INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SERGIPE - CAMPUS LAGARTO

SISTEMA DE RECOMENDAÇÃO HÍBRIDO REKOMENNDADOR HYDRA

Kaiki Mello dos Santos

Orientador: João Paulo Dias de Almeida
Coorientadora: Catuxe Varjão de Santana Oliveira

TRABALHO DE CONCLUSÃO DE CURSO

COORDENAÇÃO DO BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Lagarto - Sergipe

2023

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SERGIPE
COORDENAÇÃO DO BACHARELADO EM SISTEMAS DE INFORMAÇÃO

KAIKI MELLO DOS SANTOS

SISTEMA DE RECOMENDAÇÃO HIBRIDO REKOMENNDADOR HYDRA

Trabalho de Conclusão de Curso submetido à Coordenação de Bacharelado em Sistemas de Informação do Instituto Federal de Sergipe - Campus Lagarto como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador(a):João Paulo Dias de Almeida
Coorientador(a): Catuxe Varjão de Santana Oliveira

Lagarto - Sergipe
2023

AGRADECIMENTOS

Prezados,

Gostaria de expressar minha imensa gratidão a Deus e a Santo Expedito, que me guiaram e me abençoaram em cada etapa da minha jornada acadêmica. Sem a orientação e a força divinas, eu não teria chegado até aqui.

Agradeço aos meus pais, irmãos, sobrinhos e demais familiares por seu amor incondicional e por seu apoio durante toda a minha vida. Obrigado por me ensinar a perseverar e a nunca desistir dos meus sonhos. Eu amo demais vocês.

Ao meu orientador João Paulo e coorientadora Catuxe Varjão, minha gratidão por me ajudarem a recuperar a confiança que havia perdido há anos. Seu apoio, orientação e paciência foram fundamentais para o sucesso deste trabalho.

Aos meus amigos Vanilton, Rafaela, Rita e Josiane, que estiveram ao meu lado em todos os momentos, meu sincero agradecimento. Josiane, em especial, foi uma das mais queridas amigas que encontrei nesta jornada, se tornou uma irmã, e seu apoio e amizade foram essenciais para mim durante esta etapa da minha vida e carreira.

A Junior, aos meninos da QueroDelivery e a Alex do IFS, agradeço por suas ideias e contribuições valiosas para o meu trabalho. Agradeço também ao Instituto Federal de Sergipe por fornecer uma excelente plataforma para meus estudos.

Minha gratidão à banca de avaliação por seu tempo e dedicação em avaliar meu trabalho e fornecer seu feedback valioso. Agradeço também a Anne Vasconcelos por ter escutado meus problemas e por me ajudar a superar os desafios.

Por fim, quero agradecer a mim mesmo por perseverar e não desistir durante essa jornada. Houve momentos difíceis, mas com determinação e foco, consegui superá-los.

Agradeço a todos que cruzaram meu caminho nesta jornada, seja por um momento ou por um longo período, e que de alguma forma contribuíram para o meu crescimento pessoal e acadêmico. Espero poder retribuir de alguma forma no futuro.

Muito obrigado a todos!

RESUMO

A disponibilidade digital de filmes atualmente é abundante quando comparada ao passado. Os serviços de streaming estão em ascensão, permitindo que filmes possam ser acessados de diferentes dispositivos (e.g. tablet, computador, celular) em qualquer local, online ou offline. A grande quantidade de filmes disponíveis pode resultar em efeitos colaterais, tais como a dificuldade em classificar, buscar e organizar essa abundância de obras cinematográficas. Desta maneira, o desenvolvimento de um sistema de recomendação de filmes que possa sugerir filmes semelhantes ao perfil do usuário é altamente útil. Os sistemas de recomendação utilizam o histórico de preferências e perfil do usuário para prever e sugerir filmes adequados, considerando a similaridade de características entre os filmes. Assim, o objetivo deste trabalho é criar um sistema de recomendação de filmes que leve em consideração a similaridade dos metadados que descrevem as obras cinematográficas. Além disso, busca-se utilizar a interação do usuário com as obras para gerar recomendações personalizadas e adequadas a cada indivíduo.

Palavras-chaves: Sistemas de recomendação; K-NN; Filmes.

ABSTRACT

The digital availability of movies is currently abundant compared to the past. Streaming services are on the rise, enabling movies to be accessed from various devices (e.g. tablet, computer, phone) anywhere, online or offline. The large number of available movies can result in side effects, such as difficulty in classifying, searching, and organizing this abundance of cinematographic works. Therefore, the development of a movie recommendation system that can suggest similar movies to the user's profile is highly useful. Recommendation systems use the user's preference history and profile to predict and suggest suitable movies, considering the similarity of characteristics between the movies. Thus, the objective of this work is to create a movie recommendation system that takes into account the similarity of metadata describing the cinematographic works. Additionally, it seeks to use the user's interaction with the works to generate personalized and suitable recommendations for each individual.

Keywords: Recommender systems; K-NN; Films.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Os filmes são uma forma de arte que tem o poder de emocionar e inspirar pessoas de todas as idades e origens. Eles contam histórias que transportam o espectador para outros mundos e provocam reflexões sobre a vida, a sociedade e a humanidade. Alguns filmes se tornam clássicos e são lembrados por gerações, enquanto outros são esquecidos logo após o lançamento. A indústria cinematográfica é um ambiente altamente competitivo e exigente, onde o sucesso requer uma combinação de talento, criatividade e inovação. Nesse contexto, tanto a indústria do cinema quanto o mundo do entretenimento em geral precisam se reinventar continuamente para se destacarem. Com a crescente adoção de formatos digitais, os serviços de streaming de filmes e séries, como Prime Video, Netflix, Apple TV e outros, surgiram rapidamente, transformando a forma como o público consome conteúdo audiovisual

Estes vêm se destacando por motivarem cada vez mais ganhos no faturamento do setor cinematográfico. De acordo com relatório publicado em 2022 pela Motion Picture Association, Inc. (MPA), o comércio global de entretenimento teatral e doméstico, apresentou uma recuperação significativa, com um total de US$99,7 bilhões em gastos por parte dos consumidores. Esse valor supera os números registrados em 2019, indicando um forte impulso de retomada para o setor. Ademais, é importante destacar que o streaming tem sido um dos principais impulsionadores do crescimento da indústria do entretenimento. De fato, o número de assinaturas de serviços de streaming alcançou a marca de 1,3 bilhão em todo o mundo, o que representa um aumento de 14% em comparação a 2020.

A disponibilidade de filmes na era digital é abundante nos serviços de streaming. Nunca foi tão fácil assistir filmes e séries como nos dias atuais em virtude da facilidade de acesso desses serviços através dos diversos dispositivos (e.g. tablet, computador, celular), de qualquer local, online ou offline. A facilidade de acesso a todo esse conteúdo trouxe um novo problema para o usuário, visto que o mesmo, se encontra com dificuldades para decidir o que consumir. Além disso, a simplicidade de acesso e sobrecarga de informação dificulta a descoberta de novas obras (JORDÃO, 2016). Sendo assim, esse excesso de informação causa alguns efeitos colaterais, como dificuldade na classificação, busca e organização desse grande catálogo de obras disponíveis.

Então organizar todo esse aglomerado de conteúdo é muito custoso e causa fadiga de informação (ADIYANSJAH; GUNAWAN; SUHARTONO, 2019). Desta maneira, é de grande utilidade o desenvolvimento de um sistema de recomendação de filmes que possa sugerir itens semelhantes ao perfil do usuário. Segundo Ricci et al. (2011) os sistemas de recomendação são ferramentas e técnicas que provêm sugestão de itens para os clientes. Esses sistemas baseiam-se no histórico de preferências do usuário e em seu perfil para prever e ofertar itens (e.g. filmes, músicas, vídeos, produtos) adequadas aos indivíduos levando em conta a semelhança das características para surpreender o usuário com itens que atendam suas necessidades no momento, ou para facilitar a utilização dos serviços.

Os sistemas de recomendação encontram sugestões com base na comparação entre itens e/ou usuários, filtram as informações relevantes ordenando itens de acordo com a preferência dos usuários, viabilizando assim a tomada de decisão (SILVA, 2021). Ao considerar essa problemática acerca da dificuldade que o processo de seleção de filmes gera ao usuário diante da abundância do acervo digital de obras disponíveis, o presente trabalho objetiva o desenvolvimento de um sistema de recomendação de filmes que possa fornecer recomendações tendo como base a semelhança dos metadados que descrevem as mídias e visa utilizar a interação do indivíduo com os filmes para gerar melhores recomendações personalizadas para cada usuário.

O sistema de recomendação construído neste projeto utilizou diversas técnicas para fornecer sugestões personalizadas aos usuários. Entre elas, foram utilizadas técnicas de filtragem baseada em conteúdo, filtragem colaborativa com SVD e filtragem colaborativa com KNN. Para treinar e testar o sistema de recomendação, foi utilizado o The Movies Dataset disponibilizado por Banik (2017), que é um subconjunto de dados do MovieLens Dataset mantido pela GroupLens. O The Movies Dataset é um conjunto de dados público com informações detalhadas sobre filmes, avaliações de usuários, entre outras informações relevantes. O uso desse Dataset permitiu que o sistema de recomendação fosse desenvolvido com base em dados reais e confiáveis, garantindo maior precisão e relevância nas sugestões oferecidas aos usuários.

1.1 JUSTIFICATIVA

Tendo em vista a evolução tecnológica do mercado cinematográfico, a ascensão dos serviços web e de streaming, e a grande quantidade de acervo digital disponível nestas plataformas, os sistemas de recomendação estão cada vez mais presentes no nosso dia-a-dia para que o acesso a estes recursos seja realizado de maneira facilitada. Desta forma, esse projeto irá desenvolver um sistema de recomendação que classifique as informações disponíveis, para facilitar o processo de seleção de filmes pelo usuário diante da abundância de obras cinematográficas disponíveis. O surgimento desses serviços causam uma sobrecarga de informação para o utilizador devido ao amontoado de dados disponível e sua vasta gama de opções. Com essa diversidade de serviços à disposição, buscar novos itens exige esforço e atenção, podendo levar o usuário a tomar decisões por itens que não o agradem (SILVA, 2021).

Os problemas relacionados ao fato da abundância de filmes na era digital vão desde a dificuldade de o sujeito escolher os filmes, encontrar novos gêneros que se encaixem no seu perfil em meio a esse emaranhado de obras até a demora na busca do conteúdo com informações relevantes para o seu consumo.

Dessa forma, é possível notar que o projeto de desenvolver um sistema de recomendação de filmes pode impactar diretamente na experiência do usuário, pois este irá despender menos tempo para escolher os títulos mais atrativos ao seu perfil. Além de que, os sistemas de recomendação em alguns ramos podem ser bastante lucrativos quando efetivos, ou então, um modo de se destacar consideravelmente das plataformas concorrentes (ROCCA; ROCCA, 2019).

1.2 OBJETIVO GERAL

Este trabalho tem como objetivo implementar um sistema de recomendação de filmes, tendo como base a análise da similaridade dos metadados que descrevem os filmes mais apreciados pelo usuário. Além disso, o objetivo visa utilizar a interação do indivíduo com os filmes para gerar as recomendações, sendo estas personalizadas para cada usuário.

1.3 OBJETIVO ESPECÍFICOS

- Realizar uma revisão da literatura para compreender o estado da arte de sistemas de recomendação no contexto do ramo cinematográfico.
- Selecionar técnicas e tecnologias para o desenvolvimento de um sistema de recomendação de filmes.
- Definir quais informações compõem o perfil do usuário.
- Estabelecer critérios e métricas para a recomendação de filmes baseados no perfil do usuário.
- Construir um de sistema de recomendação de filmes que utilize os critérios e métricas estabelecidos para recomendar títulos de obras cinematográficas aos usuários.

2 METODOLOGIA

Este estudo propõe o desenvolvimento de um sistema de recomendação de músicas que faz uso dos metadados que descrevem as faixas e das interações do usuário com o sistema para sugerir músicas de forma personalizada ao usuário. A natureza do desenvolvimento do software segue a metodologia aplicada.

Em busca de referencial teórico foi realizada uma pesquisa bibliográfica direcionada em sistemas de recomendação de filmes. As fontes utilizadas na pesquisa foram livros, sites, artigos científicos disponíveis no Google Scholar, realizadas entre maio e junho de 2022.

