INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Sistema de Exploração de Venues

CARLOS ANDRÉ CASTILHO OLIVEIRA
Outubro de 2017

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Sistema de Exploração de Venues

Carlos André Castilho Oliveira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas de Informação e Conhecimento

Orientadora: Fátima Rodrigues

Porto, Outubro 2017

Resumo

Encontrar venues adequadas a um artista pode ser demorado. Existem muitas venues espalhadas por todo o mundo, e uma forma de filtrar todas essas venues permitiria aos agentes de artistas, encontrar, rapidamente, as venues adequadas ao artista que representa.

A Musicverb disponibiliza uma plataforma web onde os atores da indústria da música ao vivo colaboram e partilham informação.

Este documento descreve a implementação de um Sistema de Exploração de Venues, que usa os dados disponíveis para prever as melhores venues para um artista. Este sistema implementa um sistema de recomendação híbrido com algumas funcionalidades adicionais para melhorar a integração com a plataforma da Musicverb.

O sistema de recomendação proposto usa uma abordagem híbrida entre uma abordagem baseada em conteúdo e uma abordagem colaborativa, usando uma adaptação para este domínio do algoritmo k-NN (PKNN) e um algoritmo colaborativo baseado na similaridade entre utilizadores (CFUB).

Este sistema é integrado na plataforma web da Musicverb, logo os tempos de execução dos algoritmos são muito importantes. Em conclusão, o sistema proposto obteve um bom desempenho, com tempos de execução adequados.

Palavras-chave: Sistema de Recomendação, *Venues*, Música ao Vivo, Agentes, Artistas

Abstract

Finding new venues suitable for an artist to perform at can be cumbersome. There are a lot of venues around the world, and a way to filter all those venues would allow the artist's agent to, quickly, find what he was looking for.

Musicverb provides a web platform where actors of the live music industry can come together and work collaboratively.

To help both parties, this document describes the implementation of a Venue Explorer system, using the available data to predict the best venues for a given artist. This system implements a hybrid recommender system with some additional functionalities to integrate the system into the Musicverb's platform.

The proposed recommender system implements a hybrid approach between content based and collaborative filtering, using a domain specific k-NN algorithm (PKNN) and a user based collaborative filtering algorithm (CFUB).

This recommender system is to be integrated with Musicverb's web platform, therefor execution times are very important. In conclusion, the proposed system performed well, with the adequate execute times.

Keywords: Recommender System, Venues, Live Music, Agents, Artists

Índice

Lista de Figuras

Lista de Tabelas

Glossário

MVC	*Model View Controller* (Modelo Apresentação Controlador)
HTML	*HyperText Markup Language*
CSS	*Cascading Style Sheets*
JS	*JavaScript*
PHP	*Hypertext Processor*
MySQL	*My Structured Query Language*
AJAX	*Asynchronous Javascript And XML*
JSON	*JavaScript Object Notation*
Frontend	Lado do Cliente
Backend	Lado do Servidor
Venue	Local onde se realiza um evento

1 Interpretação do Problema

Neste capítulo aborda-se, resumidamente, o contexto, o problema, os objetivos, a análise de valor e a abordagem preconizada. Também se descreve os resultados esperados e, por fim, apresenta-se a organização do documento.

1.1 Contexto

Os eventos musicais, como festivais, concertos, festas ou eventos privados são cada vez mais numerosos não só pela grande procura destes por parte do público, como também pelo facto de constituírem um instrumento de promoção e de divulgação de artistas e do seu reportório musical.

A organização de eventos musicais é um processo logístico complexo que começa pela definição da data e do local onde se irá realizar (denominado *venue*), pela procura de artistas mais adequados aos géneros musicais do evento, entre outros aspetos.

Por outro lado, os agentes de artistas têm cada vez mais *venues* por onde escolher. O que torna difícil selecionar as melhores *venues* onde o artista que representa deva atuar.

Surgiram várias organizações no sector da música ao vivo que identificaram como oportunidade de negócio o desenvolvimento de plataformas para facilitar a criação de eventos musicais, permitindo reservar artistas e gerir uma rede de contactos. Aliaram a estas necessidades, a oportunidade da *web*, que, permite a gestão e partilha de informação, e, ao mesmo tempo, permite reduzir custos e tempo de execução das tarefas dos profissionais da área, não só organizadores de eventos como também artistas e respetivos agentes.

A Musicverb[1] é uma destas organizações. A Musicverb vem desenvolvendo uma plataforma de gestão e colaboração para artistas, agências e organizadores de eventos musicais.

[1] http://www.musicverb.com/home/

A plataforma web da Musicverb permite a organizadores de eventos encontrarem artistas para os seus eventos e permite a agentes de artistas gerirem uma rede de contactos que é explorada através do envio de *emails* personalizados para promoção do seu artista.

O sistema proposto neste projeto vai ser integrado na plataforma da Musicverb.

1.2 Problema

Os agentes dos artistas despendem muito tempo a encontrar e negociar *venues* onde os artistas que representam possam atuar. Esta procura é feita através da rede de contatos profissional do agente. Este método é muito trabalhoso e pode não gerar novas oportunidades.

1.3 Objetivos

A Musicverb pretende possibilitar aos agentes e aos seus artistas encontrar as *venues* mais adequadas à música que fazem, de acordo com os seus objetivos e o seu momento de carreira. Para isso, pretende desenvolver um sistema de recomendação de *venues*. Assim, o presente projeto tem um objetivo principal com dois subobjetivos:

- Implementação de um sistema de recomendação de *venues*;
    - Permitir a adição de um contacto relativo a uma *venue* recomendada, na rede de contactos do agente de artistas;
    - Permitir gerir os contactos de *venues* adicionados.

O objetivo principal é a parte mais importante do projeto, e justifica-se em virtude do número muito elevado de *venues* que a plataforma dispõe, o que dificulta os agentes visualizarem todas as *venues*. Com o sistema de recomendação será apresentada ao agente uma lista de *venues* personalizada. Deste modo, pretende-se que o agente encontre *venues* adequadas de forma rápida e simples. Um sistema de recomendação ajuda neste aspeto, ao determinar uma lista personalizada de *venues*, que tenham mais potencial de serem relevantes para o artista do agente. Sintetizando, este sistema de recomendação, com base em vários critérios, irá filtrar as *venues*, produzindo uma lista de recomendação personalizada.

O primeiro subobjetivo refere-se ao propósito do sistema de recomendação, adicionar o contacto de uma *venue* encontrada. A elaboração de uma rede de contactos do agente para o artista é muito relevante principalmente no início da sua atividade.

O segundo subobjetivo refere-se à gestão de contactos e consiste em adicionar uma nova categoria de contactos para facilitar a gestão dos contactos de *venues* obtidos.

A adição do contacto da *venue*, proposta pelo sistema, à rede de contactos do agente, significa que a recomendação foi aceite, sendo considerada uma boa recomendação.

1.4 Análise de Valor

O sistema será uma mais-valia para os agentes dos artistas uma vez que irá proporcionar redução do tempo despendido na tarefa de procura das *venues* mais adequadas aos artistas que representa. Adicionalmente beneficiará também os organizadores de eventos, porque dá visibilidade às *venues* e indiretamente irá apoiar no processo de procura e seleção de artistas.

Por outro lado, permite aos agentes enriquecerem a sua rede de contactos.

1.5 Resultados Esperados

No final do projeto é esperado, com a conclusão dos três objetivos, que a plataforma tenha uma ferramenta desenhada para os agentes dos artistas onde estes possam obter sugestões de *venues* interessantes e adequadas aos artistas que representam.

Os resultados do sistema de recomendação serão avaliados de acordo com o número de contactos adicionados pelos agentes às suas listas de contactos.

1.6 Abordagem Preconizada

A abordagem preconizada para este projeto é um sistema de recomendação integrado numa plataforma web.

De acordo com os perfis dos artistas, eventos e *venues* onde atuou, o sistema de recomendação, através de algoritmos de Data Mining irá encontrar as *venues* mais adequadas ao perfil do artista.

Serão usados vários algoritmos de Data Mining e avaliadas as recomendações encontradas por forma a escolher aquela que melhor se adapte ao perfil do artista.

1.7 Organização do Documento

A organização deste documento rege-se pela divisão lógica das matérias abordadas em sete capítulos, cujos assuntos são sucintamente descritos em seguida:

No presente capítulo enquadra-se o tema e o contexto do projeto desenvolvido, apresenta-se de seguida, em forma sumária, o problema, os objetivos, a análise de valor e a abordagem preconizada.

No capítulo 2 contextualiza-se o projeto. Aborda-se a área de negócio da Musicverb e a sua plataforma. Também se descreve mais a fundo o problema e a análise de valor. De seguida apresenta-se o estado da arte das abordagens comerciais concorrentes da plataforma Musicverb. É também feito um resumo do estado de arte de sistemas de recomendação em várias áreas como, e-commerce, e-business e na área da música.

No capítulo 3 analisam-se diferentes técnicas identificadas no estado da arte da tecnologia relevante.

No capítulo 4 apresentam-se os vários casos de uso e descreve-se a solução pensada.

No capítulo 5 apresenta-se pormenorizadamente as várias fases de desenvolvimento do projeto, recorrendo a esquemas, excertos de algoritmos, e imagens para ilustrar o processo.

No capítulo 6 descreve-se como vai ser feita a avaliação da solução.

No capítulo 7 são retiradas conclusões sobre o trabalho realizado, apresentadas as suas limitações, e identificadas linhas de pensamento para desenvolvimentos futuros.

2 Contexto e Estado da Arte

Este capítulo contextualiza o conteúdo do relatório, introduzindo o conhecimento necessário para uma melhor compreensão do trabalho realizado e descrito nos capítulos seguintes.

Começa-se por abordar a área de negócio e a plataforma *web* da Musicverb, apresenta-se detalhadamente o problema e faz-se uma análise de valor do projeto a desenvolver. Segue-se o estado da arte das abordagens comerciais concorrentes existentes e da tecnologia relevante.

2.1 Área de Negócio

Existem centenas destas organizações a trabalhar na indústria da música organizando milhares de eventos, com lucros globais, em 2016, de mais de 25 mil milhões de euros (aumento de 3.2%). Este lucro deve continuar a aumentar devido à explosão das aplicações de *streaming* de música, que representam 45% da receita global da indústria. Existem cada vez menos consumidores a comprar álbuns (os lucros baixaram 4.5%), fazendo com que as receitas provenientes dos concertos sejam mais importantes (Domingo, 2016) (Rutherford, 2016) (Statista, 2016).

O consumidor prefere ouvir música em aplicações de streaming como o *spotify*[2] (de graça ou por um preço baixo comparado com o preço de todos os álbuns que pode ouvir) e participar em concertos ao vivo, pois estes concertos são experiências musicais diferentes e únicas (Domingo, 2016).

A Musicverb insere-se no domínio da música ao vivo.

Neste domínio trabalham várias pessoas como agentes de artistas e organizadores de eventos. Todos eles pretendem comunicar e trabalhar entre si mais facilmente de forma a proporcionar os melhores eventos ao vivo (“About the Live Music Industry”, 2016).

[2] https://www.spotify.com/

Estes profissionais estão interessados em ter os seus processos de trabalho automatizados, pois estes podem ser demorados e pouco organizados, colocando em causa o rendimento das empresas. Exemplos de alguns destes processos são as negociações de propostas de reserva de grupos musicais (artistas) que, seriam feitas por correio eletrónico; e a gestão de contactos feita em editores de texto ou Excel. O Excel é ainda muito usado e, apesar de suprimir algumas das necessidades, é ao utilizador que cabe maior parte do trabalho.

Assim, os profissionais da música ao vivo estão interessados em otimizar as tarefas e ter uma plataforma única onde possam trabalhar, partilhando informação relevante e aproveitando as oportunidades criadas devido ao aumento da eficiência dos processos.

Esta área de negócio tem vários fatores críticos de sucesso: rapidez de acesso à informação, exatidão da informação e facilidade na integração da plataforma com os processos das organizações.

2.2 Plataforma Web

A plataforma *web* da Musicverb está dividida em várias *in-apps* (i.e. aplicações internas da plataforma) de entre as quais a *in-app* "Explore" é a mais relevante para este projeto, pois é aqui que este projeto se vai integrar de forma mais clara.

A *in-app* "Explore" (Figura 1) permite a pesquisa de artistas por nome. É usada pelos organizadores de eventos para encontrar artistas para os eventos que estão a organizar.

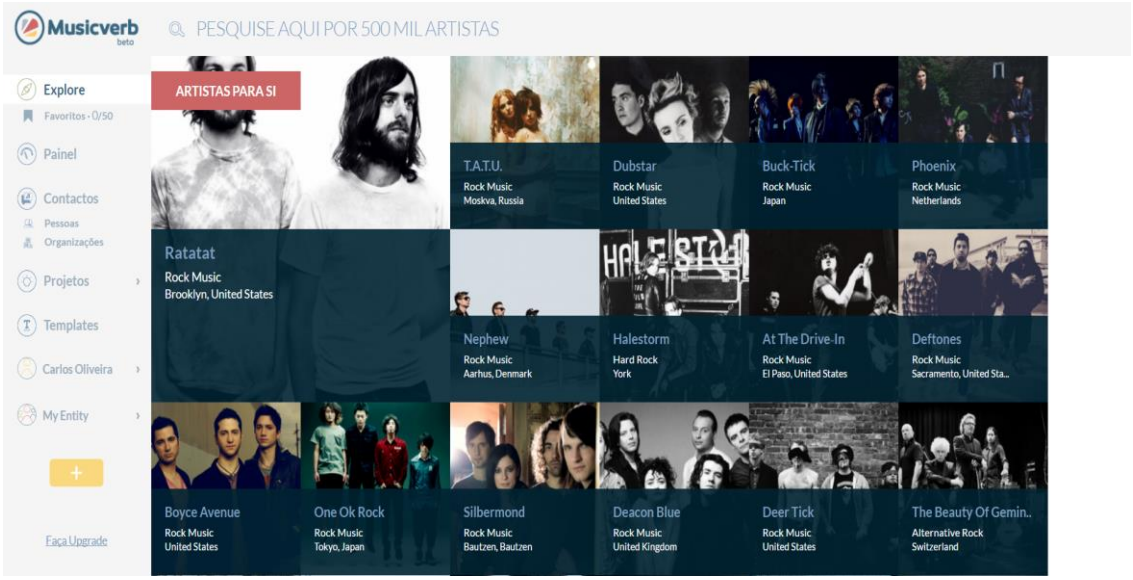

Figura 1 – Exemplo da interface do "Explore" na página da pesquisa de artistas por nome

Quando o utilizador navega para esta página, antes de começar a introduzir o nome do artista, é-lhe apresentada uma lista de artistas. Esta lista é personalizada e baseia-se na correspondência entre os géneros musicais definidos como favoritos pelo utilizador e os

géneros musicais dos artistas. Pode-se considerar que é um sistema de recomendação simples baseado, apenas, nos géneros musicais.

Portanto, os organizadores de eventos usam a plataforma para encontrar artistas para eventos que estejam a planear através do "Explore". O organizador de eventos pode adicionar artistas à sua lista de favoritos para um acesso mais fácil. Ao evento que organiza está associado a uma *venue*, a uma data e aos artistas que vão atuar (o cartaz). A plataforma tem registados os eventos com cartaz fechado, que vão sendo publicados.

Desta forma, os organizadores de eventos conseguem encontrar artistas e negociar a sua reserva com o seu agente. Este projeto pretende fazer este processo no sentido inverso, permitir aos agentes encontrar *venues* e negociar com o organizador de eventos para que este reserve o artista que representa.

Outra *in-app* é o "Mailer" (Figura 2).

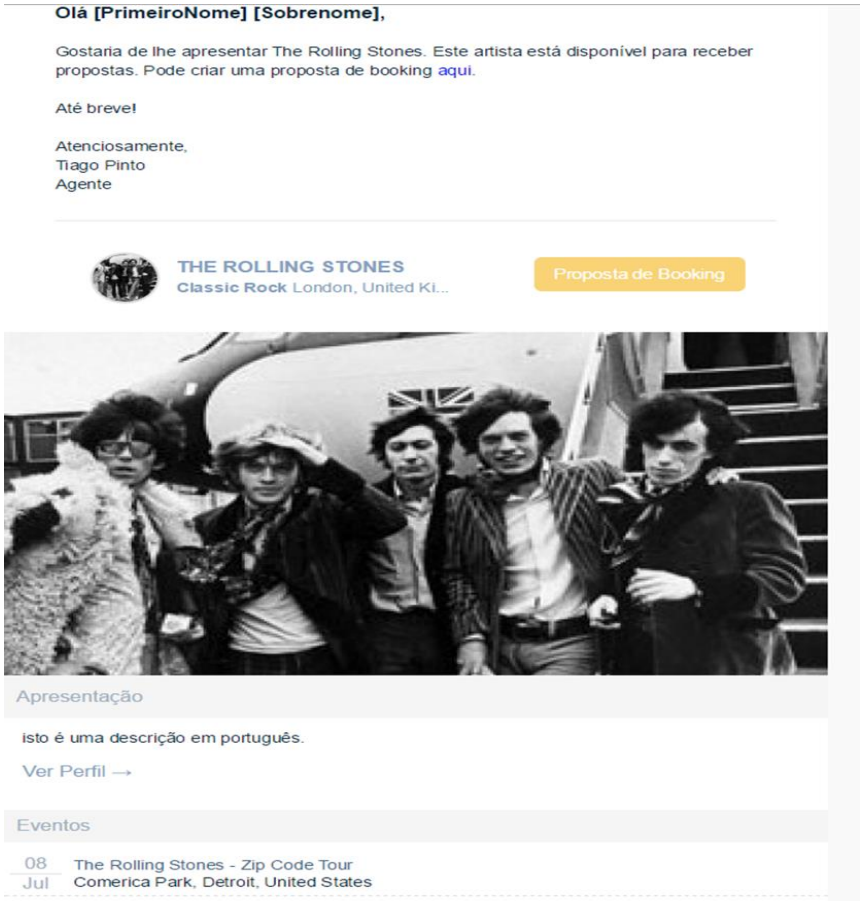

Figura 2 – Exemplo de um *email* personalizado pelo "Mailer"

Esta ferramenta permite, aos agentes de artistas, enviar *emails* personalizados com a informação relevante do artista que representa, para os organizadores de eventos

adicionados à sua rede de contactos. Estes *emails* têm o objetivo de promoverem o artista respetivo, para que, no limite, o organizador reserve o artista.

Na Figura 2 apresentou-se o *template* básico de *email*, mas é possível adquirir novos *templates* de emails para outras funções.

Para além disso, o "Mailer" ajuda a gerir os dados relativos ao envio desses *emails*. Na Figura 3 ilustra-se essa interface.

Figura 3 – Exemplo da interface do "Mailer"

São registadas métricas como número de *emails* enviados e taxa de abertura.

É devido a esta *in-app* "Mailer" que é necessário adicionar o contacto da *venue* encontrada à rede de contactos do agente, pois deste modo é agilizado o processo de *booking* (processo de negociação de reserva de artista).

Para essa gestão de contactos, existe a *in-app* "Network" que divide a rede de contactos do utilizador entre várias categorias (contactos de pessoas e contactos de organizações). Na Figura 4 apresenta-se a interface desta *in-app*.

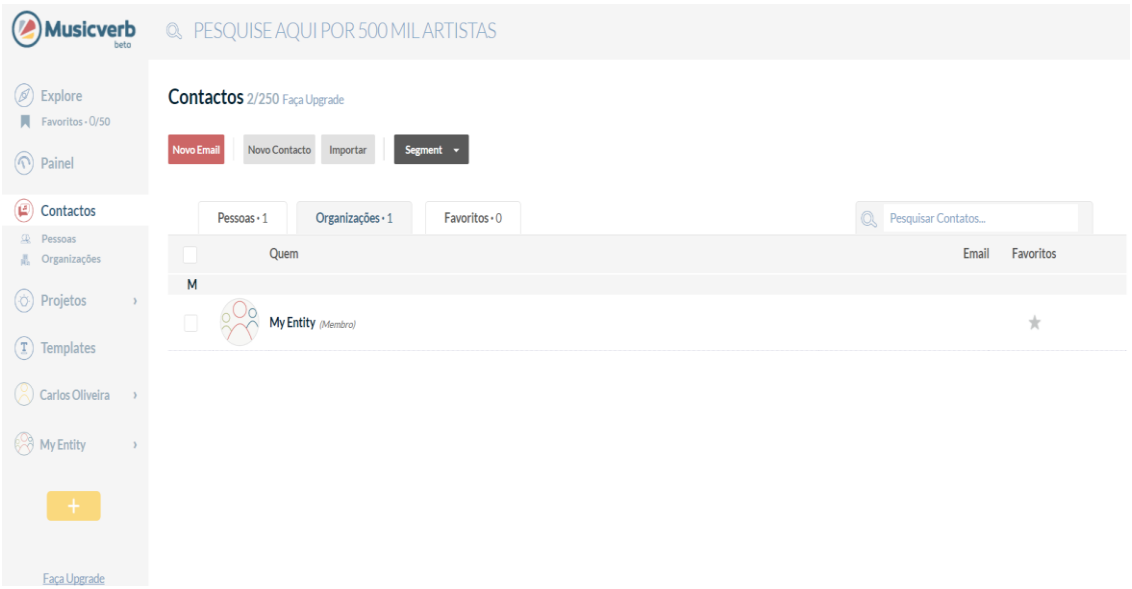

Figura 4 – Exemplo da interface do "Network"

Os contactos contidos na *network* de um agente são usados pela *in-app* "Mailer", na medida em que, permite o envio de *emails*, para esses contactos. Para além de simplesmente selecionar os contactos que devem receber o *email*, o agente pode segmentar os contactos, de forma a tornar o processo eficiente e eficaz.

