**UNIVERSIDADE FEDERAL DO CEARÁ**

**CAMPUS QUIXADÁ**

**CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO**

**DANRLEY DA SILVA TEIXEIRA**

# PREENCHIMENTO DE PLAYLISTS UTILIZANDO TÉCNICAS DE SISTEMAS DE RECOMENDAÇÃO BASEADAS EM FILTRAGEM COLABORATIVA

**QUIXADÁ**

**2018**

DANRLEY DA SILVA TEIXEIRA

# PREENCHIMENTO DE PLAYLISTS UTILIZANDO TÉCNICAS DE SISTEMAS DE RECOMENDAÇÃO BASEADAS EM FILTRAGEM COLABORATIVA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação. Área de Concentração: Computação

Orientador: Prof. Me. Regis Pires Magalhães

QUIXADÁ

2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

---

T265p Teixeira, Danrley da Silva.
Preenchimento de Playlists Utilizando Técnicas de Sistemas de Recomendação Baseadas em Filtragem Colaborativa / Danrley da Silva Teixeira. – 2018.
35 f.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Sistemas de Informação, Quixadá, 2018.
Orientação: Prof. Me. Regis Pires Magalhães.

1. Sistemas de Recomendação (filtragem de informações). I. Título.

CDD 005

---

DANRLEY DA SILVA TEIXEIRA

# PREENCHIMENTO DE PLAYLISTS UTILIZANDO TÉCNICAS DE SISTEMAS DE RECOMENDAÇÃO BASEADAS EM FILTRAGEM COLABORATIVA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação. Área de Concentração: Computação

Aprovado em: ___/___/_____

BANCA EXAMINADORA

---

Prof. Me. Regis Pires Magalhães (Orientador)
Universidade Federal do Ceará (UFC)

---

Profa. Dra. Ticiana Linhares Coelho da Silva
Universidade Federal do Ceará (UFC)

---

Prof. Me. Victor Aguiar Evangelista de Farias
Universidade Federal do Ceará (UFC)

À minha família, em especial minha Mãe e meu falecido Pai pelo suporte e incentivo. À minha namorada e melhor amiga Camilla Almeida pelo apoio e companheirismo. À todos meus amigos, professores, orientadores, colegas de trabalho.

## AGRADECIMENTOS

Aos meus pais, que nos momentos de minha ausência dedicados ao estudo superior, sempre fizeram entender que o futuro é feito a partir da constante dedicação no presente!

A minha namorada e melhor amiga Camilla Almeida, pelo incentivo, broncas e companheirismo.

Aos Prof. Regis Pires Magalhães e o Prof. Lucas Ismailly por me orientarem nas duas etapas deste trabalho.

Aos meus grandes amigos Alan Ribeiro, Matheus Pereira, Yago Alves, por todas as palavras de incentivo e motivação.

Aos grandes amigos que a graduação me proporcionou sendo eles, Emanuel Eduardo, Guilherme Estevão, Neto Deolino, Marcelo Gonçalves, Richallyson Lima.

Aos servidores técnicos administrativos do campus Quixadá que me ajudaram muito na graduação, em especial Ryanne Paz e Venício Oliveira.

Agradeço a todos os professores por me proporcionar o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, por tanto que se dedicaram a mim, não somente por terem me ensinado, mas por terem me feito aprender, em especial Ricardo Reis e Davi Romero.

E à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e ao Programa Ciência sem Fronteiras, pelo financiamento da bolsa de graduação sanduíche em Berlim na Alemanha.

"Não deveríamos estar buscando heróis, deveríamos estar buscando boas ideias."

(Noam Chomsky)

## RESUMO

Com a rápida expansão das músicas em formato digital, gerenciar e pesquisar por músicas se tornou significante. Com esta expansão, os sistemas de recomendação de música têm papel fundamental neste processo. Porém pela subjetividade das músicas e seus ouvintes, não é possível determinar com precisão total qual música recomendar. Deste modo *playlists*, listas de reprodução de músicas conseguem se adequar melhor as necessidade de um usuário, quanto ao seu humor ou momento. Atualmente uma abordagem de recomendação que se tem provado eficiente é a chamada filtragem colaborativa, onde são utilizados dados de outros usuários para criação de recomendações. Com base nisso, este trabalho apresenta um sistema recomendador de músicas para preenchimento de *playlists* utilizando filtragem colaborativa, baseada em popularidade de músicas e similaridades entre *playlists*.



**Palavras-chave:** Sistemas de Recomendação. Músicas. Playlists. Filtragem Colaborativa. Similaridade. Popularidade.

**ABSTRACT**

With the rapid expansion of digital music, manage and search for songs has became significant. With this expansion, the music recommender systems plays a fundamental role in this process. But due to subjectivity of music and its listeners, it is not possible do determine with total precision which music to recommend. This way playlists, list of songs, can be more precise to the user's need, as to their mood and moment. Currently a recommendation approach that has been proved efficient is the collaborative filtering, where the data from different users is used to build recommendations. This paper presents a music recommender system to fill playlists using collaborative filtering, based on songs popularity and similarity between playlists.

**Keywords:** Recommendation Systems. Songs. Playlists. Collaborative Filtering. Similarity. Popularity.

## LISTA DE QUADROS

**SUMÁRIO**

# 1 INTRODUÇÃO

Com a expansão da internet nas últimas décadas, a rede mundial de computadores se tornou a maior fonte de acesso a informação multimídia como vídeos, livros, músicas e etc. Muitas pessoas consideram que música é um importante fator em suas vidas e escutam música frequentemente. Uma pesquisa (SCHÄFER *et al.*, 2013) indicou que muitos participantes escutam música mais frequentemente que quaisquer outras atividades como, assistir televisão, ler livros, e ver filmes.

No entanto, organizar e gerenciar milhões de músicas produzidas pela sociedade acaba se tornando um problema. Neste âmbito serviços de músicas por *streaming* surgem como mecanismo de organização e facilidade no momento de ouvir músicas, alguns dos mais famosos serviços de *streaming* são: Spotify[1], Deezer[2], Google Play Music [3] etc.