Os passos para realizar o desenvolvimento desse estudo estão listados nos tópicos metodológicos apresentados a seguir:

- **Revisão bibliográfica:** foram utilizados artigos científicos, livros, tutoriais e artigos de sites para analisar o estado da arte em sistemas de recomendação. Os resultados mostraram diversas abordagens, técnicas e aplicações dos sistemas de recomendação, destacando a relevância e a constante evolução da área. A avaliação dos sistemas também foi discutida, com a utilização de métricas para medir a eficácia.
- **Definição do Dataset:** os Datasets são conjuntos de dados específicos que servem de amostras para treinamento de algoritmos de Inteligência Artificial e recomendação, nessa etapa será selecionado um ou mais Datasets que possam ser utilizados no processo de desenvolvimento do sistema de recomendação.
- **Limpeza e análise do Dataset:** após a seleção do Dataset serão aplicadas técnicas de limpeza dos dados visando remover possíveis ruídos (e.g. outliers, valores nulos ou duplicados) que prejudiquem no processo de análise do Dataset, em seguida será feita uma análise do Dataset.
- **Modelagem do sistema de recomendação:** por conseguinte será iniciado o processo de modelagem e desenvolvimento do sistema de recomendação.
- **Treinamento de algoritmos de classificação:** os algoritmos de classificação precisam ser ajustados de acordo com os dados, eles precisam ser treinados, para entender os padrões e poder classificá-los.

- **Personalização do sistema de recomendação:** processo de personalização do sistema de recomendação com o algoritmo de classificação treinado.

2.1 DATASETS

Para o desenvolvimento do sistema de recomendação o presente estudo fez uso do conjunto de dados *The Movies Dataset* para análise e construção do sistema de recomendação. O *The Movies Dataset* é uma fonte abrangente de dados relacionados a filmes compilados por Rounak Banik e disponibilizados no Kaggle (BANIK, 2017). Essa fonte de dados foi criada a partir de dados do Dataset *MovieLens* que é uma base de dados de acesso gratuita e pública mantido pelo GroupLens e utilizada em estudos anteriores, incluindo os trabalhos de Silva (2021), Steck (2018) e Kaya e Bridge (2019).

O *The Movies Dataset* contém uma vasta gama de informações sobre filmes, incluindo detalhes sobre títulos, elenco, equipes de produção, gêneros, sinopses, datas de lançamento, idiomas, contagem de votos TMDB, médias de votos, avaliações de usuários, dados de palavras-chave, tags, links para informações em outras bases e muito mais. A base de dados foi coletada do site de competição Kaggle (https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset) que oferece um rico conjunto de informações interessantes para pesquisas como: análises exploratórias, estudos de mercado, recomendação de filmes e muitas outras aplicações de pesquisa científica no campo de cinema e indústria cinematográfica.

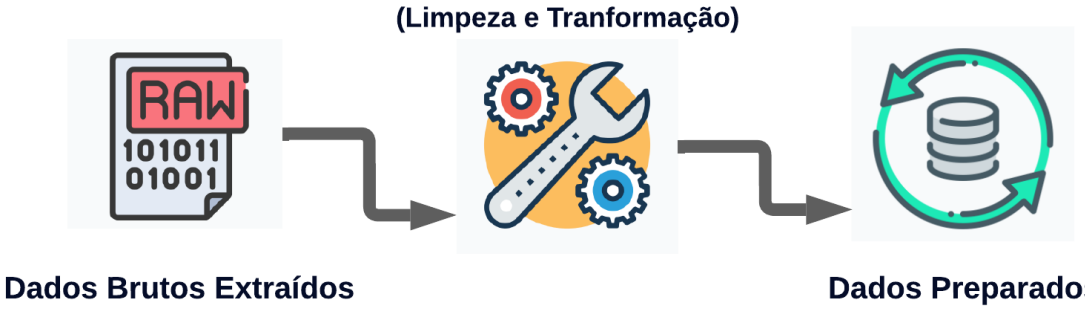
A base de dados é apresentada em formato CSV (Comma-separated values - valores separados por vírgulas) e é composta por vários arquivos, cada um contendo informações específicas sobre diferentes aspectos e características dos filmes. O conjunto de dados é composto por filmes que foram lançados até julho de 2017 e é dividido em **The Full Dataset** e **The Small Dataset**.

O **The Full Dataset** é o conjunto de dados completo que consiste em aproximadamente 26.000.000 classificações fornecidas por mais de 270.896 usuários para os 45.466 filmes. Já o **The Small Dataset** é um subconjunto composto por mais 100.004 classificações a 9.066 filmes, provenientes de 671 usuários. Para o desenvolvimento deste trabalho foram utilizados todos os arquivos do **The Small Dataset** e os arquivos de palavras-chave e créditos do **The Full Dataset** pois esses

não possuíam um subconjunto menor. Logo abaixo são listadas informações sobre cada Dataset utilizado no trabalho.

- **movies_metadata:** contém informações gerais sobre os filmes, tais como título, gênero, sinopse, idioma, país de origem, orçamento, receita e diversas outras informações relevantes. Além disso, ele traz detalhes sobre as empresas envolvidas e países de produção. Contém informações de 45.466 filmes.
- **credits:** traz informações sobre o elenco e a equipe de produção de cada filme, como atores, diretores, roteiristas e demais membros da equipe técnica. Nele, são disponibilizados dados como: nome, papel e ID dos profissionais envolvidos. Essas informações estão presentes na forma de uma string JSON codificada.
- **keywords:** contém palavras-chave associadas a cada filme, que podem ser usadas para descrever os temas ou os assuntos abordados nos filmes. Essas informações estão presentes na forma de uma string JSON codificada.
- **links_small:** contém uma seleção de 9.125 filmes e apresenta informações relativas à identificação de filmes em outros sites de filmes, tais como o ID do IMDB (do Internet Movie Database), o ID do TMDB (do The Movie Database) e o ID do filme no site MovieLens.
- **ratings_small.csv:** contém uma seleção de aproximadamente 100.000 avaliações, provenientes de 671 usuários e contém avaliações de usuários para os filmes, incluindo o ID do usuário, o ID do filme, a classificação atribuída pelo usuário e o horário em que a avaliação foi feita.

Figura 1 - Sequência de etapas do pré-processamento



Fonte: Figura elaborada pelo autor.

As bases de dados brutas passaram por um pré-processamento, durante o qual uma sequência de transformações foi aplicada, com o objetivo de torná-las mais limpas e passíveis de filtragem. A Figura 1 demonstra a sequência de etapas realizadas durante o pré-processamento.

O pré-processamento é uma etapa muito importante, pois envolve a preparação dos dados para qualquer projeto que envolva análise de dados, além de ser uma etapa fundamental para garantir a qualidade dos dados e reduzir erros na análise (FACELI et al., 2021). A seguir estão listadas algumas etapas de preparação dos dados utilizadas nesse projeto:

- Organização na forma da apresentação dos dados da coluna de gêneros dos filmes (exemplo: era “[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'name': 'Romance'}]”, foi para ”[Comedy, Romance]”).
- Criação de uma coluna que possua o dado do ano de lançamento do filme, o ano foi extraído da coluna que contém a data de lançamento (release date).
- Foram removidas as linhas [19730, 29503, 35587], pois possuíam dados mal formatados dos filmes (ex. linhas do dataframe onde foi inserido id no atributo data, além de valores ausentes nos demais atributos).
- Atribuição de tipos de dados corretos as colunas (ex. md_filmes['id'].astype('int')).
- Remoção de linhas duplicadas do Dataframe de filmes.
- Filtragem dos filmes com base nos dados do Dataset de **links_small**, este filtro foi aplicado porque a base de links é extensa e, devido à limitação da capacidade de processamento disponível, foi decidido por trabalhar com um subconjunto dos filmes disponíveis.

Após a realização de todas essas etapas, os dados resultantes encontram-se limpos, preparados e foram modelados de acordo com cada modelo de dados, apresentando, portanto, uma estruturação adequada (SILVA, 2021). A base de dados resultante possui 9.086 registros de filmes, 100.004 avaliações e 671 registros de usuários.

3 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta uma fundamentação teórica com o objetivo de introduzir os conceitos fundamentais que contribuem para a melhor compreensão do trabalho e que foram utilizados como base para o desenvolvimento do sistema de recomendação.

3.1 SISTEMAS DE RECOMENDAÇÃO

De acordo com a FACELI et al. (2021), na década de 1990, surgiu o conceito de sistemas de recomendação, que consistem em ferramentas que utilizam informações relacionadas aos usuários e aos itens para encontrar um conjunto de itens que possam agradá-los. Com o amadurecimento desses sistemas ao longo dos anos, o seu uso se tornou cada vez mais comum, sendo difícil navegar pela internet sem encontrar alguma ferramenta que faça uso desse conceito.

Os sistemas de recomendação são algoritmos que fornecem sugestões personalizadas de itens para usuários com base em suas preferências e comportamentos passados. Para compor a lista de sugestões, os sistemas de recomendação utilizam dados históricos de interação entre usuários e itens para prever e sugerir itens que os usuários possam estar interessados (SILVA, 2021).

Segundo FACELI et al. (2021), a propagação dos sistemas de recomendação levou as empresas a reconhecerem cada vez mais a relevância de proporcionar uma experiência singular para cada cliente por meio de sugestões personalizadas. Em várias organizações, a grande maioria dos produtos ou serviços acessados diariamente pelos seus usuários decorrem dos seus algoritmos de recomendação.

Quanto à construção de sistemas de recomendação existem várias abordagens de desenvolvimento, sendo algumas das principais: filtragem baseada em conteúdo, filtragem colaborativa, filtragem híbrida. Cada uma dessas abordagens são brevemente explicadas nas próximas seções.

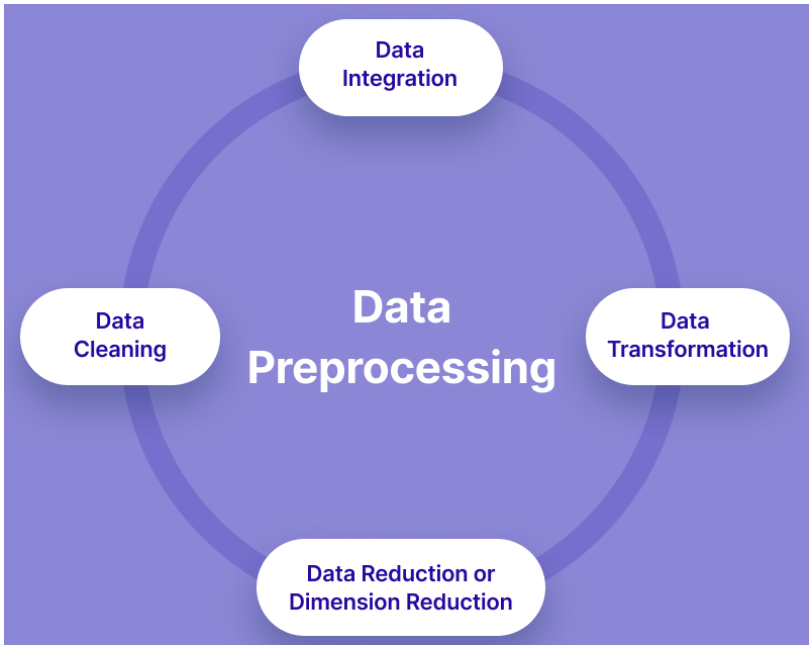
3.1.1 Pré-Processamento

A etapa de pré-processamento é uma das etapas essenciais para o desenvolvimento de sistemas de recomendação. Os conjuntos de dados podem apresentar diferentes características, formatos, dimensões e seus atributos podem possuir dados faltantes, duplicados, valores incorretos ou inconsistentes. Além disso, conjunto de dados pode conter um número pequeno ou elevado de atributos e Faceli

et al. (2021) afirmam que o estado do dado afeta no desempenho dos algoritmos de aprendizado de máquina.

As técnicas de pré-processamento de dados listadas na Figura 2 são frequentemente empregadas para melhorar a qualidade dos dados, seja eliminando ou minimizando problemas existentes citados anteriormente (FACELI et al., 2021). O objetivo dessa etapa é preparar os dados para que possam ser consumidos pelos algoritmos de aprendizado de máquina, algumas das tarefas realizadas nessa etapa são: limpeza, normalização, transformação dos dados, além de redução de dimensionalidade e criação de recursos.

Figura 2 - Técnicas de pré-processamento de dados



Fonte: (BAHETI, 2023, p. 1)

Dessa forma, as técnicas de pré-processamento são úteis não apenas por minimizar ou eliminar problemas do conjunto de dados, mas também por tornar os dados mais adequados para serem utilizados por determinados algoritmos de aprendizado de máquina (FACELI et al., 2021). Em resumo, a etapa de pré-processamento é fundamental para a construção de sistemas de recomendação eficazes. Ela ajuda a limpar, normalizar, transformar e resumir os dados, tornando-os prontos para serem usados pelo algoritmo de recomendação.

