UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS DOIS VIZINHOS
CURSO DE ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS

ANA PAULA APARECIDA SCHAITLER

DESENVOLVIMENTO DE UM ALGORITMO DE RECOMENDAÇÃO BASEADO EM AVALIAÇÕES USANDO GRAFOS

TRABALHO DE CONCLUSÃO DE CURSO DE ESPECIALIZAÇÃO

DOIS VIZINHOS
2022

ANA PAULA APARECIDA SCHAITLER

DESENVOLVIMENTO DE UM ALGORITMO DE RECOMENDAÇÃO BASEADO EM AVALIAÇÕES USANDO GRAFOS

Trabalho de Conclusão de Curso de Especialização apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Orientador: Prof. Dr. Ives Renê Venturini Pola

DOIS VIZINHOS
2022

ANA PAULA APARECIDA SCHAITLER

DESENVOLVIMENTO DE UM ALGORITMO DE RECOMENDAÇÃO BASEADO EM AVALIAÇÕES USANDO GRAFOS

Trabalho de Conclusão de Curso de Especialização apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Data de aprovação: 12/novembro/2022

Ives Renê Venturini Pola
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Pato Branco

Fernanda Paula Barbosa Pola
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Dois Vizinhos

Jefferson Tales Oliva
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Pato Branco

DOIS VIZINHOS
2022

AGRADECIMENTOS

Ao meu orientador Ives, que me auxiliou e ajudou com o tema a ser abordado nesse trabalho.

À todos os meus professores durante os todos esses anos, que contribuíram com o conhecimento obtido até aqui.

À Universidade Tecnológica Federal do Paraná, pela oportunidade de cursar a Especialização em Ciência de Dados e por proporcionar uma gama enorme de conhecimento e renomados professores para realizar com maestria o repasse de conhecimento no curso.

Aos meus amigos que estiveram me apoiando em todo o processo.

Por fim, agradeço aos meus pais, Alecir e Vilma, meus maiores apoiadores, por todo o ensino, incentivo e apoio em todos os momentos da minha vida.

RESUMO

Sistemas de Recomendação são muito utilizados em várias plataformas para uma melhor usabilidade pelo usuário. Utilizando-se de interações históricas do usuários ou usuários com os mesmos interesses é realizado a busca para apresentar as informações de maneira que o usuário tenha uma melhor qualidade na hora de buscar informações para comprar, assistir, escutar, etc. Há vários estudos na área que propõem técnicas e métodos de busca que visam tanto o usuário realizando as predições quanto aos itens que estão elencados de maneira a serem associados uns aos outros conforme suas pontuações realizadas pelo usuários. Em trabalhos realizados, são realizados em sua grande maioria a utilização de filtragem coletiva que visa a apresentação de resultados conforme avaliações de grupos usuários baseados no usuário alvo, e na recomendação baseada em conteúdo que realiza a predição com base em conteúdos já avaliados ou consumidos pelo usuário. Este trabalho ao analisar essa questão, propõe-se a criação de uma recomendação com os dados de um conjunto de avaliações realizadas previamente, visando assim apresentar aos usuários conjuntos únicos de resultados com avaliações recomendadas baseando-se em interações anteriores.

Palavras-chave: Sistemas de Recomendações, Vídeos, Avaliações de Usuários.

ABSTRACT

Recommendation systems are widely used in various platforms for better usability by the user. Using historical interactions of users or users with the same interests, the search is performed to present the information so that the user has a better quality when looking for information to buy, watch, listen, etc. There are several studies in the area that propose search techniques and methods that aim both at the user, making predictions, and at the items that are listed in order to be associated to each other according to their scores by the users. In the majority of the works performed, the use of collective filtering is used, which aims at presenting results according to the evaluations of user groups based on the target user, and the recommendation based on content that makes predictions based on content already evaluated or consumed by the user. This work, while analyzing this issue, proposes the creation of a recommendation with the data from a set of evaluations previously performed, thus aiming to present users with unique sets of results with recommended evaluations based on previous interactions.

Keywords: Recommendation Systems, Videos, User Rating.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CWI	*Centrum Wiskunde & Informatica*
FC	Filtragem Colaborativa
IDF	*Inverse Document Frequency*
k-NNq	Consulta aos k-vizinhos mais próximos (*k-nearest neighbor query*)
RBC	Recomendação Baseada em Conteúdo
Rq	Consulta por abrangência (*Range query*)
SGBD	Sistema Gerenciador de Banco de Dados
SQL	*Structured Query Language*
SR	Sistemas de Recomendação
TF	*Term Frequency*
TF-IDF	*Term Frequency-Inverse Document Frequency*
UCI	*University of California Irvine*

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

No mundo globalizado que vivemos temos cada vez mais a geração de dados, sendo os mesmos dos mais diferentes tipo, como informações geo-referenciadas, imagens, documentos de texto, grafos, sequências genéticas e séries temporais. Como exemplo, as estimativas de janeiro/2022 para os usuários ativos mensais utilizando as redes sociais, indicam que o Instagram, tem cerca de 1,478 milhões de usuários ativos mensais. O Twitter teria cerca de 436 milhões de usuários ativos mensais, com uma estimativa que as redes sociais atinjam o número de 3,96 bilhões de usuários totais em 2022 (STATISTA, 2022). Com esse número imenso de usuários, é gerado um imenso volume de dados das postagens realizadas pelos usuários dessas redes.

Com essa gama enorme de dados gerados são utilizados cada vez mais diferentes métricas e modelos para realização da analise dos dados gerados. Visando os dados complexos, como imagens ou textos com maior quantidade de caracteres, verificando a busca por palavras-chaves ou semelhanças entre as imagens analisadas, foram desenvolvidas técnicas de busca de similaridade objetivando atender às necessidades dos usuários em diversos segmentos do conhecimento (CHEN; CHIANG; STOREY, 2012).

Com tantos dados gerados pelos usuários, é possível realizar recomendações futuras aos mesmos conforme o histórico de interação e seus gostos informados para os sistemas utilizados. Os Sistemas de Recomendação (SR) auxiliam no aumento da eficácia e capacidade nesses processos de indicação já conhecido entre as interações dos seres humanos. Os dados de gostos de usuário e seus históricos de visualização auxiliam também os sistemas a recomendarem itens para outros usuários (LINDEN; SMITH; YORK, 2003).

A coleta dos dados relacionados aos usuários ocorrem de maneira implícita onde o usuário realiza interações com o sistema contribuindo para os SR tenham dados de interesse, tendo assim os dados de necessidades e preferências do usuário, como exemplos de interações implícita, temos o usuário favoritando uma página Web, movimentos do *mouse*, compartilhar uma publicação, assistir um filme ou curtir uma publicação. Outra maneira de coleta de dados ocorre de maneira explícito, quando o usuário realiza a avaliação, indicando o quanto o item é importante para ele, normalmente sendo informado um valor numérico para essa interação (RICCI et al., 2011).

A partir dessa coleta de informação dos usuários, é construído um perfil de interesse individual, com base nesse perfil, são realizadas técnicas de análise de similaridade entre os usuários ou itens, conhecidas como filtragem colaborativa, e filtragem baseada em conteúdo para gerar recomendações aos usuários.

1.1 Problema de Pesquisa

Neste trabalho, o principal problema a ser investigado é a apresentação de recomendação e extração de vídeos como proposto conforme interações anteriores do usuários com vídeos apresentados. Nesse caso, é utilizado uma proposta para totalizar as interações de usuário sobre os vídeos apresentados aos mesmos e a decisão deles sobre qual é o escolhido entre duas opções oferecidas.

As soluções existentes na literatura abordam essa situação aplicando técnicas como a filtragem colaborativa, visando encontrar k-vértices adjacentes, baseando-se sobre métricas de interações dos usuários com os itens apresentados, procurando seus k-vértices adjacentes para retornar itens com avaliações similares ou realizando previsões de valores baseando em interações de outros usuários, com o usuário-alvo

Considerando o desafio delineado e o cenário atual da literatura específica, este trabalho implementa uma solução que realiza a totalização das avaliações dos usuários conforme a escolha realizada pelo mesmo e realiza recomendações ao usuário.

1.2 Objetivos

Os principais objetivos do trabalho são apresentados a seguir.

1.2.1 Objetivo Geral

Desenvolvimento de algoritmo para recomendação aos usuários baseado em avaliações.

1.2.2 Objetivos Específicos

- Estabelecer score dos vídeos conforme predileção dos usuários;
- Extrair conjuntos de vídeos dos mais gostados únicos
- Baseado nos vídeos, conforme as avaliações, apresentar recomendações considerando os vértices adjacentes mais próximos.

1.3 Justificativa

Tendo como base a utilização de filtragem colaborativa e recomendação baseada em conteúdo a forma de apresentação das recomendações tem limitações que podem ser aprimoradas, nesse caso, temos por exemplo na filtragem colaborativa que utiliza as avaliações dos usuários que possuem o mesmo perfil dos usuários alvo ou utilizando conjuntos de avaliações para aprender modelos para realizar a predição dos modelos.