Desta forma, um agente consegue enviar centenas de *emails* para os contactos certos, com o *template* certo. No início é possível importar contactos automaticamente para que seja fácil e rápido começar a utilizar as funcionalidades.

Todas estas ferramentas têm limitações dependendo do plano de pagamento do utilizador. Por exemplo, limite de número de *email* diários enviados com o "Mailer" ou limite no número de contactos.

2.2.1 Arquitetura

A plataforma está desenvolvida com o padrão de arquitetura *Model View Controller* (MVC) usando a *framework* CakePHP[3]. O lado do cliente (*frontend*) usa HTML, CSS e JS (Camada Apresentação). O lado do servidor (*backend*) usa PHP (Camada Controlador) e MySQL (Camada Modelo). Na Figura 5 apresenta-se uma representação da arquitetura da plataforma.

[3] https://cakephp.org/

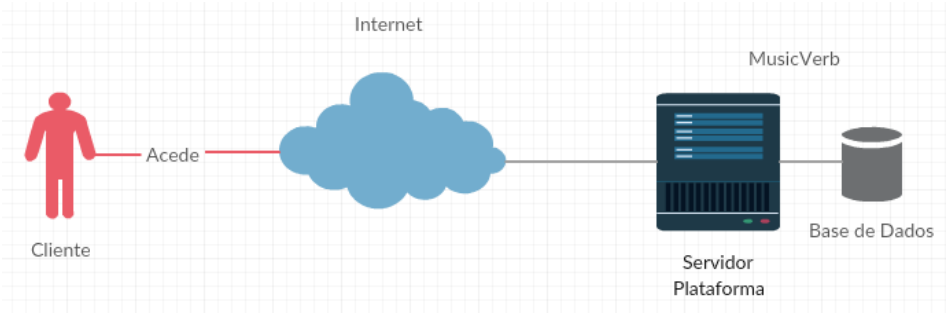

Figura 5 – Arquitetura da Plataforma

No lado esquerdo da nuvem temos o *frontend* e do lado direito o *backend*.

O cliente pode usar qualquer dispositivo para aceder à nuvem através de um navegador. Através da nuvem acede ao servidor da plataforma que responde aos pedidos com as páginas solicitadas. O servidor da plataforma está suportado por uma base de dados.

2.3 Descrição do Problema

Nesta secção descrevem-se vários aspetos do problema e do projeto a resolver.

O conceito básico do problema a resolver por este projeto é o tempo consumido pelos agentes na tarefa de procurar *venues* para os seus artistas. Os agentes gastam muito tempo nesta procura e, por vezes, o conhecimento do agente limita a procura. A plataforma tem todas as condições para ajudar a resolver estes problemas.

Desta forma, este projeto tem o propósito de desenvolver um sistema que permite aos agentes dos artistas encontrarem *venues* onde os artistas que representam devam atuar, de forma rápida e eficaz.

Para concretizar este sistema não é necessário a utilização de novas tecnologias. A adaptação de abordagens existentes bem desenvolvidas e fiáveis é preferível.

Para o primeiro objetivo é necessário fazer um levantamento do estado da arte para identificar abordagens existentes de sistemas de recomendação que possam ser adaptadas.

Para o segundo e terceiro objetivos adapta-se abordagens já utilizadas na plataforma.

Do ponto de vista da Musicverb o sistema proposto é uma adição às suas propostas de valor para os seus clientes. Significa mais uma funcionalidade, que é adicionada à plataforma, que será vendida aos utilizadores subscritos (pagantes).

Este sistema também vai permitir à plataforma tornar a informação adquirida sobre *venues* útil.

Por outro lado, do ponto de vista dos clientes o sistema proposto vai ser uma ferramenta muito útil na tarefa de procura de novas *venues* onde o artista que representa deva atuar.

Como esta tarefa faz parte das funções da sua profissão, os agentes querem que seja fácil integrar o uso do sistema na sua procura e que esta integração reduza, de facto, o tempo despendido na tarefa ou, pelo menos, que ajude a encontrar locais interessantes desconhecidos.

De uma forma breve, pode-se resumir o problema do projeto na questão "Onde deve o meu artista atuar?".

2.4 Análise de Valor

Nesta secção é feita a análise de valor do projeto.

Nas subsecções seguintes aborda-se os elementos do *New Concept Development Model* (NCD), define-se os conceitos de valor, valor para o cliente e valor percebido pelo cliente; e, finalmente, apresenta-se o modelo canvas com especial detalhe para as propostas de valor. De seguida, apresenta-se uma rede de valor para ilustrar as interações entre os vários intervenientes e o projeto. Por fim, aborda-se o processo hierárquico analítico.

2.4.1 Modelo de Desenvolvimento de um Novo Conceito

O modelo de desenvolvimento de um novo conceito de Peter Koen (*New Concept Development Model*) (P. a Koen, 2004) tem cinco elementos chave: Geração e Enriquecimento de Ideias, Seleção de Ideias, Identificação de Oportunidades, Análise de Oportunidades e Definição do Conceito. É possível começar no elemento Geração e Enriquecimento de Ideias ou na Identificação de Oportunidades para chegar à definição do conceito. Neste caso começou-se pela geração de ideias que é auxiliada pela cultura de colaboração que fomenta a geração de ideias inovadoras com envolvimento do *business-executive champion* (P. A. Koen et al., 2002).

A ideia para este projeto (Elemento geração de ideias) surgiu a partir de uma reunião com o supervisor sobre as necessidades dos agentes de artistas (o cliente). Chegou-se à conclusão que os agentes precisavam de saber onde deveriam colocar o seu artista a atuar e a plataforma tinha informação sobre *venues* que não estava a ser explorada. Assim definiu-se a ideia de criar uma ferramenta que permitisse responder a esta necessidade, isto é, este projeto.

Depois de se considerar várias alternativas, selecionou-se (Elemento seleção de ideias) a elaboração de um sistema de exploração de *venues* implementando um sistema de recomendação. Esta alternativa era a única que mostrava potencial para dar bons resultados, pois existem mais de trinta mil *venues* na base de dados. Era fundamental que a ferramenta

conseguisse filtrar toda essa informação, para apenas apresentar os resultados relevantes ao cliente. Um sistema de recomendação apresenta-se muito adequado, pois tem mostrado, ao longo dos anos, que tem bom desempenho a filtrar informação e que pode ser integrado numa plataforma *web*.

2.4.2 Valor

O agente de artistas (cliente) valoriza uma ferramenta que o ajuda nas suas funções profissionais. A ferramenta proposta permite reduzir o tempo e custo de encontrar e explorar novas *venues*, de uma forma conveniente. O cliente encontra novas *venues* onde pode ter a oportunidade de colocar o seu artista a atuar.

Na Tabela 1 indicam-se os benefícios e sacrifícios percebidos pelo cliente.

Tabela 1 – Benefício e sacrifícios do produto para o cliente

Benefícios	**Sacrifícios**
Redução Tempo Despendido	Preço
Aumento Oportunidades	
Integração na Plataforma	

Com esta ferramenta, o cliente consegue encontrar *venues* que, de outra forma, poderia não encontrar e diminui o tempo despendido na execução da tarefa de procura de *venues*. Isto permite o aumento da taxa de conversão de contactos em oportunidades de negócio.

Por último, o cliente terá o benefício de ter todas as funcionalidades interligadas na plataforma, facilitando a execução de tarefas.

2.4.3 Modelo Canvas

Como esta ferramenta é apenas uma adição às propostas de valor da Musicverb, decidiu-se fazer o canvas no ponto de vista do projeto. O cliente considerado é o agente de artistas e a proposta de valor é o sistema proposto e os seus benefícios. Na Figura 6 apresenta-se o modelo canvas.

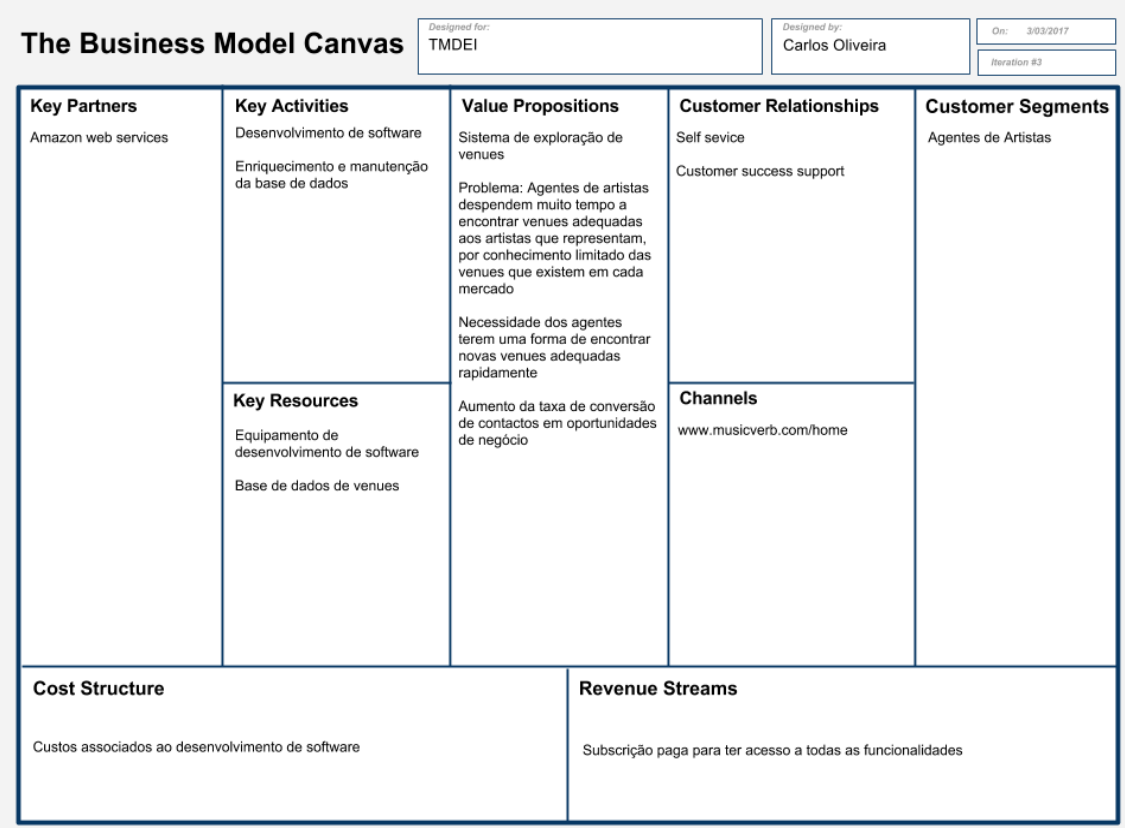

Figura 6 – Modelo Canvas do projeto

Começando da esquerda para a direita, o utilizador alvo do projeto são os agentes de artistas (*Customer Segment*). Estes profissionais têm necessidade de terem ajuda informática para facilitar as suas tarefas de trabalho e encontrar oportunidades.

A relação com o cliente é feita através do supervisor na empresa. Este age como uma interface com os clientes, na medida em que, levanta os requisitos com os clientes. O cliente usa a plataforma autonomamente (*self service*) e existe apoio ao cliente para que possa usar a plataforma corretamente e tomar o máximo partido de todas as funcionalidades.

O canal de comunicação com o cliente é a plataforma em si.

Em relação às propostas de valor temos a implementação do sistema de exploração de *venues* (produto).

Esta proposta de valor pretende resolver o problema de tempo despendido na procura de *venues* pelos agentes de artistas. Esta ferramenta pode resultar em novas oportunidades de negócios para os clientes, de uma forma simples e rápida.

O produto inova no contexto da plataforma Musicverb (onde vai ser integrado), na medida em que, preenche uma lacuna nas funcionalidades disponibilizadas pela plataforma.

Tendo esta proposta de valor em mente, as atividades chave resumem-se ao desenvolvimento do sistema e à manutenção da base de dados. Enquanto os recursos chave e estrutura de custos estão relacionados com essa atividade (e.g. internet, ambiente de desenvolvimento…).

No que toca a receitas, este projeto tem como objetivo acrescentar uma funcionalidade à proposta de valor da Musicverb para os seus clientes, portanto, as receitas estão associadas a isso. Na medida em que, a funcionalidade será paga por acrescento, ao valor de subscrição do cliente.

Finalmente, nas parcerias chave identifica-se a Amazon web services que aloja o servidor da plataforma.

2.4.4 Rede de Valor

Este projeto apresenta uma rede de valor centrada no desenvolvimento da solução. Esta rede de valor está aplicada especificamente a este projeto.

Os intervenientes neste projeto são os Colaboradores, o Supervisor na Musicverb e a Orientadora no ISEP.

As interações entre os participantes e o projeto formam uma rede de valor ilustrada na Figura 7.

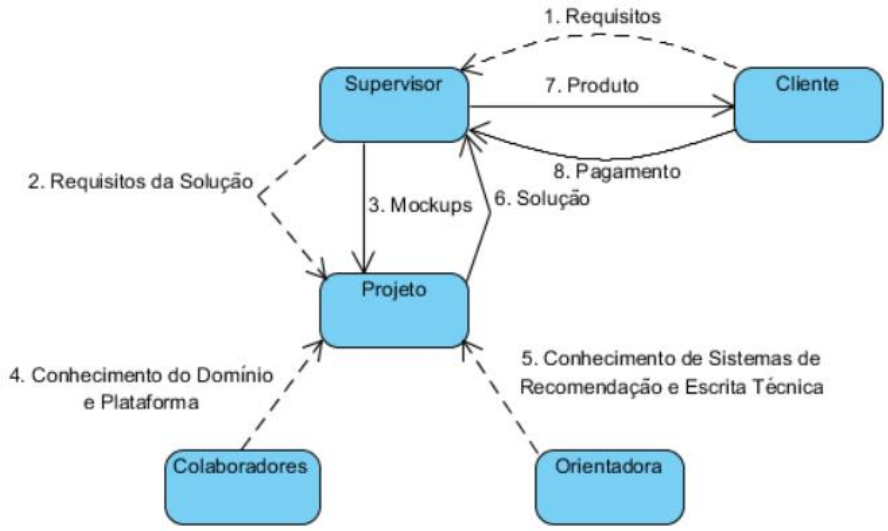

Figura 7 – Rede de valor do projeto

Na rede de valor percebe-se melhor de que forma o supervisor faz de interface com o cliente.

O supervisor levanta os requisitos do cliente que são usados na definição do problema a resolver durante o projeto (ponto 1 e 2). Também desenha a interface gráfica tendo em conta

esses requisitos. Os rascunhos da interface gráfica (*mockups*) resultantes são usados na implementação da solução (ponto 3).

No desenvolvimento da solução, o projeto usa o conhecimento do domínio e da plataforma dos colaboradores (ponto 4) e o conhecimento de sistemas de recomendação e escrita técnica da orientadora (ponto 5).

Quando a solução está finalizada, esta é entregue ao supervisor e integrada com a plataforma (ponto 6).

A plataforma (produto) é disponibilizada ao cliente que paga de acordo com o seu plano de pagamento (ponto 7 e 8).

2.4.5 Processo Hierárquico Analítico

O processo hierárquico analítico (*Analytic Hierarchy Process*; AHP) ajuda a tomar uma decisão quando se tem de escolher entre várias alternativas, tendo em conta diferentes critérios, com pesos diferentes.

O projeto não tem uma decisão deste género para fazer. Mas pode-se considerar que o cliente tem, quando escolhe uma plataforma para usar. Na secção seguinte comparam-se os vários concorrentes da Musicverb por vários aspetos que podem ser usados como critérios de decisão entre plataformas.

2.5 Estado da Arte em Abordagens Comerciais

Existem várias organizações a desenvolverem plataformas como a Musicverb. Estes são os concorrentes da Musicverb. Poucos desses concorrentes disponibilizam a pesquisa de *venues*.

Na Tabela 2 as várias organizações serão comparadas pelos aspetos seguintes:

- Aplicação: refere-se às aplicações que disponibilizam;
- Network: refere-se à possibilidade dos utilizadores gerirem uma rede de contactos;
- *Mailer*: refere-se à possibilidade de enviar *emails* personalizados a outras pessoas (mesmo que não estejam registadas nas respetivas plataformas);
- Pesquisa de *venues*: refere-se é a possibilidade de pesquisar *venues*;
- Licença: refere-se ao facto das funcionalidades da plataforma só ficarem disponíveis após a compra de uma licença.

Tabela 2 – Comparação dos concorrentes da Musicverb

Plataforma	**Aplicação**	***Network***	***Mailer***	**Pesquisa de *venues***	**Licença**
Musicverb http://www.musicverb.com/home/	Plataforma web	Sim	Sim	Não	Sim É possível usar sem pagar, com limitações
gigwell https://www.gigwell.com/	Plataforma Web com aplicação móvel	Sim	Sim	Sim	Sim
VIP vip-booking.com http://www.vip-booking.com/	Plataforma Web	Sim	Sim	Sim	Sim
overture https://overturehq.com/	Plataforma Web	Sim	Não	Não	Sim
SystemOne http://www.bookingagencysoftware.com/	Plataforma Web com aplicação móvel	Sim	Sim	Não	Sim

beatswitch https://www.beatswitch.com/	Plataforma Web com aplicação móvel	Sim	Não	Não	Sim
GIGATOOLS https://pro.gigatools.com/	Plataforma Web	Sim	Sim	Não	Sim
detəils http://www.detailsdetails.eu/	Plataforma Web	Sim	Não	Não	Sim
INDIE ON THE MOVE https://www.indieonthemove.com/venues	Plataforma Web	Não	Sim	Sim Apenas EUA	Sim É possível usar sem pagar, com limitações
venuefinder.com BLUE&GREEN http://www.venuefinder.com/music-venues/	Plataforma Web	Não	Sim	Sim Apenas RU	Sim É possível usar sem pagar, com limitações

Todas as organizações disponibilizam uma plataforma web que permite, de alguma forma, conectar o agente do artista e o organizador de eventos.

No que diz respeito a este projeto, as plataformas mais interessantes, por permitirem pesquisar *venues*, são:

- A VIP-Booking permite pesquisar *venues* por nome.
- A IndieOnTheMove usa uma abordagem colaborativa na qual os utilizadores da plataforma comentam e avaliam as *venues*. Esta avaliação consiste na atribuição de 1

a 5 estrelas (sendo 5 estrelas a melhor avaliação). Tem a limitação de apenas ter informação sobre *venues* nos Estados Unidos da América.

- A VenueFinder permite a pesquisa de *venues* por localização (área e cidade) dentro do Reino Unido.

- O Gigwell disponibiliza uma pesquisa de *venues* por localização, limitada por um raio à volta da cidade escolhida. Também permite a pesquisa de *venues* por nome.

Pode-se concluir que o sistema proposto por este projeto é inovador, pois apenas um concorrente (IndieOnTheMove) disponibiliza um sistema de recomendação de *venues*, o que torna a plataforma Musicverb mais apelativa e diferenciadora.

2.6 Estado da Arte em Sistemas de Recomendação

Nesta secção define-se o que são sistemas de recomendação e descrevem-se várias abordagens distintas de sistemas de recomendação. Por fim faz-se uma revisão da literatura sobre o tema.

2.6.1 Sistemas de Recomendação

Os sistemas de recomendação são aplicações de *software* que determinam sugestões de itens a utilizadores (online) com o objetivo de ajudar no processo de decisão. Tentam prever os itens mais adequados ao utilizador, tendo em conta as suas preferências (Lu et al., 2015; Ricci et al., 2011).

São sistemas que usam opiniões, ações ou historial de membros de uma comunidade para encontrar o conteúdo mais relevante para cada utilizador.

Desde a publicação do primeiro artigo sobre esta área, na década de 90, que tem vindo a crescer em todos os aspetos. Principalmente nas várias aplicações práticas que têm obtido resultados muito bons, como sugestões de vídeos no Youtube ou de produtos na Amazon. Estes sistemas de recomendação fazem um perfil para cada utilizador baseando-se nas ações passadas desse utilizador. Com base nisso, nas opiniões da comunidade e na descrição dos itens o sistema determina itens com alta probabilidade de serem considerados relevantes pelo utilizador (Lu et al., 2015).

É usual dividir sistemas de recomendação em três tipos fundamentais: Baseado em Conteúdo (*Content-based*), Filtro Colaborativo (*Collaborative-filtering*) e Híbridos.

Os sistemas baseados em conteúdo usam o historial de compras do utilizador para recomendar itens semelhantes aos que o utilizador já adquiriu no passado (Park et al., 2012). Estes sistemas têm dificuldade em recomendar itens diferentes dos habitualmente comprados

pelo utilizador (este problema é referido, na literatura, como "*cold-start*" (Schein et al., 2002)). Na Figura 8 ilustra-se esta abordagem.

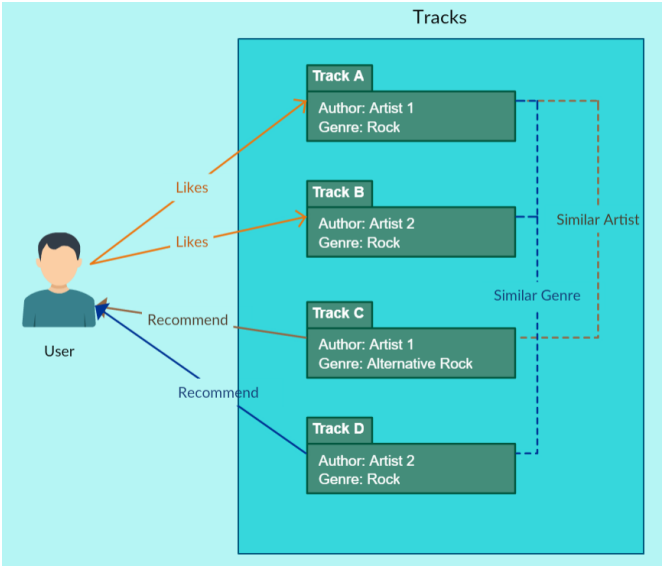

Figura 8 – Exemplo de um Sistema de Recomendação Baseado em Conteúdo

Neste exemplo, os itens são *tracks* de música. O utilizador gosta da música A e B, então o sistema recomenda a música C, por ter o mesmo artista. Também recomenda a música D, por ter o mesmo género da música A e B. Estas recomendações são feitas com base na descrição dos itens.

