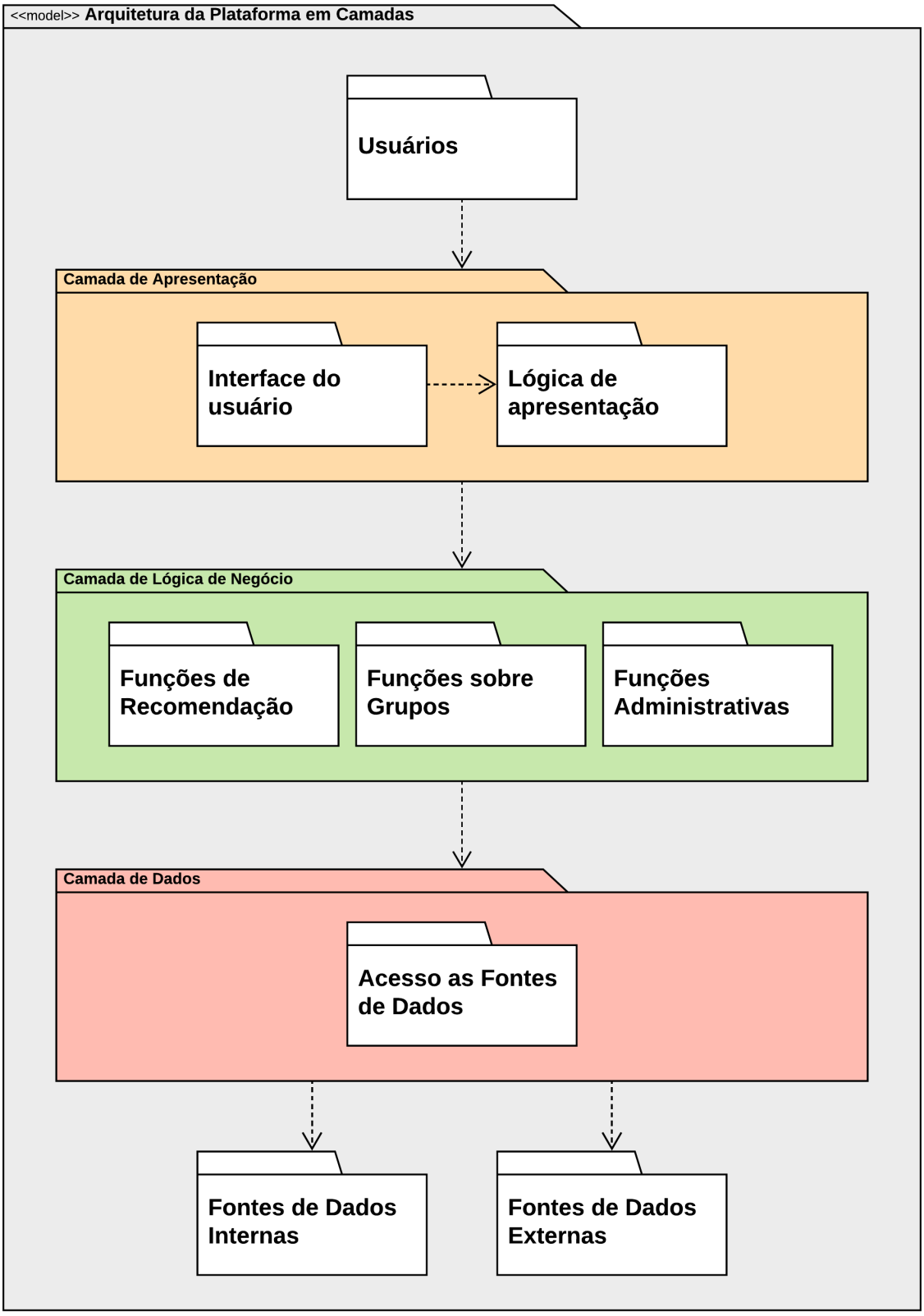

Figura 4: Arquitetura em três camadas.

4.4 Casos de Uso

Nesta seção são apresentados os casos de uso padrão do sistema. As tabelas 4.4 a 4.4 apresentam os fluxos relacionados com as atividades básicas do projeto.

Nome	Autenticar Usuário
Atores	Usuário
Descrição	Usuário faz autenticação na API do Spotify
Requisitos Relacionados	RF2, RF3
Pré-Condição	Nenhuma
Fluxo Básico de Eventos	
Ações do Ator	**Ações do Sistema**
1. Acessa a interface do sistema	
2. Entra com os dados de login	3. Sistema se conecta com o Spotify
	4. Sistema pede permissões ao Usuário
5. Aceita permissões	6. Sistema mostra as informações públicas do usuário

Tabela 1: Caso de Uso 1 - Autenticação.

Nome	Criar Playlist
Atores	Usuário
Descrição	Usuário cria uma sala para gerar uma playlist
Requisitos Relacionados	RF1, RF5
Pré-Condição	Estar autenticado no sistema
Fluxo Básico de Eventos	
Ações do Ator	**Ações do Sistema**
1. Cria um nova sala	
2. Convida outros usuários	3. Sistema informa convidados
4. Convidados entram na sala	
5. Dono da sala gera a playlist	6. Sistema gera uma playlist para o grupo de usuários

Tabela 2: Caso de Uso 2 - Criação de Playlist.

Nome	Acessar Histórico de Playlists
Atores	Usuário
Descrição	Usuário acessa o histórico de playlists
Requisitos Relacionados	RF4, RF5
Pré-Condição	Estar autenticado no sistema
Fluxo Básico de Eventos	
Ações do Ator	**Ações do Sistema**
1. Acessa o histórico de playlists	2. Sistema mostra histórico de playlists
3. Seleciona uma playlst	4. Sistema carrega a playlist
	5. Sistema acessa o spotify para reproduzir a playlist

Tabela 3: Caso de Uso 3 - Acesso ao Histórico de Playlists.