Nas recomendações baseada em conteúdo, uma grande diferença com a filtragem colaborativa é a recomendação de itens que não tenham tido nenhuma interação com o usuário, podendo assim ser apresentado uma maior quantidade de itens. Sendo assim, como prerrogativa para o trabalho, foi elaborado uma busca para retornar um conjunto de resultados

únicos que retornem conforme o *score* de vídeos escolhidos entre outras opções para o usuário, apresentando vídeos que tenham avaliação com pontuações maiores as escolhida pelo usuário.

Desta maneira, será realizado um estudo exploratório verificando a eficácia do algoritmo desenvolvido, apresentando melhores resultados para recomendação aos usuários. Como maneira alternativa é apresentado outra busca que tem como base a investigação pelos K vértices adjacentes do nó pesquisado pelo usuário final.

A busca dessas recomendações é realizada em cima de um sistema de grafos, podendo assim verificar os relacionamentos entre as arestas conforme a ligação que foi realizada entre as avaliações realizadas pelos usuários, tendo uma maior facilidade para comparar os vídeos que estão relacionados entre si verificando os mesmos pelas adjacências.

1.4 Organização do Trabalho

O restante do trabalho está dividido da seguinte forma: o Capítulo 2 apresenta revisão de literatura necessários para o desenvolvimento da solução proposta neste trabalho; o Capítulo 3 descreve detalhadamente a metodologia experimental empregada para a condução dos experimentos; o Capítulo 4 apresenta e discute os principais resultados obtidos após a realização do(s) experimento(s); e, por fim, as considerações finais, limitações e propostas de trabalhos futuros são apresentadas no Capítulo 5.

2 REVISÃO DA LITERATURA

Este capítulo apresenta os conceitos teóricos necessários para o desenvolvimento da solução proposta neste trabalho. Serão apresentados conceitos sobre as áreas utilizadas nesse trabalho: Sistemas de avaliações de vídeos e teoria de conjuntos.

2.1 SISTEMAS DE RECOMENDAÇÃO DE VÍDEOS

Sistemas de recomendação são utilizados como uma solução para apresentar aos usuários conteúdos certos e com relevância de conteúdo. São mecanismos utilizados para analisar e compreender o comportamento dos usuários de uma plataforma através de seu histórico de utilização do sistema para realizar recomendação entre uma grande gama de conteúdo, contendo sugestões precisas com o intuito de aumento de interação e satisfação do usuário, aplicando técnicas como mineração de dados e aprendizagem de máquina mapeando os interesses de cada usuário do sistema.

O primeiro sistema de recomendação surgiu na década de 90, o *Tapestry* (GOLDBERG et al., 1992), desenvolvido pela *Xerox Palo Alto* com o objetivo de seleção de e-mails. Os usuários criavam regras (filtros) para relacionar suas preferências e os e-mails eram enviados somente para os usuários de acordo com o respectivo filtro selecionado no e-mail. Esse trabalho criou a expressão "filtragem colaborativa", no qual visava a colaboração entre grupos de interessados para realizar a filtragem de preferência dos usuários.

Em comércios eletrônicos a utilização de sistemas de recomendações gera um aumento de vendas e de *ticket* médio, estreitamento de relação cliente-empresa e uma maior lealdade dos seus usuários. Como exemplo de sites que utilizam a sistemas de recomendação como um diferencial no seu negócio temos *Amazon* [1], *Netflix* [2], e *Youtube* [3] (LINDEN; SMITH; YORK, 2003).

Na última década, sistemas de recomendação tem motivado diversos trabalhos e tem evoluído bastante desde o seu surgimento nos anos 1990 e mais recentemente tem sido associados a técnicas de aprendizado de máquina (*machine learning*), tais como aprendizado por transferência, ativo ou por reforço, especialmente com a utilização de aprendizagem profunda (*deep learning*) empregando redes neurais profundas (*Deep Neural Network*) (AZAMBUJA; MORAIS; FILIPE, 2021), tendo como objetivo principal nesse trabalho sistemas de recomendação visando plataformas de vídeos, existem diversos métodos para efetuar essas recomendações com suas vantagens e desvantagens. Como métodos utilizados temos por exemplo: Filtragem Colaborativa (FC), Recomendação Baseada em Conteúdo (RBC) e Sistemas Híbridos.

[1] https://www.amazon.com.br/
[2] https://www.netflix.com/
[3] https://www.youtube.com/

É destacado o surgimento de plataformas que centraliza conteúdos com o temas de SR como Recommender-systems.com[4], Beta-recsys[5] entre outras, visto que teve um grande impacto de investigações e publicações relevantes sobre o tema pelo mundo, principalmente entre a Europa, América, e países asiáticos como China e Índia (AZAMBUJA; MORAIS; FILIPE, 2021).

2.1.1 Filtragem Colaborativa (FC)

A filtragem colaborativa começou a surgir no início de 1990 como uma solução para lidar com uma grande quantidade de e-mails recebido pelos usuários, sendo o *Tapestry* o primeiro sistema a adotar tal método, com uma filtragem colaborativa manual, sendo necessário a criação de buscas em uma linguagem especificamente projetada, o problema ao utilizar essa linguagem, é a necessidade de conhecer pessoa com os gostos semelhantes aos seus para definir a "similaridade" em relação ao usuário. A filtragem colaborativa automática foi introduzida pelo GroupLens (KONSTAN et al., 1997), em 1992, com o objetivo de filtrar *netnews* visando ajudar as pessoas a encontrar artigos que as pessoas iriam gostar.

Sistemas de filtragem colaborativa constitui-se em uma das mais populares técnicas de recomendação, baseando-se no histórico dos usuários e suas indicações no passado para fazer indicações a outros usuários com preferências similares. No contexto de recomendação de vídeos, (DAVIDSON et al., 2010) apresentam um método de recomendação de vídeos no Youtube, utilizando vídeos que o usuário consumiu e/ou indicou como tendo gostado para realizar técnicas de associação para identificar quais vídeos devem ser recomendados.

Apesar de ser uma das utilizadas e possuir diversos benefícios, alguns dos desafios comumente encontrados em FC são (SCHAFER et al., 2007):

- **Privacidade e Segurança**: para prover informações personalizadas para o usuário, sistemas FC necessitam saber informações sobre o usuário, nesse caso, os sistemas mantém dados do usuário para que possa realizar uma melhor predição de preferência. Em arquiteturas CF, são centralizadas em um único repositório que armazena as classificações dos usuários. Caso esse servidor seja comprometido ou corrompido os dados de anonimatos dos usuários serão destruídos. Os usuários necessitaram confiar que o sistema FC vai usar suas preferencias somente para classificações e recomendações.
- **Cold Start**: refere-se a uma quantidade não suficiente de dados (*ratings*) que permitam que recomendações confiáveis possam ser feitas, ou seja não se tem avaliações feitas por usuários. Ao introduzir novos itens em um sistema de recomendação, geralmente não tem avaliações realizadas, portanto, não são recomendados. Por sua vez, um item não recomendado passa despercebido por grande parte dos usuários do sistema e como elas não tem conhecimento sobre o item, ele não irá utilizá-lo ou avaliá-lo
- **Shilling Attacks**: Sistemas baseados em FC podem quebrar a partir de avaliações

[4] https://recommender-systems.com
[5] https://beta-recsys.readthedocs.io/en/latest/

maliciosas realizadas pelos usuários, tendo as avaliações não representado suas verdadeiras preferencias. Tendo também a prática realizada por usuários mal-intencionados que buscam, por interesses próprios, promover determinados itens e/ou dissuadir usuários de consumir outros itens, o fazendo através de suas próprias avaliações.

- **Esparsidade dos dados (Data Sparsity)**: É um conceito que implica na capacidade de um sistema suportar um número cada vez maior de elemento. Na filtragem Colaborativa nós temos graves problemas de escalabilidade quando o volume de usuários, itens e avaliações são muito grandes, pois os sistemas têm quer fazem o cálculo da vizinhança para cada cálculo de predição, isso pode gerar um tempo de resposta inaceitável.
- **Escalabilidade**: Sobrecarga da aplicação ao realizar a comparação para gerar a recomendação ao usuário, visto que sistemas que empregam técnicas de recomendação costumam ter seu banco de dados atualizados continuamente, por via de inserção de novas avaliações por usuários ou atualizações de avaliações pregressas; assumindo-se que um banco de dados pode conter dezenas de milhões de usuários e milhões de itens em seu catálogo, e que este número é em geral crescente, esta não pode ser considerada uma tarefa simples;
- **Sinônimos**: há situações em que um mesmo item, ou itens muito idênticos, têm representações distintas em um mesmo sistema, isto é, são referenciados por identificadores, nomes, diferentes. Esse fenômeno causa uma perda de performance no algoritmo pois as similaridades desses sinônimos com os demais itens do banco de dados podem ser diferentes umas das outras, ou seja, alguns itens podem ou não ser recomendados ao usuário, variando em acordo com qual dos sinônimos o cálculo do ranking da recomendação foi baseado. Abordagens baseadas em conteúdo não apresentam este tipo de problema, visto que suas recomendações são calculadas com base nas características comuns inerentes aos produtos;

Usando o FC, podem ser feitas previsões/recomendações para itens que são desconhecidos para o usuário alvo, mas que são avaliados como relevantes por usuários com preferências semelhantes. Para isso, os algoritmos de FC são baseados em vizinhanças (*neighborhood-based* ou *memory-based*) ou modelos aprendidos (*model-based*).