Por outro lado, os sistemas de filtro colaborativo usam o historial de compras de utilizadores com perfil semelhante ao do utilizador para propor novos itens. No entanto, esta técnica revela dois problemas principais: o problema da escassez, uma vez que as avaliações dos utilizadores são em número muito reduzido, comparadas com o número de itens disponíveis; e o problema de escalabilidade, relacionado com o elevado número de produtos e clientes (Sarwar et al., 2000). Na Figura 9 apresenta-se um exemplo desta abordagem.

Figura 9 – Exemplo de um Sistema de Recomendação Colaborativo
Fonte: (Johnson, 2014)

Neste exemplo, os itens são *tracks* de música. O utilizador da esquerda gosta das músicas P, Q, R e S. O utilizador da direita gosta as músicas Q, R, S e T. O algoritmo identifica a semelhança entre estes utilizadores (músicas Q, R e S) e produz as recomendações: o sistema recomenda a música P ao utilizador da direita e a música T ao utilizador da esquerda.

Os sistemas híbridos juntam ambas as abordagens, de forma a colmatar as limitações de ambas as abordagens (Ricci et al. , 2011). Existem outros tipos de sistema de recomendação, mais complexos, que foram desenvolvidos com o mesmo objetivo, como o uso de *active learning* (Elahi, Ricci, & Rubens, 2016), abordagens *fuzzy* (Wu, Zhang, & Lu, 2015), *context-aware* (Abbas, Zhang, & Khan, 2015), baseado em redes sociais (Lah et al., 2016), e multicritério (L. Cui et al., 2016).

Percebe-se que é muito importante para os sistemas de recomendação conseguir calcular semelhanças, quer seja entre itens ou entre utilizadores. A técnica mais simples e mais conhecida de calcular similaridades é a distância euclidiana (1), que determina a distância entre dois pontos/objetos (e.g. entre dois itens), num plano bidimensional.

$$d(x_i, x_j) = \sqrt{\sum_{t=1}^{T} [x_{ti} - x_{tj}]^2} \quad (1)$$

Outra técnica muito utilizada é a distância do cosseno (2), que mede o ângulo entre dois vetores. Cada um desses vetores representa um objeto (e.g. um item).

$$\cos(x, y) = \frac{(x \cdot y)}{||x|| * ||y||} \quad (2)$$

Pearson correlation (3) é também outra técnica utilizada para calcular similaridade que tenta descobrir se existe uma relação linear entre dois registos (Ricci et al., 2011).

$$r = \frac{\sum_i (x_i - x)(y_i - y)}{\sqrt{\sum_i (x_i - x)^2}\ \sqrt{\sum_i (y_i - y)^2}} \quad (3)$$

Para comparações de vetores binário, a similaridade de jaccard (4) é muito utilizada.

$$J = \frac{M_{11}}{M_{01} + M_{10} + M_{11}} \quad (4)$$

Dados dois vetores binários, com a mesma dimensão, A e B. O M_{11} é o número de vezes onde ambos os vetores têm o valor 1 na mesma posição do vetor. O M_{01} é o número de vezes onde o vetor A tem um 0 e o B tem um 1. Por outro lado, o M_{10} é o inverso do M_{01} (A tem 1, B tem 0). As posições do vetor onde ambos têm o valor 0 são ignoradas.

Outro aspeto importante na implementação de um sistema de recomendação é saber avaliá-lo. Existem várias fases de avaliação no processo de desenvolvimento de um sistema de recomendação. A primeira fase consiste em executar diferentes algoritmos nos mesmos dados e comparar o desempenho. A segunda fase consiste em perceber se as recomendações

estão a ser aceites pelos utilizadores, analisando os registos (*logs*) das ações dos utilizadores (Ricci et al., 2011).

2.6.2 Revisão Literária

Nesta secção apresentam-se vários artigos relevantes que identificam as técnicas mais utilizadas. Na Tabela 3 detalha-se os aspetos mais importantes dos artigos analisados para o estado da arte e respetivas técnicas.

Tabela 3 – Revisão Literária em Sistemas de Recomendação

Artigo	**Resumo**	**Técnicas**
Recommender system application developments: A survey (Lu et al., 2015)	Este artigo faz um estado da arte de sistemas de recomendação por área (e-gov, e-business, e-commerce, e-library, e-learning, e-tourism, e-resource services, and e-group ativities), e as técnicas mais bem-sucedidas em cada área	Sistemas baseados em Palavra-Chave, Ontologias, Rede de Crenças Bayesiana, *Fuzzy*, *Context-aware*, Redes sociais
A literature review and classification of recommender systems research (Park et al., 2012)	Este artigo faz um estado da arte de sistemas de recomendação, por área de aplicação, baseado em artigos publicados entre 2001 e 2010	Sistemas baseados em Métodos Heurísticos, k-NN, Regras de Associação, Árvores de Decisão, Redes Neuronais, Clustering
Recommender Systems Handbook (Ricci et al., 2011)	Este livro é um agregado de vários artigos relevantes a sistemas de recomendação. Define o que são sistemas de recomendação e os seus tipos. Aborda cada tipo e cada fase no desenvolvimento de um sistema de recomendação, em grande detalhe	Árvores de Decisão, k-NN, *Vector Space Model* e Naïve Bayes, *Context-aware*, Rede social, *Active learning*, Multicritério

A novel multi-objective evolutionary algorithm for recommendation systems (L. Cui et al., 2016)	Este artigo propõe um algoritmo novo que usa uma abordagem multicritério e um algoritmo evolucionário para que tenha em conta a diversidade e novidade das recomendações (em detrimento da exatidão). Obteve 69% de exatidão usando o *dataset* Movielens com 100 000 avaliações de 943 utilizadores sobre 1682 filmes	Algoritmo Evolucionário Multiobjectivo Probabilístico (AEMOP)
An effective collaborative movie recommender system with cuckoo search (Katarya & Verma, 2016)	Este artigo tenta aliviar o problema da dimensionalidade e *data sparsity* usando *k-means* e o algoritmo de otimização de *clusters cuckoo search*. Este algoritmo foi avaliado usando o dataset Movielens com 100 000 avaliações de 943 utilizadores sobre 1682 filmes e obteve um erro absoluto de 68%	K-means com algoritmo de otimização *cuckoo search*
Research on recommender system based on ontology and genetic algorithm (Lv, Hu, & Chen, 2016)	Tenta resolver o problema de *cold-start* e *sparcity* usando dados relacionais num domínio ontológico e usando um algoritmo genérico para obter as recomendações. Esta *framework* obteve 90% de exatidão usando um *dataset* ontológico baseado no Movielens com 100 000 avaliações	Baseado em Ontologias e Algoritmo Genético (BOAG)
A hybrid fuzzy-based personalized recommender system for telecom products/services (Zhang et al., 2007)	Este artigo usa uma abordagem colaborativa hibrida baseada em itens e utilizadores, estruturando os dados com técnicas *fuzzy*. Obteve um erro absoluto de 78% usando o *dataset* Movielens com 100 000 avaliações de 943 utilizadores sobre 1682 filmes	*Fuzzy based* k-NN

An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data (Mild & Reutterer, 2003)	Este artigo apresenta uma modificação à abordagem colaborativa quando apenas existe informação binária sobre as preferências do utilizador. Avaliam este algoritmo usando 2441 transações de clientes de entre 54 produtos. Cada transação tem, pelo menos, 5 itens. Obtém 26% de exatidão.	CF_{mod} (*Collaborative Filtering Modification*)
E-Commerce Recommendation Applications (Schafer et al., 2001)	Faz um estado da arte de técnicas e aplicações de sistemas de recomendação na área de e-commerce.	k-NN, Redes Bayesianas, Clustering, Classificação, Regras de Associação
Collaboration Filtering Recommendation Optimization with User Implicit Feedback (H. Cui & Zhu, 2014)	Abordagem colaborativa híbrida entre *feedback* explícito (e.g. avaliações) e implícito (e.g. viu o filme). O algoritmo foi testado com o *dataset* Movielens com 100 000 avaliações de 943 utilizadores sobre 1682 filmes e usando a métrica grau de concordância, definida para este artigo.	Sistema colaborativo com o algoritmo HITS (*Hyperlink-Induced Topic Search*)
A Survey of Music Recommendation Systems and Future Perspectives (Song, Dixon, & Pearce, 2012)	Faz um estado da arte de várias abordagens usadas para recomendar músicas, descrevendo várias abordagens colaborativas e baseadas em conteúdo. Para além dessas, também refere abordagens *context-based* e baseado nas emoções das músicas. Para cada abordagem, problemas e limitações são indicados.	k-NN

Collaborative Filtering Recommender Systems In Music Recommendation (Kuzelewska & Ducki, 2013)	Descreve um sistema de recomendação de músicas colaborativo baseado em avaliações feitas pelos utilizadores.	k-NN

Também é pertinente referenciar sistemas de recomendação usados em plataformas na área da música.

No *spotify* é usada uma abordagem colaborativa, mas também uma abordagem baseada nas *tags* das músicas, para ajudar o utilizador a descobrir novas músicas. Usam a técnica k-NN (Johnson, 2014).

Outras plataformas como o Gnoosic[4] constroem um perfil do utilizador perguntando o que ele gosta e não gosta. Depois, este perfil é usado para comparar com artistas e músicas semelhantes aos gostos indicados.

Por outro lado, Allmusic[5] permite a descoberta de artistas, álbuns e músicas por géneros, ou estados de espírito, ou temas. Para isso usa a popularidade desses itens. Para além disso, tem recomendações personalizadas ao utilizador, sendo que este tem de avaliar vários itens de forma a obter recomendações.

A plataforma Pandora[6] usou uma abordagem diferente. Em vez de recorrer a algoritmos computacionais, criou o *Music Genome Program*. No âmbito deste projeto, uma equipa de musicólogos estudou milhares de músicas, descrevendo cada uma, usando 450 atributos diferentes (Pandora Music, 2016). Com itens descritos de forma tão detalhada, faz com que os cálculos de similaridade entre itens sejam muito significativos.

[4] http://www.gnoosic.com/
[5] http://www.allmusic.com/
[6] https://www.pandora.com/

3 Técnicas Existentes

Neste capítulo descrevem-se várias técnicas identificadas no estado da arte em sistemas de recomendação, com mais detalhe técnico de forma a ser possível perceber que técnicas devem ser usadas e como se devem avaliar.

Quando se trabalha em sistemas de recomendação, o bom senso dita que se deve começar com a abordagem mais simples e só adicionar complexidade se o ganho de desempenho o justificar (Ricci et al., 2011). Soluções mais complexas tendem a não obter resultados muito melhores do que técnicas mais simples.

Irão ser consideradas técnicas supervisionadas, em que os algoritmos aprendem com dados previamente classificados tais como, k vizinhos-mais-próximos (k-NN), classificação, máquinas de suporte vetorial (*Support Vector Machines*; SVM); e técnicas não-supervisionadas como, clustering e regras de associação.

3.1 k vizinhos-mais-próximos (k-NN)

A abordagem k vizinhos-mais-próximos (K *Nearest Neighbor* k-NN) é a técnica mais simples e usada, com bom desempenho. Esta técnica determina os *k* itens mais semelhantes com base numa métrica de proximidade ao item de referência (ver funções (1), (2), (3) e (4)) (Ricci et al., 2011). Na Figura 10 está um exemplo da determinação dos vizinhos mais proximos com *k* diferentes.

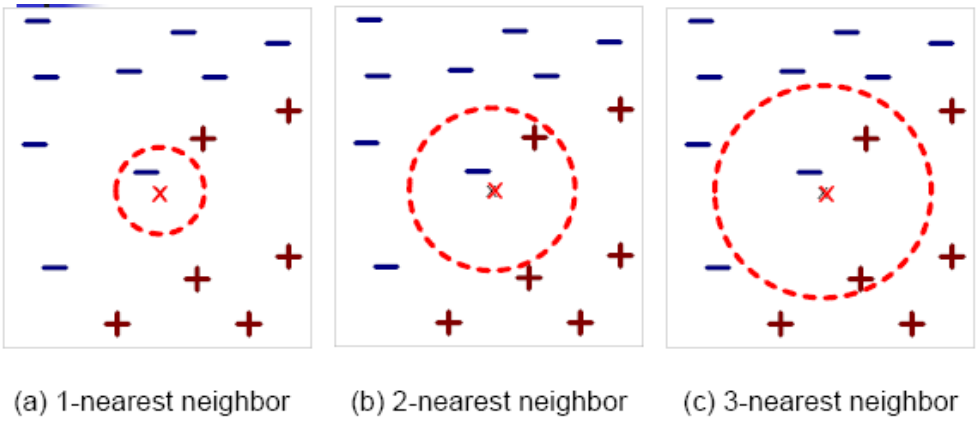

Figura 10 – Exemplo da determinação dos k vizinhos mais próximos
Fonte: (Rodrigues, 2016a)

Esta técnica é facilmente modificada para se adaptar a uma grande variedade de domínios, quer com dados numéricos, quer com dados categóricos.

3.2 Classificação

Esta abordagem desenvolve um modelo, com base em dados de treino, que permite identificar um item com uma classe objetivo. Algumas técnicas de classificação usadas são: Árvores de Decisão, Redes Neuronais e Naïve Bayes.

As Árvores de Decisão constroem uma árvore (modelo) de forma a conseguir classificar um item. Cada nó da árvore testa um atributo do item para decidir a classe apropriada. Um algoritmo muito usado para a construção de árvores é o C4.5 (Quinlan, 1993). Esta técnica adequa-se bem a dados categóricos.

As Redes Neuronais imitam a arquitetura do cérebro humano com nós conectados com ligações pesadas. Os nós são chamados neurónios e são dispostos em rede resultando numa arquitetura com os nós interligados entre si, com pesos nas ligações. Uma arquitetura possível é exemplificada na Figura 11.

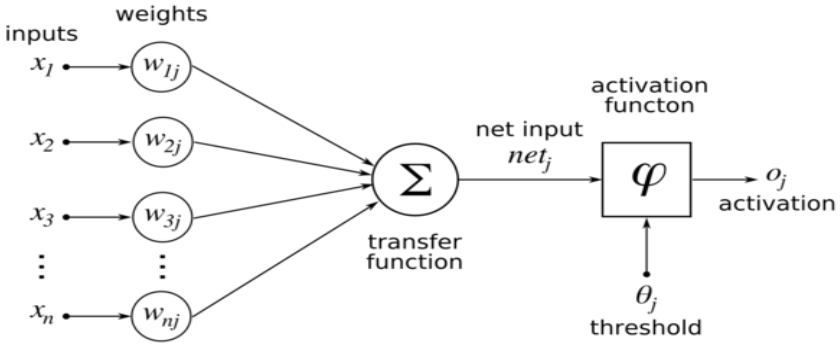

Figura 11 – Arquitetura de uma rede neuronal
Fonte: (Ricci et al., 2011)

As Redes Neuronais precisam de muitos dados para serem treinadas e o modo como classificam não é percetível, é uma caixa negra. Iteração a iteração, dependendo da saída da rede e dos valores esperados os pesos da rede são adaptados. Os dados de entrada são os vários atributos do item e o resultado é a classe a que o item pertence (Ricci et al., 2011).

Finalmente, a técnica Naïve Bayes baseia-se na teoria Bayesiana para calcular probabilidades condicionais. De forma a simplificar a fórmula, assume-se que existe uma independência probabilística entre os atributos do item. Com este pressuposto, a probabilidade condicional é calculada usando a seguinte fórmula: $P(A|b,c,d) = P(A|b) * P(A|c) * P(A|d))$, lê-se probabilidade de A dado b, c, d em que A é a classe objetivo e b, c, d são os atributos do item (Ricci et al., 2011).

3.3 Máquinas de Suporte Vetorial

Desenvolvidas por Vapnik e colaboradores (Cortes & Vapnik, 1995), as máquinas de suporte vetorial têm a capacidade de resolver problemas de classificação e regressão. Os resultados da aplicação desta técnica são comparáveis aos obtidos por outras técnicas de aprendizagem, como as redes neuronais, e em algumas tarefas até apresentam resultados superiores. As SVM constroem um classificador de acordo com um conjunto de padrões, por ele identificados nos exemplos de treino, onde a classificação é conhecida. Considerando o exemplo da Figura 12, nela existe um conjunto de classificadores lineares que separam duas classes, mas apenas um (em destaque) que maximiza a margem de separação (distância da instância mais próxima ao hiperplano de separação das duas classes em questão). O hiperplano com margem máxima é chamado de hiperplano ótimo, que será o objeto de busca do treino do classificador (Gunn, 1998).

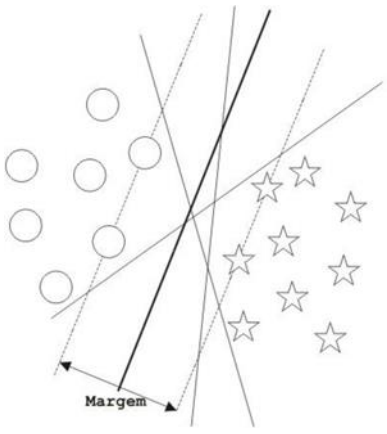

Figura 12 - Possíveis hiperplanos de separação e hiperplano ótimo

No entanto as amostras de dados nem sempre são linearmente separáveis. As funções de Kernel das SVM têm a finalidade de projetar os vetores de características de entrada num espaço de características de alta dimensão, para classificação de problemas que se encontram

em espaços não linearmente separáveis. Existem vários tipos de kernels que podem ser usados, porém as funções de Kernel mais usadas são as polinomiais, gaussiano e sigmoidal.

3.4 Clustering

Clustering é uma técnica que procura particionar os dados em grupos. A técnica tenta maximizar a distância entre os grupos e minimizar a distância entre os seus membros (maximizar a semelhança), cf. Figura 13. O algoritmo de clustering mais usado é o k-means. Este algoritmo necessita que o número k de clusters seja previamente especificado. Cada cluster tem um centro (centroid) e um conjunto de pontos próximos a esse centroid. A métrica de avaliação mais usada é a soma dos quadrados dos erros (*Sum of Squared Error*; SSE) (Rodrigues, 2015a).

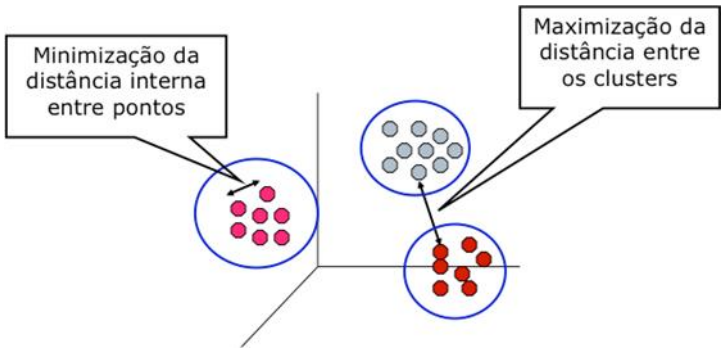

Figura 13 – Exemplo da técnica de clustering
Fonte: (Rodrigues, 2015a)

3.5 Regras de Associação

Esta técnica tem como objetivo inferir regras a partir dos dados. As regras geradas associam dois itens que tendem a aparecer na mesma transação (co-ocorrer). As regras tomam o formato $A \rightarrow B$ (Agrawal, Imieliński, & Swami, 1993). Se os dados forem os historiais dos utilizadores (em vez de transações), permite comparar os historiais e recomendar itens com base na similaridade dos perfis.

As regras são avaliadas com o suporte, confiança e interesse. O suporte refere-se à frequência da co-ocorrência dos itens. A confiança é a probabilidade dos itens antecedentes e consequentes aparecerem na mesma transação. Por último, o interesse determina a força da regra (Rodrigues, 2015b).

3.6 Avaliação Sistema de Recomendação

Os algoritmos usados no sistema de recomendação devem ser avaliados.

Para avaliar os algoritmos divide-se os dados em dois conjuntos disjuntos: um de treino e um de teste. Existem duas técnicas principais de fazer esta divisão: validação cruzada e *holdout*.

A validação cruzada parte o *dataset* em *n* partições. Uma partição é usada para teste e *n-1* partições são usadas para treino. Este processo é repetido *n* vezes, rodando a partição de teste (Ricci et al., 2011).

Na Figura 14 ilustra-se esse processo, sendo que o bloco a laranja é a partição usada para teste.

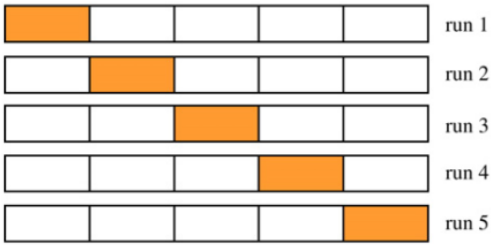

Figura 14 - Exemplo de validação cruzada usando 5 *datasets*
Fonte: (Lee et al., 2010)

A validação cruzada deve ser usada em pequenos conjuntos de dados. Para grandes conjuntos de dados divide-se os dados em dois conjuntos: um conjunto de treino e um conjunto de teste. Esta divisão denomina-se *holdout* e, normalmente, o conjunto de treino recebe 2/3 dos dados, sobrando 1/3 para teste (Rodrigues, 2016b).

Depois da divisão dos dados, usa-se os dados de treino para fazer previsões sobre os dados de teste. Estas previsões podem ser analisadas de forma a avaliar o algoritmo que as produziu.