Apesar de uma maior organização e gerenciamento com relação as músicas, os serviços de *streaming* possuem algoritmos responsáveis pela coleta de interesses musicais dos seus usuários. Com estes dados, os serviços conseguem conhecer mais os usuários e recomendá-los conteúdos baseados em seus históricos de reprodução. Estes algoritmos que consistem em uma série de técnicas que trabalham através do uso de dados de usuários para recomendação de itens, são os chamados Sistemas de Recomendação (RICCI *et al.*, 2011).

Nos serviços de *streaming* de música, os sistemas de recomendação têm como objetivo, ajudar usuários a filtrar e descobrir novas músicas de acordo com seus gostos. Um bom sistema de recomendação de músicas deve ser capaz de detectar preferências e gerar *playlists* (listas de reprodução de músicas) automaticamente. O desenvolvimento de sistemas de recomendação de músicas fornece uma grande oportunidade para a indústria em agregar usuários que possuem interesse em músicas. E mais importante, levanta desafios para pesquisadores no âmbito de entenderem melhor e modelarem as preferências de usuários em músicas.

Atualmente sistemas de recomendação de músicas conseguem bons desempenhos utilizando filtragem colaborativa baseada nos comportamentos de reprodução dos usuários e avaliações (SONG *et al.*, 2012). No entanto música é subjetiva e universal, podendo não só transmitir emoções, mas também pode modular o humor de um ouvinte. É com este aspecto que as *playlists* tem um importante papel, pois em sua maioria elas são compostas de músicas que

[1] http://www.spotify.com
[2] http://www.deezer.com
[3] http://play.google.com/music

possuem alguma similaridade em comum, como por exemplo, *playlists* com músicas sucessos nos anos 80, ou melhores músicas de 2018.

Este trabalho utiliza-se de técnicas de recomendação baseada em filtragem colaborativa utilizando *playlists* feitas por usuários do Spotify. O objetivo deste trabalho é, realizar a recomendação de músicas para preenchimento de *playlists* utilizando abordagens baseadas em filtragem colaborativa sobre dados de *playlists*, e comparar resultados e precisões dos métodos utilizados, visando analisar técnicas populares quanto ao seus desempenhos. Ao longo deste trabalho é explicado com mais detalhes os procedimentos tomados para se atingir o objetivo citado.

Este trabalho está estruturado da seguinte forma: o Capítulo 2 descreve os principais conceitos utilizados para realização deste trabalho; o Capítulo 3 apresenta trabalhos relacionados a abordagem que este trabalho toma; o Capítulo 4 descreve a metodologia utilizada para execução do trabalho; o Capítulo 5 descreve a execução dos passos apresentados na seção anterior; Por fim, o Capítulo 6 apresenta uma conclusões sobre o trabalho, seguida de alguns pontos a serem considerados para trabalhos futuros.

# 2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os principais conceitos utilizados para entendimento e execução deste trabalho.

## 2.1 Sistemas de Recomendação

Sistemas de Recomendação (SR) são ferramentas e técnicas de software que fornecem sugestões de itens a serem usados por usuários. Esta sugestão pode estar relacionada a diversos processos de decisão, como por exemplo, qual item comprar, qual música ouvir, ou qual livro ler (RICCI *et al.*, 2011).

Item é o termo geral usado para denotar o que o sistema recomenda para o usuário. Um SR normalmente foca em um tipo específico de item (i.e., filmes, ou músicas) em conformidade com seu *design*, sua interface gráfica, e o núcleo da técnica de recomendação usada para gerar as recomendações.

Em (HERLOCKER *et al.*, 2004) são definidas onze tarefas populares em que um SR pode ser útil, são elas:

- Achar algum bom item;
- Achar todos os bons itens;
- Verificação de contexto;
- Recomendar uma sequência de itens;
- Recomendar um grupo de itens;
- Ajudar na navegação do usuário;
- Fornecer recomendações confiáveis;
- Melhorar o perfil do usuário, através da análise de dados do usuário;
- Ajudar outros usuários através da análise de perfil de usuários mais experientes;
- Influenciar outros usuários.

Desse modo, sistemas de recomendação possuem diversas características, funções e diferentes fontes de dados. Vale adicionar que existem diferentes técnicas de recomendação, como, técnicas baseadas em conteúdo, demografia, filtros colaborativos, etc. Atualmente com os avanços na área de Inteligência Artificial e a necessidade de recomendar itens mais específicos para os usuários, os sistemas de recomendação vêm ganhando cada vez mais complexidade e demanda.

### *2.1.1 Dados e Fontes de Conhecimento*

SRs são sistemas de processamento de informação que ativamente utilizam diversos tipos de dados para construir suas recomendações. Dados são primordialmente sobre os itens a serem sugeridos e os usuários que irão receber estas recomendações. Mas, como a fonte de dados e conhecimento disponíveis podem ser diversos, de certa forma, se eles podem ser explorados ou não, dependendo da técnica de recomendação.

Em geral existem diversas técnicas de recomendação que são pobres de conhecimento, i.e., elas usam dados básicos e bastante simples, como por exemplo, notas e/ou avaliações de usuários para itens. Outras já possuem uma maior dependência do conhecimento, e.g., usando descrições ontológicas de usuários ou itens, ou restrições, ou relações sociais e atividades dos usuários. Em quaisquer dos casos, como uma classificação generalizada, dados usados por Sistemas de Recomendação referem-se à três tipos de objetos: Itens, usuários e transações, isto é, relações entre usuários e itens.

**Itens**: Itens são os objetos que são recomendados. Itens podem ser caracterizados pela sua complexidade, e seu valor ou utilidade. O valor de um item pode ser positivo ou negativo, dependendo do usuário e suas preferências.

SRs, de acordo com sua tecnologia essencial, pode usar uma série de propriedades e características de um item. Por exemplo, em um sistema de recomendação de filmes, o gênero, diretor e atores podem ser usados para descrever um filme, e aprender como a utilidade de um item depende de suas características. Itens podem ser representados usando várias informações e abordagens representativas, e.g., de uma maneira minimalista um simples código de identificação (id), ou de uma maneira mais rica, como um conjunto de atributos, ou até mesmo como um conceito em uma representação ontológica de um domínio.