Algoritmos de FC categorizados como *neighborhood-based* são bastante populares. Nesse modelo, é realizado previsões de valores baseando a identificação dos usuários mais semelhantes a um usuário-alvo (empregando-se alguma medida de similaridade), sendo comum definir um valor K para o tamanho da vizinhança (SARWAR et al., 2001; SCHAFER et al., 2007). O sucesso desse método depende, entre outros fatores, da escolha dos pesos que cada vizinho vai contribuir para a predição das avaliações desconhecidas esse peso é dado a partir de um cálculo de quão similar é o vizinho de um determinado elemento. Uma vez definido o conjunto de vizinhos é calculado uma predição ou recomendação usando a combinação de notas dos vizinhos juntamente com o histórico de avaliações do usuário alvo (SARWAR et al., 2001; SCHAFER et al., 2007).

Os algoritmos de FC categorizados como baseado em modelo, diferentemente dos

baseados em memórias, usam o conjunto de avaliações para aprender um modelo que é usado então para fazer as predições e recomendações. Modelos são entidades que sintetizam o comportamento dos dados. Sistemas baseado em modelos foram criados para resolver os problemas dos algoritmos baseado em memória. Estes sistemas analisam a estrutura da matriz A que relaciona usuários e itens para encontrar relações entre os itens. A ideia por trás dessa estratégia vem da intuição de que o usuário se interessaria por itens similares aos itens bem avaliados por ele e evitar os itens similares aos itens que ele não gostou no passado. Além disso, essa técnica não precisa identificar a vizinhança de usuários similares que presenta o gargalo de desempenho dos algoritmos baseados em memória. Em consequência, tende a produzir recomendações muito mais rapidamente (SARWAR et al., 2001; LINDEN; SMITH; YORK, 2003).

2.1.2 Recomendação Baseada em Conteúdo (RBC)

Sistemas de Recomendação Baseada em Conteúdo (RBC) realizam a recomendação com base em conteúdos já avaliado ou consumido previamente pelo usuário, realizando assim o processo de personalização. RBC algoritmos realiza a recomendação de itens baseados na contagem de similaridade. A melhor comparação é resultado de itens que foram recomendados em comparação com outros itens candidatos previamente avaliado pelo usuário.

Ao realizar a RBC, o conteúdo precisa estar em sua forma original, para assim realizar o pré-processamento dos dados extraídos e limpos, podendo assim identificar com maior precisão do que o item se trata e considerando o grupo de itens avaliados pelo usuário, cria-se um perfil personalizado (RICCI et al., 2011). O fluxo do sistema é realizado através da extração das informações, servindo como entrada para a análise de conteúdo. Sendo realizado a extração dos atributos do texto não estruturado. A próxima etapa é a criação de um perfil único do usuário alvo, onde necessita-se da reação dos usuários para gerar um *feedback*. Com o perfil criado é realizado nova filtragem sobre os itens que não foram avaliados, buscando maior similaridade para recomendar ao usuário (RICCI et al., 2011).

O *feedback* do usuário pode ser coletado implicitamente, quando o usuário clica em um item, o sistema pode considerar essa interação como um item positivo, já que atraiu o usuário; ou explicitamente, nesse caso, o usuário necessita informar sua avaliação sobre o item, em uma escala numérica ou binária (RICCI et al., 2011).

Em sistemas RBC, o tipo de filtragem utilizada é descrito por palavras-chaves. É definida a importância da palavra em um conteúdo de várias maneiras, sendo a medida mais utilizada para realizar essa contagem o valor *Term Frequency-Inverse Document Frequency* (TF-IDF). O valor TF-IDF é uma medida estatística utilizada com o intuito de indicar a importância de uma palavra em um documento. Frequentemente utilizado na recuperação de informação e mineração de dados, com esse algoritmo é realizado a mensuração da importância da importância da palavra no documento. A importância aumenta proporcionalmente ao número de vezes que a palavra aparece no documento, mas é compensado pela frequência da palavra

no corpo do texto.

O *term frequency* (TF) é simplesmente a quantidade de vezes que o termo aparece no documento. Esta contagem geralmente é normalizada para evitar um viés para documentos mais longos, ou seja, pode gerar uma frequência de termo alta independente da real importância do termo no documento. Por sua vez, o termo *inverse document frequency* (IDF) é a medida da importância do termo, ou seja, mede quão comum o termo é em todo o documento analisado (ZHANG; HONG; CRANOR, 2007).

A efetividade na busca por palavra-chave é uma das limitações encontradas e vem sendo bastante estudado, visto que um documento relevante não é retornado na busca por não possuir o termo procurado como palavra-chave e sim um sinônimo. Bem como, a identificação de contexto ao qual o termo está inserido na busca, evitando a ambiguidade da consulta.

Vantagens ao utilizar RBC (THORAT; GOUDAR; BARVE, 2015):

- Fornece ao usuário uma maior independência por meio de classificações exclusivas que são usadas pelo usuário ativo para construir seu próprio perfil;
- Porvê transparência para os usuários explanando como o sistema de recomendação funciona;
- Não é necessário o item já ter sido avaliado para gerar recomendação, possibilitando assim uma maior chance dos itens serem recomendados ao usuário, visto que é dependente somente do perfil do usuário.

Limitações encontradas ao utilizar RBC (THORAT; GOUDAR; BARVE, 2015):

- Dificuldade em gerar atributos para itens em certas áreas, bem como na recomendação para o usuário é necessário que ele tenha avaliado um número suficiente de itens, tornando assim a recomendação mais efetiva;
- Problema de superespecialização, pois defende os mesmos tipos de itens, sendo listado somente ao usuário os itens similares ao seu perfil;
- Dificuldade em adquirir *feedbacks* dos seus usuários já que tipicamente não é realizado classificação sobre os itens, gerando assim a impossibilidade de determinar a recomendação correta;
- Não funcionam bem em domínios que não sejam textuais, como imagens, vídeos e áudios, por ser difícil extrair os atributos relevantes dos mesmos.

2.1.3 Sistemas Híbridos

Sistemas de recomendação híbridos combinam duas ou mais técnicas de recomendação, ganhando assim melhor performance, procurando combinar as vantagens de cada técnica e atenuar suas desvantagens. A abordagem híbrida é introduzida para superar alguns problemas associados aos sistemas de recomendação anteriores, envolvendo principalmente escassez de dados, início frio, esparsidade, super especialização, melhora da precisão e eficiência nos processos de recomendação (THORAT; GOUDAR; BARVE, 2015).

Sistemas híbridos podem ser implementados de várias maneiras, implementando méto-

dos colaborativos e baseados em conteúdo individualmente agregando suas previsões; integrar características baseadas em conteúdo em uma abordagem colaborativa; incluir características colaborativa em uma abordagem baseada em conteúdo; construir modelos consolidativo geral que integre características baseadas em conteúdo e colaborativas. Utilizando esses métodos é possível resolver problemas como início frio ou esparsidade.

Existem diferentes estratégias utilizando filtragem híbridas, as principais no que dizem respeito à forma como os componentes serão combinadas, são (BURKE, 2002):

- **Ponderada**: nessa técnica, a pontuação do item recomendado é calculada a partir dos resultados de todas as técnicas de recomendação presentes no sistema. Como exemplo, é uma combinação lineares realizada com os resultados. Utilização dessa técnica no sistema P-Tango. Benefício de utilização dessa técnica é a utilização de todos os recursos do sistema no processo de recomendação de maneira direta e a atribuição de crédito de performance a eventos não planejados
- **Mista**: nessa técnica as recomendações geradas são combinadas para gerar o resultado, tornando assim a recomendação apresentada ao usuário uma lista dos dados gerados na recomendação colaborativa e baseada em conteúdo.
- **Combinação Sequencial**: nessa técnica é realizada a criação de um perfil de usuário a partir da recomendação baseada em conteúdo, para posteriormente utilizar cálculos da similaridade colaborativa para resultados.
- **Comutação**: nessa técnica é necessário a utilização de um critério para comutar ou chavear a filtragem baseada em conteúdo e a filtragem colaborativa. Como exemplo de utilização, temos o sistema DailyLearner que utiliza o híbrido de conteúdo/colaboração, onde é empregado a recomendação baseada em conteúdo primeiro, caso não tenha uma recomendação confiável será empregado a técnica de recomendação colaborativa.
- **Featured Combination**: nessa técnica diferentes fontes de dados são combinadas e usadas para um único sistema de recomendação. O contribuidor lança recursos de uma fonte para outra de componentes, fazendo assim com que o recomendado fique dependente dos dados modificados pelo contribuinte
- **Featured Augmentation**: essa técnica é semelhante ao Featured Combination nos recursos, porém é mais elástico que o Featured Combination
- **Meta-Level**: O modelo utilizado por um sistema de recomendação é usado de entrada em outro sistema, normalmente recomendações por conteúdo e colaborativa.