4.5 Visão de Engenharia

Na figura 5, temos o diagrama da comunicação do sistema distribuído. A Camada de Apresentação ficará no servidor web e será requisitada pelos dispositivos de cada usuário. Já a Camada de Regra de Negócio ficará no Servidor da Aplicação e irá responder as solicitações da camada de Apresentação. A Camada de Acesso aos Dados também ficara no servidor de aplicação, essa camada se comunicará com as fontes de dados, tanto interna com as externas. O padrão para a comunicação de dados será o JSON que é amplamente utilizado atualmente.

Assim, como exemplo, quando um grupo de usuários solicita uma recomendação, a Aplicação-Cliente faz uma requisição HTTP ao *endpoint* específico do *Web Services* que expõe os serviços da Aplicação-Servidor. Por sua vez, a Aplicação-Servidor é responsável por buscar as informações necessárias, sobre os usuários e suas preferências, nas Fontes de dados externas e interna — ambas comunicações, também por requisições HTTP aos *endpoint* dos *Web Services*.

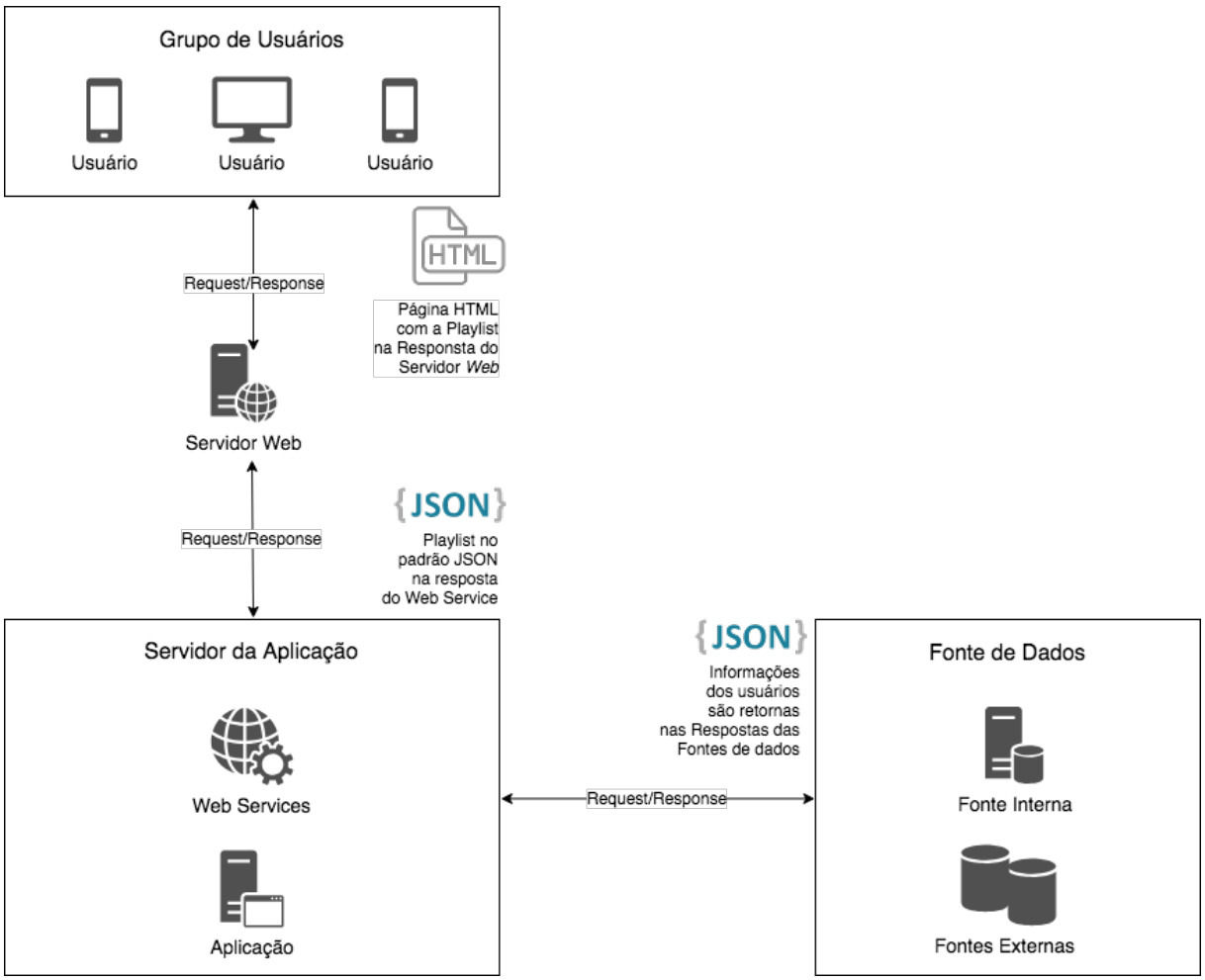

Figura 5: Diagrama da troca de informação no sistema para a recomendação de uma *playlist*.

4.6 Visão de Tecnologia

A implementação do projeto será divida em duas partes: implementação do servidor e implementação do cliente.

4.6.1 Servidor

O servidor será divido na implementação da lógica de recomendação, na implementação do acesso aos dados e na implementação da API para consumo da aplicação cliente. Será utilizado a linguagem Python para o servidor e o banco de dados será o MongoDB. A base de dados externo para obtermos as preferências dos usuários será a API do Spotify (SPOTIFY, 2018). Para rápida integração, usaremos a biblioteca em python para consumir a API do Spotify (LAMERE, 2014).

4.6.2 Cliente

O cliente será desenvolvido em HTML, CSS e JavaScript. Ele terá inicialmente três páginas: página inicial, página de salas e detalhe do sala. O termo 'Salas' será usado dentro da interface para uma comunicação mais amigável com o usuário, esse termo possui o mesmo significado de grupos que está sendo utilizado nesse trabalho.