**Usuários**: Usuários de um SR, como mencionado anteriormente, podem ter diferentes objetivos e características. Para personalizar as recomendações e a IHC, SRs exploram uma série de informações a respeito dos usuários. Estas informações podem ser estruturadas de diversas formas e novamente a seleção da informação depende da técnica de recomendação. Por exemplo, numa filtragem colaborativa, usuários são modelados como uma simples lista contendo notas providas pelos usuários para alguns itens. Em um SR demográfico, atributos sócio-demográficos como idade, gênero, profissão e educação são usados. Dados de usuários são usados para constituir um modelo de usuários.

Várias abordagens na modelagem dos usuários vêm sendo utilizadas. De certo modo,

um SR pode ser visto como uma ferramenta de geração de recomendações, ao construir e explicar modelos de usuários (RICCI *et al.*, 2011).

Como nenhuma personalização é possível sem um modelo de usuário conveniente, a menos que a recomendação seja personalizada no ato, como na seleção dos 10 melhores itens, o modelo do usuário sempre terá um papel central. Por exemplo, considerando novamente uma abordagem baseada em filtragem colaborativa, o usuário pode ser representado por um vetor de valores, onde estes valores são suas notas dadas aos itens.

Usuários podem também ser descritos pelos seus dados de padrões comportamentais, por exemplo, padrões de navegação (em um sistema de recomendação baseada em web), ou padrões de pesquisa de viagens. Além disso, dados de usuários podem incluir relações entre usuários, como em um nível de similaridade.

**Transações**: Geralmente uma transação refere-se a uma interação entre o usuários e o sistemas de recomendação. Transações são dados parecidos com registros *log*, que armazenam informações geradas durante a interação humano-computador, e que são úteis para o algoritmo de geração de recomendação que o sistema usa. Por exemplo, um *log* de transação pode conter a referência ao item selecionado pelo usuário e uma descrição do contexto (o objetivo/consulta do usuário) para aquela recomendação em particular. Se disponível, aquela transação pode também incluir um *feedback* explícito que o usuário proveu, como uma avaliação ao item selecionado. Na verdade, avaliações são a forma mais popular de transação que um sistema de recomendação coleta.

Avaliações podem ser coletadas implícita ou explicitamente. Em uma coleta explícita de avaliações, o usuário é solicitado a prover sua opinião sobre um item ou uma escala de avaliação. De acordo com (SCHAFER *et al.*, 2007), avaliações podem ser feitas de várias formas:

- Numérica como 1-5 estrelas;
- Ordinária, como “concordo plenamente, concordo, neutro”, etc.;
- Binária, onde o usuário é simplesmente perguntado se um item é bom ou ruim;
- Unário, que pode indicar que o usuário observou ou comprou um item, ou avaliou um item positivamente;

### *2.1.2 Técnicas de Recomendação*

Com o objetivo de implementar sua função principal, ou seja, identificar os itens úteis para o usuário, um sistema de recomendação deve predizer se um item vale a pena ser recomendado. Para fazer isto, o sistema deve ser capaz de predizer a utilidade de alguns dos itens, ou ao menos comparar a utilidade dos itens, e então decidir qual item recomendar baseado nessa comparação. A etapa de predição pode não ser explícita em um algoritmo de recomendação, mas pode-se ainda aplicar este modelo uniforme para descrever o papel genérico de um sistema de recomendação.

Para ilustrar a etapa de predição de um SR, pode-se considerar, por exemplo, um simples algoritmo de recomendação não personalizado que recomenda a música mais popular. A razão para usar esta abordagem é que na ausência de uma informação mais precisa sobre a preferência de um usuário, uma música popular, isto é, algo que tem preferência (alta utilidade) de muitos usuários, há uma chance maior de ser preferido por um usuário genérico, ao menos mais que uma música selecionada randomicamente.

Como mencionado anteriormente, alguns sistemas não estimam totalmente a relevância de um item antes de fazer a recomendação, mas eles podem aplicar algumas heurísticas para supor que um item é relevante para um usuário. Isto é típico, por exemplo, em sistemas baseados em conhecimento. Estas predições de relevância são computadas com algoritmos específicos e usam vários tipos de conhecimentos sobre usuários, itens e a função de predição de relevância. Por exemplo, um sistema pode assumir que uma função de relevância é booleana e, deste modo, irá determinar se um item é relevante ou não para o usuário. Consequentemente, assumindo que existe algum conhecimento disponível (possivelmente nenhum) sobre o usuário que está solicitando a recomendação, conhecimento sobre itens, e outros usuários que receberam recomendações, o sistema irá utilizar este conhecimento com um algoritmo apropriado para gerar várias predições de relevância e, consequentemente, recomendações. (BURKE, 2007)

Também é importante ressaltar que algumas vezes a relevância de um item pode depender de outras variáveis, nas quais é chamada geralmente de “contextual” (ADOMAVICIUS *et al.*, 2005). Por exemplo, a relevância de um item para um usuário pode ser influenciada de acordo com o domínio do usuário sobre determinado item ( usuários iniciantes vs. usuários experientes), ou pode depender do momento em que a recomendação é requisitada. Ou o usuário pode estar mais interessado em itens próximos de sua atual localização. Consequentemente, as recomendações devem se adaptar a estes detalhes específicos e torna-se cada vez mais difícil de

estimar quais recomendações são relevantes.

### *2.1.3 Tipos de Sistemas de Recomendação*

Os tipos de sistemas de recomendação podem variar de acordo com o domínio no qual ele é inserido, o conhecimento usado, mas especialmente em relação ao algoritmo de recomendação, i.e., como é feita a recomendação. Outras diferenças referem-se a como as recomendações são montadas e apresentadas ao usuário. (BURKE, 2007) cita algumas taxonomias que vêm se mostrando uma nomenclatura clássica para diferenciar sistemas de recomendação. Elas são definidas a seguir:

**Baseada em conteúdo**: O sistema aprende a recomendar itens que são similares aos que o usuário gostou no passado. A similaridade dos itens é calculada baseada nas características associadas aos itens comparados.