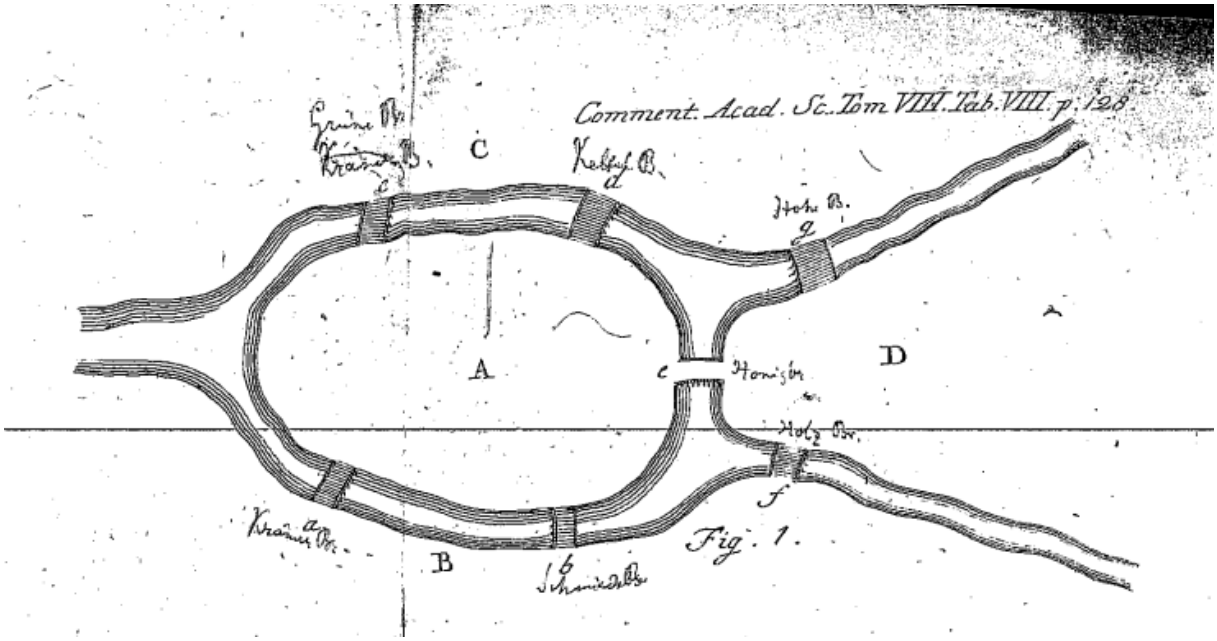
2.2 TEORIA DOS GRAFOS

A teoria dos grafos é um ramo da Matemática Discreta que estuda os objetos denominados grafos. O pioneiro desta teoria foi o matemático suíço Leonhard Euler que formulou e resolveu o problema das pontes de Königsberg (OSTROSKI; MENONCINI, 2009).

Publicada em 1736, a resolução de Euler é baseado na cidade Königsberg (território da Prússia até 1945, atual Kaliningrado (Rússia)), que é cortada pelo Rio Prególia, onde haviam

duas ilhas que, juntas com a parte do continental eram ligadas por sete pontes, conforme mostra a Figura 1. Discutia-se nas ruas da cidade a possibilidade de atravessar todas as pontes sem repetir nenhuma (OSTROSKI; MENONCINI, 2009).

Figura 1 – As pontes de Königsberg



Fonte: (EULER, 1741)

Euler, em 1936, usando de um raciocínio muito simples, mostrou a solução para essa situação: transformou os caminhos em retas e suas intersecções em pontos, criando possivelmente o primeiro grafo da história. Então percebeu que só seria possível atravessar o caminho inteiro passando uma única vez em cada ponte se houvesse no máximo dois potos de onde saía um número ímpar de caminhos. A razão de tal coisa é que de cada ponto deve haver um número par de caminhos, pois será preciso um caminho para "entrar"e outro para "sair". Os dois pontos com caminhos ímpares referem-se ao início e ao final do percurso, pois estes não precisam de um para entrar e um para sair, respectivamente (OSTROSKI; MENONCINI, 2009).

Além de Euler, Gustav Robert Kirchhoff, Arthur Cayley e William Hamilton foram alguns nomes que utilizaram o conceito de grafos para resolver problemas e contribuíram para o desenvolvimento teórico e prático acerca da teoria dos grafos (OSTROSKI; MENONCINI, 2009).

2.2.1 Definição

Um grafo é um par de conjuntos disjuntos $G = (V, A)$ em que V é um conjunto arbitrário e A é um subconjunto de V. Os elementos de V são chamados vértices, nós ou pontos e os de A são chamados de arestas. Duas arestas são ditas adjacentes quando estas possuem

um vértice em comum. Grafos podem ser representados por diagramas, em que cada vértice é representado por um ponto e cada aresta por uma linha ligando os pontos que representam seus extremos (SHANG et al., 2010)..

Um ponto importante é a presença ou ausência de orientação em um grafo, em uma situação dada. Em muitos casos, fica evidente de qual tipo de grafo se trata, enquanto em outros podem aparecer dúvidas; Muitos conceitos só fazem sentido em grafos orientados, o que não acontece com outros. Uma ligação que envolve apenas um vértice é chamado de *loop*. Se as arestas têm uma orientação associada (indicada por uma seta) há um grafo direcionado, ou digrafo. Um grafo com um único vértice e sem arestas é conhecido como o grafo trivial. Um multigrafo é um grafo que apresenta mais de uma aresta entre um mesmo par de vértices, chamada múltipla. Um grafo simples, é um grafo não direcionado, sem laços, onde existe apenas uma aresta entre cada vértice. Um grafo completo é quando todos os vértices estão conexos a todos os outros vértices (SHANG et al., 2010).

Em certas situações, os grafos podem ser utilizados para modelar relacionamentos entre elementos de classes diferentes. Esse tipo de grafo normalmente, é utilizado na representação de SRs e é conhecido como grafo bipartido. Uma classe especial de grafo bipartido é conhecida como rede usuário-objeto baseado em web e desempenha um papel em muitos serviços online que requerem a interação dos usuários (SHANG et al., 2010).

3 METODOLOGIA EXPERIMENTAL

Neste capítulo são descritos o contexto e os métodos de coleta de dados, bem como, todas as etapas da elaboração da solução do problema de pesquisa proposto.

3.1 Obtenção do Conjunto de dados

Os dados utilizados neste trabalho são provenientes de um *dataset* intitulado "YouTube Comedy Slam Preference Data Data Set" (DUA; GRAFF, 2012). Tal conjunto é composto por três colunas, contendo nas duas primeiras colunas os ID dos vídeos a serem escolhidos pelo usuário (colunas left e right) e na última coluna é listado a escolha realizado pelo usuário, tendo a informação carregada com as opções "*left*" ou "*right*". Este *dataset* é disponibilizado publicamente e pode ser acessado pela *University of California Irvine (UCI) Machine Learning Repository*[1] (DUA; GRAFF, 2012).

As informações contidas neste *dataset* são relativas a alguns meses dos anos de 2011 e 2012. Foi realizado um experimento, onde era apresentado pares de vídeos para o usuário e ele escolhia o mais engraçado entre as duas opções apresentadas. Cada vídeo é representado pelo ID do vídeo no *Youtube*, podendo acessar os vídeos informando os ID na URL da plataforma (DUA; GRAFF, 2012). Tal *dataset* foi escolhido para realizar as análises por conta da maneira como é listado os itens, tendo a predileção do usuário, bem como as opções que ele pode escolher

Os dados retornados pelo *dataset* foram carregados para uma tabela criada no banco de dados PostgreSQL [2], um poderoso sistema gerenciador de banco de dados (SGBD) objeto relacional de código aberto, originado em 1986, como parte do projeto POSTGRES da Universidade da Califórnia, em Berkeley, que utiliza e estende a linguagem *Structured Query Language* (SQL) combinada com vários recursos que armazenam e dimensionam com segurança as cargas de trabalho de dados mais complicadas (POSTGRESQL, 2022).

Com mais de 35 anos de desenvolvimento ativo tendo uma reputação por sua arquitetura robusta, confiabilidade, integridade de dados, conjunto robustos de recursos, extensibilidade e dedicação da comunidade de código aberto, por trás do software, para fornecer soluções inovadoras e de desempenho consistentes. Além de ser gratuito, o PostgreSQL é altamente extensível, sendo possível definir o próprio tipo de dado, criar funções personalizadas e escrever código de diferentes linguagens de programação sem recompilar o banco de dados (POSTGRESQL, 2022).

O PostgreSQL está com a última versão estável como a 15, neste trabalho foi utilizado a versão 12.8. A tabela criada na base de dados leva o nome de "youtubeslam" e possui as

[1] https://archive.ics.uci.edu/ml/datasets.php
[2] https://www.postgresql.org/

colunas id, vleft, vright, choice. Nela foi carregado os dados do *dataset* para poder manipular as informações retornadas. Na Tabela 1 é mostrado o esquema da tabela criado.

