UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Lucca Alexandre Schrammel

FERRAMENTA DE RECOMENDAÇÃO HÍBRIDA DE OBJETOS DE APRENDIZAGEM COM PREDIÇÃO DAS NECESSIDADES PERSONALIZADAS DE CADA ESTUDANTE

Santa Maria, RS
2023

Lucca Alexandre Schrammel

FERRAMENTA DE RECOMENDAÇÃO HÍBRIDA DE OBJETOS DE APRENDIZAGEM COM PREDIÇÃO DAS NECESSIDADES PERSONALIZADAS DE CADA ESTUDANTE

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientadora: Profª. Drª. Lisandra Manzoni Fontoura

Santa Maria, RS

2023

Lucca Alexandre Schrammel

FERRAMENTA DE RECOMENDAÇÃO HÍBRIDA DE OBJETOS DE APRENDIZAGEM COM PREDIÇÃO DAS NECESSIDADES PERSONALIZADAS DE CADA ESTUDANTE

Dissertação apresentada ao curso Mestrado Acadêmico em Ciência da Computação, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do título de **Mestre em Ciência da Computação.**

Aprovada em [dia] de [mês] de [ano].

_____

Nome Completo

_____

Nome Completo

_____

Nome Completo

Santa Maria, RS

2023

À minha família, especialmente aos meus pais, que sempre me apoiaram e permitiram que eu pudesse priorizar meus estudos.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, por dar-me saúde e forças para seguir em busca de meus objetivos.

Aos meus pais, Roberto Schrammel e Jane Schrammel, por sempre permitirem que eu pudesse priorizar meus estudos e manterem uma estrutura familiar em que isso fosse possível.

A minha namorada, Josieli Sarmento da Rosa, que nos momentos mais difíceis e que tudo parecia impossível, me apoiou, me orientou e esteve sempre ao meu lado em todas as etapas desta pesquisa, sem o seu apoio, certamente eu não teria chegado com sucesso ao final deste ciclo.

Agradeço também minha orientadora, Lisandra Manzoni Fontoura, por sempre me orientar e conduzir da melhor forma possível, tornando estes mais de dois anos de parceria da forma mais leve possível. Além disso, agradeço pela oportunidade de aprendizado que me foi dada durante a execução do estudo.

Por fim, agradeço a Universidade Federal de Santa Maria, pelo grande privilégio de fazer parte de uma das instituições de ensino mais importantes do nosso país, bem como pela obtenção do meu título de mestre em ciência da computação.

Grato a todos que de alguma forma me apoiaram e estiveram juntos comigo durante esta jornada.

Obrigado!

RESUMO

FERRAMENTA DE RECOMENDAÇÃO HÍBRIDA DE OBJETOS DE APRENDIZAGEM COM PREDIÇÃO DAS NECESSIDADES PERSONALIZADAS DE CADA ESTUDANTE

Autor: Lucca Alexandre Schrammel

Orientadora: Lisandra Manzoni Fontoura

Com a grande quantidade de dados disponíveis, está cada vez mais difícil identificar informações que contribuirão no processo de aprendizagem dos estudantes. Neste cenário, o presente estudo busca a aplicação de um *framework* de modelo híbrido para recomendação de objetos de aprendizagem com base nas preferências e nas necessidades de cada estudante. Para identificação das necessidades dos estudantes, o *framework* realiza um processo de predição de desempenho para identificar eventuais dificuldades futuras que um estudante poderá apresentar. Dessa forma, as recomendações geradas antecedem uma eventual necessidade de objetos de aprendizagem mais específicos para as necessidades de cada um dos estudantes. As recomendações são geradas a partir de métodos de filtragem colaborativa, filtragem baseada em conteúdo e filtragem baseada em conhecimento. Com o *framework* implementado, a validação dos algoritmos apresentou uma grande contribuição em relação as recomendações geradas, tendo em vista que pôde-se evidenciar que o *framework* foi capaz de integrar métodos de filtragem e algoritmos de predição de dados, para recomendação de objetos de aprendizagem. O *framework* gerou recomendações com precisão acima de 80% em todos os cenários de teste. Além disso, gerou um número baixo de recomendações, o que destaca que, mesmo que consideradas várias possibilidades de recomendações, apenas os objetos de aprendizagem com maior probabilidade de aceitação pelos estudantes foram recomendados.