**Filtragem Colaborativa**: Segundo (SCHAFER *et al.*, 2007) a implementação mais simples e original desta abordagem recomenda ao usuário os itens que outros usuários com gostos similares preferiram. Esta é a razão pela qual (SCHAFER *et al.*, 2001) refere-se a filtragem colaborativa como "correlação pessoa-para-pessoa". Filtragem colaborativa é considerada a mais popular e amplamente implementada técnica em sistemas de recomendação.

**Demográfica**: este tipo de sistema recomenda itens baseado no perfil demográfico do usuário. A suposição é que diferentes recomendações deveriam ser geradas para diferentes nichos demográficos. Muitos sites adotam simples e efetivas soluções personalizadas baseadas em demografia.

**Baseada em conhecimento**: estes sistemas recomendam itens baseados em domínios específicos de conhecimento sobre como certas características de um item atendem as necessidades e preferências dos usuários, e sobretudo, como o item é útil para o usuário. Nestes sistemas, uma função de similaridade estima o quão as necessidades de um usuário correspondem às recomendações. Neste tipo de sistema, o valor da similaridade pode ser interpretado diretamente como a relevância da recomendação para o usuário.

**Baseado em comunidade**: este tipo de sistema recomenda itens baseado nas preferências dos amigos do usuários. Esta técnica segue o epigrama "Diga me com quem tu andas, e eu te direi quem és" (ARAZY *et al.*, 2009). Evidências sugerem que pessoas tendem a apoiar-se mais em recomendações de amigos que em recomendações de indivíduos anônimos (SINHA *et al.*, 2001). Esta observação, combinada com o aumento da popularidade de redes sociais abertas,

tem gerado um crescente interesse em sistemas baseados em comunidade, também chamados de sistemas de recomendação social.

## 2.2 Medidas de Similaridade

Sistemas de recomendações possuem diversas medidas de similaridade usadas na área de aprendizagem de máquina. É possível utilizar diferentes técnicas de similaridade para computar similaridade entre usuários. Como cada similaridade possui diferentes fórmulas, elas retornam diferentes resultados. As similaridades utilizadas neste trabalho são explicadas nas subseções seguintes.

### *2.2.1 Similaridade Cosseno*

Quando é possível representar a relação entre dois usuários como vetores, a similaridade dos dois vetores corresponde a correlação entre os dois. Isto é quantificado como o cosseno do ângulo entre os vetores, a chamada similaridade cosseno. A similaridade cosseno é uma das similaridades mais utilizadas em inúmeras aplicações de recuperação de informação e *clustering* (RICCI *et al.*, 2011).

Dados dois vetores $\overrightarrow{t_a}$ e $\overrightarrow{t_b}$, sua similaridade cosseno é dada por :

$$sim = \frac{\overrightarrow{t_a} \cdot \overrightarrow{t_b}}{||\overrightarrow{t_a}||||\overrightarrow{t_b}||}$$

Onde $\overrightarrow{t_a}$ e $\overrightarrow{t_b}$ são vetores de $m$ dimensões sobre o conjunto $T = (t_1, t_2, \ldots, t_m)$, onde $(t_1, t_2, \ldots, t_m)$ são elementos de $T$. Cada dimensão representa um termo com seu peso no modelo, que é não-negativo, para este tipo de problema apresentado no trabalho. Como resultado, a similaridade cosseno é não negativa e entre 0 e 1.

### *2.2.2 Coeficiente Jaccard*

O coeficiente Jaccard, algumas vezes referido como o coeficiente Tanimoto, mede a similaridade como a cardinalidade da interseção, dividida pela cardinalidade da união dos objetos. No caso de duas *playlists*, o coeficiente Jaccard compara a soma das músicas em comum sobre a soma das músicas (não repetidas) nas duas *playlists*. A definição formal do coeficiente Jaccard é (RICCI *et al.*, 2011):

$$sim = \frac{|A \cap B|}{|A \cup B|}$$

Onde A e B são conjuntos são dois conjuntos.

O coeficiente Jaccard é uma medida de similaridade que varia entre 0 e 1. É 1 quando $A = B$ e 0 quando $A$ e $B$ são totalmente distintos, onde 1 significa que os dois objetos possuem os mesmos elementos e 0 significa que eles são completamente distintos quanto aos seus elementos.

## 2.3 Métrica de Avaliação

Neste trabalho, as recomendações são avaliadas para obter um indicador de performance das recomendações, bem como sua precisão. Para tal foi utilizada a métrica de Precisão em R. O problema deste trabalho possibilitou o uso desta métrica, pois o valor da relevância dos itens recomendados é binária (relevante ou não relevante). Esta métrica é utilizada não só no âmbito dos sistemas de recomendação, mas também nos sistemas de recuperação de informação (LARSON, 2010). A métrica de avaliação é explicada a seguir.

### *2.3.1 Precisão em R*

A métrica de Precisão em R leva este nome por ser o cálculo da precisão em um corte R dos dados, onde R são os itens relevantes a serem preenchidos em uma consulta, e o corte R um subconjunto dos dados contendo apenas os itens relevantes.

Para cálculo da Precisão em R é necessário saber quais itens são relevantes para determinada recomendação (R). O número de itens relevantes é usado como o corte para cálculo desta precisão, que varia de consulta e consulta. No caso deste trabalho, é possível realizar este cálculo, pois se sabe o número de músicas retiradas de uma *playlist*. Considerando o escopo deste trabalho, podemos considerar, $P$ uma *playlist* incompleta, onde $N$ é o número de músicas que faltam nesta *playlist*, ou seja, $N$ é o número de documentos relevantes que pretende-se acertar durante o processo de recomendação. Supondo que é realizada uma recomendação onde foram recomendados $K$ itens onde, $K$ é igual a $N$, e supondo e que um número $r$ de itens de $K$ são relevantes, é possível calcular a Precisão em $R$ utilizando.

$$Pr = \frac{r}{N}$$

Onde $N$ é o número de itens relevantes e, consequentemente, o corte R da precisão. Diante do problema apresentado neste trabalho, sendo ele uma classificação binária e o número de documentos a serem recomendados é conhecido, esta métrica tem uma boa aplicação e dispensa o cálculo de precisão e *recall*, pois na Precisão em $R$ a precisão e a *recall* são iguais.