Página inicial Nessa tela teremos as informações principais sobre a plataforma. Também terá uma forma de o usuário se autenticar usando uma conta do serviço de música Spotify.

Página de Salas Nessa tela será possível visualizar todos as salas criadas (grupos) e também será possível criar uma nova sala.

Detalhe da Sala Nessa tela será possível ver os detalhe de uma sala (grupo) como os participantes e a lista de reprodução, caso tenha sido gerada. Se ainda não foi gerada, é possível requisitar a geração da lista de reprodução desse grupo.

5 Implementação

A implementação do projeto foi feita em Python e foi dividida em duas partes: implementação do servidor e implementação do cliente. O servidor foi dividido entre implementação da lógica de recomendação e implementação da API para consumo pelo cliente.

5.1 Busca de Bases de Dados

Sistemas de recomendação precisam de bases extensas, especialmente quando se trata de uma recomendação colaborativa ou baseada em conteúdo, caso contrário o sistema pode sofrer do problema *cold-start*. Para contornar tal problema, optamos por usar uma base de dados que facilitasse o desenvolvimento de recomendadores, contendo uma grande quantidade de usuários e de músicas.

Cogitamos utilizar o *dataset* da last.fm (LAST.FM, 2011), porém esta não apresentava todas as características que buscávamos, pois os dados eram antigos e não permitiria que usasemos o serviço deles para executar as playlist geradas. Optamos por consumir dados do Spotify (SPOTIFY, 2018) agregando dados de músicas, artistas, usuários e features em um banco MongoDB, localizado no servidor, pra fácil acesso e agilidade de busca.

5.1.1 Consumo da API do Spotify

O Spotify proporciona uma API para consumo dos seus dados e, em python, a comunidade desenvolveu uma biblioteca para facilitar seu uso (LAMERE, 2014). Usando esta biblioteca conseguimos extrair dados do Spotify como:

Users Dados de um usuário, incluindo library pessoal, artistas mais ouvidos, músicas mais ouvidas e perfil.

Tracks Abstração de música composto de diversos metadados, incluindo artistas participantes, popularidade e álbum.

Artists Abstração de um artista composto de diversos metadados, incluindo gêneros relacionados, músicas mais ouvidas, artistas relacionados e popularidade.

Features Análise de uma música contendo diversos valores variando de 0 a 1 que representam características da música, como dancability e acousticness.

Como as requisições à API do Spotify são limitadas por quantidade de itens trafegados, optamos por agregar todos os dados necessários na base de dados, proporcionando uma consulta mais ágil e permitindo uma rápida expansão do conteúdo da plataforma. Conforme a nossa base de dados ganhe massa crítica, podemos integrar outros serviços como Apple Music, Deezer e Play Music, já que o problema do cold-start estará superado e não dependeremos mais dos dados providos pelo Spotify para recomendar músicas.

5.2 Desenvolvimento Aplicação-Servidor

Para desenvolvimento do código, optamos por começar usando o Jupyter Notebook (JUPYTER, 2018), uma ferramenta muito usada para desenvolvimento em Python com ênfase em análise de dados. Esta ferramenta proporciona uma rápida prototipação, nos permitindo grande agilidade nas pesquisas e testes relacionados tanto ao desenvolvimento da integração com o Spotify quanto à implementação dos agentes recomendadores.

A implementação do cerne do sistema de recomendação é o próprio recomendador, portanto, o dividimos em quatro partes para pesquisa e implementação. Segue abaixo as quatro partes em detalhe.

5.2.1 Integração com o Spotify

Para integrarmos nosso sistema com o Spotify, fez-se necessário uma pesquisa dos serviços providos pela API para levantamento dos endpoints relevantes e análise da estrutura de dados retornada. Após esta pesquisa começamos a implementar métodos para uso dos recursos da API e a popular nossa base de dados que seria usada nas próximas etapas da implementação.

5.2.2 Agente Recomendador Baseado em Clusterização por Features

Este agente tem como base o fato de que músicas parecidas apresentam features similares, portanto, se dividirmos o espaço amostral em *clusters*, podemos identificar esferas de preferência de um usuário e recomendar com base nisso.

O agente foi implementado usando o método de clusterização KMeans (PIECH; NG, 2013), um método simples mas eficaz de clusterização de dados não classificados. Como fonte de dados extraímos as features mais relevantes tanto das músicas da base de dados quanto das músicas do usuário, gerando um vetor de 7 dimensões para cada música.

O algoritmo se inicia pela clusterização das músicas da biblioteca do usuário, com o objetivo de achar os maiores e mais relevantes *clusters*. Após a geração dos *clusters*, o algoritmo analisa-os por meio da quantidade de músicas, desvio padrão de cada feature e

popularidade das músicas presentes. Na figura 6 podemos observar como as músicas ficam distribuídas em torno de um centroide.

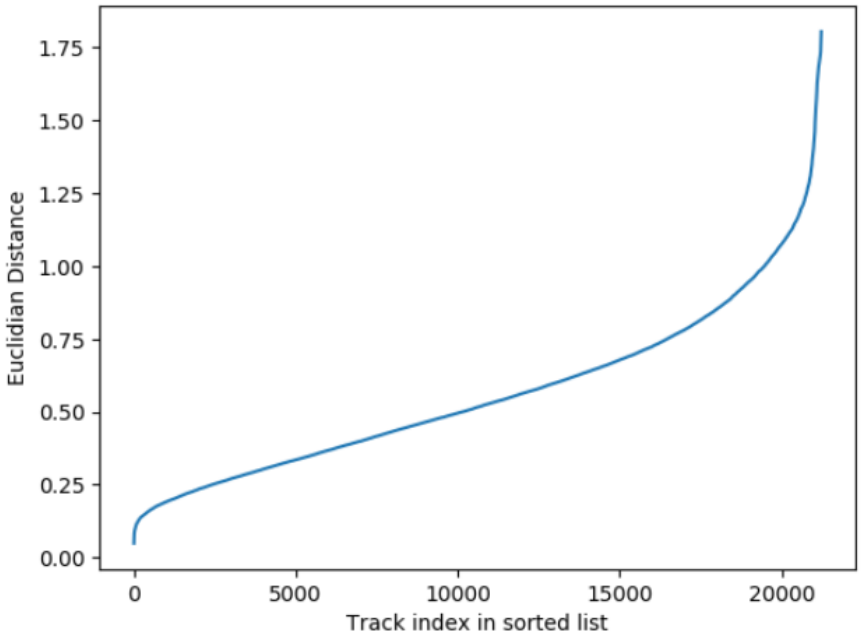

Figura 6: Distribuição de distâncias euclidianas em relação ao centroide de um *cluster*.