Cada algoritmo pode usar técnicas diferentes, técnicas estas que têm métricas de avaliação diferentes, o que torna difícil comparar resultados de forma a determinar os melhores algoritmos. Por isso, desenvolveu-se várias métricas com o objetivo de uniformizar os resultados, para que possam ser comparáveis:

- **Accuracy (5):** esta métrica calcula o rácio entre as boas recomendações obtidas (num_gr) e o total de recomendações obtidas (num_tr). Permite perceber a percentagem de boas recomendações.

$$accuracy = \frac{num\_gr}{num\_tr} \quad (5)$$

- **Gr_Coverage (6):** esta métrica calcula o rácio entre as boas recomendações obtidas (num_gr) e o total de boas recomendações que seria possível obter (num_tgr). Permite perceber se o algoritmo consegue descobrir todas as boas recomendações.

$$gr\_coverage = \frac{num\_gr}{num\_tgr} \quad (6)$$

- **Mg_Coverage (7):** esta métrica calcula o rácio entre o número de artistas que receberam recomendações (num_mg_r) e o número de artistas total (num_mg). Permite perceber se o algoritmo produziu as recomendações para um grande número de artistas.

$$mg\_coverage = \frac{num\_mg\_r}{num\_mg} \quad (7)$$

- **Tempo de execução por artista:** tendo em conta que o sistema deve ser rápido, esta é uma métrica muito importante. Representa o tempo, em segundos, que o algoritmo demora a obter recomendações para um artista.

Estas métricas não foram pensadas para serem usadas externamente, isto é, dificilmente se fará comparações entre os algoritmos implementados, com algoritmos em algum artigo científico. Para que isso fosse possível, os algoritmos externos teriam de usar os mesmos dados (ou muito parecidos), mas como os dados são propriedade da Musicverb, não estão públicos. Logo, estas métricas têm o objetivo de permitir a comparação de algoritmos internamente. Sendo que, futuramente, um novo algoritmo de recomendação pode ser implementado e, usando estas métricas, é possível compará-lo com os já existentes.

3.7 Discussão

Cada *venue* é descrita usando dados categóricos, por isso algumas técnicas não são adequadas a este tipo de dados, como por exemplo redes neuronais. Além disso, o tempo de execução tem de cumprir os requisitos definidos (algoritmos que necessitem de saber o artista de referência para obter recomendações podem ter tempos de execução elevados). Devido a estes fatores, nem todas as técnicas vão ser implementadas e avaliadas. Outras serão implementadas e avaliadas, mas descartadas devido aos seus tempos de execução serem demasiado altos.

As técnicas que consigam ser adaptadas ao problema a resolver, com tempos de execução apropriados, serão avaliadas e comparadas usando os mesmos dados, de forma a identificar as melhores técnicas para usar no sistema de recomendação.

4 Design da Solução

Neste capítulo apresenta-se a solução pensada e as suas alternativas.

Começa-se por analisar os requisitos identificando e descrevendo os vários casos de uso associados aos objetivos propostos. De seguida descreve-se solução pensada e identificam-se alternativas.

4.1 Análise de Requisitos

Os requisitos da solução são definidos pelo Supervisor, como dito na rede de valor descrita na secção 2.4.4.

Os requisitos não funcionais são gerais a todos os casos de uso e são os seguintes:

- Desempenho: tempo de resposta (<20 segundos);
- Interface gráfica deve ser consistente com o resto da plataforma;
- Interface deve adaptar-se à língua do utilizador (localização);

Os requisitos funcionais estão divididos pelos três casos de uso ilustrados no diagrama da Figura 15.

Figura 15 – Diagrama de Casos de Uso

O único utilizador que pode obter recomendações é o agente do artista, porque é necessário o perfil do artista que representa. Depois de obter essas recomendações, o agente pode adicionar o contacto de uma dessas *venues*. Posteriormente, o agente pode gerir os contactos adicionados como uma nova categoria de contactos (i.e. separadamente dos restantes contactos).

4.1.1 UC1 Obter *Venues* Recomendadas

- **Ator principal**

Agente do Artista

- **Partes interessadas e seus interesses**

Agente do Artista: rapidez na obtenção das recomendações com qualidade e que resultem em novas oportunidades de negócio.

- **Pré-condições**

O agente do artista está autenticado e na página correta.

O agente está subscrito a esta funcionalidade.

- **Pós-condições**

Resultados apresentados.

- **Cenário de sucesso principal (ou fluxo básico)**

1. O agente do artista pede recomendações definindo um artista que representa e filtros.

2. O sistema determina as *venues* mais relevantes ao utilizador com base no artista selecionado e os filtros.

3. O sistema apresenta as recomendações.

- **Extensões (ou fluxos alternativos)**

*a. A qualquer momento o agente do artista pode sair da página.

- O caso de uso termina.

*b. A qualquer momento o agente pode alterar o artista selecionado ou a localização.

- O caso de uso reinicia (Passo 1).

3a. O agente do artista pede mais resultados.

- O sistema obtém mais resultados (Passo 2) e apresentam-se os resultados (Passo 3).

❖ **Requisitos Especiais**

Registar todos os pedidos de recomendação na base de dados;

- Recomendações para os mesmos filtros deverão devolver os mesmos resultados.

4.1.2 UC2 Adicionar Contacto *Venue*

❖ **Ator principal**

Agente do Artista

❖ **Partes interessadas e seus interesses**

Agente do Artista: adiciona o contacto da *venue* para poder promover o seu artista ao organizador de eventos da *venue* selecionada.

❖ **Pré-condições**

O agente do artista está autenticado.

O agente do artista selecionou uma *venue* usando o UC1.

❖ **Pós-condições**

Contacto da *venue* adicionado à rede de contactos do agente do artista.

❖ **Cenário de sucesso principal (ou fluxo básico)**

1. O agente do artista adiciona o contacto da *venue*.

2. O sistema adiciona o contacto da *venue* à rede de contactos do agente do artista.

❖ **Extensões (ou fluxos alternativos)**

*a. A qualquer momento o agente do artista pode sair da página.

- O caso de uso termina.

4.1.3 UC3 Gestão Contactos *Venues*

❖ **Ator principal**

Agente do Artista

- **Partes interessadas e seus interesses**

Agente do Artista: gere os contactos das *venues* adicionados separadamente.

- **Pré-condições**

O agente do artista está autenticado.

O agente do artista está na "Network".

- **Pós-condições**

Gestão dos contactos concluída.

- **Cenário de sucesso principal (ou fluxo básico)**

1. O agente do artista seleciona a categoria "Venues".

2. O sistema apresenta os contactos adicionados sob essa categoria.

3. O agente gere os contactos.

- **Extensões (ou fluxos alternativos)**

*a. A qualquer momento o agente do artista pode sair da página.

- O caso de uso termina.

4.2 Design da Solução

A solução está dividida pelos elementos *frontend* e *backend*, seguindo o padrão MVC.

Os componentes *frontend* são: componente Apresentação (engloba HTML e CSS) e componente JavaScript (código que altera o componente Apresentação do lado do cliente e que permite comunicar com o servidor).

Os componentes *backend* são: componente Controlador (recebe e processa os pedidos do cliente de forma a enviar a resposta adequada), componente Biblioteca (contém classes que auxiliam os outros componentes), componente Modelo (interage diretamente com a Base de Dados), componente Base de Dados (estrutura de dados necessária) e o componente Sistema de Recomendação (usa os dados na base de dados para recomendar *venues*).

Na Figura 16 ilustra-se a interação entre os componentes indicados.

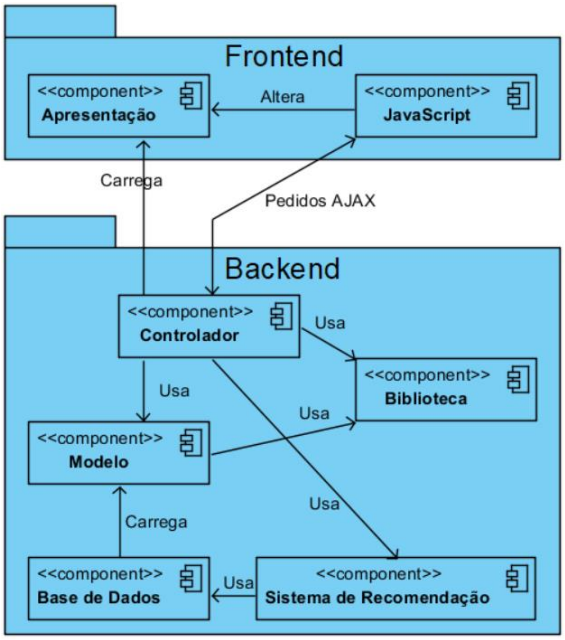

Figura 16 – Interação entre Componentes

O componente Apresentação consiste em páginas *web* desenhadas pelo supervisor. Esta interface gráfica deve ser consistente com o resto da plataforma e deve permitir efetuar todas as interações previstas com o utilizador (e.g. apresentação de resultados, introdução de dados…).

O componente JavaScript tem a responsabilidade de validar os dados introduzidos pelo utilizador e fazer pedidos AJAX para o componente Controlador, de forma a atualizar o componente Apresentação com a nova informação, sem a necessidade de atualizar a página.

O componente Controlador recebe os pedidos do *frontend* e valida os dados. Processa o pedido e responde com a informação solicitada.

O componente Modelo representa a lógica do negócio, interage diretamente com os dados.

O componente Biblioteca contém classes auxiliares aos componentes Controlador e Modelo.

O componente Base de Dados refere-se à estrutura e armazenamento dos dados.

O componente Sistema de Recomendação é o maior foco deste projeto, pois é neste componente que os algoritmos de recomendação vão estar implementados.

Na secção seguinte descreve-se a solução.

4.2.1 Descrição da Solução

A solução consiste em integrar uma nova pesquisa, a pesquisa por *venues*, na plataforma, usando um sistema de recomendação. Esta nova funcionalidade estará no centro da nova *in-app* "VenuesExplorer".

Um utilizador que esteja autenticado como agente de um artista, define uma localização e escolhe um artista que representa. Quando seleciona um artista, as recomendações relativas a esse artista são apresentadas. Esta apresentação mostra as *venues* recomendadas num mapa interativo, tendo como referência a localização definida.

Pretende-se que os dados necessários ao sistema de recomendação estejam guardados num ficheiro RData. Isto facilita a transferência de dados entre os vários componentes do sistema de recomendação. Desta forma, os dados pré-processados podem ser lidos sem ser necessário gerá-los novamente. Assim, a obtenção das recomendações terá um tempo de resposta menor e os dados podem ser atualizados periodicamente, para que os resultados se mantenham relevantes e adequados às alterações nos perfis dos artistas.

O agente pode usar estas recomendações para chegar a uma *venue* do seu interesse. Quando seleciona a *venue*, o agente pode despoletar a adição do contacto da *venue* à sua rede de contactos através de uma funcionalidade própria para este efeito.

Todas as ações do utilizador executam o javascript que valida os filtros inseridos pelo utilizador e fazem os pedidos necessários, atualizando a página com a informação recebida.

Em relação ao servidor, este estará preparado para receber os seguintes pedidos:

- Pedido pela página inicial da pesquisa de *venues*;
- Pedido das *venues* recomendadas tendo em conta os filtros;
- Pedido por mais recomendações;
- Pedido da adição do contacto da *venue* selecionada à rede de contactos do agente;
- Pedido da apresentação e gestão dos contactos das *venues*.

Para responder a estes pedidos é necessário o suporte da base de dados. A base de dados contém as tabelas seguintes:

- Tabela dos Logs: registos de ações do utilizador;
- Tabela das Venues: informação sobre as *venues*;
- Dados processados RData: informação usada para obter as recomendações;

- Tabela das Redes de Contactos: informação sobre as redes de contactos do utilizador.

Felizmente é possível reutilizar tabelas/métodos já existentes na base de dados da plataforma para registar as ações do utilizador, para obter informações sobre *venues*, para adicionar o contacto da *venue* à rede de contactos do agente e para gerir os contactos das *venues*.

Nas secções seguintes especifica-se o que é necessário implementar em cada componente, de acordo com a solução descrita.

4.2.2 Componente Apresentação

Neste componente é necessário criar o ficheiro HTML para a página inicial do "VenuesExplorer".

É necessário ajustar a interface da "Network" de forma a acrescentar a categoria "Venues". Também é preciso ajustar a SideBar de forma a ter a nova categoria aninhada em "Network" e uma hiperligação para a página inicial do "VenuesExplorer".

A página usada para criar um novo contacto tem de ser alterada para suportar o novo tipo de contacto. Este novo tipo de contacto precisa de ter uma página própria onde se consiga visualizar toda a informação relativa a esse contacto, uma página de perfil.

Todos os textos devem ser adaptados à língua do utilizador. As línguas suportadas são o português e o inglês. Para fazer esta adaptação atualiza-se os ficheiros de tradução da plataforma.

4.2.3 Componente JavaScript

Neste componente é necessário implementar um ficheiro JS (venues_explorer.js) que irá estar ativo na página da pesquisa por *venues*.

Por último, é necessário fazer alterações ao ficheiro JS network.js, de forma a suportar a nova categoria de contactos.

Assim, o venues_explorer.js suporta três ações por parte do utilizador:

1. Pesquisar;
2. Carregar mais resultados;
3. Adicionar novo contacto.

Quando o utilizador define os filtros e pesquisa (Ação 1), o javascript valida os filtros e faz o pedido AJAX ao VenuesExplorerController, depois processa o resultado e adiciona marcadores

ao mapa com as recomendações obtidas. No diagrama de sequência da Figura 17 mostra-se o fluxo desta ação.

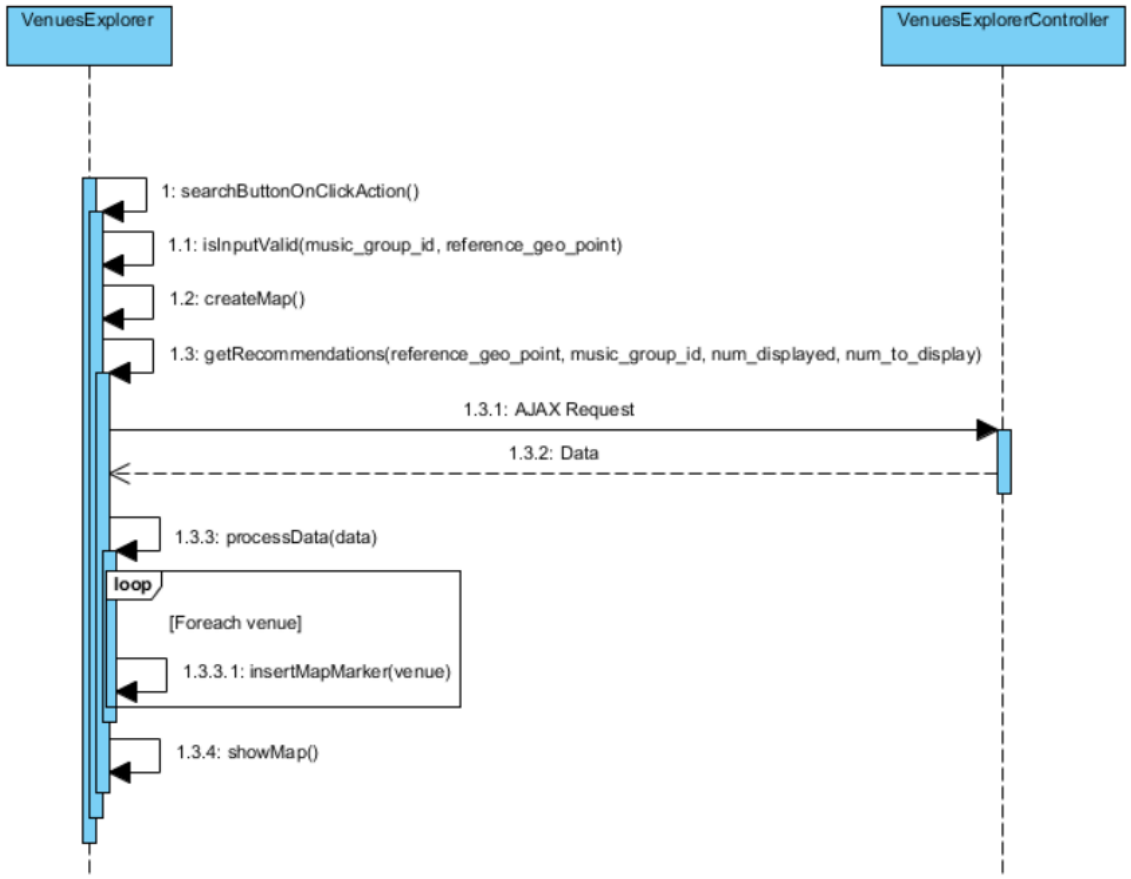

Figura 17 – Diagrama de Sequência da Ação 1

Cada vez que o utilizador clica no botão para pesquisa, o método searchButtonOnClickAction é chamado. Este método valida os filtros (isInputValid) e instância o mapa, depois, chama o método getRecommendations. Este método faz o pedido AJAX ao VenuesExplorerController de forma a obter as recomendações relativas aos dados de entrada. Caso o pedido não seja bem-sucedido é chamado o método recommendationErrorMessage que mostra uma mensagem de erro adequada.

Se o pedido for bem-sucedido, o método getRecommendations chama o método processData que processa os dados devolvidos pelo controlador, para que eles sejam usados pelo método insertMapMarker. O método insertMapMarker recebe a informação de uma *venue* e constrói o marcador com a posição no mapa (geográfica) e a janela informativa respetiva. Por fim, o mapa fica visível na página.

Se existirem mais resultados para mostrar, o utilizador pode carregar mais resultados (Ação 2). Nesse caso, é chamado o loadMoreButtonOnClickAction. Este método valida os filtros novamente e chama o método getRecommendations (seguindo o fluxo consequente; ponto 1.3).

Em relação à última ação (ação 3), o fluxo é curto e está ilustrado na Figura 18.

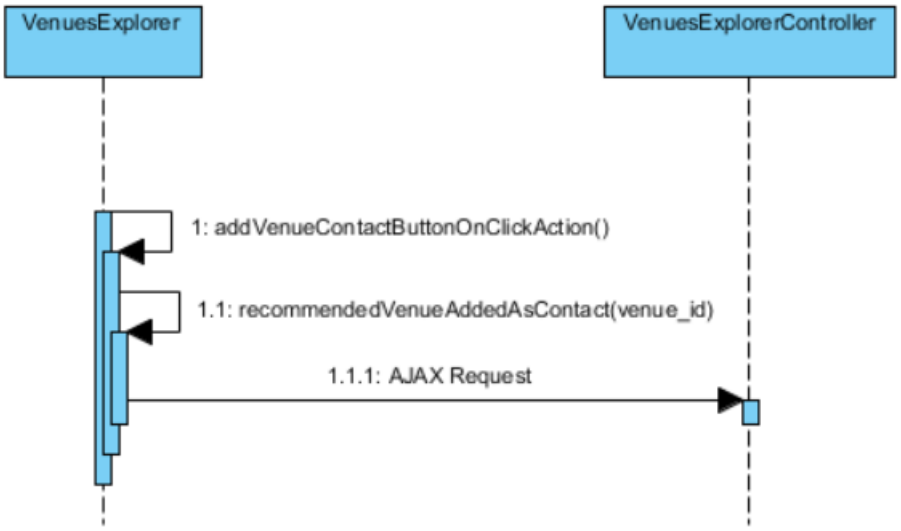

Figura 18 – Diagrama de Sequência da Ação 3

Quando o utilizador clica no botão de adicionar o contacto de uma *venue* (presente na janela informativa, respetiva), o método addVenueContactButtonOnClickAction é chamado. Este método chama o método recommendedVenueAddedAsContacts que apenas faz um pedido AJAX ao VenuesExplorerController. Ao mesmo tempo, o método addVenueContactButtonOnClickAction abre uma nova página no navegador do utilizador onde ele deve colocar a informação sobre o novo contacto, concluindo a sua adição.

4.2.4 Componente Controlador

Neste componente será necessário implementar um novo controlador, VenuesExplorerController, com a responsabilidade de tratar dos pedidos referentes ao "VenuesExplorer". Também é necessário acrescentar métodos ao controlador VenuesController e alterar os controladores OrganizationsController, ProjectsController e AjaxController.

No VenuesExplorerController são reutilizados os métodos do AppController (superclasse de todos os controladores). Estes servem para registar as ações do utilizador na base de dados (logs), validar sessões e obter todos os artistas representados por um agente. Estes métodos auxiliam o suporte das novas funcionalidades do "VenuesExplorer".

O VenuesControler implementa a vista *profile* que cria a página de perfil de um contacto do tipo *venue*.

Em relação ao OrganizationsController, ao ProjectsController e ao AjaxController será necessário alterar métodos existentes para suportar esse novo tipo de contacto.

4.2.5 Componente Modelo

Neste componente é necessário adicionar métodos à classe Venue, que representa a tabela ”venues” da base de dados. Esta tabela contém toda a informação das *venues*. Também será necessário alterar métodos na classe AppModel (superclasse de todas as classes modelo) e na classe Organization (representa a tabela “organizations” da base de dados que contém informação sobre organizações). Estas alterações têm como objetivo suportar o novo tipo de contacto.

4.2.6 Componente Biblioteca

Neste componente implementa-se um enumerador de forma a facilitar a adição de novos tipos de contacto e reduzir a duplicação de código. Também facilita a adição de novos enumeradores no futuro.

4.2.7 Componente Base de Dados

Neste componente identifica-se uma tabela e três *views* a usar e duas *views* a implementar.

A tabela venue é usada e é necessário acrescentar um novo campo: “is_contact”. A *view* “v_event_music_groups_venues” tem dois novos campos: “EventDate” e “VenueID”. Por outro lado, as *views* “v_network_contact_venues” e “v_getVenueContactProfile” são implementadas para suportar o novo tipo de contacto. Por fim, as views “v_music_group_genres” e “v_org_projects” são usadas, sem alterações.