Além disto esta também é a técnica utilizada como um dos critérios do Ranking da competição *RecSys Challenge* 2018 [1]

[1] https://recsys-challenge.spotify.com/rules

# 3 TRABALHOS RELACIONADOS

Este capítulo aborda trabalhos que se relacionam com este trabalho no âmbito de recomendação de músicas.

## 3.1 *Efficient Top-N Recommendation for Very Large Scale Binary Rated Datasets*

(AIOLLI, 2013) apresenta um algoritmo escalável para top-n recomendações para com o objetivo de com grandes conjuntos de dados e avaliação binária. Os autores focam em algoritmos de filtragem colaborativa baseada em memória, similar à técnica baseada em vizinho para *feedback* explícito (RICCI *et al.*, 2011). A maior diferença que torna o algoritmo proposto por (AIOLLI, 2013) escalável é que usa somente o *feedback* positivo, e a construção da matriz de similaridade não é necessária.

O estudo deste algoritmo foi conduzido utilizando os dados do desafio *Million Songs Dataset* (MSD), no qual a tarefa era sugerir uma lista de músicas (de mais de 380 mil músicas) para mais de 100 mil usuários, dado metade do histórico de músicas dos usuários. Como medida de similaridade, os autores utilizaram cosseno, para medida de correlação entre os vetores, porém os autores perceberam a necessidade da utilização de uma interpretação probabilística para os vetores de valores binários. Por este motivo, eles trabalharam em uma similaridade cosseno assimétrica.

Os autores então conseguem recomendar itens baseados na técnica de similaridade cosseno assimétrica combinando os dados entre usuários e itens. Como métrica de avaliação os autores utilizaram a precisão média, que consiste na média das precisões máximas em diferentes valores de *recall*.

## 3.2 *Spotify me: Facebook-assisted automatic playlist generation*

(GERMAIN; CHAKARESKI, 2013) propõem um método para gerar uma *playlist* de músicas recomendadas utilizando a aplicação de compartilhamento de música *Spotify* nas redes sociais, que os usuário têm alta probabilidade de gostarem. O método utiliza-se de múltiplos artistas sementes (artistas mais populares) como entrada, obtidos através de curtidas do artista no *Facebook* e o histórico de músicas de um usuário do *Spotify*.

Primeiramente (GERMAIN; CHAKARESKI, 2013) , construiu-se um vetor de entrada contendo todos os artistas que o usuário curtiu no *Facebook* e que escuta no *Spotify*.

Então, é realizada uma busca por outros artistas e bandas relacionadas aos artistas no vetor utilizando *EchoNest*[1], uma plataforma online de aprendizagem de máquina. É então assinalada uma pontuação para cada artista na coleção obtida, baseada na frequência de aparições.

Por fim, os autores constroem uma *playlist* contendo músicas populares associadas aos artistas com maior frequência de aparições. A recomendação do algoritmo utiliza a WTF *score* (consiste no cálculo da fração de músicas não curtidas sobre as músicas curtidas) nas *playlists* geradas por diferentes tamanhos de vetores sementes.

Os trabalhos apresentados anteriormente se relacionam com este trabalho quanto ao seu contexto, recomendação de músicas e geração de *playlists*, abordagem de recomendação utilizando filtragem colaborativa e em suas técnicas de recomendação. No entanto, este trabalho se diferencia em trabalhar na recomendação de músicas para preenchimento de *playlists* utilizando técnicas de recomendação e, em seguida uma comparação entre os desempenhos das técnicas utilizando Precisão em R.

Quadro 1 – Quadro comparativo de trabalhos relacionados

| Trabalhos | Técnica | Métrica Avaliativa | Contexto |
|---|---|---|---|
| (AIOLLI, 2013) | Similaridade Cosseno | Precisão Meia | Recomendação de Músicas |
| (GERMAIN et al., 2013) | Popularidade em Redes Sociais | WTF Score | Geração de Playlists |
| Este trabalho | Popularidade, Cosseno, Jaccard | Precisão em R | Preenchimento de Playlists |

Fonte: elaborado pelo autor.

[1] http://the.echonest.com/

# 4 METODOLOGIA

Esta seção tem o objetivo de expor a metodologia utilizada para realização deste trabalho.

## 4.1 Obtenção dos dados

Na realização deste trabalho utilizou-se os dados do *Million Playlists Dataset* disponibilizado pelo Spotify para a competição *RecSys Challenge* 2018. O arquivo contém um milhão de *playlists* criadas por usuários do Spotify.

## 4.2 Pré-processamento dos dados

Nesta etapa, a separação dos dados em treino e teste é realizada, com o intuito de trabalhar com as recomendações. Os dados são separados em dados de treino e teste, onde o treino consiste em 70% da quantidade total dos dados. Os dados de teste são compostos pelos 30% restantes dos dados do MPD, e em cada *playlist* do conjunto de teste, são removidas aleatoriamente 30% das músicas de cada *playlist*, além disso, uma cópia das *playlists* completas foram mantidas, para fins comparativos.

Após a divisão é necessária uma nova etapa de tratamento dos dados, onde para cada *playlist* apenas as músicas devem ser consideradas, sendo necessário apenas os identificadores únicos e o nome de cada música.

## 4.3 Cálculo de Popularidade e Similaridade

Para que a recomendação seja possível, é necessário realizar os cálculos das bases recomendativas. Neste trabalho é utilizada a recomendação baseada em popularidade e similaridade.

### *4.3.1 Popularidade*

Criação de um *ranking* com as músicas mais populares dos dados de treino, ordenadas pela sua popularidade (quantidade de vezes que uma música está presente nas *playlists* dos dados de treino).