Tabela 1 – Tabela youtubeslam

Coluna	Tipo	Primary Key	Permite Null?
ID	SERIAL	Sim	Não
VLEFT	VARCHAR(80)	Não	Sim
VRIGHT	VARCHAR(80)	Não	Sim
CHOICE	VARCHAR(80)	Não	Sim

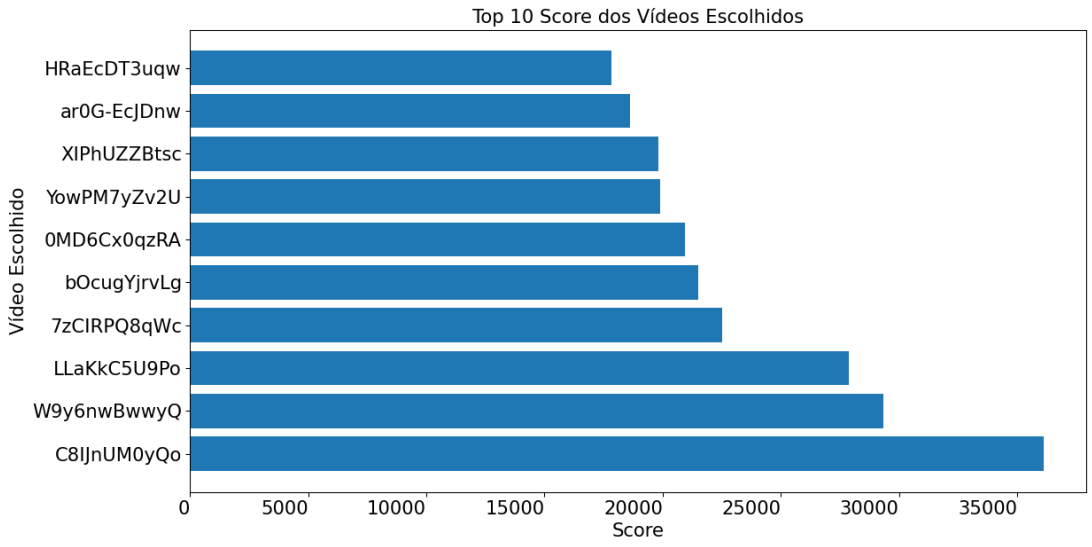
Fonte: Autoria própria.

3.2 Preparação dos dados

Foi realizada a coleta no banco de dados através de um SQL para retornar todos os dados da tabela "youtubeslam". Através desse retorno, foi adicionado os dados em um *dataframe* para realizar a manipulação dos campos. Para a coluna choice, foi alterado no SQL para que fosse validado qual dos dois vídeos apresentados ao usuário foi escolhido, e alterado para listar o nome do vídeo escolhido ao invés da escolha como "*left*" ou "*right*"

A partir dele, foi realizado o agrupamento dos vídeos, realizando a contagem do total de vezes que o vídeo foi escolhido pelos usuários, gerando assim a pontuação, *score*, do vídeo escolhido. Com isso, gerou-se um novo *dataframe* somente com os dados do ID do vídeo e o *score*. Na Figura 2, foi listado o top 10 dos vídeos com o maior número de *score* nas escolhas dos usuários e o ID relativo a ele.

Figura 2 – Top 10 dos scores de vídeos



Fonte: Autoria própria.

3.3 Desenvolvimento da Aplicação

Para o desenvolvimento da aplicação foi utilizado Python[3], uma linguagem de programação de alto nível utilizada para desenvolvimento de aplicações web, *Data Science*, *Machine Learning*, jogos, *scripts* e muito mais. O desenvolvimento da linguagem Python surgiu em 1989, por Guido Van Rossum, enquanto ele trabalhava no Centrum Wiskunde & Informatica (CWI), em Amsterdã. Enquanto trabalhava para a CWI, Rossum tinha como trabalho implementar a linguagem de programação ABC (PYTHON, 2022).

Rossum começou a procurar uma linguagem de *scripts* que tivesse uma sintaxe semelhante ao ABC, mas que tivesse acesso às chamadas de sistema do Amoeba. Após não encontrar nenhuma linguagem com essas especificações, Rossum decidiu projetar uma linguagem simples que pudesse superar as inadequações do ABC. Em 1991, Rossum lançou a versão de abertura da linguagem de programação. Esta primeira versão tinha um sistema de módulo Modula-3, linguagem que foi posteriormente denominada "Python" (PYTHON, 2022).

Python é uma linguagem aberta, disponibilizado pela licença PSF *License Agreement*, atualmente está na versão 3.11.0, e é possível realizar o download de maneira gratuita pelo site oficial da linguagem (PYTHON, 2022). Para o desenvolvimento da aplicação deste trabalho, foi utilizado a versão 3.9.5 do Python. Utilizando essa linguagem de programação, foi realizada a conexão com o banco de dados através da biblioteca psycopg2 que realiza a conexão com o SGBD PostgreSQL sendo necessário repassar o usuário, senha e schema do banco de dados.

Para criar os grafos e realizar processos de leitura, alteração e remoção dos nós e arestas, foi utilizado a biblioteca NetworkX[4], uma biblioteca em Python para criação, manipulação e estudo da estrutura, dinâmica e funções de redes complexas de grafos e redes (NETWORKX, 2022).

Para a criação dos nós e das arestas, foi utilizado o valor dos scores referente aos vídeos. Se o nó A, tem um valor maior que o nó B, vai ser criado a aresta B para o nó A e assim sucessivamente, gerando assim o grafo para a aplicação.

3.4 Soluções Implementadas

Visando a implementação de recomendações ao usuário, nesse trabalho foi feita a aplicação de duas políticas para consultas referente aos dados dos usuários para realizar a recomendação aos mesmos. Para isso, foi implementado no algoritmo a política de extração e a política de recomendação com uma abordagem de filtragem colaborativa que tem como base a avaliação realizada por outros usuários anteriormente.

Nas duas políticas o usuário informa qual o *score* do vídeo a ser consultado. Na primeira política é retornado todos os vídeos que tenham um valor maior que o informado pelo

[3] https://www.python.org/
[4] https://networkx.org/

usuário, e na segunda política, foi ajustado para realizar a busca do vizinho do valor informado, e após isso realizar a consulta neste vizinho encontrado anteriormente.

A busca é feita conforme o nó do grafo é retornado, através da interação do usuário com o algoritmo, retornando os nós que estão relacionados a aresta conforme a política adotada e o valor informado na pesquisa pelo usuário. O retorno para o usuário é a imagem do grafo com os nós e arestas resultantes dessa interação.

No Algoritmo 1 é apresentado como é realizado a busca utilizando a política de Extração para realizar a recomendação. No caso dessa política, temos a remoção dos nós que não são retornados na consulta do grafo buscando os *score* maiores que o valor informado pelo usuário. Para esse primeiro algoritmo, é realizado a inserção dos valores de *score* como nó no grafo, sendo validado se o valor em questão é maior que o valor anterior inserido, caso positivo, o mesmo é ligado com o nó anterior no grafo. Como o *dataframe* utilizado já está com os valores de *score* ordenado de maneira de decrescente, o valor verificado sempre será menos que o anterior.

Algoritmo 1: Algoritmo - Política de Extração

```
Input: Dataset com as escolhas Scores, Nó a ser procurado Valor
Output: Grafo de recomendação Graph
Grafo GraphS;
for u ∈ range(len(Score)) do
    if u == 0 then
        adicionar o nó em Graph;
    else
        if Score[u] <= Score[u-1] then
            adicionar o nó atual ligado com o nó anterior em Graph;
        end
    end
end
Valor ←Valor a ser procurado no Grafo
for u ∈ Graph do
    if u < Valor then
        Remove nó do Graph
    else
end
Retorno de Graph para o usuário
```

Após a criação do grafo com as arestas, é realizado a busca de conjunto único para retorno ao usuário, para isso é necessário que o usuário informe o valor a ser procurado no grafo, após ter esse valor informado, é verificado no grafo os nós que possuem valor de *score* maior ou igual ao informado pelo usuário, retornando assim uma lista com os valores encontrados na busca. Ao gerar essa lista, é removido do grafo todos os nós que não estão listando na lista retornada, gerando assim um grafo com os de valores de *score* dos vídeos válidos de recomendação para o usuário.

No Algoritmo 2 é apresentado a busca utilizando a política de Recomendação para realizar sugestões aos usuários. Nessa política a busca é realizado pelos vizinhos do valor imputado pelo usuário, A busca seguinte é realizada pelos vizinhos do resultado anterior da busca, gerando assim um *array* de *score* com os valores vizinhos informado pelo usuário. Assim como informado no Algoritmo 1 é realizado a criação do grafo, validando se o valor que está sendo verificado do *dataframe* é menor que o valor anterior, e a partir disso é criado o grafo.