As *views* “v_event_music_groups_venues”, “v_music_group_genres” e “v_org_projects” têm os dados necessários ao sistema de recomendação.

A *view* “v_event_music_groups_venues” liga um grupo musical a um evento e a uma *venue*. Logo, é possível construir os perfis de atuações de cada artista, também é possível obter a localização de uma *venue* a partir destes dados.

A *view* “v_music_group_genres” contém os géneros musicais de cada artista, o que permite descrever cada artista pelos seus géneros musicais. Ligando à *view* “v_event_music_groups_venues” é possível inferir géneros musicais mais populares de cada *venue*.

A *view* “v_org_projects” contém todos os artistas associados a organizações. Portanto, esses artistas terão prioridade no sistema de recomendação.

4.2.8 Fluxo dos Casos de Uso

Agora que já se identificou os vários métodos a utilizar e implementar, é possível apresentar os diagramas de sequência concretos para se perceber o fluxo de cada caso de uso. Sendo que o fluxo do UC3 não é necessário apresentar visto que as alterações são efetuadas apenas em métodos existentes.

O UC1 começa com o agente a aceder à funcionalidade. Como apresentado na Figura 19, os dados dos artistas (para que o agente possa escolher) são fornecidos no momento em que a página é carregada.

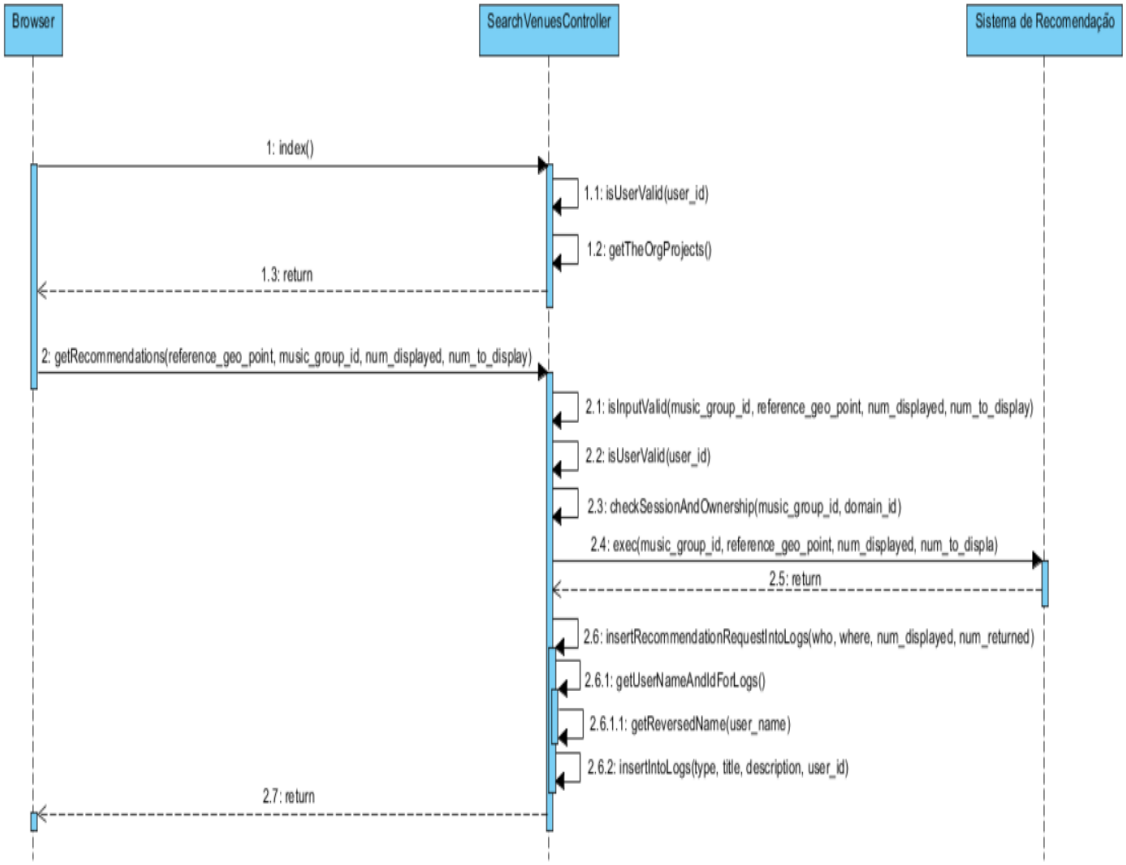

Figura 19 – Diagrama de sequência relativo ao UC1

Em seguida são validados os dados e a sessão, seguindo-se a execução do sistema de recomendação e registo da ação, atualizando a página com os resultados.

Por último, o UC2 inicia quando um utilizador quer adicionar o contacto de uma *venue*, Figura 20.

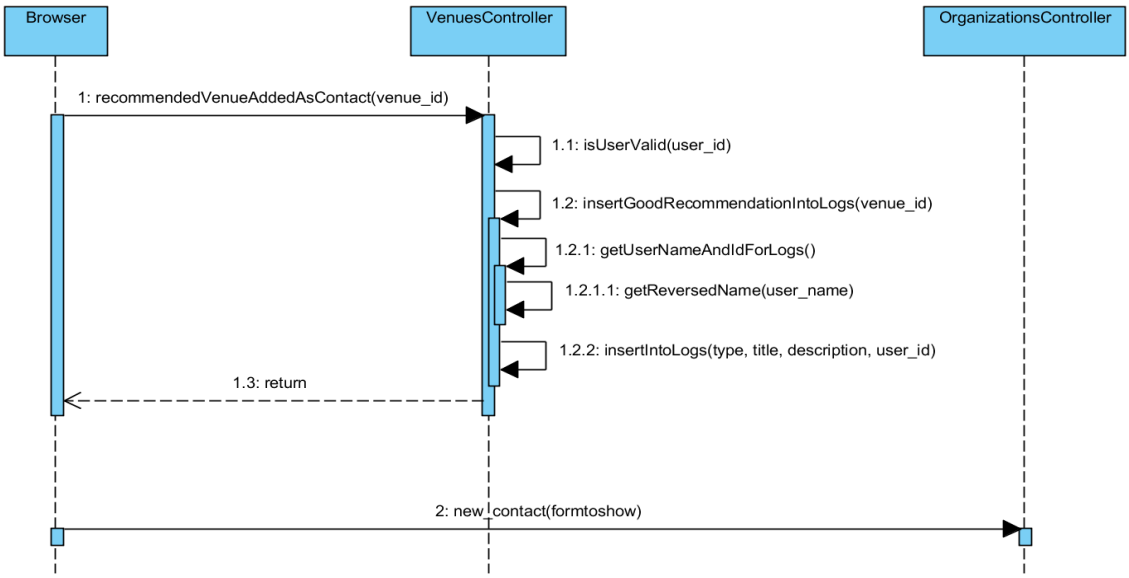

Figura 20 – Diagrama de sequência relativa ao UC2

Depois de registar na base de dados a boa recomendação, redireciona-se o agente para a vista "new_contact" do OrganizationsController que permite adicionar um novo contacto do tipo *venue* à rede de contactos do utilizador.

4.2.9 Componente Sistema de Recomendação

O sistema de recomendação será desenvolvido usando a linguagem R (versão 3.2.3), pois é uma linguagem de livre acesso com capacidade de explorar os dados, que disponibiliza bibliotecas com a maioria dos algoritmos de Data Mining.

O sistema de recomendação divide-se em três componentes: DataLoading, DataProcessing e RecommenderSystem.

Os dois primeiros são executados periodicamente de forma a atualizar os dados utilizados pelo terceiro. O RecommenderSystem é o componente que determina as recomendações, a pedido do componente Controlador.

Todos os componentes comunicam através de ficheiros RData que produzem com os resultados. Na Figura 21 ilustra-se essa transferência de dados.

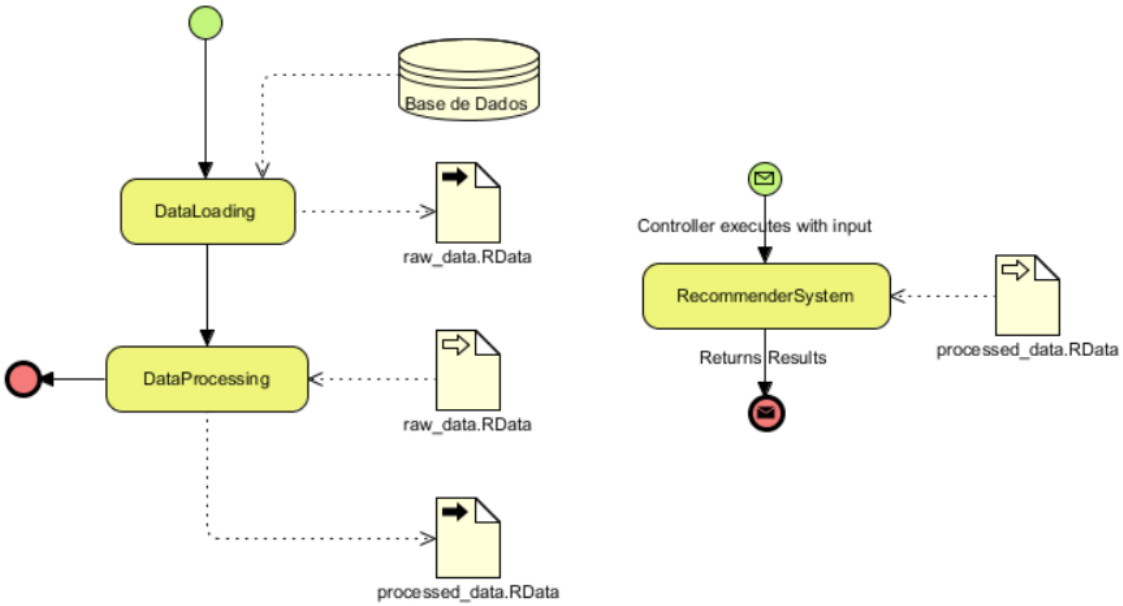

Figura 21 – Interação entre os Vários Componentes

Começando pelo DataLoading, este componente carrega os dados necessários da base de dados (e.g., os dados de descrição das *venues*) e escreve os resultados para o ficheiro raw_data.RData. Na Figura 22 apresenta-se o diagrama de sequência deste componente.

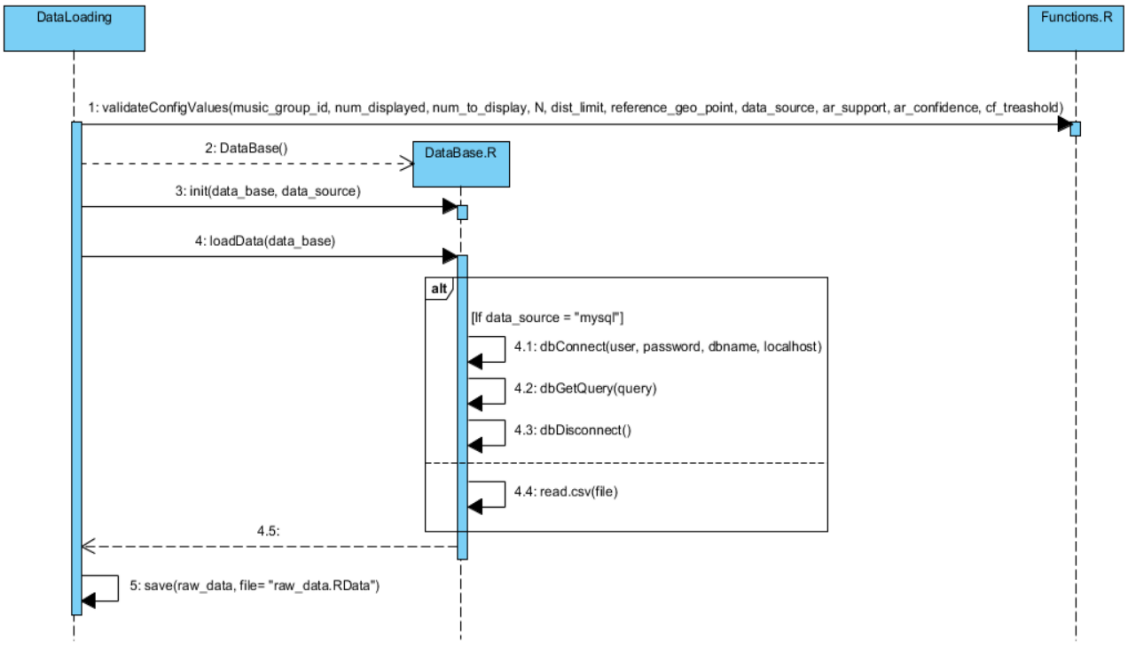

Figura 22 – Diagrama de Sequência do componente DataLoading

O componente começa por validar os dados de entrada, de seguida carrega os dados da base de dados e, por fim, gera o ficheiro raw_data.RData.

O segundo componente, DataProcessing, lê esse ficheiro e processa os dados, como se pode observar no diagrama da Figura 23.

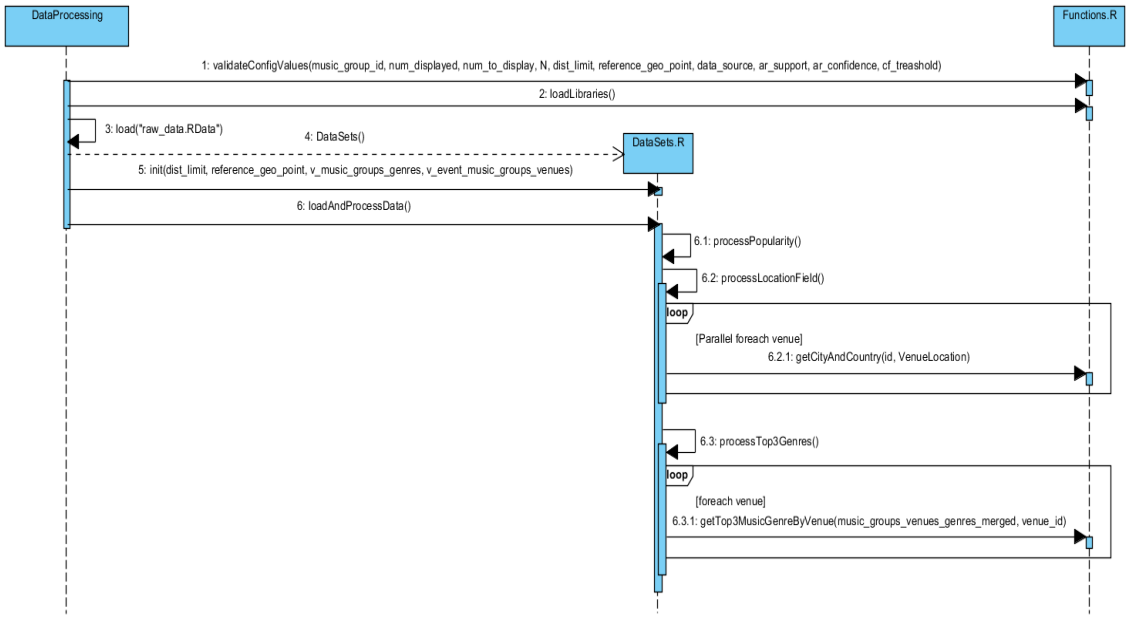

Figura 23 – Diagrama de Sequência do componente DataProcessing (Parte 1)

O processamento dos dados passa por várias fases. Começa por criar estruturas de dados adequadas (datasets) para armazenar os dados processados. Depois usa esses datasets para construir outras estruturas de dados, nomeadamente a matriz binária e as matrizes de similaridades. Na Figura 24 ilustra-se essa parte do fluxo.

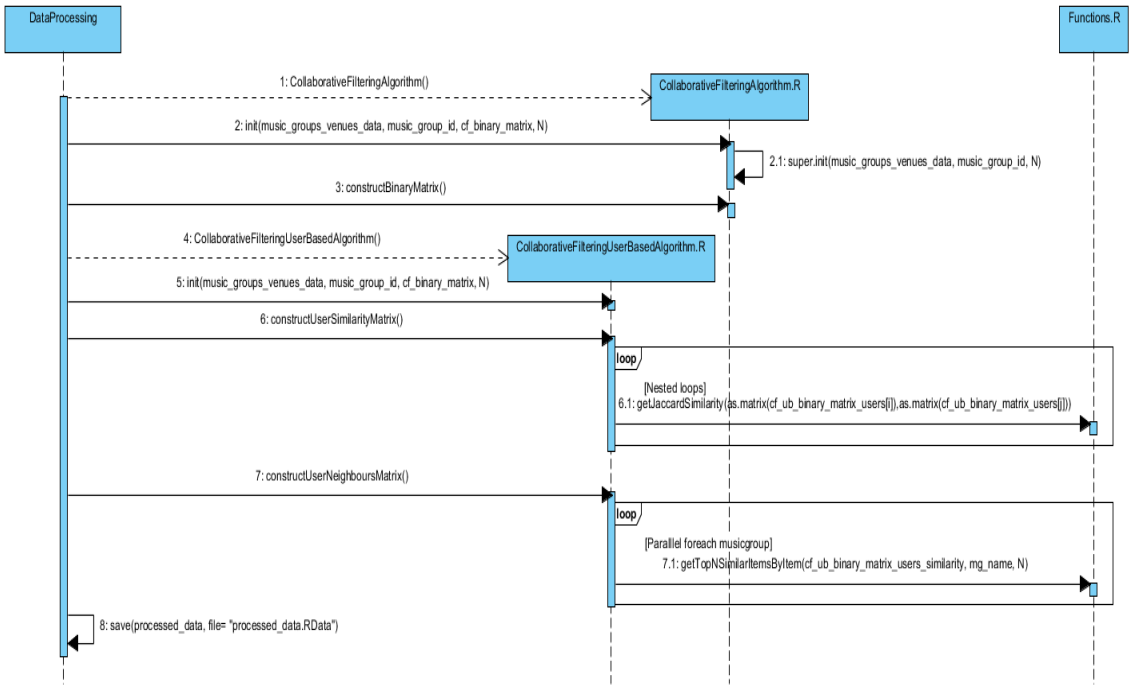

Figura 24 – Diagrama de Sequência do componente DataProcessing (Parte 2)

Por fim, os *datasets* processados são guardados no ficheiro processed_data.RData. Este processamento é importante para que o sistema de recomendação consiga chegar rapidamente à lista personalizada.

A geração das recomendações propriamente dita é feita pelo componente RecommenderSystem, que usa uma abordagem híbrida. Tanto gera recomendações baseadas nas semelhanças entre os géneros musicais de um artista e os géneros musicais das *venues* (baseadas em conteúdo), como gera recomendações com base em semelhança de perfis de artistas (colaborativa).

Cada vez que um agente pede recomendações na plataforma este componente é executado. Este componente recebe o indentificador do artista selecionado (music_group_id), o ponto geográfico de referência (reference_geo_point), o número de resultados já mostrados (num_displayed) e o número de resultados que o sistema deve obter (num_to_display).

O algoritmo a usar para a obtenção das recomendações depende da aptidão do artista. Esta aptidão é determinada pelo tamanho do histórico de atuações, isto é, artistas que tenham um historial de atuações superior a dois são elegíveis para usar a abordagem colaborativa. Por outro lado, se o historial for mais pequeno, ou inexistente, é usada a abordagem baseada em conteúdo.

Na Figura 25 apresenta-se o diagrama de sequência do componente RecommenderSystem onde se ilustra o fluxo quando o artista está apto para usar a abordagem colaborativa.

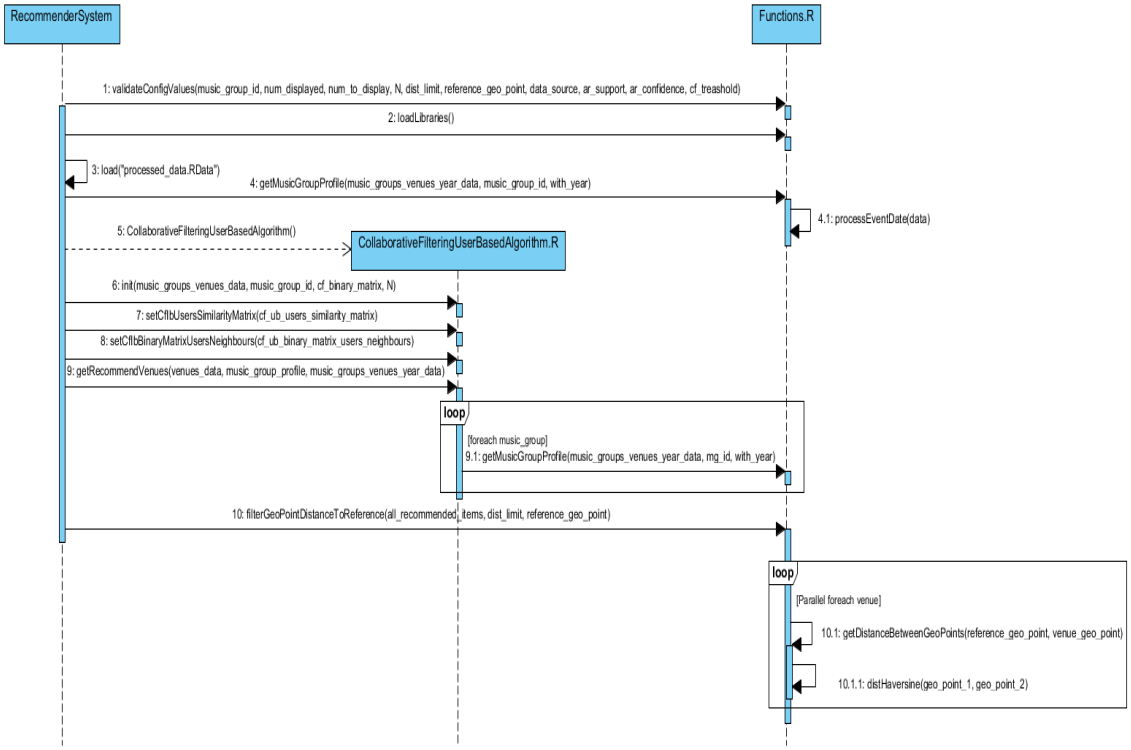

Figura 25 – Diagrama de Sequência do componente RecommenderSystem (Apto)

Este componente começa como os outros, a validar os dados de entrada e a carregar bibliotecas. Depois disso carrega os dados processados e obtém o perfil do artista (historial de atuações).