3.1.1.1 Pré-Processamento Textual

O pré-processamento textual é uma técnica específica para tratamento de dados textuais, que visa melhorar a qualidade dos dados antes de serem usados em

algoritmos de aprendizado de máquina e mineração de texto. Cazella, Drumm e Barbosa (2010) destacam que, é comum que muitas palavras em um documento não sejam relevantes para representá-lo semanticamente, sendo que, em geral, substantivos ou grupos de substantivos são as palavras mais representativas em um documento. Essa etapa de pré-processamento é importante para garantir que os algoritmos de aprendizado possam entender e extrair informações relevantes dos dados textuais.

Algumas das técnicas comuns de pré-processamento textual incluem:

- Remoção de stopwords: remove palavras comuns que não adicionam significado ao texto, como "a", "e", "o" e "de".
- Tokenização: divide o texto em unidades menores, geralmente palavras, que podem ser processadas pelo algoritmo. Esse processo pode envolver a remoção de pontuação e caracteres especiais.
- Stemming e lematização: reduzem as palavras a sua raiz ou forma básica, para que diferentes formas de uma palavra possam ser tratadas como a mesma. Por exemplo, as palavras "casa", "casas" e "casinha" podem ser reduzidas à raiz "cas".
- Normalização de caso: garante que todas as palavras no texto estejam em maiúsculas ou minúsculas para que não haja ambiguidade nas análises.
- Remoção de caracteres não alfabéticos: remove caracteres não alfabéticos como números e símbolos que não têm relação com o conteúdo do texto.
- Bag-of-words: tem como propósito criar uma representação numérica de um conjunto de palavras em um texto, atribuindo a cada única palavra um número, de acordo com sua frequência no texto. Esse número é calculado com base na quantidade de ocorrências de cada palavra no texto. Essa técnica não leva em conta as características sintáticas das palavras (JÚNIOR, 2021).

Para realizar algumas das etapas de pré-processamento, foram utilizadas funções de bibliotecas específicas. No caso da tokenização, por exemplo, foi empregada a função CountVectorizer da biblioteca Scikit Learn. Conforme descrito por Campos e Figueiredo (2021), essa função realiza a contagem de palavras em

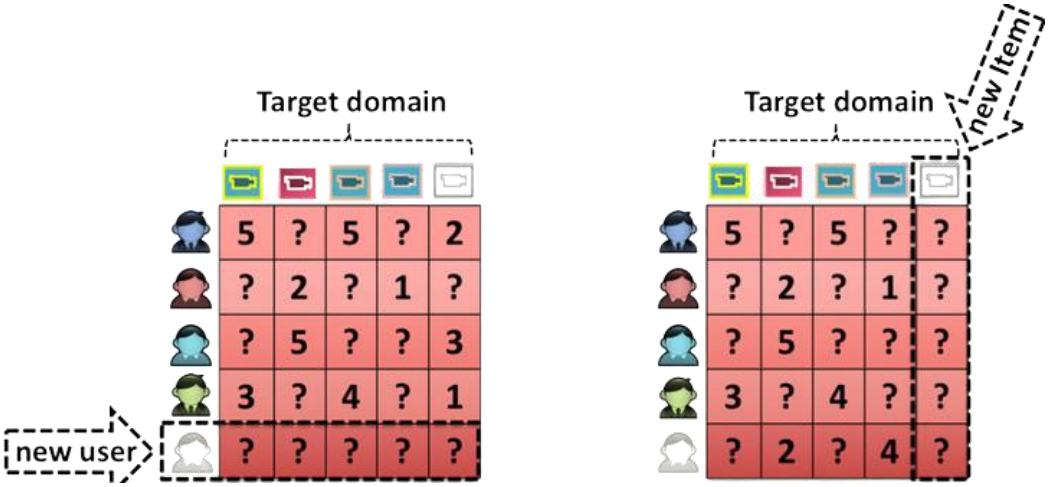
cada documento e transforma dados textuais em vetores. Seguindo o mesmo raciocínio, Júnior (2021) descreve que a função CountVectorizer utiliza o conceito de Bag-of-Words para produzir um vetor de características. Já para realização da lematização foi a SnowballStemmer biblioteca da Natural Language Toolkit (NLTK).

As técnicas pré-processamento textual ajudam na melhoria da qualidade dos textos e, consequentemente, aumentam a eficácia dos algoritmos de aprendizado e mineração de texto, essas técnicas permitem que esses algoritmos possam lidar com dados textuais de forma mais eficiente e produzir resultados mais precisos.

3.1.2 Partida A Frio (Cold Start Problem)

O problema de partida a frio, ou "cold start", é um desafio comum em sistemas de recomendação. Ele surge quando um novo usuário se inscreve em um serviço ou quando um novo item é adicionado ao catálogo de produtos, e o sistema de recomendação ainda não possui informações suficientes sobre as preferências do usuário ou as características do item para fornecer recomendações precisas.

Figura 3 - Funcionamento das duas formas da partida a frio



Fonte: (ELAHI, 2019, p. 1)

Pires (2021) afirma que, em razão de que a maior parte dos algoritmos de recomendação necessitam fortemente de um registro de interações prévias para gerar uma sugestão, caso um item ou usuário não esteja presente nos dados utilizados para treinar o sistema, torna-se impossível incluí-los na recomendação. Existem duas formas de cold start problem: o cold start do usuário e o cold start do item, como demonstra a Figura 3.

O cold start do usuário ocorre quando o sistema de recomendação não tem informações suficientes sobre as preferências de um usuário novo ou pouco ativo para fazer recomendações precisas. Já o cold start do item surge quando um item novo é adicionado ao catálogo e ainda não há informações suficientes sobre suas características para que o sistema possa fazer recomendações precisas.

De acordo com Goyani e Chaurasiya (2020), ao buscar por usuários ou itens semelhantes, é comum combiná-los com um conjunto de usuários ou itens já disponíveis. Entretanto, quando um novo usuário é cadastrado, seu perfil está vazio, o que torna difícil para o sistema fornecer recomendações personalizadas. O mesmo acontece com novos itens que ainda não foram avaliados pelos usuários. Nesse sentido, a implementação de técnicas híbridas pode ser uma solução para ambos os problemas.

O cold start problem é um desafio comum em sistemas de recomendação, mas há várias técnicas e abordagens que podem ser usadas para lidar com ele e fornecer recomendações precisas e personalizadas, mesmo para novos usuários e itens.

3.1.3 Filtragem Colaborativa

A filtragem colaborativa é uma abordagem utilizada em sistemas de recomendação que consiste em analisar as interações dos usuários com itens conforme ilustrado na Figura 4 e, a partir disso, fazer sugestões de itens que possam ser do interesse deles. Essa técnica se baseia no pressuposto de que usuários com gostos e comportamentos semelhantes tendem a ter preferências parecidas em relação aos itens.

Conforme Silva (2021) afirma, essa técnica de recomendação considera os feedbacks de todos os usuários para indicar novos produtos ou serviços a outros usuários, levando em consideração o histórico de preferências deste usuário. Esta técnica leva em conta exclusivamente os feedbacks, sem depender, portanto, do conteúdo dos itens ou informações adicionais dos usuários.

Figura 4 - Funcionamento da filtragem colaborativa

Fonte: (TECHLABS, 2021, p. 4)

De acordo com Faceli et al. (2021), as relações entre itens e usuários podem ocorrer de várias formas, como por exemplo, através de uma pontuação que varia dentro de um determinado intervalo (como uma nota atribuída a um item por um usuário), um valor binário que indica se o usuário gostou ou não do item, ou até mesmo um valor unário que representa o consumo ou compra do item. Esses dados podem ser coletados de forma explícita, com o usuário informando seu grau de relação com o item, ou de forma implícita, analisando apenas as ações do usuário.

Figura 5 - Ilustração da matriz de interações usuário-item.

Users	User-item interactions matrix	Items
suscribers	rating given by a user to a movie (integer)	movies
readers	time spent by a reader on an article (float)	articles
buyers	product clicked or not when suggested (boolean)	products

Fonte: (ROCCA; ROCCA, 2019, p. 3)

As interações entre usuários e itens são registradas e armazenadas na matriz de interações usuário-item como consta na Figura 5. Assim, a premissa fundamental dos métodos colaborativos é que as interações prévias entre usuários e itens são capazes de identificar usuários e/ou itens com características semelhantes e, com base nessas proximidades estimadas, realizar predições (ROCCA; ROCCA, 2019).

Os algoritmos da técnica de filtragem colaborativa são divididos em duas subcategorias: abordagem baseada em memória e abordagem baseada em modelo (RICCI; ROKACH; SHAPIRA, 2011; ISINKAYE; FOLAJIMI; OJOKOH, 2015; SILVA, 2021). As características de cada método serão descritas a seguir:

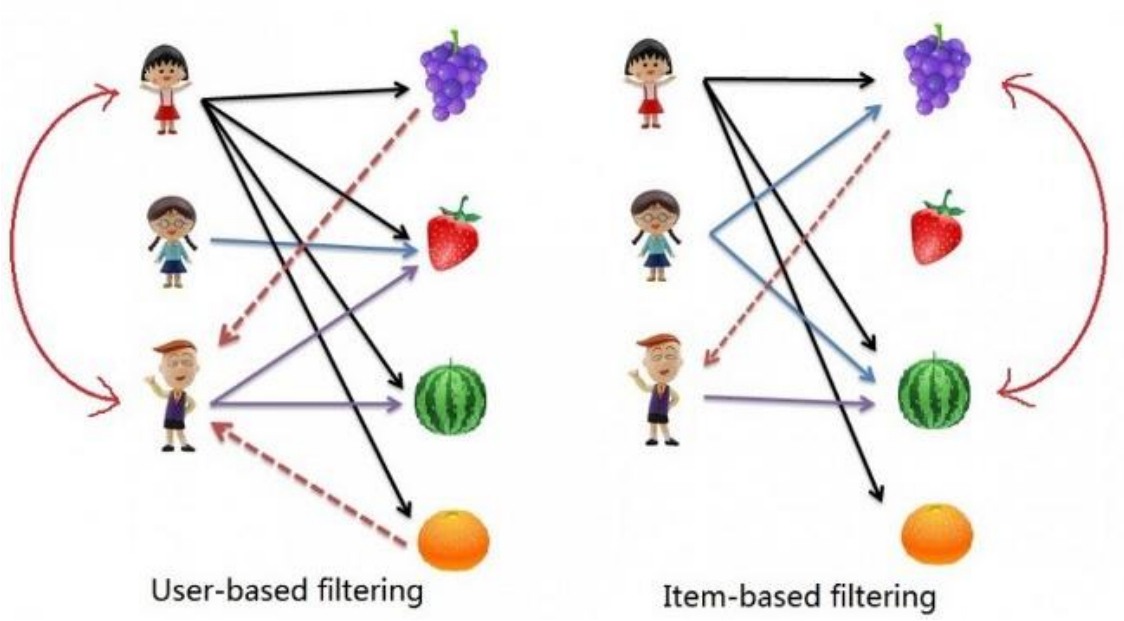
- Método baseado em memória: em algumas pesquisas, é denominado de técnica baseada em vizinhança ou baseado em heurística. Utiliza as avaliações dos usuários diretamente para identificar outros vizinhos que apresentem avaliações semelhantes. Esse método pode ser aplicado de duas maneiras distintas: i) baseado no usuário e ii) baseado no item, conforme evidenciado na Figura 6.

    i. A fim de proporcionar uma nova sugestão a um usuário, a técnica também conhecida como usuário-usuário procura, de modo geral, identificar usuários cujas interações sejam mais semelhantes ao do usuário em questão (vizinhos mais próximos), sugerindo-lhes os itens mais populares entre esses vizinhos (e que são "novos" para o nosso usuário). Este método é considerado "baseado no

usuário", já que representa os usuários com base em suas interações com os itens e avalia a distância entre eles.

ii. Com o intuito de sugerir novos itens a um usuário, a estratégia empregada pelo método, também conhecido como item-item, é identificar itens semelhantes aos que o usuário já teve interações "positivas". A similaridade entre dois itens é determinada se a maioria dos usuários que interagiram com ambos os itens o fez de maneira semelhante. Por isso, esse método é considerado "baseado no item", já que representa os itens com base nas interações que os usuários tiveram com eles e avalia as distâncias entre eles.

Figura 6 - Ilustração do método baseado em usuário e baseado em item.