### *4.3.2 Similaridade*

Em seguida, o cálculo das duas medidas usadas como similaridade sendo elas, similaridade cosseno e coeficiente Jaccard. O cálculo se dá utilizando duas *playlists*, uma de teste e incompleta, e outra de treino. Com as duas *playlists* é então calculada a similaridade entre as duas, e armazenado o valor. Este procedimento é repetido até que todas as *playlists* de treino e teste sejam processadas. No fim, cada *playlist* de teste incompleta possui um ranking das cinco *playlists* dos dados de treino com maior similaridade a elas.

## 4.4 Algoritmo de Recomendação

Nesta etapa, os *rankings* de popularidade e similaridade são utilizados para realização das recomendações.

### *4.4.1 Por Popularidade*

Nesta abordagem, as *playlists* incompletas dos dados de teste são preenchidas com as músicas mais populares, onde não devem ser incluídas músicas que já estão presentes na *playlist* a ser preenchida.

### *4.4.2 Por Similaridade*

Nesta abordagem as *playlists* incompletas são preenchidas com as músicas das *playlists* que atingiram maior valor em cada similaridade. De acordo com a posição da música nas *playlists*, não devem ser incluídas músicas que já estão presentes na *playlist* a ser preenchida.

## 4.5 Avaliação

Nesta etapa é realizado o cálculo da precisão em R de cada abordagem recomendativa com o intuito de fazer um comparativo entre os algoritmos de recomendação.

# 5 EXPERIMENTOS E RESULTADOS

Nesta capítulo, é descrito com detalhes o processo de execução do trabalho, desde a obtenção, detalhamento e tratamento dos dados, até os métodos de recomendação e suas avaliações de desempenho. A execução destas etapas foram realizadas em máquina com sistema operacional Linux, mais precisamente executando uma distribuição Ubuntu 16.04. Além disso, do ponto de vista de hardware, a máquina possui 32 Gigabytes de Memória RAM, processador Xeon de 4 núcleos e 3.0 Gigahertz.

## 5.1 Obtenção dos Dados

Neste trabalho, utilizou-se os dados do *RecSys Challenge* 2018[1], para obtenção dos dados foi necessário o cadastramento com um e-mail institucional para comprovar fins científicos. O conjunto de dados conhecido como *Million Playlist Dataset* (MPD) foi disponibilizado pela equipe do Spotify para a realização do desafio.

Para uma melhor diversidade dos dados, o *Million Playlist Dataset* (MPD) é composto de *playlists* que possuem ao menos 3 artistas únicos e 2 álbuns únicos para evitar *playlists* com apenas um artista ou álbum. Também são incluídas apenas *playlists* com ao menos 5 músicas e não mais de 250 músicas.

### *5.1.1 Million Playlist Dataset*

O MPD é filtrado das bilhões de *playlists* que os usuários do Spotify criaram durante os últimos 10 anos. As *playlists* foram randomicamente selecionadas de acordo com os seguintes critérios:

- Criadas por usuários que residem nos Estados Unidos e possuem ao menos 13 anos de idade;
- Eram públicas no momento em que o MPD foi gerado;
- Possuem ao menos 5 músicas;
- Possuem não mais que 250 músicas;
- Possuem ao menos 3 artistas únicos;
- Possuem ao menos 2 álbuns únicos;
- Não possuem músicas locais (músicas locais são músicas que não estão no

[1] http://https://recsys-challenge.spotify.com/

Spotify que um usuário possui em seu dispositivo);

- Possuem ao menos um seguidor;
- Foram criadas depois de 1 de Janeiro de 2010 e antes de 1 de Dezembro de 2017;

### *5.1.2 Dados Demográficos do MPD*

Após a geração dos dados, a equipe do Spotify conseguiu apurar e disponibilizar alguns dados demográficos a respeito do usuários que contribuíram para a construção do MPD, são eles:

Gênero:

- 45% Masculino;
- 54% Feminino;
- 0.5% Não especificado;
- 0.5% Não binário;

Idade:

- 43% entre 18-24;
- 31% entre 25-34;
- 9% entre 35-44;
- 4% entre 45-54;
- 3% entre 55 ou mais;
- 10% Outros;

País:

- Estados Unidos 100%;

### *5.1.3 Formato*

O MPD consiste em mil arquivos de formato JSON contendo mil *playlists* cada arquivo. Estes arquivos possuem uma convenção de nomenclatura como:

`mpd.slice.INICIO_PLAYLIST_ID_-_FINAL_PLAYLIST_ID.json`

Onde a primeira parte representa os dados, neste caso o mpd, seguido da palavra *slice*, e os intervalos de *playlists* contidas no arquivo.

### *5.1.4 Campos*

Cada arquivo consiste em um dicionário com dois campos, `info` e `playlists`.

#### *5.1.4.1 Campo `info`*

O campo `info` é um dicionário contendo informações gerais sobre aquele determinado arquivo. As informações usadas são descritas a seguir.

- `slice`: o intervalo de *playlists* que estão no arquivo.
- `version`: a versão do MPD (versão 1)
- `generated_on`: *timestamp* indicando quando aquele arquivo foi gerado

#### *5.1.4.2 Campo `playlists`*

Consiste em um *array* contendo 1.000 *playlists*. Cada *playlist* consiste em um dicionário contendo os seguintes campos:

- `pid` : identificador da *playlist* um número inteiro entre 0 e 999.999
- `name` : o nome da *playlist*
- `description` : contém a descrição de uma *playlist* informada pelo usuário
- `modified_at` : *timestamp* que representa a última vez que a *playlist* foi modificada
- `num_artists` : total de artistas únicos na *playlist*
- `num_albums` : total de álbuns únicos na *playlist*
- `num_tracks` : número de músicas na *playlist*
- `num_followers` : número de seguidores da *playlist*
- `num_edits` : número de edições da *playlist*
- `duration_ms` : duração total da *playlist* em milissegundos
- `collaborative` : se verdadeiro a *playlist* é colaborativa, isto é, múltiplos usuários podem contribuir adicionando músicas à *playlist*
- `tracks` : um *array* de informações sobre cada música na *playlist*. Cada elemento do *array* consiste em um dicionário com os seguintes campos:
  - `track_name` : nome da música
  - `track_uri` : identificador único da música
  - `album_name` : nome do álbum
  - `album_uri` : identificador único do álbum

  - `artist_name` : nome do artista
  - `artist_uri` : identificador único do artista
  - `duration_ms` : duração da música em milissegundos
  - `pos` : posição da música na *playlist*

## 5.2 Pré-processamento dos dados

Para a realização do trabalho, realizou-se um tratamento dos dados para ajudar na execução dos algoritmos e atender os objetivos deste trabalho.