Se o artista estiver apto para o algoritmo colaborativo (CFUB), usa-se esse algoritmo para obter recomendações. Se o artista não estiver apto ou, mesmo estando apto, o algoritmo colaborativo não conseguiu obter recomendações, aplica-se o algoritmo baseado em conteúdo (PKNN). A Figura 26 ilustra o fluxo quando isso acontece.

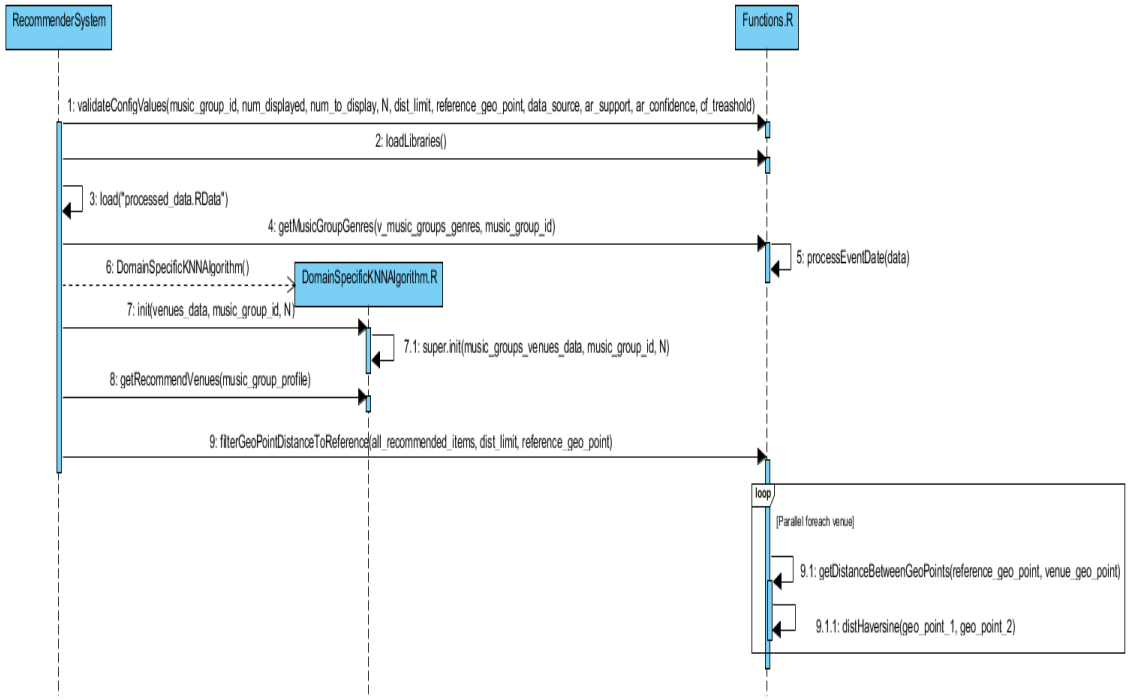

Figura 26 – Diagrama de Sequência do componente RecommenderSystem (Não Apto)

De qualquer forma, no final a lista de recomendações obtida é filtrada tendo em conta o ponto geográfico de referência. As recomendações que sobrevivem a este filtro são devolvidas para o controlador.

4.2.10 Arquitetura

Nesta secção descreve-se a arquitetura da solução recorrendo a um diagrama de componentes, semelhante ao apresentado no início da secção 4.2.

Relembrando, a solução divide-se entre os componentes *frontend* e *backend*. Os componentes Apresentação e Javascript são *frontend*. Os componentes Controlador, Modelo, Biblioteca, Base de dados e sistema de recomendação são *backend*.

Depois das descrições de cada componente, ao longo da secção 4.2, é possível detalhar a interação entre cada componente de forma mais concreta (em relação ao que foi feito na Figura 16). Na Figura 27 apresenta-se o diagrama de arquitetura.

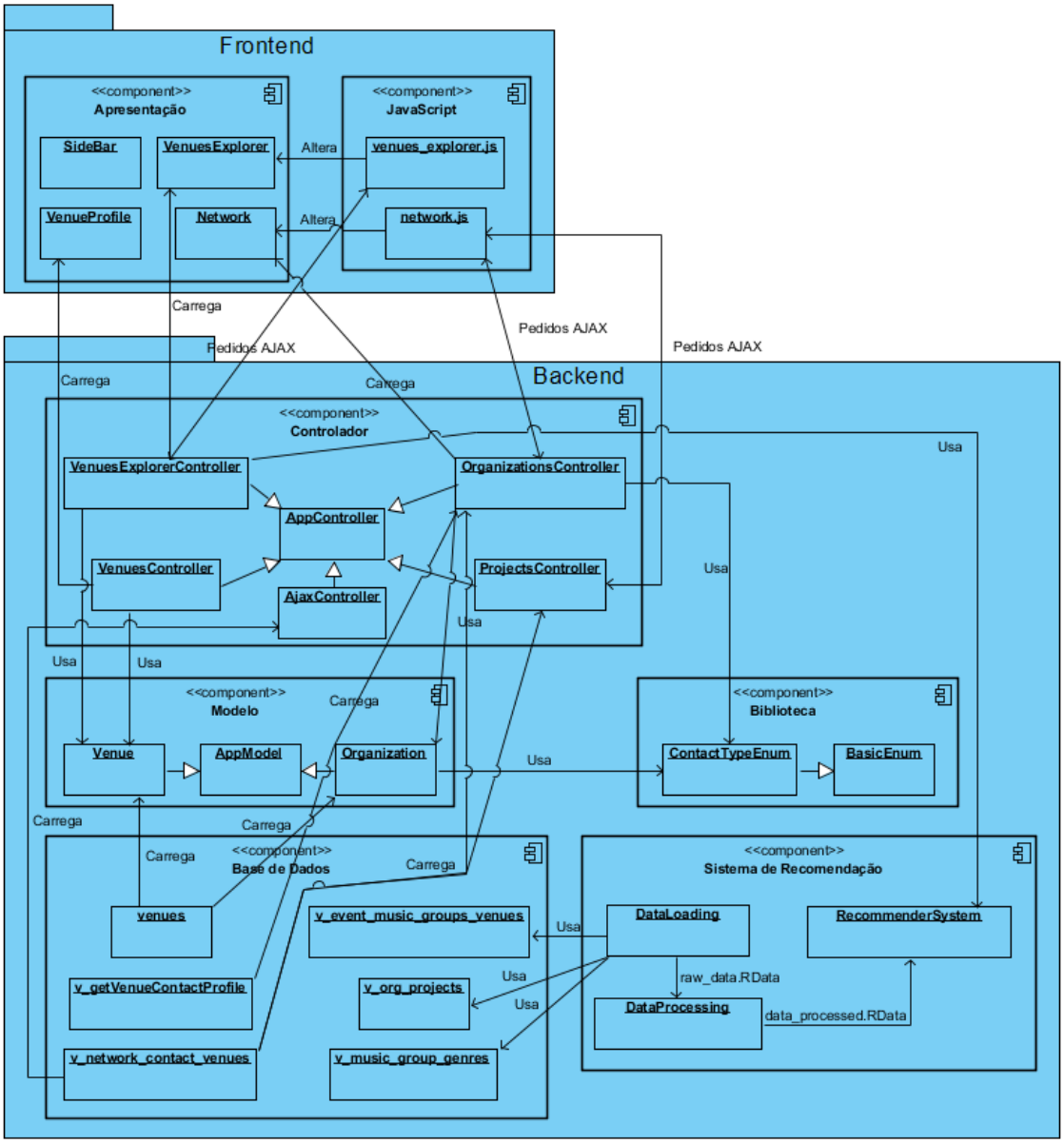

Figura 27 – Diagrama de Arquitetura

O diagrama está dividido nos componentes *frontend* e *backend*. Cada componente identificado está presente no diagrama. Os componentes estão divididos nas suas partes relevantes para o projeto.

No componente Apresentação indicam-se as vistas identificadas na secção 4.2.2. Com esta granularidade é possível perceber que ficheiros JS estão ativos em cada vista. Também se observa que controladores carregam cada vista.

Na secção 4.2.3 identificou-se os ficheiros JS relevantes. Na imagem estão identificados os controladores que recebem pedidos AJAX de cada um desses ficheiros.

No componente Controlador é apresentado o diagrama de classes, no qual AppController é a superclasse de todas as outras classes. Tendo em conta que este é um dos componentes centrais do fluxo dos casos de uso, existem ligações a todos os outros componentes.

Segue-se o componente Modelo que também tem um diagrama de classes. Esta tem a superclasse AppModel, superclasse para todos os modelos.

Tanto o componente Controlador como o componente Modelo usam o componente Biblioteca, para auxiliar no suporte dos vários tipos de contacto. O diagrama de classes do componente Biblioteca indica uma superclasse para todos os enumeradores BasicEnum e a sua subclasse ContactTypeEnum.

Em relação ao componente Base de Dados, apresenta-se as várias tabelas e *views* relevantes. Percebe-se que as *views* são carregadas diretamente pelos componentes controlador e sistema de recomendação, por outro lado, as tabelas são geridas pelas classes Modelo respetivas.

Por fim, o componente sistema de recomendação mostra a interação do seu componente DataLoading com a base de dados, que permite a entrada dos dados, e mostra a interação entre o seu componente RecommenderSystem com o VenuesExplorerController, que representa a saída das recomendações.

5 Implementação

Neste capítulo descreve-se a implementação da solução pensada no capítulo anterior.

Nas próximas secções ilustra-se as alterações feitas ao componente Apresentação e descreve-se, detalhadamente, a implementação do sistema de recomendação, o grande foco deste projeto.

Tendo em conta as descrições presentes no capítulo anterior e o facto de não ser permitida a publicação de código, considera-se que os componentes Javascript, Controlador, Modelo, Biblioteca e Base de Dados estão suficientemente descritos. Logo, não serão abordados em mais detalhe.

5.1 Componente Apresentação

Nesta secção apresenta-se as adições/alterações que foram feitas ao componente Apresentação. Este é o componente mais suscetível a alterações no futuro.

5.1.1 Página inicial do VenuesExplorer

Esta página, ilustrada na Figura 28, contém os campos necessários para a definição dos filtros e um botão para iniciar a pesquisa.

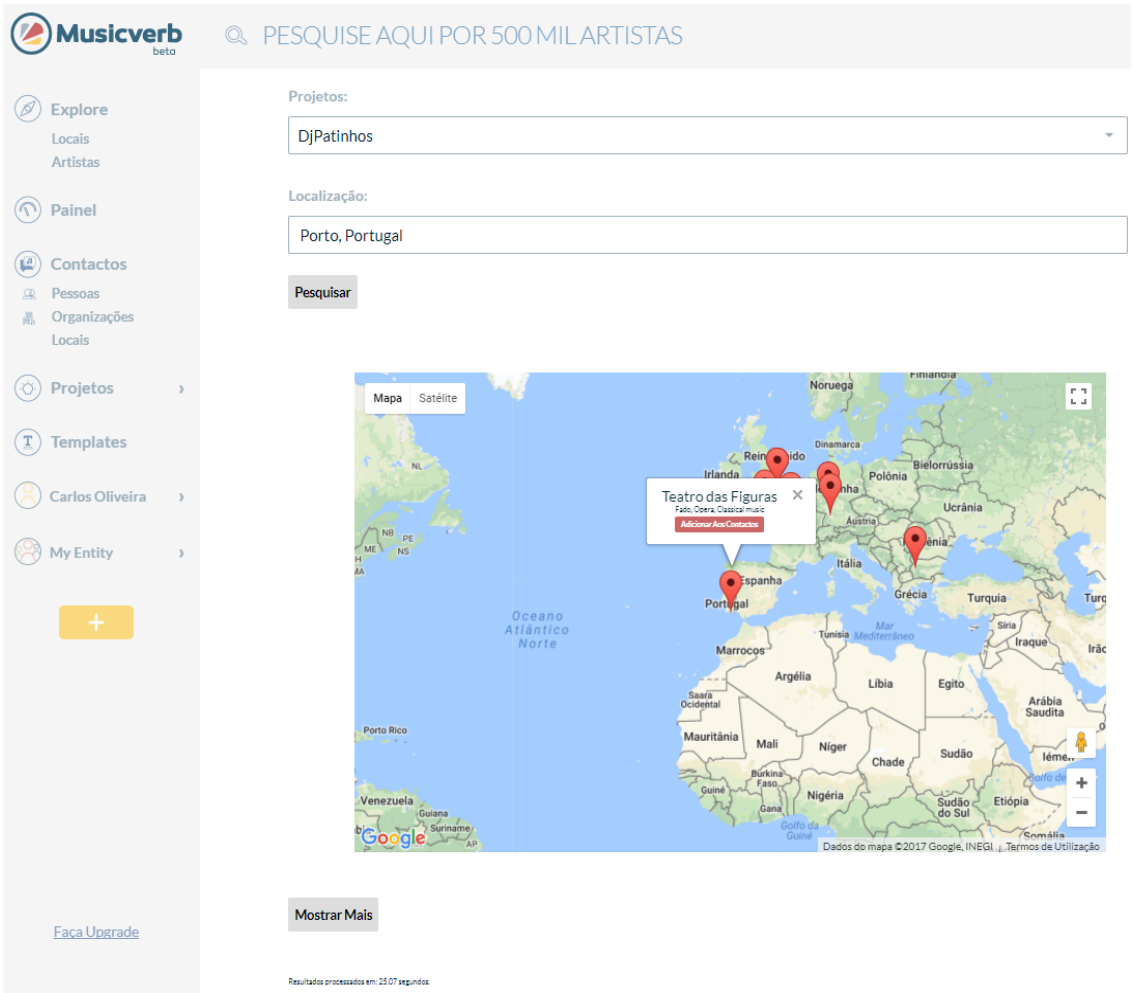

Figura 28 – Exemplo da página inicial do VenuesExplorer após uma pesquisa

Após uma pesquisa, o mapa aparece por baixo contendo os marcadores das *venues* recomendadas. No caso de não estarem todas apresentadas, o botão para mostrar mais é apresentado. Por fim, uma pequena mensagem é colocada no fundo da página com o tempo que demorou a obter as recomendações.

O agente pode selecionar um marcador para obter mais informação sobre cada venue recomendada.

5.1.2 Página inicial da Network

Esta página teve uma nova aba acrescentada: Locais. Na Figura 29 apresenta-se um exemplo contendo três contactos adicionados para teste.

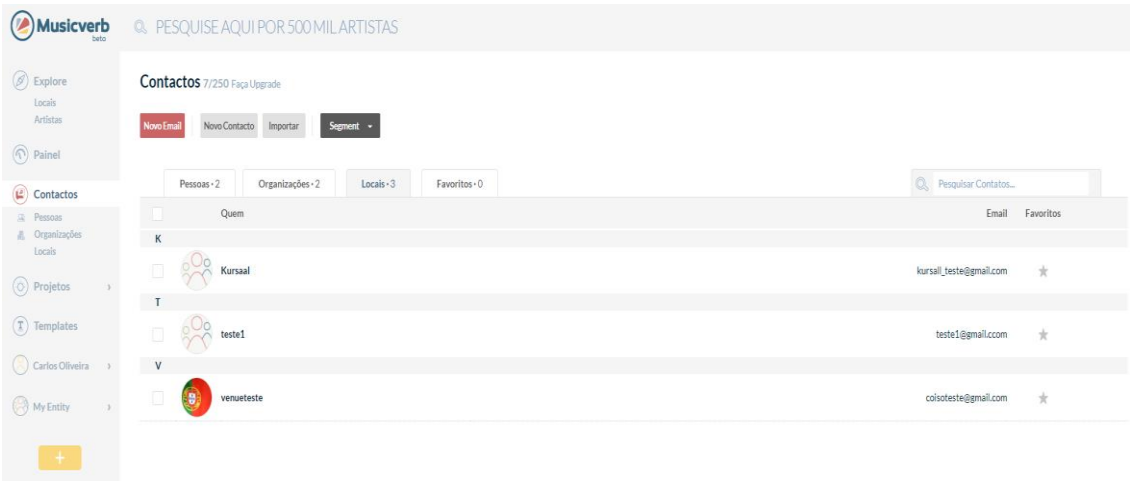

Figura 29 – Exemplo da aba Venues da Network

5.1.3 Página de adição de novo contacto

Esta página acolheu um novo formulário para adicionar contactos do tipo *venue*. Este novo formulário está apresentado na Figura 30.

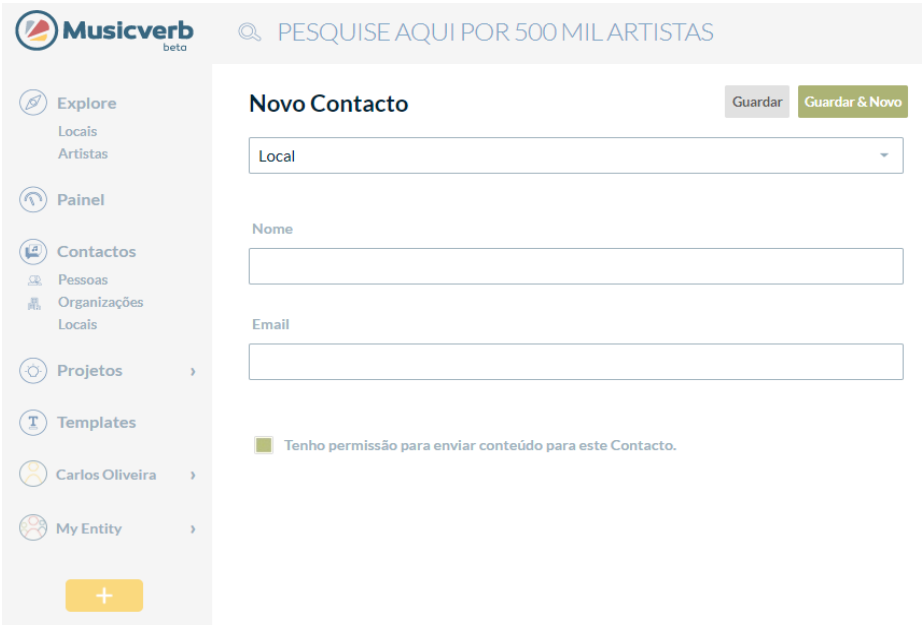

Figura 30 – Novo formulário para adicionar um novo contacto do tipo *venue*

Para adicionar um novo contacto é apenas necessário o nome e *email* do contacto. Posteriormente, na página do perfil do contacto é possível acrescentar mais dados.

5.1.4 Página do perfil de um contacto de uma *venue*

Esta página foi adaptada do perfil de um contacto do tipo Organização. Como se pode ver na Figura 31, é possível acrescentar um número de telemóvel, uma imagem e várias *tags* que são tidas em conta na segmentação dos contactos, na Network.

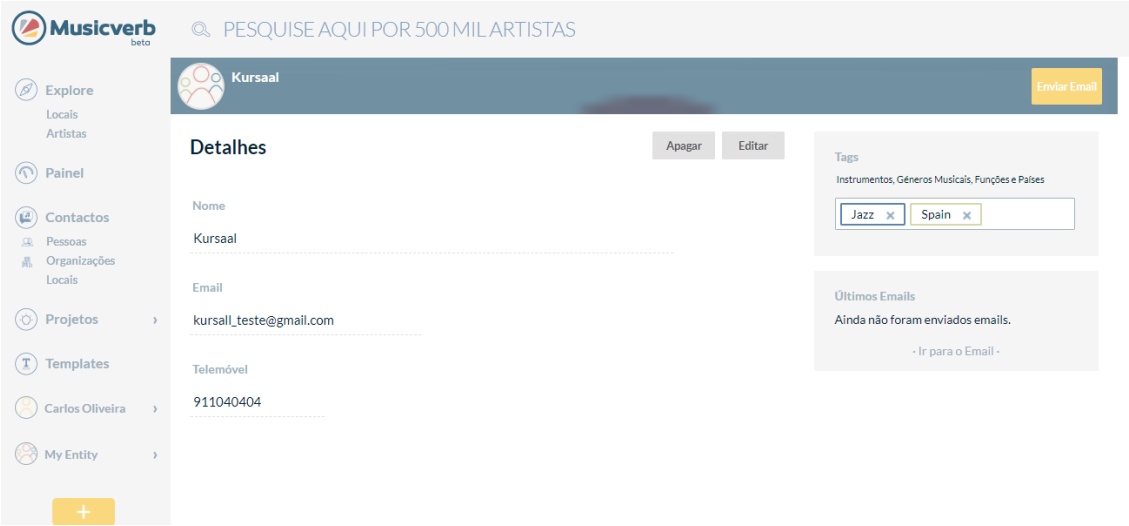

Figura 31 – Exemplo do perfil de um contacto do tipo *venue*

Também é possível observar um histórico dos últimos *emails* enviados a este contacto.

5.1.5 Barra Lateral

A alteração à barra lateral está ilustrada nas imagens anteriores, nesta secção. O Explore é desdobrado em Explore para Artistas e Locais, e a Network recebe um novo submenu Locais.

5.1.6 Localização

As novas páginas e alteração a páginas já existentes criaram a necessidade de suportar a tradução de novas frases. Na Tabela 4 apresenta-se todas as novas frases e respetivas traduções.