Palavras-chave: Recomendação. Objetos de Aprendizagem. Predição. Desempenho.

ABSTRACT

HYBRID LEARNING OBJECT RECOMMENDATION TOOL WITH PREDICTION OF EACH STUDENT'S PERSONALIZED NEEDS

AUTHOR: Lucca Alexandre Schrammel

ADVISOR: Lisandra Manzoni Fontoura

With a large amount of data available, it is increasingly difficult to identify information that will contribute to the student's learning process. In this scenario, the present study seeks to apply a hybrid model framework to recommend learning objects based on the preferences and needs of each student. To identify students' needs, the framework carries out a performance prediction process to identify any future difficulties that a student may present. In this way, the recommendations generated precede a possible need for more specific learning objects for the needs of each student. Recommendations are generated from collaborative, content, and knowledge-based filtering methods. With the framework implemented, the validation of the algorithms made a significant contribution to the recommendations generated, considering that it was evident that the framework was capable of integrating filtering methods and data prediction algorithms, for recommending learning objects. The framework generated recommendations with accuracy above 80% in all test scenarios. Furthermore, it generated a low number of recommendations, which highlights that even if several possibilities of proposals were considered, only the learning objects with the highest probability of acceptance by students were recommended.

Palavras-chave: Recommendation. Learning Object. Predict. Performance.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

SRE	Sistema de Recomendação Educacional
OA	Objeto de Aprendizagem
EA	Estilo de Aprendizagem
LMS	*Learning Management System*
FM	*Factorization Machines*
RF	*Random Forests*
AVA	Ambiente Virtual de Aprendizagem
ELARS	*E-Learning Activities Recommender System*
ER	Entidade Relacionamento
UFSM	Universidade Federal de Santa Maria
LOM	*Learning Objects Metadata*
MER	Modelo Entidade Relacionamento
KNN	*K-Nearest Neighbors*
UML	*Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO

No cenário atual, a publicação e visualização de conteúdos está cada vez mais popular e acessível, benefício gerado pela disseminação da Internet e das ferramentas que facilitam a criação e compartilhamento de informações. Entretanto, esse cenário gera uma sobrecarga de informações e os usuários sentem uma dificuldade para identificar aquilo que melhor atende às suas necessidades. Diante disso, os sistemas de recomendação surgem como alternativas cada vez mais viáveis para auxiliar as pessoas a encontrar as informações necessárias para cada uma, conforme o seu grau de interesse e necessidade (SILVA *et al.* 2022).

Agora, aplicando-se ao contexto educacional, os sistemas de recomendação são popularmente conhecidos como uma ferramenta útil para auxiliar os estudantes a encontrar informações de interesse pessoal, o que pode ser utilizado para resolver a sobrecarga de informações que os alunos recebem durante a realização de algum curso ou disciplina específica. Os sistemas de recomendação também podem filtrar aquilo que é de maior interesse do estudante, auxiliando-o a ter uma jornada de aprendizado mais ampla em relação ao que é apresentado em sala de aula (PINHO *et al.* 2019). Deste modo, os sistemas de recomendação podem ser implementados em espaços que permitam o gerenciamento do processo de ensino e aprendizagem, como ambientes virtuais de aprendizagem e ferramentas colaborativas (CAZELLA *et al.* 2010).

Os sistemas de recomendação podem ter diferentes propósitos, como: recomendar planos de estudos; personalizar o currículo dos alunos; recomendar faculdades e universidade (MORAES *et al.* 2018). É possível identificar o uso de sistemas de recomendação em várias situações, tais como: personalizar o processo de ensino-aprendizagem de acordo com o perfil de aprendizagem do aluno (AGUIAR *et al.* 2013), adaptar a experiência do aluno visando o aumento da satisfação (GULZAR *et al* 2018), ajudar estudantes a escolher matérias ou cursos (AGUIAR et al 2019).