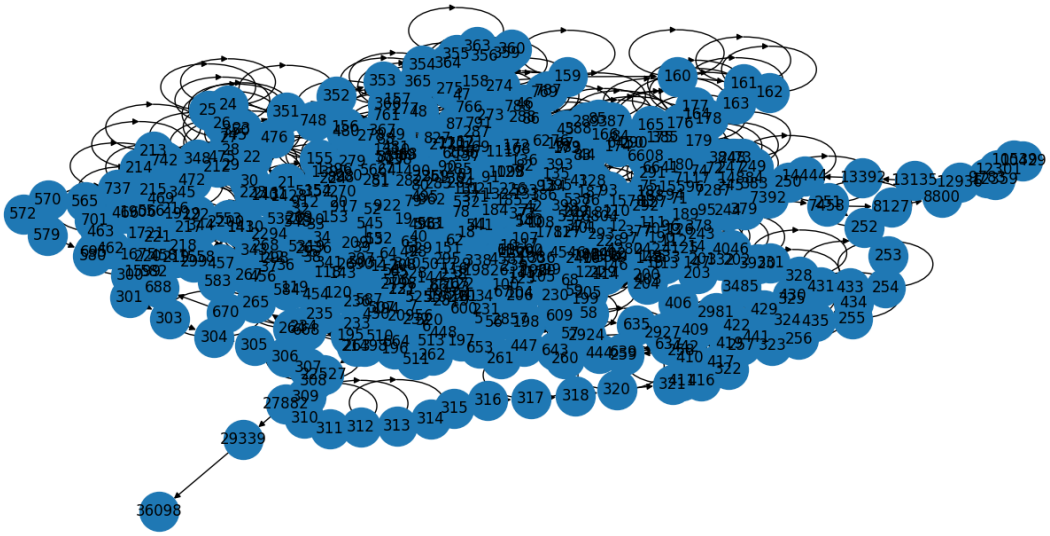
Algoritmo 2: Algoritmo - Política de Recomendação

```
Input: Dataset com as escolhas Scores, Nó a ser procurado Valor
Output: Grafo de recomendação GraphS
Grafo GraphS;
arrayResult ←' []'
for u ∈ range(len(Score)) do
    if u == 0 then
        adicionar o nó em GraphS;
    else
        if Score[u] <= Score[u-1] then
            adicionar o nó atual ligado com o nó anterior em GraphS;
        end
    end
end
Valor ←Valor a ser procurado no Grafo
Vizinhos ←Busca pelos Vizinhos do valor informado
for u ∈ range(len(Vizinhos)) do
    arrayResult vai receber os primeiros resultados da busca de vizinhos, nesse
     caso o array Vizinhos e vai acrescentar os valores de vizinhos retornados da
     busca em Vizinhos
end
GraphS ←Criação de Grafo com o resultado arrayResult
Retorno de GraphS para o usuário
```

Após a criação do grafo, é esperado que o usuário informe o valor de *score* a ser procurado pelo grafo, nesse caso, é realizado a busca de valores dos *K* vértices adjacentes do valor informado pelo usuário. A partir dessa busca, é retornado uma lista com os valores dos *score* adjacentes que tenham o valor maior que o informado pelo usuário. Com essa lista retornada, é realizado uma nova busca pelos *K* vértices adjacentes dos valores contidos na lista retornada na primeira busca. Ao final disso, é criado um novo grafo, com os valores da primeira busca junto com o resultado da segunda busca nos vértices adjacentes.

Na Figura 3 é apresentado o grafo criado para representar as ligações entre os nós e as arestas da aplicação. Para o grafo criado, há 533 nós existentes com as ligações pelas arestas uns com os outros, os mesmos são representados pelos valores de *score* relacionados ao vídeo. O valor que se sobressai como o primeiro valor apresentado é o 36098 que seria o valor mais alto de *score*.

Figura 3 – Grafo criado para a aplicação



Fonte: Autoria própria.

4 RESULTADOS

Este capítulo é apresentado resultados dos experimentos realizados no algoritmo desenvolvido, verificando sua eficácia e conjunto de resposta dos experimentos.

4.1 EXPERIMENTOS

Para o algoritmo desenvolvido foi realizado duas políticas para geração dos experimentos: a de extração e de recomendação.

Para ambas, a criação do grafo de análise seguiu os mesmos protocolos. Utilizando da biblioteca *NetworkX* foi criado o grafo com o comando *nx.DiGraph()*, gerando assim um grafo dirigido, adicionado os nós *add_node()* e realizado a ligação pelas arestas com o comando *add_edge*, onde é repassado os nós que serão realizado a ligação. Através do *dataset* foi realizado a busca pelos score e realizado a ligação entre os nós resultantes dessas avaliações conforme a validação: se o *score* percorrido tem valor menor que o *score* anterior repassado no laço do for, caso retorne verdadeiro, ambos os nós serão ligados, gerando assim o grafo que será utilizado nos experimentos.

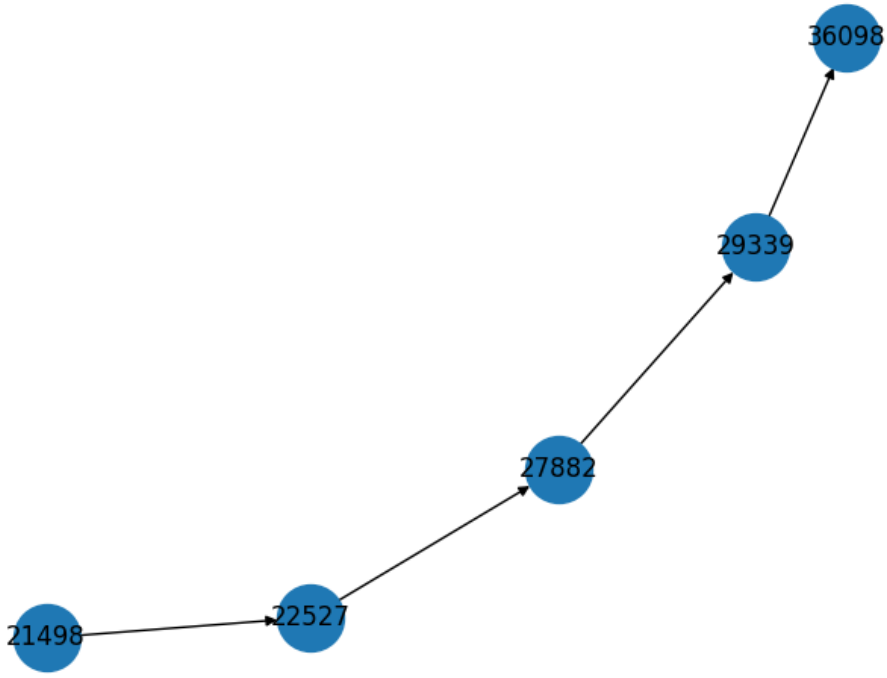
O grafo resultante dessa criação contém 533 nós, ao adicionar o nó no grafo se o valor já existir no grafo ele não é adicionado novamente, sendo assim, mesmo tendo como retorno 17046 resultados do *dataset*, como são adicionados no grafo os valores por *score*, não é adicionado valores repetidos.

4.1.1 Política de Extração

Nesse experimento, o objetivo é encontrar os nós do conjunto de escolha do usuário. A ideia dessa política é realizar a busca e excluir as arestas do nó encontrado, pois os mesmos perderam em comparação com o vídeo utilizado pelo usuário. Para essa política, a realização do experimento ocorreu, solicitando para que o usuário informasse o score a ser filtrado no grafo, listando assim os nós com score maiores ou iguais que estão ligados ao *score* do nó informado. Nos experimentos executados, é realizado a remoção dos nós (*remove_nodes_from()*) que tem um valor menor que o informado pelo usuário, gerando assim um grafo que leva somente os nós onde as avaliações dos vídeos é maior que o valor informado pelo usuário

Como exemplo, no primeiro experimento executado, o valor informado para o algoritmo pelo usuário foi 21498. O algoritmo realizou a busca entre os nós, verificando qual deles possui um valor maior ou igual à 21498, como retorno ele obteve 5 nós com os valores 36098, 29339, 27882, 22527 e 21498. Na Figura 4 é possível verificar o grafo que foi gerado desse experimento. Para essa política os dados a serem retornados são avaliados conforme o *score* apresentado na consulta. O retorno são os vídeos ligados ao nó retornante da consulta tendo um valor maior. O tempo de resposta para o retorno do novo grafo criado com esses valores, foi de 1 segundo.