Tabela 4 – Tabela das Novas Traduções

Frase	Em Inglês	Em Português
Search	Search	Pesquisar
Input Invalid!	Input Invalid!	Dados de entrada inválidos!
Add To Contacts	Add To Contacts	Adicionar aos Contactos
seconds	seconds	segundos
Results Processed In	Results Processed In	Resultados processados em
No Recommendations Found For The Given Location!	No Recommendations Found For The Given Location!	Não foram encontradas recomendações para essa localização!
No Recommendations Found!	No Recommendations Found!	Não foram encontradas recomendações!
Search Venues	Search Contacts…	Pesquisar Contatos...
Venues	Venues	Locais
Venue	Venue	Local
Artists and Venues	Explore Artists and Venues	Navegue Artistas e Locais
Artists	Artists	Artistas

5.2 Componente Sistema de Recomendação

Nesta secção descreve-se a implementação do sistema de recomendação. A principal incógnita em relação ao sistema era quais os algoritmos que se deveriam usar, tanto na abordagem colaborativa, como na abordagem baseado em conteúdo. Foram implementados

e testados vários algoritmos. Por fim, dois foram selecionados: um algoritmo colaborativo baseado nos perfis dos artistas (CFUB) e uma adaptação do algoritmo k-NN para este domínio específico (PKNN). O processo de seleção de algoritmos e os resultados de cada um serão apresentados no capítulo 6.

Nas secções seguintes apresenta-se as variáveis presentes no ficheiro de configuração do sistema de recomendação, a origem dos dados, o processamento feito sobre eles, a forma como são usados para obter recomendação, e, por fim, descrevem-se os dois algoritmos selecionados.

5.2.1 Configuração

O sistema de recomendação tem vários aspetos que podem ser configurados, antes da execução de qualquer componente. Na Tabela 5 apresenta-se todas as variáveis que podem ser alteradas.

Tabela 5 – Variáveis configuráveis

Variáveis	**Descrição**	**Valor Por Omissão**
data_source	Local onde o sistema irá carregar dos dados. Valores possíveis: "csv" ou "mysql."	"mysql"
N	Valor usado para limitar o número de *venues* que cada algoritmo retorna. É o *top-n* que cada algoritmo deve retornar.	500
cf_treashold	Valor que determina se o historial de atuações é suficiente para que o artista esteja apto para a abordagem colaborativa.	2
dist_limit	Raio da circunferência desenhada à volta do ponto geográfico de referência. Usado para filtrar por localização.	2500

Estas variáveis farão mais sentido ao longo das próximas secções.

5.2.2 Exploração dos Dados

O ponto de partida para a implementação do sistema de recomendação é perceber que dados estão disponíveis e o que é possível fazer com eles.

Na secção 4.2.7 foram identificadas as *views* "v_event_music_groups_venues", "v_music_group_genres" e "v_org_projects". Estas *views* contêm a informação mais adequada para o sistema de recomendação.

A *view* "v_event_music_groups_venues" conecta um artista a todas as *venues* onde ele já atuou. O que permite descrever um evento. Na Tabela 6 identifica-se e descreve-se, sucintamente, os campos carregados (relevantes) dessa *view*.

Tabela 6 – Campos obtidos da *view* "v_event_music_groups_venues"

Campos	**Descrição**
MusicGroupID	Identificador do artista (numérico)
VenueID	Identificador da *venue* (numérico)
geo_point	Ponto geográfico da localização da *venue* (lat, long)
EventDate	Data do evento (timestamp)

Portanto, esta *view* descreve um evento, ligando um artista (MusicGroupID) a uma *venue* (VenueID), numa data (EventDate) e num lugar (geo_point). Estes dados permitem não só descrever uma *venue*, como deduzir o historial de atuações de um artista.

Esta *view* tem mais de um milhão de registos e, originalmente, 16 campos. A Figura 32 ilustra as sete localizações de *venues* (de mais de doze mil localizações diferentes) mais populares, presentes nos dados.

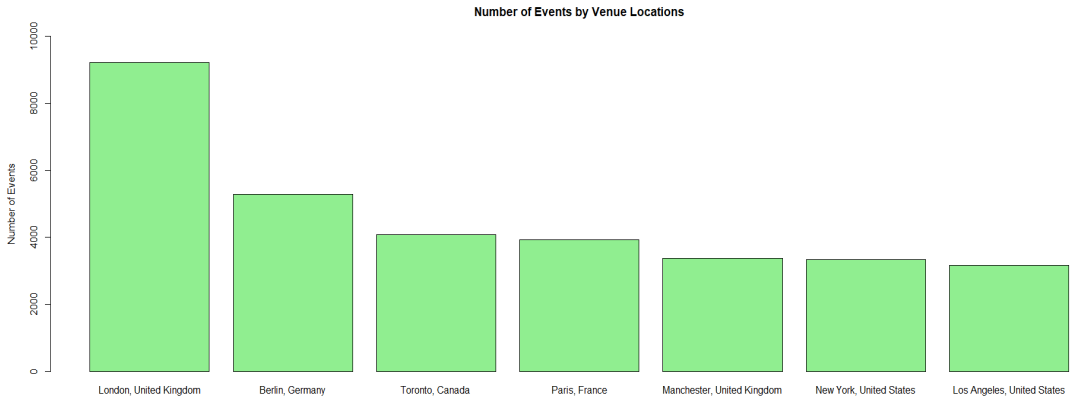

Figura 32 – Número de eventos das sete localizações de *venues* mais populares

Observa-se um grande domínio de *venues* britânicas e americanas, com especial atenção para *venues* londrinas com perto de dez mil eventos.

Em relação à localização dos artistas em si, a Figura 33 ilustra os dez países de origem de artistas (de mais de trinta países diferentes) mais populares.

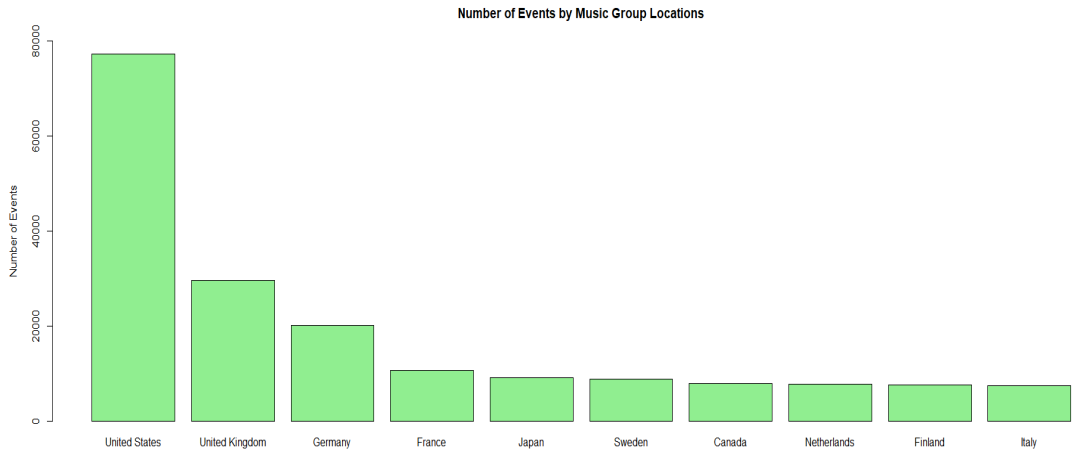

Figura 33 – Número de eventos dos dez países de origem mais populares entre artistas

Neste gráfico percebe-se, claramente, o número esmagador de artistas com a localização de origem nos Estados Unidos, perto de oitenta mil. O segundo país mais popular é o Reino Unido com menos de metade.

Relacionando os dois gráficos anteriores, é interessante observar que existem muitos artistas sedeados nos Estados Unidos, mas existem mais atuações no Reino Unido.

Outro aspeto muito relevante para o projeto é perceber a distribuição dos eventos por ano, isto é, identificar os anos mais populares de eventos presentes nos dados. A Figura 34 apresenta esse gráfico.

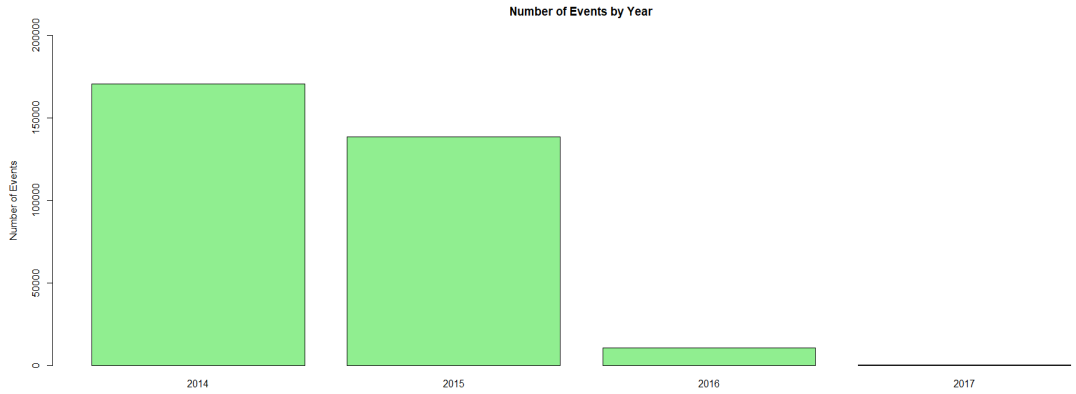

Figura 34 – Número de eventos registados, por ano

Este gráfico indica que os eventos presentes na *view* são dos anos 2014 a 2017, sendo que o número de eventos de 2017 são muito residuais (menos de 100). Deduz-se que a obtenção de dados foi efetuada, principalmente, nos anos de 2014 e 2015.

Avançando para a *view* "v_music_group_genres" (Tabela 7), esta associa géneros musicais a um artista.

Tabela 7 – Campos obtidos da view "v_music_group_genres"

Campos	**Descrição**
MusicGroupID	Identificador do artista (numérico)
Genre	Género musical (texto)

Um artista pode ter um ou mais géneros musicais. Estes dados permitem completar a descrição do perfil de um artista.

Esta *view* tem mais de 226 mil registos, e originalmente 4 campos. A Figura 35 ilustra os quinze géneros musicais (de mais de trezentos géneros diferentes) mais populares.

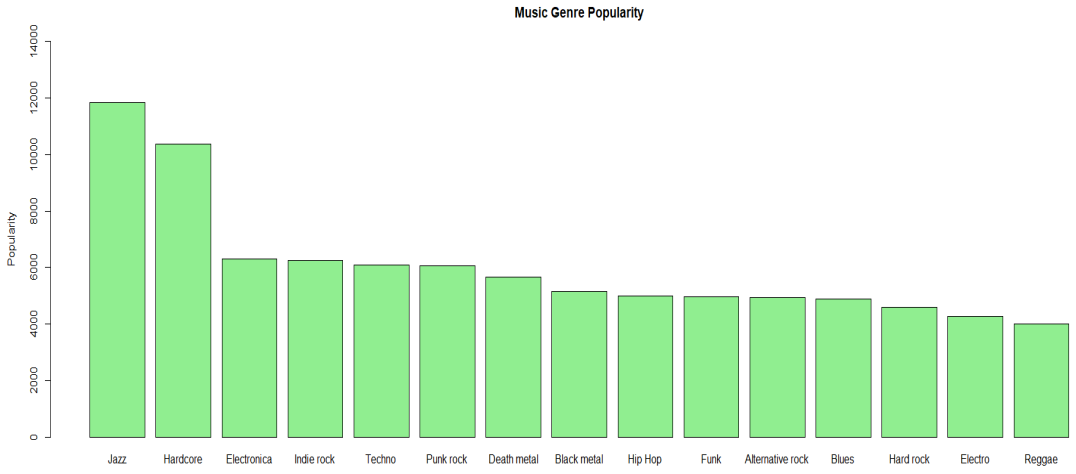

Figura 35 – Popularidade de géneros musicais entre os artistas

Observa-se uma distribuição algo uniforme com os géneros Hardcore e Jazz a isolarem-se na liderança dos géneros mais populares.

Com estas duas *views* é possível descrever *venues* e artistas, e até obter históricos de atuações. Não existe informação sobre avaliações quer de *venues*, quer de artistas, e, a maior parte dos campos são categóricos. Estes fatores limitam o número de algoritmos adequados.

Por fim, a *view* "v_org_projects". Desta view é apenas obtido um campo: "key" (numérico), que representa o identificador de um artista.

Os artistas presentes nesta *view* estão associados a organizações, por isso, são mais suscetíveis a serem usados no sistema de recomendação, logo, devem ser inseridos na lista de artistas aptos para a abordagem colaborativa.

A secção seguinte aborda o processamento dos dados descritos nesta secção.

5.2.3 Processamento dos Dados

Esta secção descreve o processamento e transformação aplicada aos dados. São produzidos três datasets: *venues_data*, *music_group_venues_data* e *v_music_group_genres*.

Em primeiro lugar, os dados obtidos das três *views* são limpos. São removidos registos que não tenham todos os valores, ou que tenham valores inválidos.

De seguida inicia-se a produção do primeiro *dataset*: *venues_data*. Os campos VenueID e geo_point são reutilizados.

Agregando as views "v_event_music_groups_venues" e "v_music_group_genres" (usando o MusicGroupID) é possível associar géneros musicais a *venues*. Isto permite uma descrição melhorada de cada *venue*. Esta associação justifica-se com conhecimento de domínio que mostra que *venues* tendem a especializar-se num conjunto de géneros musicais (e.g., Hard Club tende a ter eventos de rock e a Casa da Música tende a ter eventos de música clássica).

Mas uma *venue* pode ter eventos de um conjunto géneros musicais muito alargado, por isso é necessário determinar o número de géneros mais adequado a associar a cada *venue*. Para determinar este valor, calculou-se o número de género musicais diferentes, por *venue*. Depois, obteve-se a mediana desses valores. Usou-se a mediana em detrimento da média, pois a mediana tem em conta a distribuição probabilística e é menos influenciada por *outliers*.

O valor da mediana obtida foi 3, portanto, três novos campos foram adicionados ao *dataset venues_data* com os três géneros musicais mais populares/frequentes para cada *venue*.

Para concluir o processamento deste *dataset* e aproveitando os cálculos de frequência efetuados, acrescentou-se um novo campo: popularidade. Este campo representa o número de vezes que uma *venue* está associada a um evento, isto é, o número de vezes que uma *venue* aparece nos dados. *Venues* com popularidade de 1 são removidos para reduzir o tamanho do *dataset*.

Assim, a Tabela 8 resume o *dataset venues_data*.

Tabela 8 – *Dataset venues_data*

Campos	**Descrição**
VenueID	Identificador da *venue* (numérico)
geo_point	Ponto geográfico da localização da *venue* (lat, long)
VenueGenre_1	Género musical mais frequente (texto)
VenueGenre_2	Segundo género musical mais frequente (texto)
VenueGenre_3	Terceiro género musical mais frequente (texto)
Popularity	Número de vezes que a *venue* aprece nos dados (numérico)

Avançando para o *dataset music_group_venues_data*, este contém a informação das "transações", isto é, informação das atuações dos artistas. A Tabela 9 descreve os seus campos.

Tabela 9 – *Dataset music_group_venues_data*

Campos	**Descrição**
MusicGroupID	Identificador do artista (numérico)
VenueID	Identificador da *venue* (numérico)
EventDate	Data do evento (timestamp)

Como se pode ver, é um subconjunto da *view* original. Este dataset possui os historiais de atuações dos artistas.

O último *dataset*, *v_music_group_genres*, é a *view* "v_music_group_genres" limpa, sem mais alterações. Este dataset é usado para descrever um artista pelos seus géneros musicais.

O carregamento das *views* é feito pelo componente DataLoading. O posterior processamento é efetuado pelo componente DataProcessing.

Estes dois componentes podem, então, ser executados periodicamente, atualizando os três *datasets*, de forma a manter os resultados do sistema de recomendação adequados e válidos.

5.2.4 Sistema de Recomendação

Nesta secção aborda-se a parte do sistema de recomendação que, de facto, obtém as recomendações. Corresponde ao componente RecommenderSystem que é executado pelo componente Controlador com os filtros definidos pelo agente.

Relembrando o que foi dito na secção 4.2.9, esses filtros (music_group_id e reference_geo_point) e a informação sobre o estado das recomendações (num_displayed e num_to_display) vão validados.

Depois o algoritmo a usar baseia-se na aptidão do artista. O artista está apto se o tamanho do seu historial de atuações for superior ao valor configurável cf_threashold. Se estiver apto, usa-se o CFUB. Caso não esteja, usa-se o PKNN. De qualquer forma, no final, a lista intermédia de *venues* a recomendar (tamanho determinado pelo valor configurável N) é filtrada, usando o reference_geo_point e o valor configurável dist_limit. A equação usada para calcular a distancia entre dois pontos geográficos é a distância de haversine[7].

Devido a este filtro por localização, é necessário que os algoritmos devolvam um elevado número de recomendações, para que seja menos provável que o filtro exclua todas as recomendações.

Por fim, usando os valores num_displayed e num_to_display, o sistema retorna os resultados pedidos.

Nas secções seguintes descreve-se tanto o algoritmo PKNN como o CFUB. Para permitir uma melhor compreensão, serão apresentados exemplos baseados em datasets de exemplo, descritos na próxima secção.

5.2.5 Dados Exemplo

Nesta secção apresenta-se um exemplo para cada um dos três *datasets*, descritos anteriormente: *venues_data*, *music_group_venues_data* e *v_music_group_genres*.

[7] http://wordpress.mrreid.org/2011/12/20/haversine-formula/

Começando pelo *dataset venues_data*, Tabela 10.

Tabela 10 – Exemplo do *dataset venues_data*

VenueID	geo_point	VenueGenre_1	VenueGenre_2	VenueGenre_3	Popularity
1305	40, -8	Rock	Rock Alternativo	Metal	1
1306	38, 10	Música Clássica	Pop	Rock Clássico	1
1307	42, -4	Rock and Roll	Heavy Metal	Rock Clássico	3

O *dataset music_group_venues_data* tem o exemplo apresentado na Tabela 11.

Tabela 11 – Exemplo do *dataset music_group_venues_data*

MusicGroupID	VenueID	EventDate
1	1307	02/12/2014 15:00:00
2	1306	02/10/2015 18:00:00
2	1307	02/12/2014 17:00:00
3	1305	20/01/2014 20:00:00
3	1307	20/01/2015 19:30:00

Por último, o *dataset v_music_group_genres* está na Tabela 12.

Tabela 12 – Exemplo do *dataset v_music_group_genres*

MusicGroupID	Genre
1	Heavy Metal
2	Rock Clássico
2	Rock and Roll
3	Rock
3	Rock Alternativo

Os exemplos apresentados nas secções seguintes vão ter em conta estes exemplos.

5.2.6 Personalized k-NN

Este algoritmo baseia-se nos princípios do algoritmo k-NN e tem como objetivo recomendar *venues* usando apenas o *dataset venues_data*, isto é, sem necessitar das "transações". Logo, é uma abordagem baseada em conteúdo, isto é, baseado na descrição dos itens, neste caso *venues*. Como se observou na Tabela 8, os campos que descrevem uma *venue* são categóricos (géneros musicais), portanto, o tão importante cálculo de similaridade (de forma a determinar as *n venues* mais semelhantes) assenta-se na comparação de *strings*.

Como este algoritmo tem o objetivo de não necessitar de históricos de atuações dos artistas, o perfil de um artista é, então, definido pelos seus géneros musicais (obtidos a partir do *dataset v_music_group_genres*). Nesse caso, os campos relevantes, para o cálculo da similaridade, são os géneros musicais das *venues*. Tendo em conta que cada *venue* tem 3 géneros musicais, a similaridade máxima é 3.

Para determinar as *n venues* mais semelhantes ao perfil de um artista, é necessário percorrer todas as *venues* presentes no *dataset venues_data*. Mesmo após toda a limpeza e filtragem, este *dataset* contém mais de 4000 *venues*. Chegando a este ponto, é essencial implementar otimizações de forma a determinar as recomendações rapidamente. Sem otimizações o algoritmo teria problemas de escalabilidade.

A primeira otimização foi o uso de paralelismo no ciclo que percorre todas as *venues*. Este ciclo foi adaptado a partir dos ciclos paralelos utilizados no processamento de dados.

Esta otimização é possível devido à independência dos vários cálculos de similaridade. Mas não é suficiente, é necessário reduzir, de alguma forma, o número de *venues*.

Para resolver este problema analisou-se o *dataset venues_data* e percebeu-se que estavam a ser calculadas similaridades com *venues* que não tinham qualquer género musical em comum com o artista. Então, efetuou-se um simples filtro à lista de *venues*, mantendo apenas as *venues* com pelo menos um género musical em comum.

Com o problema do tempo de execução resolvido, foi necessário decidir como ordenar e filtrar as *venues*, pois a lista de recomendações obtida nesta fase é ainda muito extensa.

Para isso, a lista de *venues* a recomendar é ordenada, primeiro pelo valor da similaridade, e segundo pela popularidade. Depois, o *top-n* dessa lista ordenada é retornada.

Por exemplo, para o artista com identificador 2 o seu perfil contém "Rock and Roll e Rock Clássico". Então o algoritmo irá percorrer as *venues* 1306 e 1307 (1305 é descartada por não ter géneros em comum com o artista), calculando as similaridades 1 e 2, respetivamente. Então, para este exemplo, a lista ordenada de *venues* a recomendar seria "1307 e 1306".

Neste exemplo, as duas *venues* recomendadas estão presentes no histórico do artista 2 e não são removidas, porque este algoritmo é feito para artistas que não possuem um histórico de atuações.

5.2.7 Collaborative Filtering User Based

Este algoritmo baseia-se nos princípios de algoritmos colaborativos, na medida em que, calcula similaridades, neste caso entre utilizadores, com base nas avaliações que os utilizadores efetuaram aos itens.