Conforme destacado anteriormente, da forma que ocorre um aumento significativo de informações a cada dia, também existe uma quantidade considerável em relação aos recursos educacionais disponíveis para os estudantes (MARQUES 2022). Diante disso, as instituições de ensino estão cada vez mais buscando recursos digitais para

utilização por parte dos estudantes. Essa situação pode gerar uma sobrecarga de dados a serem processados pelos estudantes, o que faz com que os sistemas de recomendação possuam um papel importante no auxílio aos estudantes, tendo em vista que os sistemas de recomendação podem facilmente apoiar a tomada de decisão em relação à metodologia do professor e preencher lacunas na aprendizagem dos alunos, com base na recomendação personalizada e nas necessidades individuais (CAMPOS *et al.*, 2017).

Este trabalho propõe a criação de um *framework* de modelo híbrido para recomendação de objetos de aprendizagem para estudantes matriculados em uma disciplina de algum curso. O foco é fazer com que os estudantes possam receber recomendação de materiais didáticos que fará com que as chances de êxito e aprovação sejam otimizadas, uma vez que o processo de recomendação considerará as preferências e as necessidades do estudante durante o processo de aprendizagem de uma determinada disciplina, antecipando eventuais dificuldades e reforçando assuntos já aprendidos para melhor fixação.

A proposta visa contribuir para a evolução dos sistemas de recomendação na educação, partindo não apenas das preferências do estudante, mas incrementando critérios de necessidades e deficiências dos estudantes para a geração de recomendações mais precisas e personalizadas, com base no perfil de cada estudante. Esta motivação ocorre, pois, segundo estudo realizado por Silva (*et. al* 2022), 50% dos sistemas de recomendação na educação partem de uma abordagem tradicional, ou seja, as recomendações são geradas apenas com base nas preferências dos estudantes, o que pode desconsiderar as reais necessidades de cada estudante.

O texto da pesquisa está organizado da seguinte forma: No Capítulo 2 é apresentada a fundamentação teórica utilizada para basear a pesquisa, apresentando detalhes em relação a métodos de filtragem para recomendação, aplicação de sistemas de recomendação na educação, estrutura de objetos de aprendizagem e conceitos iniciais para a pesquisa. No Capítulo 3 são apresentados os trabalhos relacionados a pesquisa em questão. O Capítulo 4 aborda a metodologia utilizada para estudo, desenvolvimento e validação da pesquisa em questão. O Capítulo 5 apresenta com detalhes a implementação do *framework* proposto pelo estudo,

juntamente com detalhes das classes responsáveis pela geração do *dataset* e pela filtragem e recomendação de objetos de aprendizagem. O Capítulo 6 apresenta todo o processo de validação que foi realizado para verificar a real aplicação do *framework* proposto. Por fim, o Capítulo 7 apresenta a conclusão da pesquisa, juntamente com possibilidades de trabalhos futuros para otimização deste estudo.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos necessários para compreensão de algumas informações essenciais que compõem o presente trabalho, desta forma, é possível identificar os conceitos já existentes no estado da arte e como estes conceitos são referenciados no decorrer da pesquisa em questão.

2.1 SISTEMAS DE RECOMENDAÇÃO

Com a crescente exponencial de informações disponíveis atualmente, muitas vezes há a necessidade de tomada de decisão sem a experiência necessária para definir se a decisão é realmente a mais assertiva. Durante toda nossa existência, contamos com recomendações de outras pessoas para realização de alguma determinada ação no mundo real, porém não são todos os casos nos quais conseguimos uma recomendação humana, tendo como base que existe a necessidade de conhecer uma determinada pessoa para que a mesma faça uma recomendação com base em sua própria experiência ou conhecimento. Para facilitar e agilizar esse processo, um sistema de recomendação pode nos assistir na tomada de decisões baseada na experiência de outras pessoas (RESNICK et al. 1997).