Fonte: (ILUMEO, 2019, p. 9)

- Método baseado em modelo: usa técnicas de aprendizado de máquina para construir um modelo que preveja as preferências dos usuários com base em seus comportamentos anteriores. Através do uso de algoritmos de aprendizado de máquina, o perfil do usuário é aprendido e pré-computado no sistema, o que facilita o processo de recomendação. No entanto, é necessário um período de aprendizado para que o sistema adquira o conhecimento necessário. Por sua vez, esse método analisa a relação usuário-item a fim de sugerir novos itens. Esse método é mais eficiente em lidar com a esparsidade do conjunto de dados em comparação com o método baseado em memória. Diversos tipos de

aprendizado podem ser empregados, incluindo os probabilísticos, fatoração de matrizes, redes neurais, entre outros.

Os sistemas de recomendação que fazem uso da técnica de filtragem colaborativa levam em consideração tanto as vantagens quanto as desvantagens dessa abordagem. Algumas vantagens são específicas dos métodos baseados em vizinhança, enquanto outras estão relacionadas aos métodos baseados em modelos. Da mesma forma, cada método possui suas próprias desvantagens.

Tanto os prós quanto os contras são abordados nos estudos realizados por Silva (2021); Isinkaye; Folajimi; Ojokoh (2015), Desrosiers; Karypis (2011) e Su; Khoshgoftaar (2009).

Algumas vantagens da técnica colaborativas são:

- Os sistemas de recomendação baseados em memória são caracterizados pela facilidade de compreensão, implementação e adaptação.
- Os sistemas de recomendação baseados em modelos apresentam alta precisão em suas recomendações.
- Ao contrário da técnica de filtragem baseada em conteúdo, a filtragem colaborativa não requer informações detalhadas sobre os itens para gerar recomendações. Em domínios com poucos dados disponíveis, essa técnica pode fornecer resultados melhores.

Algumas desvantagens da técnica colaborativas são:

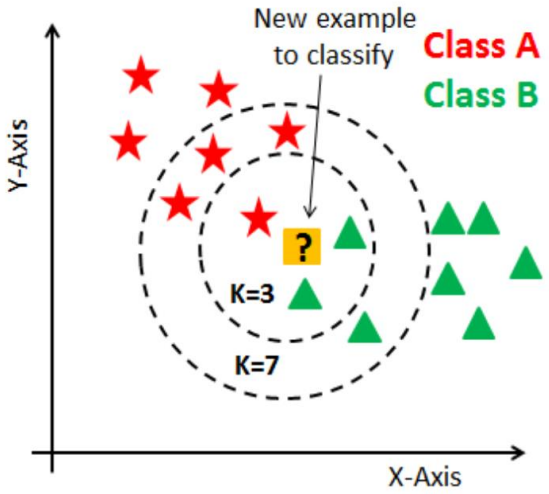
- Um dos principais desafios da técnica de filtragem colaborativa é quando um usuário novo se inscreve no aplicativo e ainda não forneceu nenhum feedback, problema conhecido como partida a frio (“cold start"). Nessa situação, a técnica não é capaz de gerar recomendações personalizadas para o usuário.
- O cadastro de um novo item na aplicação pode representar um problema para a técnica de recomendação, já que o item não possui nenhum feedback do usuário. Dessa forma, a técnica não é capaz de incluir o item na lista de recomendação de algum usuário que possa vir a ter interesse nele.
- O emprego de feedbacks pode gerar enviesamentos na técnica de recomendação, resultando em sugestões que não correspondem às preferências reais dos usuários. Dentre os enviesamentos mais comuns, destaca-se o enviesamento por popularidade, no qual as recomendações são formadas por um conjunto de itens populares entre os usuários mais antigos do sistema. Outro enviesamento é a

sobreposição, que ocorre quando um gênero de maior preferência sobrepõe um gênero de menor preferência.

3.1.3.1 Vizinhos Mais Próximos ou K-Nearest Neighbor (k-NN)

O algoritmo dos vizinhos mais próximos (K-Nearest Neighbor (k-NN)) é um algoritmo de aprendizado de máquina frequentemente utilizado para tarefas de classificação e regressão. O k-NN é um dos algoritmos mais simples de entender e implementar, e é conhecido por sua eficácia em problemas de classificação de dados. O algoritmo k-NN é baseado na ideia de que itens semelhantes são classificados juntos e seu funcionamento é relativamente simples como demonstrado na Figura 7.

Figura 7 - Exemplo de classificação usando k-NN.



Fonte: (SHAH, 2021, p. 1)

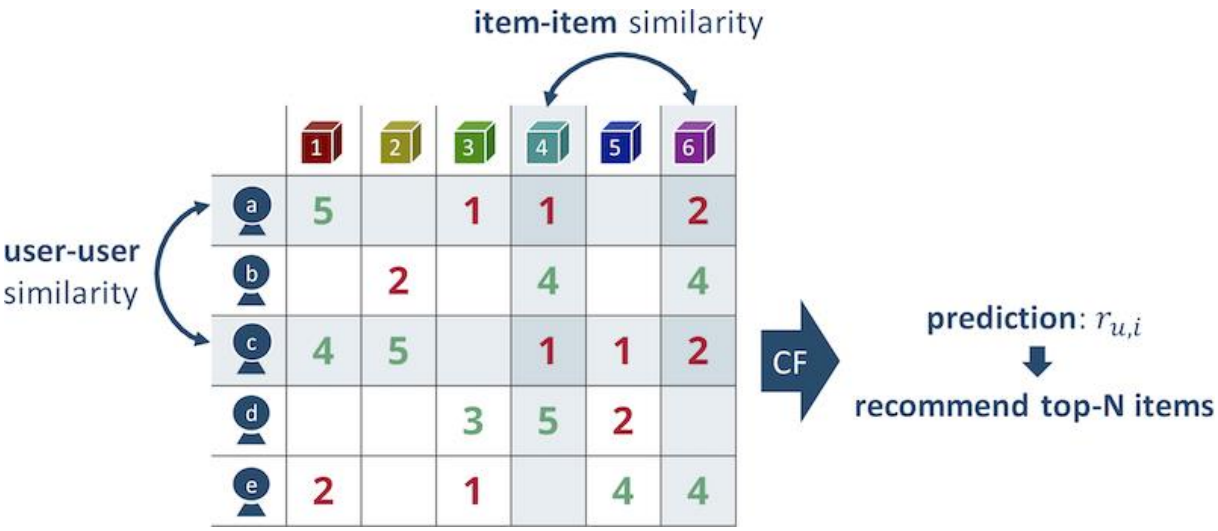
O algoritmo é alimentado com um conjunto de dados que consiste em um conjunto de objetos com suas respectivas classes ou valores de saída e para realizar uma classificação de um novo objeto, o k-NN calcula a distância entre ele e todos os objetos de treinamento. A distância pode ser medida por várias métricas, como a distância do cosseno ou a distância de Manhattan, entre outras. Em seguida, o k-NN seleciona os k objetos mais próximos do novo objeto com base em sua distância e usa suas classes ou valores de saída para prever a classe ou valor de saída do novo objeto (FACELI et al., 2021).

Algumas das vantagens do k-NN incluem sua facilidade de implementação e interpretação, flexibilidade para trabalhar com diferentes tipos de dados e o algoritmo de treinamento é simples. No entanto, o k-NN tem algumas desvantagens. Uma desvantagem é que ele pode ser computacionalmente caro para grandes conjuntos

de dados, uma vez que é necessário calcular a distância para cada ponto no conjunto de treinamento. Além disso, o k-NN pode ter problemas com a chamada "maldição da dimensionalidade", que ocorre quando o número de dimensões dos dados aumenta e o espaço de características fica muito espalhado, tornando difícil encontrar vizinhos próximos (FACELI et al., 2021).

De acordo com Silva (2021), os métodos de recomendação baseados em memória, também chamados de métodos baseados em vizinhança, utilizam o feedback dos usuários para determinar a similaridade entre os vizinhos. Quanto maior a similaridade, mais próximos estão esses vizinhos. Esses métodos podem ser divididos em dois grupos: um que verifica a similaridade entre os usuários (baseado em usuários) e outro que verifica a similaridade entre os itens (baseado em itens) conforme a Figura 8. No presente estudo é utilizada a abordagem baseada em usuários.

Figura 8 - Funcionamento do método baseado em usuário e baseado em item.



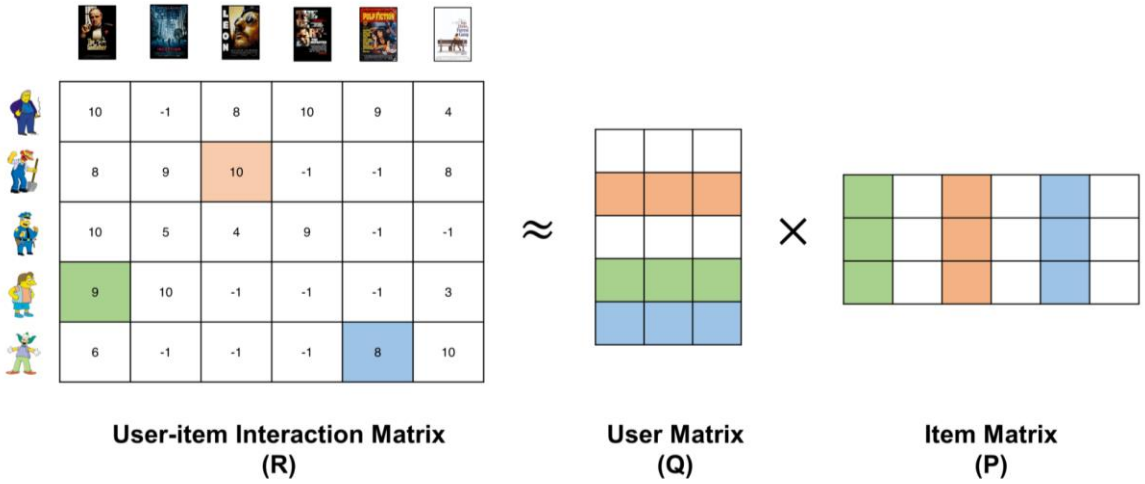
Fonte: (SINHA, 2021, p. 3)

Essa implementação do k-NN se baseia em identificar os usuários que têm as maiores similaridades. A relação entre todos os usuários e todos os itens é representada por meio de uma matriz usuário-item, na qual cada linha corresponde a um usuário e cada coluna corresponde a um item. Ao escolher um usuário para gerar recomendações, são verificados os outros usuários que apresentam semelhanças nos feedbacks. Essa abordagem avalia o grau de similaridade entre o usuário e outro usuário usando qualquer medida de distância (SILVA, 2021).

3.1.3.2 Fatoração De Matrizes

De acordo com Silva (2021), existem algoritmos dentro dos métodos baseados em modelos que utilizam a técnica de fatoração de matrizes. Os algoritmos de fatoração de matriz têm como objetivo decompor uma matriz incompleta de interações entre usuário e item em duas matrizes menores, uma para representar usuários e outra para representar itens. Essa decomposição é realizada de tal forma que é possível reconstruir a matriz original usando as matrizes resultantes, como demonstrado na Figura 9. Durante a reconstrução, os valores das avaliações existentes são mantidos e os valores das avaliações faltantes são previstos, gerando recomendações.

Figura 9 - Ilustração da técnica de fatoração de matrizes.



Fonte: (KUMAR, 2021, p. 2)

Faceli et al. (2021) afirmam que, existem vários algoritmos que utilizam a técnica de fatoração de matriz, e um deles é o SVD (o Singular Value Decomposition, decomposição em valores singulares). O SVD é um método amplamente utilizado para a redução de dimensionalidade e análise de dados, e também pode ser aplicado para recomendação de itens em sistemas de recomendação. Segundo Koren, Bell e Volinsky, (2009) SVD ficou famoso pelo seu desempenho no Netflix Prize, competição promovida pela Netflix.