A partir dos *clusters* organizados por relevância, podemos recomendar músicas a partir do centroide dos *clusters*, usando uma função de avaliação por música que leva em conta metadados da música, sua popularidade e sua posição no espaço relativo ao centroide. Esta recomendação se provou bastante eficiente em gerar playlists com músicas similares em experiência, porém, cerca de 10 a 20% das músicas desviam razoavelmente de estilo musical.

5.2.3 Agente Recomendador Baseado em Conteúdo

A base de um recomendador baseado em conteúdo geralmente é a recomendação de items similares usando metadados. Para este recomendador optamos por focar nos gêneros músicais, com o objetivo de obtermos um recomendador mais consistente que o primeiro e que, quando eles fossem unidos pelo Moderador, houvesse uma sinergia tanto de experiência das músicas quanto de gênero.

Apesar das músicas não conterem explicitamente metadados sobre o seu gênero, podemos inferir este dado a partir dos artistas participantes. Notamos que os artistas participantes tem relação direta com o estilo da música, até em composições com diversos artistas.

O agente inicia extraindo gêneros das músicas da biblioteca e das músicas mais ouvidas do usuário e simplificando a lista para minimizar o efeito de gêneros muito obs-

curos. Por exemplo, *Brazillian Raggae* seria separado em *Brazil* e *Raggae*, enquanto *Dark Metal* seria simplificado para *Metal*.

Partindo da lista de gêneros, podemos elencá-los para obter os gêneros que o usuário prefere e, a partir dos mais comuns, podemos recomendar músicas similares. O agente então passa para a fase de pesquisa, onde são buscadas músicas na base de dados e no Spotify que apresentem os gêneros desejados.

Após a geração das músicas relevantes, o agente passa para a fase de avaliação e recomendação, onde todas as músicas levantadas são analisadas usando seus metadados, artistas envolvidos, gêneros e popularidade para gerar uma nota única de 0 a 1, para cada música, que define a confiança do método na recomendação. Então são retornadas como a recomendação do método as músicas com confiança mais alta.

5.2.4 Moderador

Após a pesquisa inicial do projeto, concluímos que o melhor método para implementar um recomendador para múltiplos usuários seria por meio de um Modelo Híbrido Cooperativo (BURKE, 2007). Tal modelo parte de diversos agentes de recomendação, executados para cada usuário de forma a unificar seus resultados em uma recomendação única. Escolhemos um modelo baseado em pesos para fazer a agregação dos resultados, tal modelo é divido em três partes. Segue abaixo as três partes em detalhe.

O algoritmo Moderador, implementação deste modelo, gere a execução geral do sistema de recomendação, sendo o orquestrador da inicialização dos agentes, geração de músicas candidatas a recomendação, valoração das músicas candidatas e, por fim, a combinação ponderada dos valores a a partir dos pesos de cada agente.

5.2.4.1 Fase de Treino

Esta fase consiste no treino dos agentes usando os dados de cada usuário para cada método, preparando os agentes para gerar recomendações. Ao final desta fase temos os agentes configurados com os perfis dos usuários e podemos gerar seus candidatos a recomendação.

5.2.4.2 Geração de Candidatos

A partir dos agentes treinados na fase anterior, geramos recomendações com cada um dos N agentes, para cada um dos M usuários, resultando em $N \cdot M$ listas de músicas, que serão os candidatos finais a recomendação pelo Moderador. Como podem haver intersecções entre as listas é importante eliminar ocorrências repetidas para garantir o funcionamento do método.

Ao final desta fase temos uma extensa lista de músicas candidatas que não apresentam grande coerência, porém contém todas as músicas relevantes para a parte final de pontuação e recomendação.

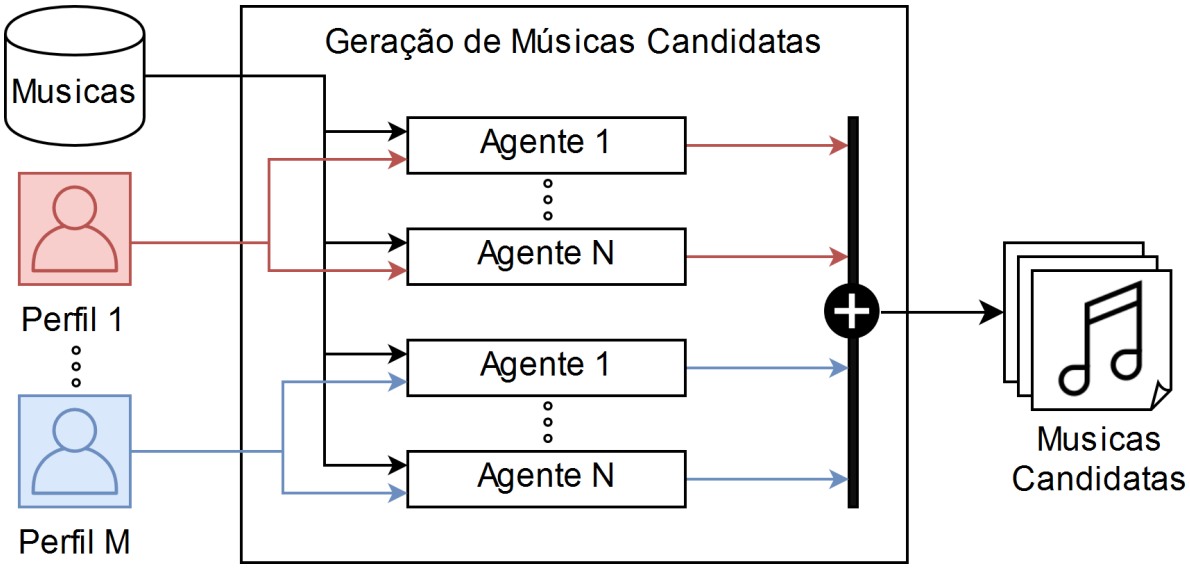

Figura 7: Diagrama da geração de músicas candidatas a recomendação.