O primeiro passo se deu ao separar os dados em dados de treino e teste. Os dados de treino consistem em 70% dos dados do MPD, ou seja, 700 mil *playlists* selecionadas aleatoriamente. A geração dos dados de testes consiste em duas etapas, sendo elas, arquivamento dos 30% restantes dos dados (300 mil *playlists*). Depois disso, foi feita uma cópia destes dados arquivados e em cada *playlist* do conjunto de dados, foram removidas aleatoriamente 30% das músicas do *array* de *tracks* de cada *playlists*.

Além disso, no escopo deste trabalho utilizou-se apenas as músicas (`track_uri`, `track_name` do dicionário `tracks`) de cada *playlist*.

Ao fim deste procedimento, os dados quanto a seus conjuntos ficaram como demonstrado a seguir:

- **Treino**: Composto de 700 mil *playlists* dos dados do MPD;
- **Teste**: Composto de 300 mil *playlists* incompletas;
- **Teste Original**: Composto de 300 mil *playlists* dos dados de teste completas.

Quanto aos campos as *playlists* se encontraram desta forma:

- `pid`
- `name`
- `num_tracks`
- `tracks`
  - `track_name`
  - `track_uri`
  - `pos`

## 5.3 Cálculo de Popularidade e Similaridade

Utilizando conceitos de filtragem colaborativa, utilizou-se analogamente o modelo de uma *playlist* como um usuário e os itens as músicas que estão inseridas nesta *playlist*, por exemplo, uma dada *plalylist P* que contém um *array* de músicas *M*, pode ser vista como um usuário *U* que classificou um número *N* de itens, como adequados às suas preferências, onde *N* é o número de músicas de *M*. Nesta seção, é descrita a forma na qual as popularidades e similaridades foram calculadas neste contexto.

### *5.3.1 Popularidade*

O cálculo de popularidade se deu através da contagem de músicas nas *playlists* dos dados de treino como um todo, ou seja, foi calculado a frequência em que cada música do conjunto de dados de treino se repetia, gerando no final um *ranking* de músicas ordenado por popularidade.

### *5.3.2 Similaridade*

Para similaridade foram realizados dois cálculos de similaridade para fins comparativos: similaridade cosseno e coeficiente de Jaccard.

#### *5.3.2.1 Similaridade cosseno*

No cálculo da similaridade cosseno, a função recebe duas *playlists* e retorna a similaridade cosseno entre 0 e 1. Considerando a $p_1$ e $p_2$ como duas *playlists* distintas, se a união dos dois *arrays* de músicas de $p_1$ e $p_2$ sem repetições resulta em um *array* de músicas *M* que possui todas as músicas de $p_1$ e $p_2$. Em seguida são gerados dois *arrays* binários que representam $p_1$ e $p_2$ com o mesmo tamanho de *M*, e os *arrays* são preenchidos com 0 e 1. Seguindo a mesma ordem de *M* preenche-se com 1, se a *playlist* possui a música de *M* e 0, se a *playlist* não possuir a música. Ao fim, são utilizados os dois novos *arrays* para calcular a similaridade cosseno entre os dois.

#### *5.3.2.2 Coeficiente Jaccard*

No cálculo do coeficiente Jaccard, é possível considerar duas *playlists* $p_1$ e $p_2$ e, então, utilizando o *array* de músicas de cada *playlist*, calcular a interseção $I$ de $p_1$ e $p_2$ (músicas que são comum entre os conjuntos) e a união $U$ de $p_1$ e $p_2$ (total de músicas de $p_1$ e $p_2$ sem repetições), em seguida calcular o coeficiente Jaccard $C_j$ através da divisão $\frac{I}{U}$.

## 5.4 Algoritmo de Recomendação

Nesta etapa, são realizadas as recomendações. Para tal utilizou-se de duas abordagens: recomendação por popularidade e recomendação por similaridade, que são abordadas a seguir.

### *5.4.1 Popularidade*

Na abordagem utilizando popularidade, as *playlists* dos dados de treino foram preenchidas com as músicas mais bem colocadas no *ranking* de popularidade. Estas músicas, no entanto, só eram adicionadas se as mesmas não fizessem parte da *playlist* a ser preenchidas. Ao final do processo as *playlists* foram preenchidas e armazenadas em um diretório separado com a identificação de que foram preenchidas utilizando popularidade.

### *5.4.2 Similaridade*

Nesta abordagem, foi realizado o preenchimento das *playlists* de treino, utilizando os *rankings* de similaridade gerados na etapa de cálculo de similaridades. O processo consiste em uma *playlist* incompleta dos dados de treino ser preenchida com a *playlist* mais similar, de acordo com o *ranking* de similaridade e o tipo de similaridade, cosseno e Jaccard. O critério de preenchimento foi o da música mais acima na ordem da *playlist* mais similar e não repetida.

## 5.5 Avaliação e Resultados

Nesta etapa foram utilizadas as *playlists* preenchidas baseadas em popularidade, similaridade cosseno e coeficiente Jaccard, e calculadas a métrica de Precisão em R para cada abordagem. Para cada precisão foram utilizadas as *playlists* completas dos dados de treino para medir acertos.

### *5.5.1 Precisão em R*

O cálculo da precisão em R se deu através da seguinte fórmula:

$$Pr = \frac{|Mr|}{|Me|}$$

Onde $Mr$ são as músicas acertadas e $Me$ as músicas que deveriam ter sido recomendadas.

Em ambos os casos foram realizadas as médias das precisões, tendo em vista que foi realizado um cálculo de precisão para cada *playlist* dos dados de treino.

### *5.5.2 Resultados*

Após a realização das etapas a anteriores, obteve-se os seguintes resultados.