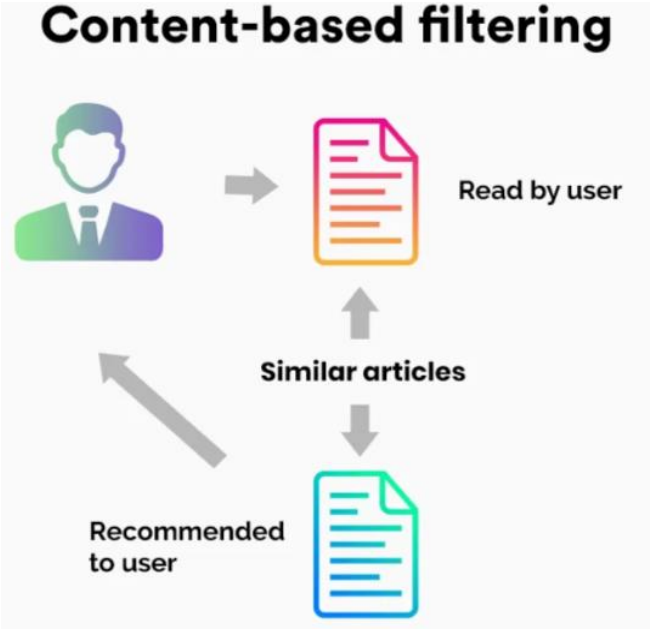
3.1.4 Filtragem Baseada em Conteúdo

A abordagem baseada em conteúdo é uma técnica de filtragem de informações baseada que usa características dos conteúdos para filtrar para recomendar itens semelhantes. De acordo com Dias et al. (2018), o sistema aprende a recomendar itens

semelhantes aos que o usuário apreciou anteriormente, levando em consideração as características associadas a esses itens, por exemplo, se um filme de ação foi avaliado positivamente pelo usuário, o sistema aprende a recomendar outros filmes do mesmo gênero.

A ideia por trás da filtragem baseada em conteúdo é que se um usuário gostou de um item específico, é provável que ele goste de outros itens que possuem características semelhantes. Segundo Silva (2021), os sistemas que adotam essa técnica examinam os metadados que constituem os itens, criando uma estrutura de informações que é utilizada no processo de recomendação. Esse processo se baseia na utilização dos metadados dos itens e das preferências do usuário para encontrar itens desconhecidos com conteúdo semelhantes, classificando-os em uma lista de recomendação.

Figura 10 - Funcionamento da filtragem baseada em conteúdo



Fonte:(TECHLABS, 2021, p. 5)

Os sistemas baseados em conteúdo são amplamente utilizados em várias áreas, como comércio eletrônico, música, filmes e mídia social. Eles podem ser implementados de várias maneiras, fazendo uso dos metadados e analisando seu conteúdo como: palavras-chave, sinopses, tags, gêneros, elencos, músicas entre outros. Como exemplo, na Figura 10 é possível verificar que o usuário comprou um determinado livro e devido à similaridade das informações que constituem o livro foi recomendado outra obra similar.

Assim como outras técnicas de recomendação, os sistemas que utilizam filtragem baseada em conteúdo têm suas vantagens e desvantagens. Algumas das desvantagens são listadas nos estudos de Silva (2021) e Pires (2021). Entre as vantagens temos a rápida adaptação, onde o sistema pode se adaptar rapidamente às mudanças nas preferências do usuário, independentemente da magnitude dessas alterações. Outra vantagem é que o sistema pode fornecer recomendações de novos itens mesmo que o usuário não tenha muitos itens no perfil ou não forneça uma quantidade de feedbacks significativos.

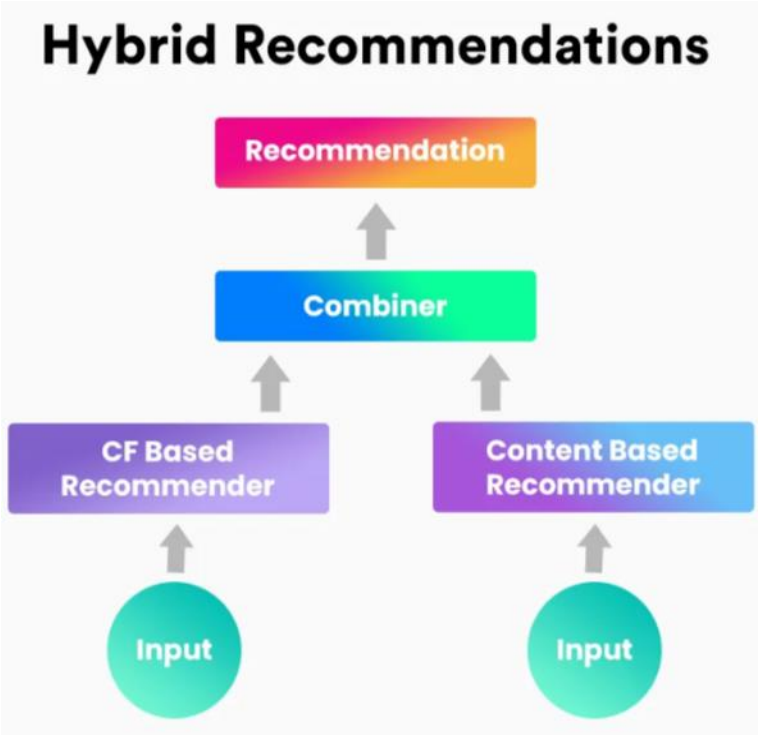
Já entre as desvantagens estão superespecialização onde os itens recomendados têm o máximo de similaridade com os itens já presentes no perfil do usuário, levando o mesmo a ficar preso em uma "bolha de preferências", temos a dependência de informações em que os itens recomendados dependem de metadados que os descrevem, quanto menos informações sobre os itens estiverem disponíveis, mais imprecisa será a recomendação.

A abordagem de filtragem baseada em conteúdo enfrenta menos problemas de partida a frio (cold start problem) do que a abordagem colaborativa. Isso decorre devido que novos usuários ou itens podem ser descritos por suas características (conteúdo), permitindo que sugestões pertinentes sejam feitas para essas novas entidades. Com informações sobre as características dos itens ou usuários, o sistema consegue gerar recomendações personalizadas desde o início, sem precisar depender de interações anteriores do usuário com o sistema.

3.1.5 Filtragem Híbrida

Os sistemas de recomendação que usam a abordagem de filtragem híbrida são aqueles que combinam múltiplas técnicas de recomendação para gerar sugestões personalizadas para os usuários. De acordo com as pesquisas realizadas por Dias et al. (2018) e Silva (2021), um sistema de recomendação híbrido consiste na combinação de duas ou mais técnicas de recomendação, visando melhorar o desempenho e minimizar as desvantagens de cada técnica individualmente. Essa abordagem pode ser realizada de diversas maneiras, como por exemplo, combinando sistemas baseados em conteúdo com filtragem colaborativa conforme demonstrado na Figura 11.

Figura 11 - Exemplo de abordagem filtragem híbrida.



Fonte:(TECHLABS, 2021, p. 6)

A precisão e diversidade de itens sugeridos podem ser aprimoradas pela combinação de múltiplos métodos de recomendação, visto que essa abordagem leva em conta informações de diferentes fontes e técnicas de recomendação. Além disso, Dias et al. (2018), em sua pesquisa, explica que o uso dessa abordagem reduz o problema de "cold-start", onde o sistema tem dificuldades para fazer recomendações para novos usuários ou itens. Devido aos resultados superiores em comparação com os métodos utilizados individualmente, essa abordagem tem se tornado mais amplamente utilizada no mercado.

Sistemas de recomendação híbridos são uma abordagem promissora para a geração de recomendações personalizadas para os usuários. A combinação de diversas abordagens de recomendação pode resultar em sugestões de melhor qualidade, mas é importante escolher as técnicas apropriadas e integrar as informações de forma eficaz.

Figura 12 - Sistema de Recomendação Hydra

Fonte: Figura elaborada pelo autor.

O sistema desenvolvido neste trabalho, ilustrado na Figura 12, é um sistema de recomendação híbrido que combina diferentes técnicas de filtragem, incluindo filtragem baseada em conteúdo com similaridade de cosseno, filtragem colaborativa com SVD e filtragem colaborativa com KNN. Essas técnicas trabalham juntas para oferecer recomendações personalizadas aos usuários, levando em consideração as características dos itens e os feedbacks dos usuários.

4 ALGORITMOS

Nesta seção são apresentados os algoritmos desenvolvidos para o sistema de recomendação híbrido. Foram combinadas duas estratégias de recomendação (baseado em conteúdo e colaborativo) com o algoritmo de classificação vizinho mais próximo (*Nearest Neighbor* (k-NN)). O conjunto desses algoritmos irão resultar no sistema de recomendação híbrido final (ReKomeNNdador Hydra).

4.1 ESTRATÉGIA BASEADA EM CONTEÚDO

Inicialmente para customizar as recomendações foi calculada a semelhança dos filmes com base nos atributos: elenco, equipe, palavras-chave e gênero. Estes atributos são utilizados para sugerir filmes similares a um determinado filme que o usuário gostou. Chama-se de filtragem baseada em conteúdo quando os metadados dos filmes (conteúdo) são utilizados. Nesta etapa foram utilizados os datasets (movies_metadata, links_small) e criada a variável *ds_links* (que contém os dados de *links_small*) que será usada no k-NN.

Durante a análise do conteúdo dos filmes foi realizada uma etapa de pré-processamento dos dados, nessa etapa foi percebido a necessidade de excluir algumas linhas que possuíam dados mal formatados (ex. linhas do dataframe onde na coluna id foi inserido data, além de valores ausentes nos demais atributos), criar colunas ano e organizar dados em formato de lista da coluna de gênero além de remover dados duplicados presentes no dataframe de filmes, como haviam linhas inteiras duplicadas e linhas que só se diferenciavam pela popularidade, foram considerados os dados dos filmes com maior popularidade.

Para implementação deste algoritmo, uniu-se o dataframe filmes (que contém a coluna gêneros) com os dataframes de créditos (contém o elenco e a equipe) e palavras-chave do dataframe de palavras-chave (keywords), após isso é criada a variável *smd* (similar movie metadata). *smd* é um subconjunto de dados do *md_filmes* (movie data filmes) filtrado pelo conjunto de dados do dataframe links_small, também é criado dataframe *ds_filmes* que será utilizado na implementação do k-NN.

Com todos esses dados em um dataframe foi definido que do atributo equipe (crew) será utilizado apenas informações relacionadas ao diretor, já que os outros não contribuem muito para a impressão do filme (ex. equipe de animação, efeitos especiais, produtores, editores entre outros). No caso do atributo elenco (cast) foram selecionados os três principais atores. Logo após, é aplicado nas colunas a função

literal_eval, que visa avaliar qualquer string passada para ela e converte em seu objeto python correspondente.

Em seguida, é criada a função *obter_diretor()* (Algoritmo 1) para obter o nome dos diretores que constam na coluna “crew” e cria uma nova coluna “director” que vai receber esse dado. Posteriormente, os nomes dos atores que constam no elenco são organizados e são selecionados os três principais atores.

ALGORITMO 1: OBTER DIRETOR

```
Input: CelulaCrew //Célula da série que contém a equipe(crew).
Output: Retorna nome_diretor que contém o nome do Diretor.
1: begin
2:     i ← [] //Lista que irá receber os valores de CelulaCrew a cada iteração
3:     for each i ∈ CelulaCrew do
4:         if i.job = "Director" then
               nome_diretor ← i.name
5:             return nome_diretor
6:         end
7:     end
8:     return NULL
9: end
```

A abordagem utilizada é criar um despejo (dump) de metadados que consiste em gêneros, diretor, atores principais e palavras-chave para cada filme. Aplicou-se remoção dos espaços e conversão para minúsculas das características diretor, atores principais e palavras-chave, além de mencionar o diretor três vezes para obter maior peso em relação a todo o elenco, pretendendo aumentar a similaridade de filmes com o mesmo diretor.

Seguidamente é realizado um pré-processamento nas palavras chaves antes de calcular a similaridade cosseno. Após calcular a contagem de frequência de cada palavra, é criada uma série com essas informações. As palavras-chave ocorrem em frequências que variam de 1 a 610. Palavras que ocorrem uma única vez não tem utilidade e podem ser removidas com segurança, resultando numa nova série contendo estas palavras-chaves filtradas. Então é aplicada a função *filtrar_palavras_chave()* (Algoritmo 2) na coluna de palavras-chave do dataframe *smd* visando selecionar apenas palavras que estejam na série filtrada anteriormente.

ALGORITMO 2: FILTRAR PALAVRAS-CHAVE

Input: *palavrasChave* //Célula da série que contém as palavras-chave (keywords).
palavrasFrequentes //Série que contém as palavras chaves com frequência maior que 1.