5.2.4.3 Pontuação dos Candidatos

A maioria das músicas contidas na lista de candidatos contém só a pontuação respectiva para cada execução que a gerou, portanto devemos executar a pontuação das músicas para cada um dos N agentes e M usuários, de forma a obter $N \cdot M$ valores de 0 a 1 para cada música candidata.

O próximo passo é agregar os valores gerados usando uma combinação ponderada baseada em pesos dos agentes. Os pesos poderiam ser definidos dinamicamente, utilizando a confiança média de cada agente ou por meio de feedbacks dos usuários, porém por simplicidade, optamos por utilizar valores iniciais estáticos obtidos por testes empíricos das recomendações, que são ajustados a medida que os usuários indicam se uma música recomendada foi satisfatória ou não.

Ao final da execução do moderador, conseguimos obter uma lista de músicas que leva em conta tanto as preferências individuais de cada usuário quanto a harmonia entre os gostos do grupo. Na figura 8 está exemplificado o funcionamento desta etapa.

5.2.5 Desenvolvimento da API

O desenvolvimento da API foi feito em paralelo com o cliente da aplicação, provendo os endpoints necessários para o funcionamento da interface Web. Entre os endpoints criados, estão chamadas de autenticação, adição de usuários a uma sala e chamadas para

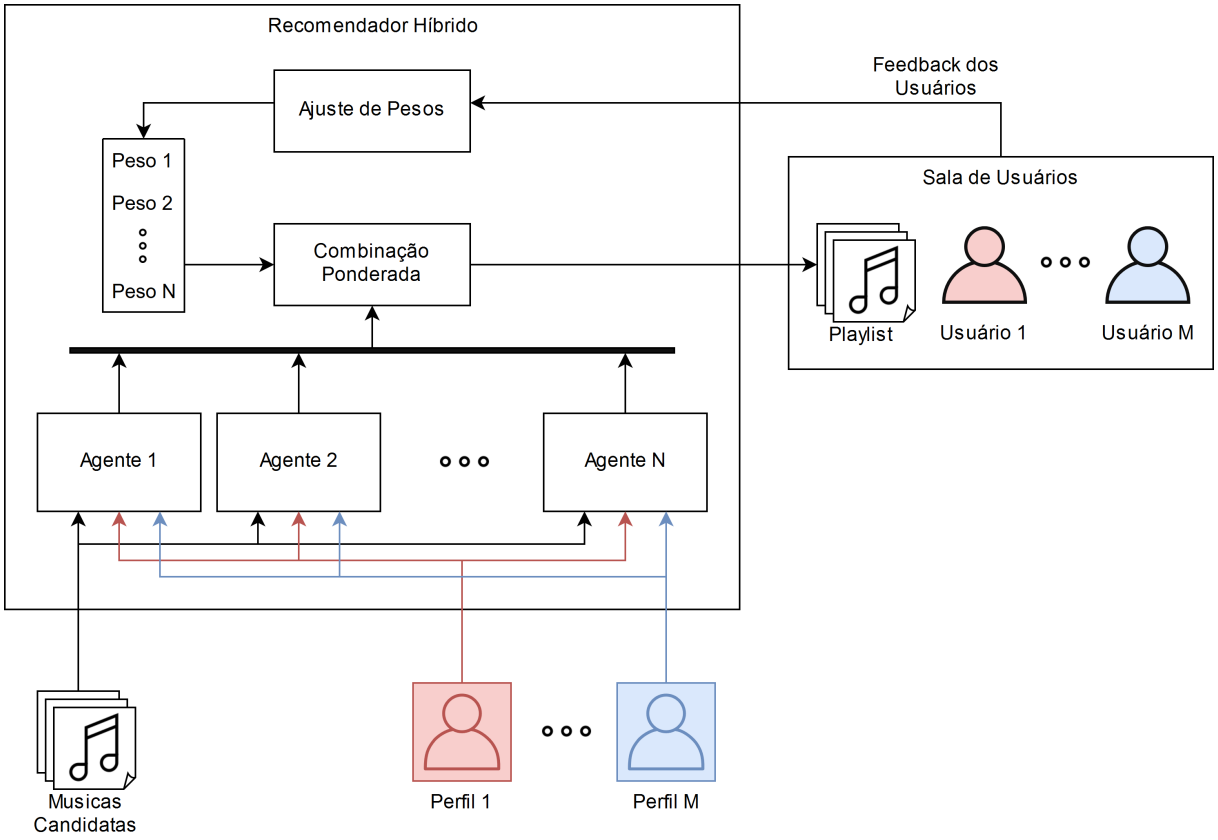

Figura 8: Diagrama do processo de valoração das músicas candidatas.

execução da recomendação. A API foi desenvolvida em Python e serve como a interface entre o cliente e os métodos re recomendação e controle geral de uma sessão.

5.3 Desenvolvimento Aplicação-Cliente

A Aplicação-Cliente foi desenvolvida em HTML, CSS e JavaScript. Optamos por utilizar essas tecnologias pela sua simplicidade e fácil customização. Também optamos por utilizar um template o que facilitou o desenvolvimento da aplicação sem a necessidade de desenvolver as estruturas básicas de um site (Menu, Rodapé, Banner, etc). Com o intuito de demostrar as funcionalidades principais e especificadas neste documento, foram desenvolvidas três páginas na aplicação como demostradas no Apêndice A.