Quadro 2 – Quadro comparativo de melhor caso de preenchimento segundo suas Precisões em R

| | Popularidade | Similaridade Cosseno | Jaccard |
|---|---|---|---|
| Precisão em R | 0.033 | 0.32 | 0.38 |

Fonte: elaborado pelo autor.

Quando foi executado o preenchimento utilizando popularidade, foi possível notar que várias *playlists* eram compostas de músicas populares, porém muitas *playlists* criadas eram de conteúdo específico, por exemplo, uma *playlist* contendo músicas infantis foram preenchidas com músicas populares, independente do gênero, ocorrendo assim uma distorção completa do contexto da *playlist*.

Como apresentado no Quadro 2, analisando o desempenho da similaridade Cosseno, pode-se notar uma melhor precisão R quando comparada com a popularidade, porém, como foram utilizadas apenas as faixas, não considerando álbum e artista, a similaridade cosseno não conseguiu atingir o melhor desempenho. Apesar disso, a recomendação usando similaridade cosseno conseguiu um desempenho satisfatório, quando comparado com as duas abordagens.

A recomendação utilizando coeficiente Jaccard foi a que obteve o melhor resultado. Isto se deu, pois as operações foram realizadas como conjuntos, e em alguns casos foi possível determinar *playlists* muito similares, apesar de ter obtido casos onde todas as músicas recomendadas foram acertadas. A precisão R utilizando Jaccard e a da similaridade Cosseno obtiveram valores bem próximos, como apresentado no Quadro 2.

# 6 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, foi descrito um processo de criação de um sistema de recomendação de músicas para preenchimento de *playlists*. O sistema de recomendação aqui descrito é baseado em filtragem colaborativa, isto é, utiliza-se de dados de outras *playlists* para realizar a recomendação. Para geração de recomendações, foram utilizadas duas abordagens, a primeira utilizando popularidade onde as *playlists* a serem preenchidas foram completadas com as músicas que possuíam maior frequência de aparições em outras *playlists*. A segunda abordagem foi utilizando medidas de similaridades, sendo elas cosseno e Jaccard, em seguida as *playlists* eram preenchidas com as músicas não repetidas e mais no topo da *playlist* mais similar.

Nos resultados dos experimentos a recomendação utilizando similaridade Jaccard foi a que obteve melhor performance, devido as operações análogas a operações de conjunto, onde as *playlists* eram os conjuntos e suas músicas eram seus elementos. A similaridade cosseno atingiu resultados satisfatórios quando comparada com as demais abordagens. E, por fim, a recomendação baseada em popularidade das músicas foi a que se demonstrou com pior performance apesar de ter havido casos onde grande quantidade de músicas recomendadas foram certas, porém muito raros.

Como sugestão de trabalhos futuros pode-se considerar como principal sugestão a de utilizar mais campos como critério de relevância. Neste trabalho, utilizou-se apenas um critério de relevância, que foi o acerto quanto a música. Para trabalhos futuros seria interessante o uso de outras características como álbum e artistas nos critérios de relevância, podendo assim a similaridade cosseno apresentar uma performance melhor. Além disso, para melhor desempenho a clusterização dos dados seria de extrema ajuda para se ter uma maior rapidez nos cálculos de similaridade, pois com os dados clusterizados de acordo com suas similaridades, seria possível montar os *rankings* com maior velocidade, tendo em vista que os dados estariam já agrupados.

# REFERÊNCIAS

ADOMAVICIUS, G.; SANKARANARAYANAN, R.; SEN, S.; TUZHILIN, A. Incorporating contextual information in recommender systems using a multidimensional approach. **ACM Transactions on Information Systems (TOIS)**, ACM, v. 23, n. 1, p. 103–145, 2005.

AIOLLI, F. Efficient top-n recommendation for very large scale binary rated datasets. In: ACM. **Proceedings of the 7th ACM conference on Recommender systems**. [S.l.], 2013. p. 273–280.

ARAZY, O.; KUMAR, N.; SHAPIRA, B. Improving social recommender systems. **IT professional**, IEEE, v. 11, n. 4, 2009.

BURKE, R. Hybrid web recommender systems. In: **The adaptive web**. [S.l.]: Springer, 2007. p. 377–408.

GERMAIN, A.; CHAKARESKI, J. Spotify me: Facebook-assisted automatic playlist generation. In: IEEE. **Multimedia Signal Processing (MMSP), 2013 IEEE 15th International Workshop on**. [S.l.], 2013. p. 025–028.

HERLOCKER, J. L.; KONSTAN, J. A.; TERVEEN, L. G.; RIEDL, J. T. Evaluating collaborative filtering recommender systems. **ACM Transactions on Information Systems (TOIS)**, ACM, v. 22, n. 1, p. 5–53, 2004.

LARSON, R. R. Introduction to information retrieval. **Journal of the American Society for Information Science and Technology**, Wiley Online Library, v. 61, n. 4, p. 852–853, 2010.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: **Recommender systems handbook**. [S.l.]: Springer, 2011. p. 1–35.

SCHAFER, J. B.; FRANKOWSKI, D.; HERLOCKER, J.; SEN, S. Collaborative filtering recommender systems. In: **The adaptive web**. [S.l.]: Springer, 2007. p. 291–324.

SCHAFER, J. B.; KONSTAN, J. A.; RIEDL, J. E-commerce recommendation applications. **Data mining and knowledge discovery**, Springer, v. 5, n. 1-2, p. 115–153, 2001.

SCHÄFER, T.; SEDLMEIER, P.; STÄDTLER, C.; HURON, D. The psychological functions of music listening. **Frontiers in psychology**, Frontiers, v. 4, p. 511, 2013.

SINHA, R. R.; SWEARINGEN, K. *et al.* Comparing recommendations made by online systems and friends. In: **DELOS workshop: personalisation and recommender systems in digital libraries**. [S.l.: s.n.], 2001. v. 106.

SONG, Y.; DIXON, S.; PEARCE, M. A survey of music recommendation systems and future perspectives. In: **9th International Symposium on Computer Music Modeling and Retrieval**. [S.l.: s.n.], 2012. v. 4.