Output: *Retorna* ***words*** *lista contendo as palavras-chaves filtradas*

```
1:  begin
2:      i ← "" //String contido em keywords a cada iteração
3:      words ← [] //Lista de palavras-chave filtradas
4:      for each  i  ∈ palavrasChave do
5:          if  i in palavrasFrequentes then
6:              words.append(i)
7:          end
8:      end
9:      return words
10: end
```

Logo depois, as palavras passam pelo processo de Stemming onde serão convertidas em seu radical para que palavras como Cats e Cat sejam consideradas iguais e para isso é feito o uso do objeto *SnowballStemmer*. E então é criada uma coluna contendo o saco de palavras (*bag of words*) que é a união das colunas palavras-chave, elenco, diretor e gênero. Na linha 2 (Algoritmo 3), é usado o *CountVectorizer* da biblioteca do Scikit-Learn para criar uma matriz de contagem de tokens, como está descrito no algoritmo abaixo:

ALGORITMO 3: SIMILARIDADE DE COSSENO

```
1:  begin
2:      count ← CountVectorizer() // Converte uma coleção de documentos de texto
        em uma matriz de contagem de tokens
3:      count_matrix ← count.fit_transform(smd['soup']) //Retorna uma matriz
        termo documento
4:      cosine_sim ← cosine_similarity(count_matrix, count_matrix) // Retorna à
        similaridade de cosseno entre os filmes
5:  end
```

Os parâmetros utilizados no *CountVectorizer* são: *analyzer* informando que o recurso deve ser feito de n-grams de palavras, *ngram_range* se refere ao intervalo de n-gramas do texto que será incluído no saco de palavras, nesse caso são unigramas e bigramas, *min_df* que é usado para remover palavras que aparecem com pouca frequência e *stop_words* que irá remover palavras de parada da língua inglesa dos tokens resultantes (ex. "a", "the", "this", "and").

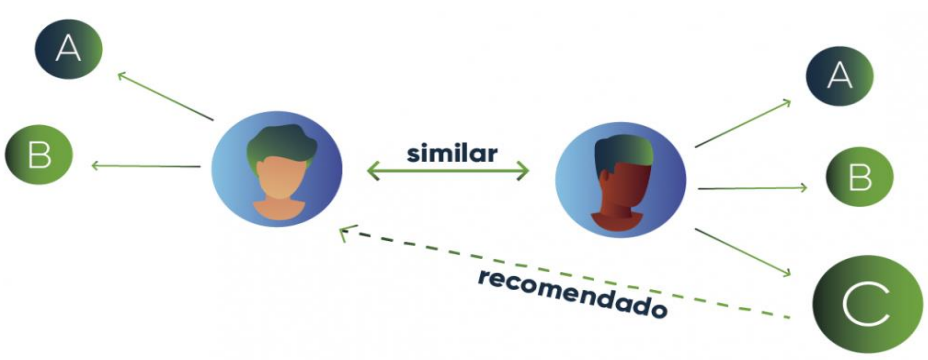
Na linha 3 (Algoritmo 3), é aplicada a função *fit_transform()* que irá aprender o dicionário de vocabulário e retornar uma matriz documento-termo, posteriormente na linha 4 (Algoritmo 3) é calculada a similaridade cosseno e criada uma série chamada de *indices* que contém os títulos e seus respectivos índices no dataframe *smd*, essa série será utilizada para buscar o id do filme através de seu título.

4.2 ESTRATÉGIA COLABORATIVA COM SVD

O algoritmo baseado em conteúdo sofre a limitação de somente ser capaz de sugerir filmes *próximos* a um outro determinado filme, ou seja, não é capaz de capturar gostos e preconceitos pessoais de um usuário, o que torna essa técnica um mecanismo impessoal (RICCI et al., 2010). Qualquer usuário independente de quem seja, que usar esse mecanismo, irá obter as mesmas recomendações caso realize uma consulta para um determinado filme.

Desse modo, foi implementado um algoritmo (Algoritmo 4) que usa a técnica de *Filtragem Colaborativa* para oferecer recomendações mais customizadas e pessoais para o usuário. Essa técnica é baseada na ideia de que indivíduos semelhantes a um usuário, podem ser usados para prever o quanto esse usuário vai gostar de um determinado produto (ex. os filmes) ou serviço que esses indivíduos consumiram/experimentaram, mas o usuário não, como ilustrada na Figura 13.

Figura 13 - Ilustração da Filtragem Colaborativa.



Fonte: (SILVA, 2019, p. 5)

A estratégia *colaborativa* foi implementada utilizando a biblioteca *Surprise,* usando algoritmos como *SVD* (Singular Value Decomposition - Decomposição de Valor Singular) para minimizar o *RMSE* (Root Mean Square Error - Raiz do Erro Quadrático Médio) e fornecer recomendações com maior precisão.

Na linha 2 (Algoritmo 4), foi criado um objeto *Reader* que é usado para analisar um arquivo contendo avaliações, e que em resumo define as configurações de leitura dos dados de entrada para o algoritmo de recomendação. Logo após, é carregado um conjunto de dados de avaliações dos filmes usando a biblioteca Pandas e criado o dataframe *ds_avaliacoes* que será usado na implementação do k-NN (linha 3).

ALGORITMO 4: FILTRAGEM COLABORATIVA SVD

```
1: begin
2:     reader ← Reader() // Analisa um arquivo contendo avaliações
3:     avaliacoes = pd.read_csv('ratings_small.csv')
4:     dados = Dataset.load_from_df(avaliacoes[], reader) // Carrega o conjunto
       de dados de um dataframe do pandas
5:     svd = SVD() // Objeto que realiza a fatoração de matrizes
6:     cross_validate(svd, dados, measures=['RMSE', 'MAE'], verbose=True)
7:     trainset = dados.build_full_trainset() // Cria um objeto trainset
8:     svd.fit(trainset) // Treina o algoritmo com o conjunto de treinamento
9: end
```

Na linha 4 (Algoritmo 4), é feito o uso da biblioteca Surprise para carregar esses dados em um objeto *Dataset* fazendo o uso da função *load_from_df(),* que carrega um conjunto de dados de um dataframe pandas, onde são passados como parâmetros as colunas que contém as informações relevantes do dataframe *avaliações* (‘user_id’, ‘movie_id’, ‘rating’) e o objeto *Reader* para que a biblioteca possa ler corretamente os dados do dataframe.

Logo após na linha 5 (Algoritmo 4), é criado um objeto *SVD,* que é usado para realizar a fatoração de matrizes, e utilizada função *cross_validate (linha 6),* que execute um procedimento de validação cruzada para um determinado algoritmo, passando como parâmetros o objeto *SVD* (algoritmo a ser avaliado), os dados, as medidas (as medidas de desempenho a serem computadas) e a flag verdadeiro para o atributo verbose, essa função é usada para avaliar o desempenho do modelo com

base nas métricas de erro *RMSE (Root Mean Square Error - Raiz do erro quadrático médio) e MAE (Mean Absolute Error - Erro Absoluto Médio)*.

Por fim na linha 7 (Algoritmo 4), é feito o uso da função *build_full_trainset()* para criar um objeto *trainset* que contém todos os dados disponíveis de treinamento, logo após na linha 8, o modelo é treinado com esse conjunto de dados completo usando a função *SVD.fit* que treina o modelo de recomendação usando a técnica de decomposição de valores singulares (SVD).

4.3 ESTRATÉGIA COLABORATIVA COM VIZINHOS MAIS PRÓXIMOS (K-NN)

Em adicional foi desenvolvido o algoritmo de recomendação usando o K-NN (K - Nearest Neighbors ou K - Vizinhos mais próximos) que usa a técnica de filtragem colaborativa baseada em usuários (Algoritmo 5). Este algoritmo tem por objetivo realizar a recomendação de filmes para um usuário específico com base nas avaliações de outros usuários. Esse algoritmo faz uso de *ds_filmes*, *ds_links e ds_avaliacoes* dataframes criados nas seções anteriores e que são passados como parâmetros da função *recomendacao_KNN()*.

Para implementação desse algoritmo são verificados se os identificadores *id* e *imdb_ids* são nulos no *ds_filmes* (criada na Seção 4.1), essa verificação dos identificadores é importante para garantir que o algoritmo possa funcionar corretamente, pois eles são usados para identificar cada entrada no dataset e associá-las a informações adicionais sobre o filme. Caso esses identificadores estejam nulos ou ausentes, pode haver erros ou resultados inconsistentes no algoritmo.

Em seguida é extraída a coluna imdbId e convertida para o tipo de dado inteiro e então feito o merge com *ds_links* (criada na Seção 4.1)*,* além disso são selecionadas as colunas *movieId* e *title* do *ds_filmes* e do *ds_avaliacoes* (dataset de avaliações criado na Seção 4.2) foram selecionadas as colunas *userId*, *movieId* e *rating.*

- É realizado um merge entre *ds_filmes* e *ds_avaliações* usando a coluna de *movieId* e então realizado um pivot[1] no dataframe onde os dados ficam distribuídos no formato de uma matriz onde as linhas são os usuários, as colunas são os títulos dos filmes e os valores são as

[1]O método Pivot é utilizado para transformar um Dataframe em outro, ele permite que os usuários transformem seus dados em uma nova estrutura de tabela para melhor compreensão e análise, reorganizando as informações de acordo com o formato desejado pelo usuário (PANDAS, [s.d.]).

avaliações dadas pelos usuários aos filmes, e os valores NA ou NaN (Not Avaliable e Not a Number respectivamente) são substituídos pela nota zero. Em seguida é instanciado o modelo de vizinhos mais próximos *Nearest Neighbors* - que implementa o aprendizado não supervisionado dos vizinhos mais próximos - como mostrado no Algoritmo 5.

ALGORITMO 5: FILTRAGEM COLABORATIVA COM K-NN

```
1: begin
2:     modelo_knn ← NearestNeighbors(metric ← "consine") // Instanciado o
       modelo que implementa o aprendizado não supervisionado do KNN
3:     modelo_knn.fit(df_recommender) // Ajuste o estimador de vizinhos mais
       próximos do conjunto de dados de treinamento - Calcula as distancias
4:     qtde_vizinhos ← 4 // Quantidade de vizinhos próximos definida
5:     idx_usuario_knn←df_recommender.index.get_loc(df_recommender.loc[usuar
       ioId].name // Recupera a linha que o usuário se encontra
6:     distancia_vizinhos, indices_vizinhos ← modelo_knn.kneighbors(
       df_recommender.iloc[idx_usuario_knn].values.reshape(1, -1),  n_neighbors
       ← qtde_vizinhos) // Retorna a distância e os índices dos vizinhos
7: end
```

Por conseguinte na linhas 2 (Algoritmo 5), é passado como hiperparâmetro a métrica de distância do cosseno, é realizado o treinamento com *df_recommender* (linha 3), é definida a quantidade de vizinhos a serem avaliados (linha 4), em seguida na linha 5 é buscado o índice do usuário no *df_recommender* que passa pelo *reshape* que reformula o formato da série retornada para um Array 2D pois é isso que a função *kneighbors* espera como parâmetro (linha 6). E então na linha 6 (Algoritmo 5), é executada a função *kneighbors* que irá encontrar os vizinhos mais próximos do usuário com base nas avaliações dadas pelos outros usuários.

Na linha 6 (Algoritmo 5), a função *kneighbors* retorna dois arrays: um contendo as distâncias dos vizinhos mais próximos e o outro contendo os índices desses vizinhos. A seguir é criada uma lista de dataframes, onde cada dataframe é composto pelos títulos de filmes e as avaliações dadas pelo usuário e os vizinhos mais próximos. Posteriormente é feito uma mescla entre o dataframe do usuário e seus vizinhos mais próximos utilizando a função *reduce* do módulo *functools* que aplica uma função de dois argumentos cumulativamente aos itens da sequência.

Logo depois, a variável ds_titulos (dataset títulos) que foi criada após a aplicação do *reduce*, é ordenada de forma decrescente pelos vizinhos mais próximos

e então é feito um filtro de todos os filmes que os vizinhos mais próximos assistiram e avaliaram e o usuário não. Após isso, é criada a coluna de média que em *ds_titulos* que é calculada através da soma das avaliações de cada vizinho dividida pela quantidade de vizinhos mais próximos.

A seguir é resetado o índice do *ds_titulos* (que antes eram os títulos dos filmes) para inteiros e os títulos passam a ser uma coluna, após isso é realizada a normalização dos valores da média para que os valores fiquem entre 0 e 1 conforme a função *normalizacao_valores()* (Algoritmo 6). Essa função faz uso do *MinMaxScaler* do Scikit Learn que irá transformar os valores dimensionando cada valor para um determinado intervalo, nesse caso deixando os valores padronizados entre o intervalo de 0 e 1 (linha 5), em seguida é criada uma coluna chamada *est* (estimativa) onde os valores retornados pela função são atribuídos a essa coluna. A função de normalização descrita acima está ilustrada no Algoritmo 6.