6 Testes e Avaliação

Este capítulo descreve os métodos de avaliação das recomendações e os testes realizados.

6.1 Agentes de Recomendação

Para avaliar as recomendações geradas pelos agentes, usamos 90% das músicas das bibliotecas dos usuários para construir os perfis e calculamos as pontuações esperadas para os 10% restantes. Como a avaliação de uma música varia entre 0 e 1, julgamos que valores acima de 0.5 seriam dados como acertos e valores inferiores como erros.

Em 10 testes, ambos os agentes tiveram acima de 70% de acerto, em média 76% para o agente baseado em clusterização e 81% para o agente baseado em conteúdo, evidenciando que os algoritmos são capazes de gerar recomendações alinhadas com o gosto individual de um usuário.

6.2 Moderador

Como o moderador tem por objetivo unir recomendações de múltiplos agentes e múltiplos usuários, torna-se complexo fazer uma avaliação quantitativa que contemple o alinhamento com os perfis dos usuários e o desempenho dos agentes. Portanto optamos por focar em testes retroativos, além de avaliar seu desempenho pelo mesmo método de avaliação dos agentes.

Ao se avaliar o Moderador usando a mesma métrica aplicada aos agentes, constatou-se em 10 testes uma taxa de acerto média de 57%, porém, como o valor único gerado para cada música deve contemplar o perfil de todos os usuários, é natural que músicas que se alinhem com somente um dos perfis tenham valores baixos.

6.3 Testes Retroativos

Para avaliar o desempenho geral do modelo, optamos por fazer dois testes retroativos que consigam avaliar a sinergia entre as músicas e alinhamento entre as músicas geradas e os gêneros preferidos dos usuários. Para realizar estes testes, implementamos dois algoritmos que recebessem o perfil dos usuários do grupo e a lista de músicas recomendadas e, por fim, retornassem quantas músicas da amostra desviavam significativamente do esperado.

O primeiro teste avalia o alinhamento entre as músicas recomendadas e os perfís dos usuários, para isso, faz uso dos gêneros mais ouvidos pelos usuários, avaliando quais músicas não apresentam estes gêneros. Constatamos que cerca de 10% das músicas das playlists geradas desviavam do esperado.

O segundo teste avalia a sinergia das playlists geradas, calculando a variância das features das músicas, ou seja, sua dispersão. Este teste foi menos conclusivo devido a variabilidade dos fatores, porém, avaliamos que a dispersão das músicas era menor do que a média dos *clusters* gerados pelo agente baseado em clusterização, indicando que a maioria das músicas foram retiradas de um mesmo *cluster*, comum a ambos os usuários.

Parte III

Considerações Finais

7 Considerações Finais

7.1 Cumprimento dos Objetivos

O objetivos inicial de construir um sistema de recomendação híbrido foi atingido. O algoritmo híbrido de recomendação gerou *playlists* com resultados satisfatórios de similaridade com as preferências dos usuários. Mesmo assim, ainda há espaço para diversas melhorias, como novos algoritmos de recomendação e novas funcionalidades para os usuários.

O cumprimento do segundo objetivo, o de elaborar um plataforma digital, também foi satisfatório e permitiu aos integrantes explorar a criação de um modelo de negócio e de implementar um protótipo para a plataforma digital (aplicação-cliente).

7.2 Contribuições

As pesquisas sobre sistemas de recomendação tendem a crescer devido às diversas possibilidades que esses sistemas permitem. Ainda mais, com os avanços das áreas de inteligência artificial e *machine learning*, os sistemas de recomendação passam a ser mais acessíveis e eficientes, facilitando a sua utilização.

Nesse contexto, este trabalho contribuiu ao implementar um sistema de recomendação híbrido com o intuito de criar sugestões para grupos de pessoas, e não apenas para um único indivíduo. A implementação da arquitetura híbrida, com mecanismo ponderado, pode servir de comparação para novas pesquisas com sistemas híbrido com diferentes mecanismos. Já a implementação da recomendação para grupos, pode gerar uma contribuição para essa área específica, que ainda está começando a ser desenvolvida.

7.3 Trabalhos Futuros

Há diversas possibilidades de evolução desse trabalho, a seguir listamos algumas delas.

- **Aumentar quantidade de recomendadores**

  A criação de novos recomendadores seria uma ótima forma de evoluir este trabalho. Principalmente, se fossem algoritmos de recomendação que trouxessem outras qualidades como diversidade e harmonia. Dado que foi implementado um sistema híbrido, a arquitetura já permite uma rápida integração de outros algoritmos. Tra-

balhos futuros poderiam focar em criar algoritmos de recomendação de filtragem colaborativa, de filtragem social e de filtragem demográfica, que podem trazer recomendações com características diferentes das implementadas.

- **Configurações iniciais da recomendação**

  Outra possibilidade de expansão para este trabalho, seria acrescentar funcionalidades de customização da recomendação pelos usuários. Por exemplo, o grupo poderia escolher um gênero ou características musicais (como energia, dançabilidade, etc) para a afunilar os gêneros da lista de recomendação musical. Outro exemplo, seria ao possibilitar que o usuário definisse a qualidade desejada para a *playlist* e assim e a geração da recomendação selecionaria os recomendadores que aumentassem essa qualidade desejada — por exemplo, se desejassem uma lista de reprodução com maior diversidade, o recomendador poderia aumentar o peso de recomendadores que geram essa qualidade ou retirar recomendadores que aumenta a qualidade de homogeneidade.

- **Integração com outras fontes de dados**

  Há uma grande oportunidade ao se conectar com outros serviços que possam disponibilizar mais informações sobre os usuários. Com essas informações é possível criar novos recomendadores que não seriam possíveis apenas com a API escolhida. Um caso interessante de expansão, seria se conectar a redes sociais e utilizar as informações dos usuários na rede para gerar recomendações baseada em filtragem social, que é um novo tipo de recomendação que começa a cada vez mais ser usada pelos sistemas de recomendação.