Figura 4 – Política de Extração - Resultado valor 21498



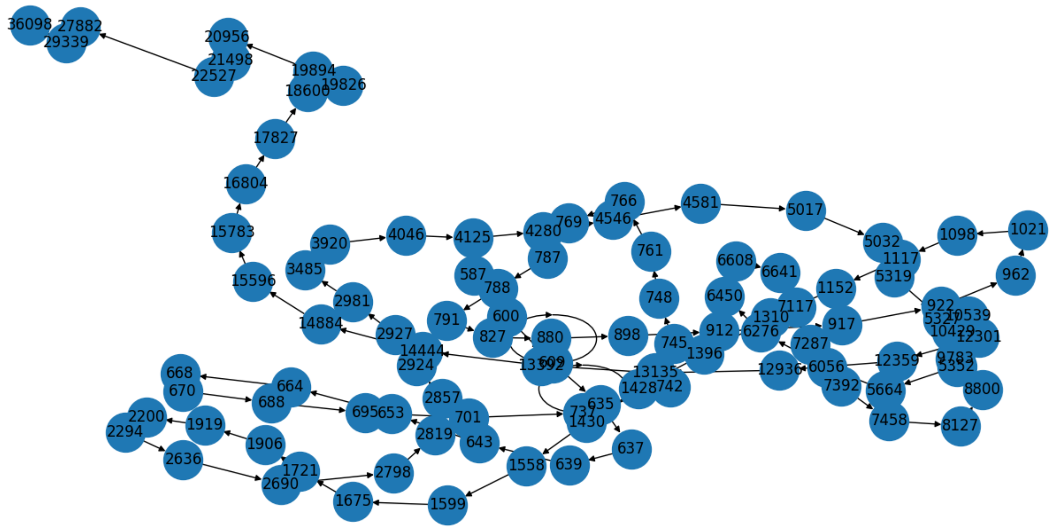
Fonte: Autoria própria.

Ao realizar um segundo experimento, foi repassado para o algoritmo o valor 589, para esse caso, é possível verificar que houve um aumento significativo na quantidade de nós resultantes da pesquisa. Ao realizar a pesquisa entre os nós do grafo que possui *score* maior ou igual ao informado pelo usuário, foi retornada a quantidade de 101 nós ligados pelas arestas com o valor informado, gerando assim um grafo com maior número de nós e ligações entre eles. Na Figura 5 é apresentado o retorno do grafo criado por esse experimento. Conforme o primeiro experimento, o tempo de retorno do algoritmo foi de 1 segundo. Mesmo aumentando a quantidade de valores de resposta o algoritmo se manteve com o mesmo tempo de retorno.

Ao realizar um terceiro experimento, foi repassado para o algoritmo o valor 1, para esse caso, a quantidade de nós resultantes da pesquisa é de 533 nós ligados pelas arestas com o valor informado, gerando assim um grafo com maior número de nós e ligações entre eles. Na Figura 6 é apresentado o retorno do grafo criado por esse experimento. Diferente dos dois primeiros resultados, teve um aumento no momento de gerar um grafo de 2 segundos.

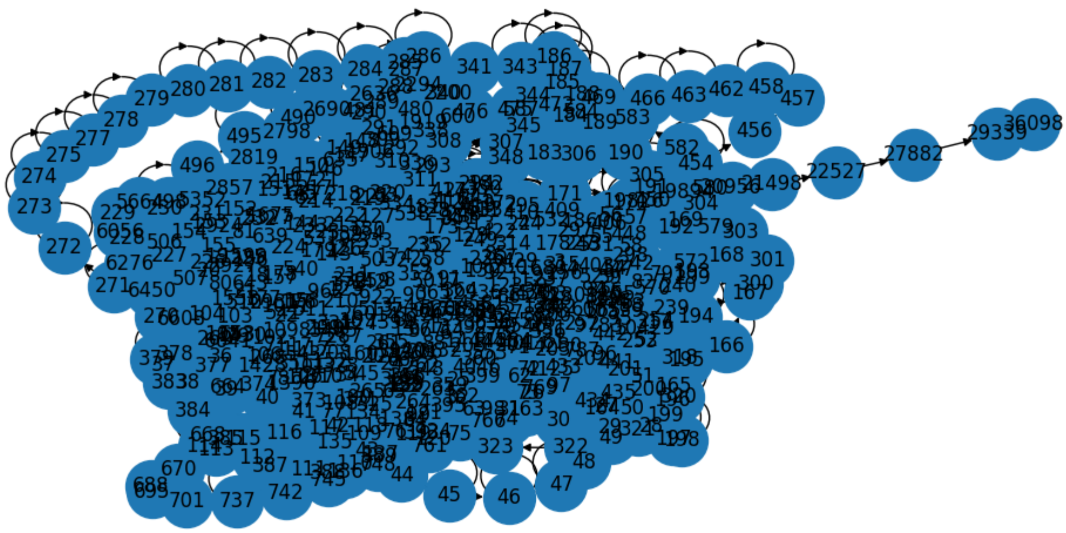
Essa política não tem como base outros estudos, visto que a única informação relevante para o retorno dos dados é o valor de *score* a ser buscado. Nesse caso, podemos verificar que conforme a quantidade de nós existente no grafo, maior será o tempo dado que será utilizado para retorno ao usuário. Levando isso em consideração, e o tempo de retorno dos grafos do experimento, é possível verificar que não tem muita demora, mesmo aumentando a quantidade de valores retornados, visto que se manteve em segundos para a geração. Outro ponto importante dessa política é que não há limitação da quantidade de retorno do novo grafo

Figura 5 – Política de Extração - Resultado valor 589



Fonte: Autoria própria.

Figura 6 – Política de Extração - Resultado valor 1



Fonte: Autoria própria.

criado para a listagem ao usuário.

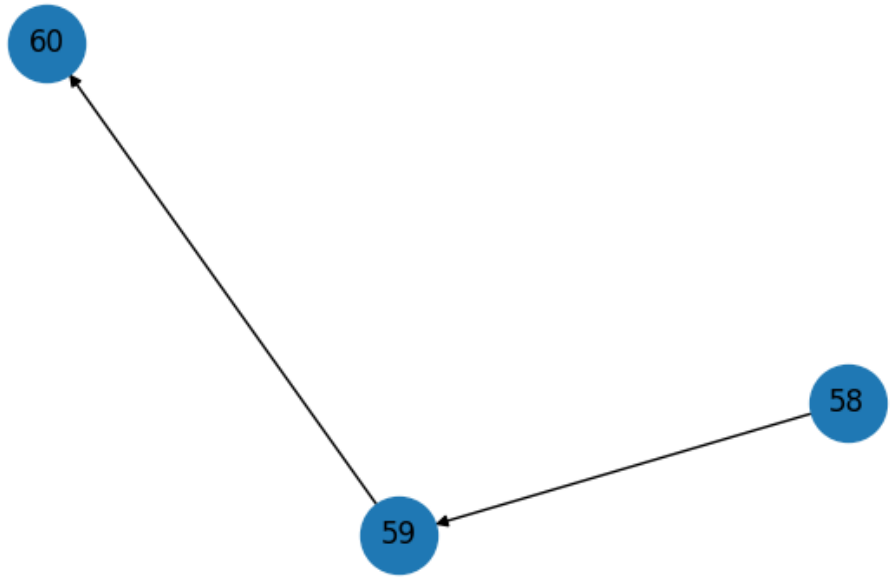
4.1.2 Política de Recomendação

Na política de recomendação, a tática utilizada para a implementação é a utilização da busca pelos vizinhos. Nesse caso, o usuário informa o valor de *score* a ser buscado no grafo e através dele é realizado a busca pelo vizinho através do comando *neighbors* da biblioteca *NetworkX*, repassando o valor informado pelo usuário. Através disso, é retornado os vizinhos

que estão ligados ao nó informado. Com essa informação retornada, é realizado a busca pela vizinhança novamente, porém com a lista retornada anteriormente na primeira busca pelos vizinhos, sendo repassado dessa vez para a função *neighbors*, os valores vizinhos que foram retornados.

Para verificar o resultado dessa política, foi realizado o primeiro experimento informando ao algoritmo o valor numérico como 58, nesse caso, é realizado a busca de vizinhos primeiramente no valor 58, como esse *score* tem como vizinho o valor 59, será ele o retorno. A partir do valor 59, é realizada uma nova busca, verificando assim o vizinho desse valor, tendo como retorno o valor 60. Na Figura 7 é listado grafo com os dados de retorno da busca. Para essa política, o tempo de retorno dos dados é de 1 segundo e tem como retorno 3 nós com os valores de 58, 59 e 60.

Figura 7 – Política de Recomendação - Resultado do valor 58



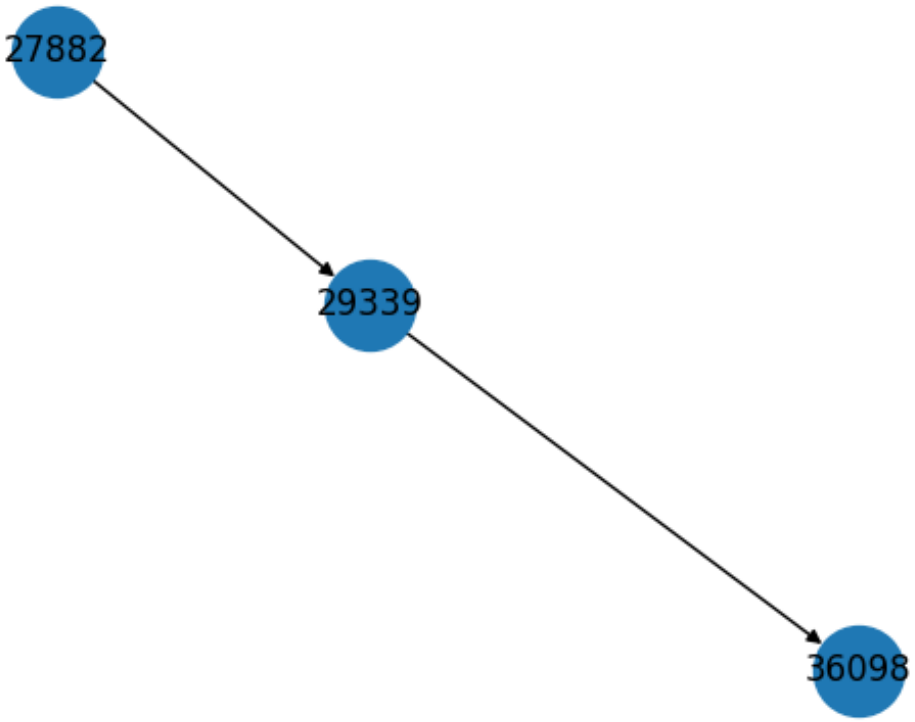
Fonte: Autoria própria.