ALGORITMO 6: NORMALIZAÇÃO DE VALORES

```
    Input: serie //Recebe uma coluna de valores em formato de série
    Output: Retorna serie_normalizada com valores padronizados entre 0 e 1
1:  begin
2:      serie_index ← serie.index
3:      array  ← serie.values
4:      y ← array.reshape(-1, 1) // Redimensiona o array
5:      scaler ← MinMaxScaler(feature_range= (0, 1)) // Instancia MinMaxScaler
        e informa o intervalo desejado para os dados serem transformados
6:      rescaled  ← scaler.fit_transform(y) // Função que ajusta aos dados e, em
        seguida, transforme-os, normaliza os dados no intervalo de 0 e 1
7:      array  ← rescaled // Atualiza array com valores normalizados
8:      serie_normalizada ←  pd.DataFrame(array, index=serie_index) // Cria
        um dataframe contendo os índices e seus respectivos valores normalizados
9:      return serie_normalizada
10: end
```

Enfim, são selecionadas as colunas *títulos* e *est* do *ds_titulos* ordenados os títulos de forma decrescente pelos valores da coluna de estimativas e logo em seguida é retornado o dataframe contendo os dez títulos e as respectivas estimativas dos filmes que foram avaliados pelos os vizinhos mais próximos e que o usuário ainda não assistiu.

4.4 SISTEMA REKOMENNDADOR HYDRA

Para a construção do sistema de recomendação ReKomeNNdador Hydra foram reunidas as técnicas anteriormente implementadas na análise de conteúdo, colaborativa e vizinhos mais próximos. O funcionamento do sistema de recomendação se dará da seguinte forma:

- **Entrada:** Identificação do usuário (ID) e o título do filme em inglês (EN-US).
- **Saída:** Dataframe contendo os filmes semelhantes ao informado e ranqueados para um determinado usuário com base nas características deste usuário e ordenados pela proximidade do filme retornado ao passado na entrada.

Inicialmente o algoritmo começa carregando o arquivo CSV *links_small* que contém o mapeamento entre identificadores dos filmes (IDs), os IDs do The Movie Database (tmdbIds) e os IDs do Internet Movie Database (imdbIds), sendo selecionadas as colunas *movieId* e *tmdbId*, os valores da coluna *tmdbId* são convertidos para inteiros fazendo o uso da função *convert_int* que consta no Algoritmo 7. Essa função é responsável por converter um valor em inteiro, ela recebe uma variável “x” e tenta convertê-la (linha 3 Algoritmo 7), caso a conversão falhe é retornado um valor nulo (linha 6).

ALGORITMO 7: CONVERT INT - CONVERTE VALOR PARA INTEIRO

```
   Input: x //Célula da série que contém o TMDB id.
   Output: Retorna x o valor da célula convertido para inteiro.
1: begin
2:     try:
3:         x ← int(x) //Tenta converter valor de x para inteiro
4:         return x
5:     catch
6:         return NULL //Retorna nulo caso não consiga realizar a conversão
7:     end
8: end
```

Em seguida, é renomeada a coluna da *tmdbId* para *id* do arquivo carregado e realizado o merge com o dataframe smd (Similar Movie Data) que contém as

informações dos filmes, filtrada as colunas título e id, além de definir como identificador do dataframe o título e atribuido a *id_map*. Logo depois é criada uma variável (*indices_map*) que irá conter o mesmo dataframe sem a coluna de título e é definida a função que *agrupa_recomendacoes()* (Algoritmo 8), que irá juntar o resultado das recomendações das abordagens em conteúdo e filtragem colaborativa(k-NN e SVD) conforme o algoritmo a seguir.

ALGORITMO 8: AGRUPA RECOMENDAÇÕES

Input: *df_knn* //Dataframe que contém as recomendações do KNN
df_colab_cont //Dataframe que contém a recomendação combinada (conteúdo e colaborativa)

Output: Retorna ***dfs_agrupados*** os dataframes combinados.

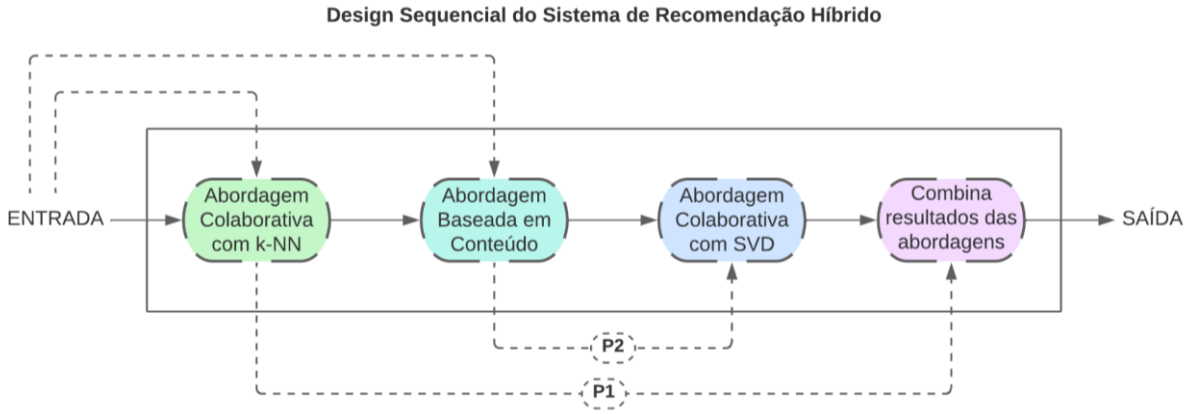
```
1: begin
2:     dfs.concatenados ←pd.concat([df_knn, df_colab_cont], ignore_index←True) // Concatena um dataframe abaixo do outro
3:     dfs_agrupados ← dfs.concatenados.groupby('titulo')[['est']].mean() //Agrupa os titulos e calcula a média da estimativa
4:     dfs_agrupados ← dfs_agrupados.sort_values('est', ascending=False) //Classifica os filmes em ordem descrecente baseado na estimativa
5:     dfs_agrupados ← dfs_agrupados.reset_index() //Redefine os índices para valores inteiros
6:     return dfs_agrupados
7: end
```

A função agrupa_recomendações() (Algoritmo 8) combina os dataframes que contém os resultados de cada abordagem (linha 2) e em seguida agrupa as recomendações com base nos títulos e média das estimativas de classificação (linha 3). Isso é feito devido que o mesmo título pode ser retornado pelas diferentes abordagens utilizadas visando evitar que o valor da estimativa (nota) eleve mais determinado filme em consequência do seu título se repetir nas recomendações. Logo após na linha 4 (Algoritmo 8), os títulos são ordenados com base nas estimativas em ordem decrescente e o índice é redefinido para valor inteiro (linha 5). Essa função é utilizada posteriormente para agrupar as recomendações geradas pelas diferentes abordagens.

Os sistemas de recomendação híbridos possuem dois designs predominantes, o paralelo e o sequencial. Para o desenvolvimento do sistema de recomendação deste

trabalho foi utilizada a abordagem sequencial onde são fornecidos os parâmetros de entrada para um único algoritmo de recomendação e a saída é passada para as próximas abordagens de recomendação em uma sequência até que as saídas sejam combinadas e as recomendações fornecidas ao usuário conforme a Figura 14.

Figura 14 - Design Sequencial do Sistema de Recomendação Hydra.



Fonte: Figura elaborada pelo autor.

Como demonstrado na Figura 14, as setas pontilhadas representam os passos (**P1** e **P2)** do percurso percorrido por cada abordagem no sistema de recomendação híbrido, mostrando a forma que as recomendações foram computadas e agrupadas. Em seguida é descrita a função *recomendador_hydra()* (Algoritmo 9) que é responsável por gerar as recomendações do ReKomeNNdador Hydra, nela serão passados o id (código) do usuário e o título do filme para qual se deseja obter a recomendação.

ALGORITMO 9: RECOMENDADOR HÍBRIDO

```
Input: userId //ID do usuário que consta no Dataset de avaliações.
       titulo //Título do filme que se deseja obter recomendação.
Output: Retorna lst_recomend contendo a lista de filmes sugeridos.
1:  begin
2:      if (userId instanceof Integer) AND (titulo instanceof String) then
3:          if (userId ∈ avaliacoes['userId']) AND (titulo ∈ id_map.index) then
            //Verifica se o userId e o titulo estão contidos nos seus respectivos datasets
4:              idx ← get_index(titulo)
5:              titulos_recomendados_KNN ← recomendacao_KNN(userId)
                //Obtém recomendações baseadas em KNN
6:              sim_scores ← list(enumerate(cosseno_sim[int(idx)])) // Lista contendo a
                similaridade de cosseno dos filmes em relação ao id do filme passado
7:              sim_scores ← sorted(sim_scores, chave=lambda x: x[1], reverse=True)
                //Classifica em ordem decrescente com base os scores de similaridade
8:              sim_scores ← sim_scores[1:26] // Seleciona os 25 filmes mais similares
9:              movie_indices ← [i[0] for i in sim_scores] // Obtém os índices dos filmes similares
10:             movies ← smd.iloc[movie_indices][['title', 'vote_count', 'vote_average',
                'year', 'id']] // Obter informações dos filmes similares
11:             movies['est'] ← filmes['id'].apply(lambda x: svd.predict(userId,
                indices_map.loc[x]['movieId']).est) // Calcula estimativa de avaliação dos filmes
                pelo usuário usando SVD
12:             movies['est'] ← normalizacao_valores(movies['est']) // Normaliza os valores
                de estimativa de avaliação
13:             movies ← movies[['title', 'est']] // Seleciona as colunas título e a estimativa
14:             movies ← movies.sort_values('est', ascending =Falso) // Ordena os filmes pela
                estimativa de avaliação em ordem decrescente
15:             movies ← movies.head(10) // Seleciona os 10 filmes
16:             lst_recomend ← agrupa_recomendacoes(titulos_recomendados_KNN,
                movies) // Combina as recomendações do KNN, Conteúdo e SVD
17:             return lst_recomend
18:         else
19:             print('Erro: O id do usuário ou nome do filme não consta no dataset')
20:         end
21:     else
22:         print('Algo inesperado aconteceu. Tente novamente!')
23:     end
24: end
```

A função *recomendador_hydra()* (Algoritmo 9) recebe como parâmetros o código do usuário e o título do filme, esses dados são passados para uma estrutura de controle que verifica se o código do usuário é inteiro e se o título do filme é um texto caso contrário uma mensagem de erro é exibida (linha 2). Então na linha 3 (Algoritmo 9), é verificado se o id do usuário está presente no dataframe de avaliações e se o título do filme está presente no dataframe *id_map,* se ambos os dados

estiverem presentes nos dataframes a função prossegue. Depois na linha 4 (Algoritmo 9), o título é passado para a função *get_index* (Algoritmo 10) para obter o índice do filme correspondente ao título que foi passado, abaixo o Algoritmo 10 descreve o código da função.

ALGORITMO 10: GET_INDEX - RETORNA OS ÍNDICES DOS FILMES

```
    Input: titulo //Título passado pelo usuário que será buscado o ID.
    Output: Retorna idx que contém o id do filme para o determinado título.
1:  begin
2:      idx ← indices[titulo] //Recupera o(s) id(s) do filme através do título
3:      if isinstace(idx, pd.Series) then   //Verifica se o valor de idx é uma série
4:          return idx[0]        // Retorna o primeiro id da série
5:      end
6:      return idx
7:  end
```

A função *get_index()* (Algoritmo 10) recebe o título do filme e será buscado o id do filme na série índices criada no final da abordagem baseada em conteúdo (linha 2). Em seguida na linha 3 (Algoritmo 10), é feita uma verificação se o valor retornado é uma série ou o código o título, como muitos filmes podem possuir o mesmo título mas seus conteúdos, histórias e enredos podem ser totalmente diferentes, em alguns casos podem ser retornados uma série contendo os índices do filmes com aquele determinado título, então foi definido que somente o primeiro índice dessa série será retornado (linha 4 do Algoritmo 10).

Logo após na linha 5 (Algoritmo 9), é chamada a função *recomendacao_KNN* passando como parâmetro para ela o id do usuário que retorna uma lista dos filmes recomendados para o usuário, empregando o método de filtragem colaborativa baseada em usuários e fazendo o uso do algoritmo de k-Nearest Neighbors (k-NN), a lista retornada é armazenada na variável *titulos_recomendados_KNN* que será usada logo mais. Depois na linha 6 (Algoritmo 9), é criada uma variável chamada *sim_scores* que contém a lista similaridade do cosseno do filme consultado e os demais filmes, a similaridade do cosseno foi calculada na abordagem baseada em conteúdo.