- **Avaliação dos usuários**

  Outra forma de melhorar e aumentar esse projeto, seria através de funcionalidades que capturassem a avaliação do usuário de cada música e pudessem aprender com essa informação. Por exemplo, se cada usuário pudesse dar uma nota para cada música sugerida, o sistema poderia responder numa próxima recomendação retirando determinado artista ou gênero.

- **Testes com grupos de usuários**

  Os testes dos sistemas de recomendação podem ser mais significantes se testes com grupos de usuários em contextos específicos fossem realizados. Dado a dificuldade de se criar esses contexto e de realizar esses testes com diversos usuários, essa seria

uma interessante expansão para esse projeto.

7.4 Conclusões

Na atualidade, os sistemas de recomendação estão se tornando cada vez mais relevantes. Esses sistemas são capazes de alterar por completo a experiência final dos usuários e representam uma grande chance de aumentar o valor gerado por diversos serviços de tecnologia. Assim, esse projeto buscou implementar um sistema de recomendação de músicas para grupos de pessoas e de elaborar a ideia de uma plataforma digital que utilizaria esse sistema para fornecer inteligência a interessados do mercado da música. De modo geral, os resultados obtidos foram satisfatórios pelo sistema de recomendação e a implementação da aplicação-cliente possibilitou simular o funcionamento da plataforma digital e a aprofundar na criação do modelo de negócio viável.

No mais, o trabalho se mostrou de muita importância para que os integrantes tivessem uma experiência mais aprofundada nas área de inteligência artificial e *machine learning* que, atualmente, se apresentam como tendências muito relevantes no campo da computação. O trabalho também foi muito significativo ao possibilitar a recapitulação e consolidação de diverso conceitos aprendidos durante a graduação e que puderam ser usados para um único objetivo. Principalmente, os conceitos relacionados à banco de dados, engenharia de software, engenharia de sistemas, sistemas de informação e empreendedorismo. Por fim, o aprendizado obtido sobre a área de sistemas de recomendação também se revelou como algo interessante e relevante para a formação acadêmica.

Referências

ALI, I.; KIM, S.-W. Group recommendations: approaches and evaluation. In: ACM. *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*. [S.l.], 2015. p. 105. Citado na página 19.

BOBADILLA, J. et al. Recommender systems survey. *Knowledge-based systems*, Elsevier, v. 46, p. 109–132, 2013. Citado na página 17.

BONNIN, G.; JANNACH, D. Automated generation of music playlists: Survey and experiments. *ACM Computing Surveys (CSUR)*, ACM, v. 47, n. 2, p. 26, 2015. Citado 2 vezes nas páginas 18 e 19.

BURKE, R. Hybrid web recommender systems. In: *The adaptive web*. [S.l.]: Springer, 2007. p. 377–408. Citado 4 vezes nas páginas 3, 15, 16 e 36.

FIELDING, R. T. et al. *Hypertext Transfer Protocol – HTTP/1.1*. [S.l.], 1999. Disponível em: <https://tools.ietf.org/html/rfc2616ref-39>. Citado na página 20.

FRANCESCO, R.; LIOR, R.; BRACHA, S. *Introduction to Recommender Systems Handbook, RecommenderSystems Handbook*. [S.l.]: Amerika Serikat: Springer, 2011. Citado na página 10.

JSON. *Introducing JSON*. 2018. Disponível em: <https://www.json.org>. Citado na página 21.

JUPYTER. *Jupyter Notebook*. 2018. Disponível em: <http://jupyter.org/>. Citado na página 34.

LAMERE, P. *spotipy*. 2014. Disponível em: <https://spotipy.readthedocs.io>. Citado 2 vezes nas páginas 32 e 33.

LAST.FM. *The Last.fm Dataset*. 2011. Disponível em: <https://labrosa.ee.columbia-.edu/millionsong/lastfm>. Citado na página 33.

LININGTON, P. F. et al. *Building enterprise systems with ODP: an introduction to open distributed processing*. [S.l.]: Chapman and Hall/CRC, 2011. Nenhuma citação no texto.

LU, J. et al. Recommender system application developments: a survey. *Decision Support Systems*, Elsevier, v. 74, p. 12–32, 2015. Citado na página 10.

MCFEE, B. et al. The million song dataset challenge. In: ACM. *Proceedings of the 21st International Conference on World Wide Web*. [S.l.], 2012. p. 909–916. Citado na página 19.

O'CONNOR, M. et al. Polylens: a recommender system for groups of users. In: SPRINGER. *ECSCW 2001*. [S.l.], 2001. p. 199–218. Citado na página 19.

PIECH, C.; NG, A. *K Means*. 2013. Disponível em: <http://stanford.edu/~cpiech/cs221-/handouts/kmeans.html>. Citado na página 34.

RAYMOND, K. Reference model of open distributed processing (rm-odp): Introduction. In: *Open distributed processing*. [S.l.]: Springer, 1995. p. 3–14. Nenhuma citação no texto.

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. [S.l.]: Prentice Hall, 2010. VIII p. Citado na página 13.

SPOTIFY. *Spotify Web API*. 2018. Disponível em: <https://developer.spotify.com-/documentation/web-api/reference/>. Citado 2 vezes nas páginas 32 e 33.

WANG, P. What do you mean by "ai"? *Artificial General Intelligence*, p. 362–373, 2008. Citado na página 13.

Apêndices

APÊNDICE A – Interface do Usuário

Figura 9: Página Inicial - Parte superior.

Figura 10: Página Inicial - Parte inferior.