No segundo experimento realizado nessa política, foi informado para o algoritmo o valor de 27882. Como o valor informado existe no grafo, ele realiza a busca dos nós vizinhos que estão ligados a ele, tendo como retorno um nó com o valor de 29339, após esse retorno, ele realiza a busca novamente pelos vizinhos que estão associados a esse nó, tendo como retorno um único nó novamente, com o valor de 36098. Na Figura 8 é listado grafo com os dados. O tempo de execução se manteve no 1 segundo para gerar a informação para o usuário

No terceiro experimento realizado, foi informado pelo usuário o valor de *score* como 5876, como o nó não existe no grafo, o sistema retorna o erro de "*not found node*" não tendo nenhum resultado gerado para esse valor, visto que ele não existe no grafo. Para retornar dados de recomendação nessa política para o usuário, é necessário que o nó exista no grafo.

Para essa política, ao utilizar o comando *neighbors*, é retornado o valor maior que o informado pelo usuário, já que é realizado a busca do nó sucessor ao informado pelo usuário,

Figura 8 – Política de Recomendação - Resultado do valor 27882



Fonte: Autoria própria.

sendo assim o nó com o score maior. Isso ocorre, pois ao inserir os dados no grafo é realizada a validação se o valor em questão é maior que o valor anterior.

Nessa política, podemos verificar que assemelha-se a filtragem colaborativa, pois é realizada a recomendação com base em avaliações de usuários e por itens, ela ainda se diferencia ao realizar a busca uma segunda vez aos vizinhos do primeiro resultado das buscas de vizinhos, tendo assim, uma diferenciação entre a maneira como é realizada a busca para retorno de recomendação ao usuário final.

5 CONCLUSÃO

Este trabalho apresenta uma nova abordagem na recomendação de vídeos utilizando grafos, baseando-se em avaliações anteriores realizadas por ele: a política de extração e a política de Recomendação, sendo essa última a busca de recomendação em nós vizinhos dos grafos e realizando a busca novamente pela vizinhança desse primeiro vizinho encontrado. Já a primeira política implementada, ela visa mostrar aos usuários todo o conjunto relacionado que seja maior ao valor de *score* pesquisado pelo usuário. Tais implementações diferente das buscas por filtragem colaborativa e recomendação baseada em conteúdo, apresenta uma maior quantidade de recomendação ao usuário, visto que na primeira política não é limitante por usuário, ele será apresentado aos usuários conforme totalização dos itens apresentados e realiza uma busca mais ampla pela vizinhança apresentada, não parando a busca na primeira vizinhança encontrada. Entretanto, a política de recomendação, pode ser melhorada a implementação, buscando apresentar uma análise mais detalhada dos vizinhos a serem listados para o usuário final.

5.1 Limitações

O trabalho tem algumas limitações que podem ser melhoradas em trabalhos. Como principais limitações e soluções propostas, é possível pontuar:

- O *dataset* utilizado na aplicação, ele tem poucas tuplas de retorno ao realizar a contagem por escolha do usuário, sendo assim, fica como sugestão, utilizar um *dataset* que ao realizar o agrupamento dos valores por escolha do usuário
- No caso da política de extração, ao listar todos os nós maiores que o valor informado pode ser que ocorra uma sobrecarga ao usuário de informações recomendadas ao mesmo, ou até mesmo uma demora na geração dos dados. Nesse caso, fica como sugestão realizar uma limitação de quantidade de nós a serem retornados pelo algoritmo.
- Na política de recomendação, ao realizar a busca e não existir o nó em questão, é apresentando para o usuário a informação que não tem o nó no grafo, nesse caso, acredita-se que pode ser alterado para buscar pelo valor mais próximo ao informado pelo usuário.
- Definição de uma métrica para para a avaliação de resultados apresentados pelo algoritmo.
- No caso do modelo de extração, buscar limitar a quantidade de itens a serem retornados para o usuário, visando assim, quando buscado por um valor muito baixo, não retorne toda a estrutura do grafo criado.

Referências

AZAMBUJA, R. X. de; MORAIS, A. J.; FILIPE, V. Teoria e prática em sistemas de recomendação. **Revista de Ciências da Computação**, n. 16, p. 23–46, 2021. Citado 2 vezes nas páginas 14 e 15.

BURKE, R. Hybrid recommender systems: Survey and experiments. **User Modeling and User-Adapted Interaction 12**, p. 331–370, 2002. Citado na página 19.

CHEN, H.; CHIANG, R.; STOREY, V. Business intelligence and analytics: from big data to big impact. **MIS Quarterly**, v. 36, n. 4, p. 1165–1188, 2012. Citado na página 11.

DAVIDSON, J. et al. **The YouTube Video Recommendation System**. 1. ed. Barcelona, Spain: In the Proceedings of the Fourth ACM Conference on Recommender systems - RecSys '10, 2010. Citado na página 15.

DUA, D.; GRAFF, C. **UCI Machine Learning Repository**. 2012. Disponível em: <https://archive.ics.uci.edu/ml/datasets/YouTube+Comedy+Slam+Preference+Data#>. Acesso em: 02 de maio de 2022. Citado na página 22.

EULER, L. Solutio problematis ad geometriam situs pertinentis. **Euler Archive - All Work**, 1741. Citado na página 20.

GOLDBERG, D. et al. Using collaborative filtering to weave an information tapestry. **Communications of the ACM**, v. 35, n. 12, p. 61–70, 1992. Citado na página 14.

KONSTAN, J. A. et al. Grouplens: applying collaborative filtering to usenet news. **Communications of the ACM**, v. 40, n. 3, p. 77–87, 1997. Citado na página 15.

LINDEN, G.; SMITH, B.; YORK, J. Amazon. com recommendations: Item-to-item collaborative filtering. **IEEE Computer Society**, v. 7, n. 1, p. 76–80, 2003. Citado 3 vezes nas páginas 11, 14 e 17.

NETWORKX. **NetworkX - NetworkX documentation**. 2022. Disponível em: <https://networkx.org/>. Acesso em: 29 de outubro de 2022. Citado na página 24.

OSTROSKI, A.; MENONCINI, L. Teoria dos grafos e aplicaÇÕes. **Synergismus scyentifica UTFPR**, 2009. Citado 2 vezes nas páginas 19 e 20.

POSTGRESQL. **What is PostgreSQL?** 2022. Disponível em: <https://www.postgresql.org/about/>. Acesso em: 29 de outubro de 2022. Citado na página 22.

PYTHON. **About Python | Python.org**. 2022. Disponível em: <https://wiki.python.org/moin/BeginnersGuide>. Acesso em: 29 de outubro de 2022. Citado na página 24.

RICCI, F. et al. **Recommender Systems Handbook**. 1. ed. New York, NY: Springer-Science+Business Media, LLC 2011, 2011. Citado 2 vezes nas páginas 11 e 17.

SARWAR, B. et al. Item-based collaborative filtering recommendation algorithms. **Proceedings of the 10th international conference on World Wide Web**, p. 285–295, 2001. Citado 2 vezes nas páginas 16 e 17.

SCHAFER, J. B. et al. Collaborative filtering recommender systems. **P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.): The Adaptive Web, LNCS 4321 Springer-Verlag Berlin Heidelberg**, p. 291 – 324, 2007. Citado 2 vezes nas páginas 15 e 16.

SHANG, M.-S. et al. Empirical analysis of web-based user-object bipartite networks. **EPL (Europhysics Letters)**, v. 90, n. 4, p. 48006, 2010. Citado na página 21.

STATISTA. **Redes sociais mais populares em todo o mundo em janeiro de 2022, classificadas pelo número de usuários ativos mensais**. 2022. Disponível em: <https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/>. Acesso em: 29 de setembro de 2022. Citado na página 11.

THORAT, P. B.; GOUDAR, R.; BARVE, S. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. **International Journal of Computer Applications**, v. 110, n. 4, p. 31–36, 2015. Citado na página 18.

ZHANG, Y.; HONG, J.; CRANOR, L. Cantina: A content-based approach to detecting phishing web sites. **Proceedings of the 10th international conference on World Wide Web**, p. 639–648, 2007. Citado na página 18.