A lista de *sim_scores* contém tuplas que comportam os valores do índice dos filmes e suas respectivas similaridades, em seguida na linha 7 (Algoritmo 9), os filmes são classificados em ordem decrescente com base nos escores de similaridade e

então são selecionados os vinte e cinco filmes mais semelhantes (linha 8), um detalhe a se notar é que nessa lista de similaridade o primeiro item será o próprio filme e devido a isso o range para selecionar esses filmes inicia do código 1 ao invés de 0. Dos filmes selecionados são extraídos os índices e armazenados na variável *movie_indices* (linha 9).

No passo seguinte linha 10 (Algoritmo 9), os índices selecionados contidos em *movie_indices* são usados para extrair informações relevantes dos filmes no Dataframe smd essas informações são: título, contagem de votos, média de votos, ano de lançamento e ID do filme, esses são armazenados na variável *movies*. E então o ID do filme é utilizado para obter uma estimativa de classificação do usuário para cada um dos filmes selecionados fazendo uso da função “predict” do SVD que foi treinado na abordagem colaborativa com SVD, esse método calcula a previsão de classificação para determinado usuário e item, retorna a previsão e então é obtida a classificação estimada (linha 11 do Algoritmo 9).

Em seguida na linha 12 (Algoritmo 9), é utilizada a função listada no Algoritmo 6 para normalizar as estimativas de classificação dos filmes para que estejam na escala entre 0 e 1, mesma escala da saída da abordagem colaborativa que usa k-NN. Depois na linha 13 (Algoritmo 9), são filtradas as colunas dos títulos e estimativas, esses dados são ordenados de forma decrescente com base nas estimativas de classificação dos filmes (linha 14) e então são selecionados os dez melhores filmes com base nas estimativas de classificação e são armazenados na variável *movies* (linha 15).

Logo após na linha 16 (Algoritmo 9), é chamada a função *agrupa_recomendacoes()* presente no Algoritmo 8, que irá receber como parâmetros as variáveis contendo recomendações obtidas da abordagem colaborativa que usa o k-NN e da abordagem de conteúdo combinada com a colaborativa que usa SVD. Desse modo são passadas as variáveis contendo as respectivas recomendações *titulos_recomendados_KNN* e *movies*, são combinadas as listas recomendadas e então é retornada à recomendação híbrida resultante da combinação dos dois Dataframes de filmes recomendados (linha 17 do Algoritmo 9).

Por fim, é observado que o sistema de recomendação ReKomeNNdador Hydra fornece lista de recomendações distintas para cada usuário, inclusive se os títulos dos filmes passados para o mecanismo de recomendação for o mesmo. A lista de recomendação possui vinte títulos ordenados de forma decrescente por suas

respectivas notas (estimativas) para que o usuário possa ter um leque variado de filmes para selecionar. Demonstrando assim, que as recomendações são personalizadas e moldadas para usuários diferentes.

5 CONSIDERAÇÕES FINAIS

Este trabalho apresenta a proposta de desenvolvimento de sistema de recomendação de filmes que faz uso das abordagens de filtragem baseada em conteúdo, colaborativa e híbrida para fornecer aos usuários recomendações personalizadas. A filtragem baseada em conteúdo é utilizada para recomendar filmes com base nas características dos filmes (conteúdo). A filtragem colaborativa é usada para identificar padrões nos comportamentos de avaliação de outros usuários semelhantes e recomendar filmes com base nessas informações. A abordagem híbrida combina essas duas técnicas para fornecer recomendações mais customizadas.

Os sistemas de recomendação permitem que os usuários descubram novos filmes de forma mais fácil e eficiente. Com base em algoritmos e análise de dados, o sistema pode sugerir filmes que possam ser do interesse do usuário com base em suas preferências e histórico de visualização. Isso não apenas melhora a experiência do usuário, mas também pode aumentar a satisfação do cliente e a fidelidade à plataforma de streaming ou serviço de aluguel de filmes. Além disso, o sistema de recomendação pode ser uma ferramenta útil para ajudar na promoção de filmes menos populares ou de nicho, aumentando sua visibilidade e potencial de sucesso.

Nesse estudo, os resultados obtidos demonstram o funcionamento da aplicação e ilustram o seu potencial para uso com usuários reais, tendo sido desenvolvido um sistema de recomendação que utiliza diferentes técnicas de filtragem. Com a aplicação de algoritmos baseados em conteúdo, colaborativos e híbridos, o sistema é capaz de realizar recomendações personalizadas para cada usuário levando em conta o histórico de interações e preferências de cada usuário.

Algumas das possibilidades de trabalhos futuros seriam a criação de um algoritmo de avaliação para medir o desempenho das recomendações, incorporar o feedback dos usuários em tempo real para ajustar as recomendações com base em suas preferências atuais. Além disso, a incorporação de dados adicionais, como a inserção de mais informações sobre o elenco, equipe de produção e votos atribuídos para cada filme, pode melhorar ainda mais a qualidade das recomendações.

REFERÊNCIAS

ADIYANSJAH; GUNAWAN, A. A. S.; SUHARTONO, D. *Music recommender system based on genre using convolutional recurrent neural networks*. Procedia computer science, v. 157, p. 99–109, 2019.

APUKE, O. D. *Quantitative research methods: A synopsis approach. Kuwait Chapter of Arabian Journal of Business and Management Review*, American University, v. 33, n. 5471, p. 1–8, 2017.

BAHETI, P. *Data preprocessing in machine learning [steps & techniques]*. V7labs.comV7, 24 abr. 2023. Disponível em: <https://www.v7labs.com/blog/data-preprocessing-guide>. Acesso em: 26 abr. 2023

BANIK, R. *Movie recommender systems*. Disponível em: <https://www.kaggle.com/code/rounakbanik/movie-recommender-systems>. Acesso em: 9 out. 2022a.

BANIK, R. *The Movies Dataset.*, 10 nov. 2017b. Disponível em: <https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset>. Acesso em: 9 out. 2022

CAMPOS, S. L. B.; FIGUEIREDO, J. M. DE. *Uso de técnicas de processamento de linguagem natural para identificação de similaridade de serviços públicos*. Anais do Workshop de Computação Aplicada em Governo Eletrônico (WCGE 2021). Anais...Sociedade Brasileira da Computação, 2021. Acesso em: 27 abr. 2023

CAZELLA, S. C.; DRUMM, J. V.; BARBOSA, J. L. V. *UM SERVIÇO PARA RECOMENDAÇÃO DE ARTIGOS CIENTÍFICOS BASEADO EM FILTRAGEM DE CONTEÚDO APLICADO A DISPOSITIVOS MÓVEIS*. RENOTE, v. 8, n. 3, 2010.

DA SILVA, A. R. *O que está por trás dos sistemas de recomendação?,* 9 out. 2019. Disponível em: <https://statplace.com.br/blog/o-que-esta-por-tras-dos-sistemas-de-recomendacao/>

DA SILVA, William Rodrigues; DOMINGUES, Marcos Aurélio. *Musipath: um sistema para exploração de relacionamentos de artistas em redes de músicas*. Brazilian Journal of Development, v. 8, n. 4, p. 27802-27820, 2022.

DESROSIERS, C.; KARYPIS, G. *A comprehensive survey of neighborhood-based recommendation methods*. In: RICCI, F. et al. (Ed.). Recommender Systems Handbook. Springer, 2011. p. 107–144. ISBN 978-0-387-85820-3.

DIAS, A. DA S. et al. *Explicação de Recomendações com Diversificação: uma Revisão Bibliográfica*. Cadernos de Informática, v. 10, n. 1, p. 45–64, 2018.

ELAHI, M. *Illustration of Cold Start problem in recommender systems: New user problem (left) and New item problem*. Disponível em: <https://www.researchgate.net/figure/Illustration-of-Cold-Start-problem-in-

recommender-systems-New-user-problem-left-and_fig2_334570701>. Acesso em: 8 abr. 2023.

FACELI, K. et al. *Inteligência artificial: uma abordagem de aprendizado de máquina*. 2021. 2. ed. Rio de Janeiro: LTC, 2021.

GOYANI, M.; CHAURASIYA, N. *A review of movie recommendation system: Limitations, Survey and Challenges*. ELCVIA Electronic Letters on Computer Vision and Image Analysis, v. 19, n. 3, p. 18, 2020.

ILUMEO*. Como funcionam os sistemas de recomendação?* Disponível em: <https://ilumeo.com.br/todos-posts/2019/08/12/como-funcionam-os-sistemas-de-recomendacao>. Acesso em: 9 maio. 2023.

ISINKAYE, F. O.; FOLAJIMI, Y. O.; OJOKOH, B. A. *Recommendation systems: Principles, methods and evaluation*. Egyptian Informatics Journal, v. 16, n. 3, p. 261–273, 2015.

JORDAO, Pedro Henrique Ribeiro. *SISTEMA DE RECOMENDAÇÃO DE MUSICAS USANDO LDA E ATRIBUTOS DE AUDIO*. 2016. Dissertação de Mestrado. Universidade Federal do Rio de Janeiro.

JUNIOR, M.; SANTOS, L. D. *Comparando predição de popularidade de Podcast a partir de metadados, conteúdo e características de áudio*. [s.l.] Universidade Federal de Pernambuco, 9 mar. 2021.

KAYA, M.; BRIDGE, D*. A comparison of calibrated and intent-aware recommendations*. Proceedings of the 13th ACM Conference on Recommender Systems. Anais...New York, NY, USA: ACM, 2019.

KOREN, Y.; BELL, R.; VOLINSKY, C. *MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS*. IEEE Computer Society, 2009.

KUMAR, S. *Fundamental of matrix factorization for recommender system*. Disponível em: <https://www.linkedin.com/pulse/fundamental-matrix-factorization-recommender-system-saurav-kumar/?trk=public_profile_article_view>. Acesso em: 15 maio. 2023.

MOTION PICTURE ASSOCIATION. *Global Theatrical, home entertainment, and pay TV market rebounds to $328.2 billion, new MPA report shows*. Disponível em: <https://www.motionpictures.org/press/global-theatrical-home-entertainment-and-pay-tv-market-rebounds-to-328-2-billion-new-mpa-report-shows/>. Acesso em: 6 abr. 2023.

PANDAS. *Pandas.Pivot_table — pandas 2.0.1 documentation*. Disponível em: <https://pandas.pydata.org/docs/reference/api/pandas.pivot_table.html>. Acesso em: 8 maio. 2023.

PIRES, P. R. *Representação vetorial distribuída em sistemas de recomendação*. 2021.

RICCI, F. et al. (EDS.). *Recommender Systems Handbook. Boston*, MA: Springer US, 2010.

RICCI, F.; ROKACH, L.; SHAPIRA, B. *Introduction to recommender systems handbook. Em: Recommender Systems Handbook*. Boston, MA: Springer US, 2011. p. 1–35.

ROCCA, B.; ROCCA, J*. Introduction to recommender systems*. Disponível em: <https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada>. Acesso em: 9 jun. 2022.

SHAH, R*. Introduction to k-nearest neighbors (kNN) algorithm*. Disponível em: <https://ai.plainenglish.io/introduction-to-k-nearest-neighbors-knn-algorithm-e8617a448fa8>. Acesso em: 7 abr. 2023.

SILVA, A. R. DA. *O que está por trás dos sistemas de recomendação?* Disponível em: <https://statplace.com.br/blog/o-que-esta-por-tras-dos-sistemas-de-recomendacao/>. Acesso em: 1 maio. 2023.

SILVA, D. C. DA. *Explorando calibragem ponderada, balanceamentos e métricas para justiça em sistemas de recomendação*. 2021.

SINHA, J. *Train your first KNN Model for Collaborative Filtering*. Disponível em: <https://blog.jaysinha.me/train-your-first-knn-model-for-collaborative-filtering/>. Acesso em: 6 mar. 2023.

STECK, H. *Calibrated recommendations*. Proceedings of the 12th ACM Conference on Recommender Systems. Anais...New York, NY, USA: ACM, 2018.
SU, X.; KHOSHGOFTAAR, T. M. *A survey of collaborative filtering techniques*. Adv. in Artif. Intell., Hindawi Limited, London, GBR, v. 2009, jan. 2009. ISSN 1687-7470.

TECHLABS, M. *Types of recommendation systems & their use cases - MLearning.Ai - medium*. Disponível em: <https://medium.com/mlearning-ai/what-are-the-types-of-recommendation-systems-3487cbafa7c9>. Acesso em: 8 maio. 2023.