No caso deste projeto, não existem dados sobre avaliações. Portanto, resolveu-se usar "avaliações" implícitas binárias. Isto é, se um artista atuou numa *venue*, avalia com um 1. Se não atuou, avalia com um 0.

Usando este principio e usando o *dataset music_group_venues_data*, uma matriz binária é construída. Um artista por linha. Uma *venue* por coluna. O exemplo da Tabela 13 é feito a partir do *dataset music_group_venues_data* de exemplo.

Tabela 13 – Exemplo de uma matriz binária

MusicGroupID	**1305**	**1306**	**1307**
1	0	0	1
2	0	1	1
3	1	1	0

Com base nesta matriz, é possível calcular a semelhança entre artistas ou entre *venues*. Neste algoritmo calcula-se a similaridade entre artistas.

Na secção 2.6.1 foram identificadas várias equações que permitem calcular similaridade entre vetores (cada linha da matriz é um vetor). Como a matriz é binária (logo, os vetores também), selecionou-se a similaridade de jaccard (4).

Assim, usando essa equação é possível construir uma matriz de similaridade entre artistas. Usando a matriz binária da Tabela 13 a matriz de similaridade resultante seria igual à da Tabela 14.

Tabela 14 – Exemplo de uma matriz de similaridade

	1	**2**	**3**
1	1	0.5	0
2	-	1	0.33
3	-	-	1

Como a similaridade entre o artista A e o artista B é igual à similaridade entre o artista B e o artista A, esta matriz é simétrica, então é apenas necessário preencher um lado da matriz.

Agora que a semelhança entre todos os artistas está calculada, é possível determinar os artistas mais semelhantes, e a partir daí, determinar as *venues* a recomendar. Devido à necessidade de tempos de execução mínimos, essa determinação tem de ser pré-processada.

Portanto, com base na matriz de similaridade, é criada uma tabela com os 5 artistas mais semelhantes, para cada artista. Usando o exemplo da Tabela 14, obtém-se a Tabela 15.

Tabela 15 – Exemplo da tabela dos artistas mais semelhantes

MusicGroupID	**Top1**	**Top2**	**Top3**	**Top4**	**Top5**
1	2	3	-	-	-
2	1	3	-	-	-
3	2	1	-	-	-

Com apenas três artistas, a tabela apenas contém o top 2, mas é suficiente para exemplificar.

Todas estas matrizes e tabelas que referi (Tabela 13, Tabela 14 e Tabela 15) são construídas no componente DataProcessing. Isto é, são pré processadas. Assim, os tempos de execução para a obtenção das recomendações é muito mais baixo. Se o número de artistas aptos para a abordagem colaborativa aumentar, este algoritmo pode encontrar problemas de escalabilidade, pois as matrizes serão demasiado grandes.

Então, quando o VenuesExplorerController executa o sistema de recomendação, o identificador do artista passado por parâmetro (de referência) é usado para determinar os 5 artistas mais semelhantes.

Depois, as listas de *venues* presentes nos historiais de atuações desses 5 artistas são agregadas numa só (removendo os duplicados). De seguida retira-se as *venues* já presentes no historial de atuações do artista de referência.

Por fim, essa lista é ordenada pela popularidade e o *top-n* é retornado.

Exemplificando para o artista identificado pelo número 2 e usando a Tabela 15, este algoritmo retornaria uma lista de recomendações com a *venue* 1305. Os artistas mais semelhantes em relação ao artista 2 são os artistas 1 e 3. A reunião da lista de *venues* onde atuaram contém as seguintes *venues*: 1305, 1306, 1307. Como o artista 2 tem as *venues* 1306 e 1307 no seu perfil, é recomendada a *venue* 1305.

5.2.8 Testes Unitários

De forma a testar a implementação efetuada, realizaram-se vários testes unitários.

Para executar estes testes unitários foi implementado um novo componente, UnitTests. Este componente começa por criar dados fictícios e estáticos, guardando-os num ficheiro RData. Estes dados artificiais são usados para fornecer a informação requerida pelos métodos e para validar o retorno desses métodos.

6 Avaliação da Solução

Neste capítulo descreve-se a avaliação dos algoritmos do sistema de recomendação e indica-se como avaliar a solução final.

6.1 Avaliação Sistema de Recomendação

Como foi referido na secção 3.6, é difícil comparar resultados de forma a determinar os melhores algoritmos, se não forem usadas métricas uniformizadoras. Por isso, desenvolveu-se várias métricas com o objetivo de uniformizar os resultados, para que possam ser comparáveis. Estas métricas foram descritas na secção 3.6 e são: *Accuracy* (5), *Gr_coverage* 6), *Mg_coverage* e tempo de execução.

A *Accuracy* representa a eficiência do algoritmo, isto é, quantas recomendações foram necessárias para obter uma boa recomendação. O valor é baixo se o algoritmo retornar um valor elevado de recomendações, em comparação com o número de boas recomendações. Por outras palavras, este valor traduz quantas boas recomendações foram obtidas, por cada 100 recomendações totais.

O *Gr_coverage* representa a eficácia, tendo em conta que o objetivo é obter todas as boas recomendações possíveis. O valor é baixo se o algoritmo não conseguir obter boas recomendações. Portanto, e tendo em conta que os algoritmos devem retornar um elevado número de recomendações devido ao filtro por localização, esta métrica é mais importante do que a *accuracy*.

O *Mg_coverage* representa a cobertura suportada pelo algoritmo. Quantos mais artistas estiverem suportados, melhor. Esta métrica permite identificar algoritmos muito eficientes e eficazes, mas que apenas obtêm recomendações para um número muito baixo de artistas.

Por fim, o tempo de execução é o tempo que o algoritmo demora a obter as recomendações.

6.1.1 Metodologia

Para que os resultados tenham significado, os dados usados para avaliar cada algoritmo devem ser iguais. Por isso, definiu-se uma lista de 1548 artistas, todos os artistas com mais de duas atuações presentes nos dados de treino. Cada algoritmo terá de obter recomendações para todos os 1548 artistas.

De forma a determinar o que é uma boa recomendação, usa-se conhecimento de domínio. Normalmente um agente prepara o ano seguinte com antecedência. Logo, e tendo em conta a distribuição de eventos por ano da Figura 34, divide-se os dados num conjunto de treino com as atuações de 2014, e num conjunto de teste com as atuações de 2015 e 2016. Simulando assim a preparação de um agente no final de 2014 para os anos seguintes.

O historial de atuações de 2014 é utilizado, pelos algoritmos, para obterem recomendações para cada artista. O historial de 2015 e 2016 é usado como dados de teste. Se o algoritmo recomendar uma *venue* que esteja no historial de teste (de 2015/16), considera-se uma boa recomendação.

Artistas que não tenham atuações em 2015/16 não são contabilizados no cálculo das métricas, pois é impossível obter boas recomendações para esse artista.

Todos os datasets e matrizes têm de ser processadas novamente, usando apenas os dados de treino. Só desta forma é possível simular que os algoritmos estão a ser executados no final de 2014.

6.1.2 Comparação

Implementaram-se vários algoritmos com sucesso variável. Na Tabela 16 apresentam-se os resultados de 5 ensaios, executados para cada algoritmo, num computador de características semelhantes ao servidor.

Estes resultados permitem perceber quais os melhores algoritmos para integrar com o sistema de recomendação.

Tabela 16 – Resultados dos vários algoritmos de recomendação

Algoritmo	Accuracy	Gr_coverage	Mg_coverage	Tempo médio por artista	Tempo médio por ensaio
Abordagem colaborativa baseada nos artistas (CFUB)	0.68%	26.52%	100%	4.85s	1h25min
Abordagem colaborativa baseada em itens (CFIB)	0.665%	17.8%	96.32%	0.61s	46min
Abordagem baseada em conteúdo específica do domínio (PKNN)	0.91%	21.08%	92.64%	3,454s	51min

Estes algoritmos obtiveram os mesmos valores para a *accuracy*, *Gr_coverage* e *Mg_coverage* em todos os ensaios, respetivos. Isto significa que são determinísticos, não há aleatoriedade na forma como determinam as recomendações.

Os valores de *accuracy* são baixos o que significa que os algoritmos têm de produzir muitas recomendações para encontrarem uma boa recomendação. Estes valores podem ser justificados com o conjunto de *venues* que podem ser recomendadas, isto é, cada algoritmo apenas consegue recomendar *venues* que conheça (não inventa). No caso destes testes só é possível recomendar *venues* que estejam presentes nos dados de 2014. *Venues* que apenas apareçam nos dados de 2015/16 nunca serão recomendadas, o que torna a obtenção de boas recomendações mais difícil.

O único algoritmo baseado em conteúdo (PKNN) obteve resultados muito aquém do desejado, pois apenas conseguiu prever 21.8% das boas recomendações. Mas é o único que obtém recomendações com base nos géneros musicais de artistas, logo, é o único que consegue recomendar *venues* a artistas sem histórico de atuações. O seu *Mg_coverage* traduz a percentagem de artistas que têm géneros musicais associados.

Em relação aos dois algoritmos colaborativos, o algoritmo CFUB obteve melhores resultados, apesar de ser mais lento e ter obtido resultados baixos. Este algoritmo conseguiu obter

recomendações para todos os artistas testados, isto significa que com capacidade de memória ilimitada, este algoritmo consegue recomendar a qualquer artista com historial de atuações.

Por outro lado, o algoritmo CFIB (muito semelhante ao CFUB, mas a matriz de similaridade processada contém a similaridade entre *venues*, em vez de artistas) obteve os resultados mais baixos. Este algoritmo consegue recomendar a artistas que tenham atuado em *venues* suportadas. Caso um artista tenha atuado em *venues* que não estejam presentes na matriz de similaridade, este algoritmo não consegue recomendar.

Em resumo, os resultados ficaram aquém das expectativas apesar de serem previsões difíceis de fazer. Espera-se que com mais dados à disposição dos algoritmos os resultados melhorem.

6.2 Avaliação Solução Final

A solução final após entrada em produção será avaliada segundo o proveito dos seus utilizadores. A solução final engloba todo o desenvolvimento feito neste projeto, não só o sistema de recomendação.

Nas secções seguintes indicam-se as métricas, hipótese e metodologia a usar.

6.2.1 Métricas

A métricas a utilizar para avaliar a solução final é a exatidão (taxa de acerto) das recomendações. Considera-se que uma recomendação "acertou" se o utilizador adicionar o contacto da *venue* recomendada.

6.2.2 Hipótese

A hipótese a ser testada é que a taxa de acerto será superior a 80%, isto é, prevê-se que os agentes adicionem os contactos de, pelo menos, 80% das *venues* recomendadas.

6.2.3 Metodologia

Para avaliar a solução serão usados dados implícitos baseados nas ações dos utilizadores. A taxa de acerto será calculada com o rácio entre o número de contactos adicionados pelos agentes, de entre todas as *venues* recomendadas. Isto é, quando um agente adiciona o contacto de uma *venue*, implicitamente considera-se que foi uma boa recomendação.

Se o aproveitamento do VenuesExplorer pelos agentes não for o previsto será necessário perceber a razão. Uma possível justificação pode dever-se ao facto de os agentes não adicionam venues que julguem estar demasiado longe. Isso pode provocar alterações ao valor configurável dist_limit ou mesmo à alteração da fórmula usada para filtrar por localização.

Outra possibilidade é alterar a forma de cálculo da taxa de acerto para ter em conta a distância ao ponto de referência (adição de contactos de venues mais próximas pesariam mais do que venues mais longe). Por último, caso o aproveitamento continue a não ser o desejado terá de se alterar ou adicionar novos algoritmos.

7 Conclusões

Este projeto foi desenvolvido no contexto da plataforma *web* da Musicverb, e teve o objetivo de reduzir o tempo gasto pelos agentes de artistas na procura de novos locais, adequados ao artista que representam. Para isso, propôs-se uma solução que consiste num sistema de exploração/recomendação de *venues* híbrido, com fácil integração na plataforma da Musicverb.

Antes do projeto avançar para implementação, analisou-se a área de negócio da Musicverb, a música ao vivo, identificando vários aspetos importantes como o volume de negócio e o seu crescimento nos últimos anos. Também se analisou a plataforma em si, de forma a perceber o que já existia, para identificar locais onde o projeto poderia integrar-se e incentivar o uso de outras funcionalidades da plataforma. Depois, fez-se uma análise de valor do projeto de forma a validar as suas premissas.

Para uma implementação informada do sistema de recomendação, fez-se o levantamento do estado da arte de sistemas de recomendação. Neste levantamento, descreveu-se as várias abordagens comuns utilizadas, esclarecendo as fraquezas de cada uma, e, por fim, fez-se uma revisão literária que permitiu identificar várias técnicas referidas em artigos científicos.

As técnicas identificadas foram, de seguida, apresentadas, com o intuito de elucidar a sua possível adaptação ao projeto, identificando as várias métricas de avaliação de cada técnica.

Com esta bagagem teórica, o *design* da solução foi feito com mais lucidez. Primeiro, foram identificados, e descritos detalhadamente, os vários casos de uso. De seguida, a solução foi dividida em vários componentes *frontend* e *backend* de forma a permitir uma descrição mais focada e detalhada de cada componente. Estas descrições foram suportadas por diagramas de arquitetura e sequência. O componente sistema de recomendação recebeu uma atenção especial, pois é fundamental para o projeto.

Depois da análise e *design*, avançou-se para a implementação. A descrição da implementação focou-se no sistema de recomendação e no componente apresentação. Aqui, foram ilustradas as alterações à interface gráfica da plataforma, incluindo as novas traduções.

A descrição da implementação do sistema de recomendação foi dividida em várias partes, desde a exploração dos dados, que permitiu compreender o volume de dados, ao processamento aplicado a esses mesmos dados. Cada algoritmo foi descrito recorrendo a exemplos simples, que ilustraram o seu funcionamento.

Estes algoritmos foram avaliados usando métricas concebidas especialmente para este projeto, para que seja possível a comparação entre os vários algoritmos. Os algoritmos PKNN e CFUB obtiveram os melhores resultados (em cada abordagem) e foram, por isso, selecionados para serem utilizados no sistema de recomendação.

Para além desta avaliação, é necessário fazer uma avaliação geral ao projeto, como um todo. Esta avaliação é apenas explicada, pois é necessário o contributo dos utilizadores para se determinar a eficácia do sistema.

No final do projeto, todos os objetivos foram alcançados com sucesso.

7.1 Limitações e Trabalho Futuro

No futuro poderá ser benéfico encontrar mais otimizações para o sistema de recomendação, em particular para os seus algoritmos. Novas formas de melhorar a descrição das *venues* pode abrir a oportunidade de usar algoritmos e técnicas diferentes, que podem obter melhores resultados.

Para além de melhores descrições é útil obter informação sobre *venues* que não estejam presentes nos dados, isto porque, recomendar *venues* para 2018 usando os dados atuais vai encontrar o mesmo problema observado na avaliação, onde se faz recomendações de *venues* para 2015/16 com os dados de 2014. Este problema será resolvido quando a base de dados contiver o registo de eventos de todas as *venues* no mundo.

O algoritmo PKNN consegue obter recomendações em tempo útil, mas com a adição de mais *venues*, o tempo de execução pode tornar-se demasiado alto.

Os algoritmos colaborativos têm a capacidade de conseguir recomendar venues a qualquer artista com um historial de atuações. O que limita estes algoritmos é a memória. Uma solução para este problema permitiria aumentar o número de artistas aptos para a abordagem colaborativa, melhorando as recomendações.

Para um futuro a mais curto prazo, fica a preparação do projeto para o seu lançamento em produção e consequente avaliação.

Referências

Abbas, A., Zhang, L., & Khan, S. U. (2015). A survey on context-aware recommender systems based on computational intelligence techniques. *Computing*, *97*(7), 667–690. https://doi.org/10.1007/s00607-015-0448-7

About the Live Music Industry. (n.d.). Retrieved from http://www.ilmc.com/index.php/about/about-the-live-music-industry

Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, *22*(2), 207–216. https://doi.org/10.1145/170036.170072

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297. https://doi.org/10.1023/A:1022627411411

Cui, H., & Zhu, M. (2014). Collaboration filtering recommendation optimization with user implicit feedback. *Journal of Computational Information Systems*, *10*(14), 5855–5862. https://doi.org/10.12733/jcis10758

Cui, L., Ou, P., Fu, X., Wen, Z., & Lu, N. (2016). A novel multi-objective evolutionary algorithm for recommendation systems. *Journal of Parallel and Distributed Computing*. https://doi.org/10.1016/j.jpdc.2016.10.014

Domingo, P. (2016). *Global Music Report 2016: State of the Industry*. *IFPI*. Retrieved from http://www.ifpi.org/downloads/GMR2016.pdf

Elahi, M., Ricci, F., & Rubens, N. (2016). A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, *20*, 29–50. https://doi.org/10.1016/j.cosrev.2016.05.002

Gunn, S. R. (1998). Support Vector Machines for Classification and Regression. *Image Speech and Intelligent Systems Technical Report*, *14*(May), 230–67. https://doi.org/10.1039/b918972f

Johnson, C. (2014). Algorithmic Music Recommendations at Spotify. Retrieved from http://pt.slideshare.net/MrChrisJohnson/algorithmic-music-recommendations-at-spotify

Katarya, R., & Verma, O. P. (2016). An effective web page recommender system with fuzzy c-mean clustering. *Multimedia Tools and Applications*. https://doi.org/10.1007/s11042-016-4078-7

Koen, P. a. (2004). Understanding the Front End: A Common Language and Structured Picture. *Stevens Institute of Technology*.

Koen, P. A., Ajamian, G. M., Boyce, S., Clamen, A., Fisher, E., Fountoulakis, S., … Seibert, R. (2002). Fuzzy Front End: Effective Methods, Tools, and Techniques. In *The PDMA Toolbook for new product development*. New York: John Wiley & Sons.

Kuzelewska, U., & Ducki, R. (2013). Collaborative Filtering Recommender Systems in music recommendation. *Advances in Computer Science Research Collaborative*, *10*, 67–79. https://doi.org/10.1561/1100000009

Lah, N. S. C., Hussin, A. R. C., Rahim, N. Z. A., & Busalim, A. H. (2016). Social Learning Approach in Designing Persuasive E-Commerce Recommender System Model. *Journal of Theoretical and Applied Information Technology*, *90*(2), 77–85. Retrieved from http://search.proquest.com/docview/1823846968?accountid=13031

Lee, H. joo, Shin, H., Hwang, S. seob, Cho, S., & MacLachlan, D. (2010). Semi-Supervised Response Modeling. *Journal of Interactive Marketing*, *24*(1), 42–54. https://doi.org/10.1016/j.intmar.2009.10.004

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: A survey. *Decision Support Systems*, *74*, 12–32. https://doi.org/10.1016/j.dss.2015.03.008

Lv, G., Hu, C., & Chen, S. (2016). Research on recommender system based on ontology and genetic algorithm. *Neurocomputing*, *187*, 92–97. https://doi.org/10.1016/j.neucom.2015.09.113

Mild, A., & Reutterer, T. (2003). An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data. *Journal of Retailing and Consumer Services*, *10*(3), 123–133. https://doi.org/10.1016/S0969-6989(03)00003-1

Pandora Music, I. (2016). About The Music Genome Project®. Retrieved from http://www.pandora.com/about/mgp

Park, D. H., Kim, H. K., Choi, I. Y., & Kim, J. K. (2012). A literature review and classification of recommender systems research. *Expert Systems with Applications*, *39*(11), 10059–10072. https://doi.org/10.1016/j.eswa.2012.02.038

Quinlan, J. (1993). *C4. 5: programs for machine learning*. *Machine Learning* (Vol. 240). https://doi.org/10.1016/S0019-9958(62)90649-6

Ricci, Francesco Rokach, Lior Shapira, Bracha Kantor, P. B. (2011). *Recommender System Handbook*. *Springer US*. https://doi.org/10.1007/978-0-387-85820-3

Rodrigues, F. (2015a). Clustering.

Rodrigues, F. (2015b). Regras de Associação Introdução.

Rodrigues, F. (2016a). Data Mining.

Rodrigues, F. (2016b). Model Evaluation.

Rutherford, T. (2016). Music Industry Revenue In 2016. Retrieved February 19, 2017, from https://www.careersinmusic.com/music-industry-revenue

Sarwar, B. M., Karypis, G., Konstan, J. a, & Riedl, J. T. (2000). Application of Dimensionality

Reduction in Recommender System - A Case Study. *Architecture*, *1625*, 264–8. https://doi.org/10.1.1.38.744

Schafer, J. Ben, Konstan, J., & Riedl, J. (2001). E-commerce recommendation applications. *Applications of Data Mining to Electronic …*, 115–153. https://doi.org/10.1007/978-1-4615-1627-9_6

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '02*, (August), 253. https://doi.org/10.1145/564376.564421

Song, Y., Dixon, S., & Pearce, M. (2012). A survey of music recommendation systems and future perspectives. *9th International Symposium on Computer Music Modeling and Retrieval*, (June), 19–22. Retrieved from http://cmmr2012.eecs.qmul.ac.uk/sites/cmmr2012.eecs.qmul.ac.uk/files/pdf/papers/cmmr2012_submission_42.pdf

Statista. (2016). U.S. Music Industry - Statistics & Facts. Retrieved February 15, 2017, from https://www.statista.com/topics/1639/music/

Wu, D., Zhang, G., & Lu, J. (2015). A fuzzy preference tree-based recommender system for personalized business-to-business e-services. *IEEE Transactions on Fuzzy Systems*, *23*(1), 29–43. https://doi.org/10.1109/TFUZZ.2014.2315655

Zhang, Z., Liu, K., Wang, W., Zhang, T., & Lu, J. (2007). a Personalized Recommender System for Telecom Products and Services. *Artificial Intelligence*, 689–693.