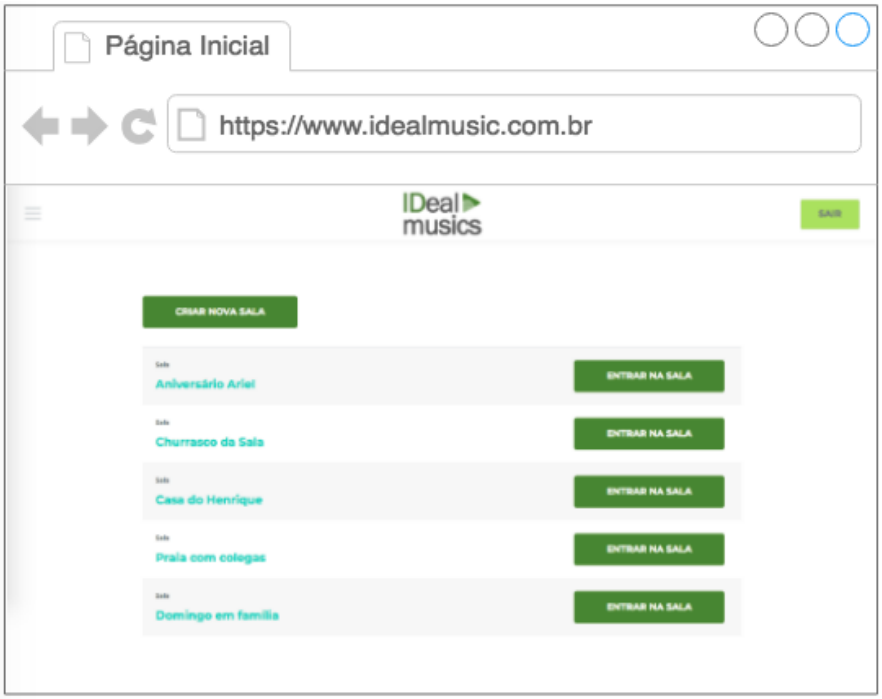

Figura 11: Listagem e criação de grupos (salas).

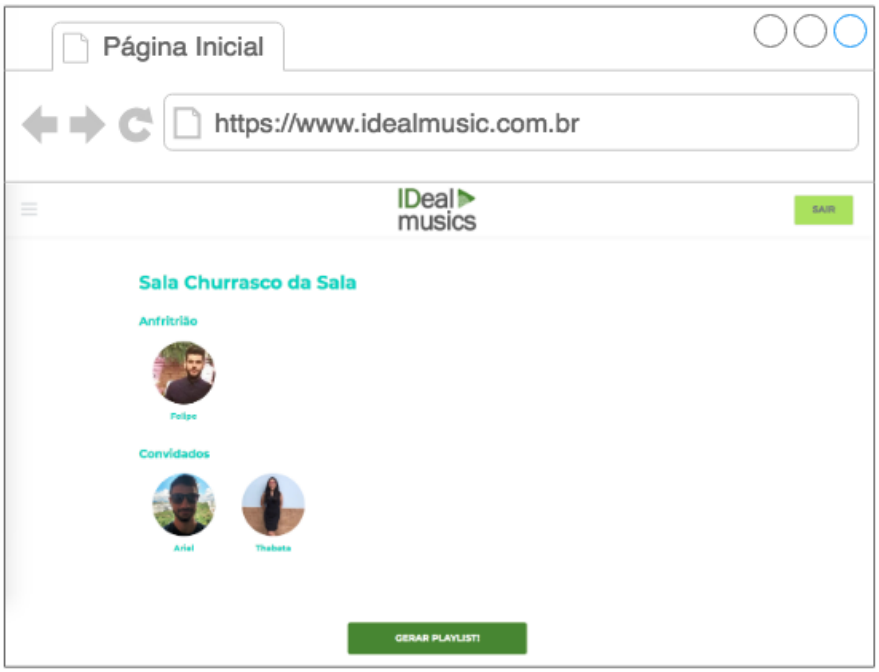

Figura 12: Página de um grupo (Churrasco da Sala) - Parte superior.

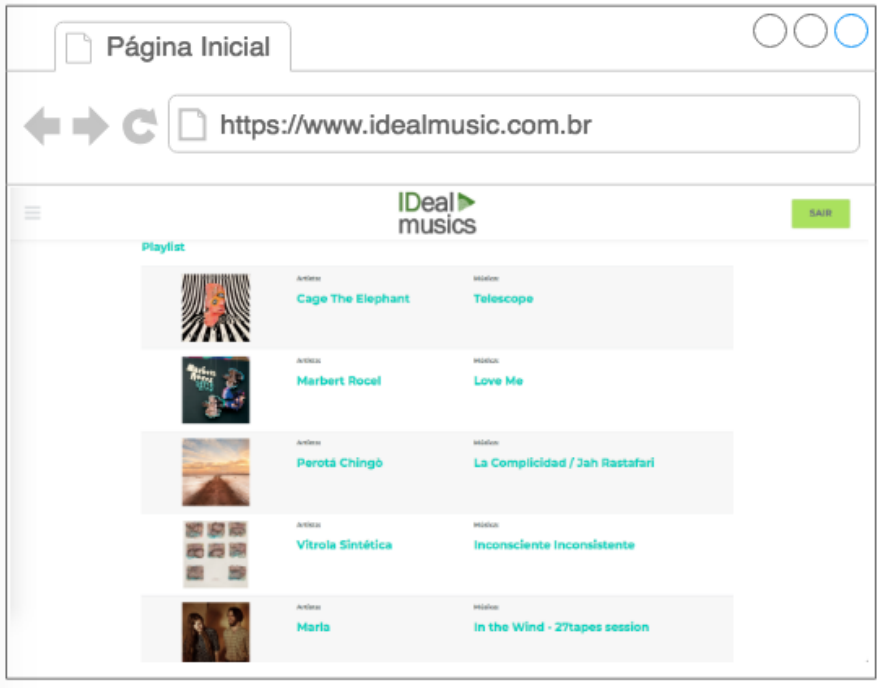

Figura 13: Página de um grupo (Churrasco da Sala) - Parte inferior.