Para Resnick (1997), os sistemas de recomendação têm como objetivo a realização de sugestões personalizadas de itens ainda não avaliados por determinado usuário e que são de potencial interesse para o mesmo. Para que seja possível um sistema de recomendação gerar uma sugestão de item para um usuário, deve ser construído um perfil de interesse individual a partir da avaliação de itens feita por este. Para realização destas recomendações, os sistemas combinam uma série de técnicas computacionais para definir itens personalizados com base nas necessidades de cada usuário, que podem ser em formatos de livros, filmes, produtos, notícias e artigos (Ricci et al. 2010).

De maneira formal, Adomavicius et al. (2005) definem que o problema da recomendação pode ser formulado como sendo *A* o conjunto de todos os usuários de um determinado sistema e *I* o conjunto de itens que podem ser recomendados por tal sistema, considerando a aplicação de uma função de utilidade *u* que mede quão útil o item *i* foi para o usuário que recebeu a recomendação:

u: A x I $\rightarrow$ R, onde R se refere a um conjunto limitado e conhecido.

O sistema de recomendação deve identificar e escolher o item i’ ∈ *I* com maior índice de utilizada para o usuário, conforme representado na Figura 1.

Figura 1 - Fórmula de utilidade de item

$$\forall a \in A, i'_a = \underset{i \in I}{\arg\max}\, u(\mathrm{a}, \mathrm{i})$$

Fonte: Gotardo et al. (2013).

De forma sintetizada, é necessário definir um *score* de avaliação da utilidade da recomendação ao usuário, com o objetivo de verificar se a recomendação realmente teve algum impacto ao usuário.

O primeiro sistema de recomendação conhecido, o *Tapestry* (GOLDBERG *et al*. 1992), refere-se a um modelo de sistema por meio de filtragem colaborativa, partindo do pressuposto que a filtragem de informações apresenta uma maior eficiência quando são envolvidas pessoas no processo de filtragem. Este sistema incluía um processo de filtragem a partir das ações que determinados utilizadores realizavam ao ler uma mensagem.

Atualmente, os sistemas de recomendação são definidos como todo e qualquer sistema que realiza a sugestão de informações que são de interesse ou utilidade para os usuários do sistema, porém para que o sistema tenha o seu devido funcionamento, é necessário entender os problemas que os utilizadores sofrem na coleta de informações, identificando suas necessidades, objetivos e preferências. Dessa forma, é necessário que os sistemas englobem esses requisitos iniciais, independente da área em que o sistema de recomendação é aplicado.

As seções seguintes apresentam as principais tarefas desempenhadas por um sistema de recomendação, bem como os tipos de recomendação existentes, além de como estas recomendações podem ser implementadas no processo de aprendizagem dos estudantes do ensino superior.

2.2 MÉTODOS DE RECOMENDAÇÃO

Conforme visto até o presente momento, sistemas de recomendação são ferramentas que visam sugerir informações para os usuários de forma personalizada,

com base em seus interesses e necessidades, porém nem todos são iguais, existem vários sistemas e com uma gama considerável de formas de geração destas recomendações. Tendo isto como premissa, esta seção apresenta um resumo das técnicas de recomendação mais comuns e utilizadas atualmente.

2.2.1 Filtragem Colaborativa

A filtragem colaborativa é umas das abordagens mais comuns em sistemas de recomendação (BOBADILLA et al. 2013). A ideia desta técnica é recomendar ao usuário itens que outros utilizadores com interesses semelhantes, consideraram como interessante em algum momento do passado.

Para Goldberg et al. (1992), este modelo de recomendação assume a hipótese de que usuários com um modelo de interações no passado terão uma tendência a manter o mesmo comportamento no futuro. Desta forma, é possível gerar um modelo de recomendação de um usuário comparando com as interações históricas de usuários com um perfil semelhante.

No método de filtragem colaborativa, alguns problemas podem ser evidenciados, conforme relatado por Reategui et al. (2005):

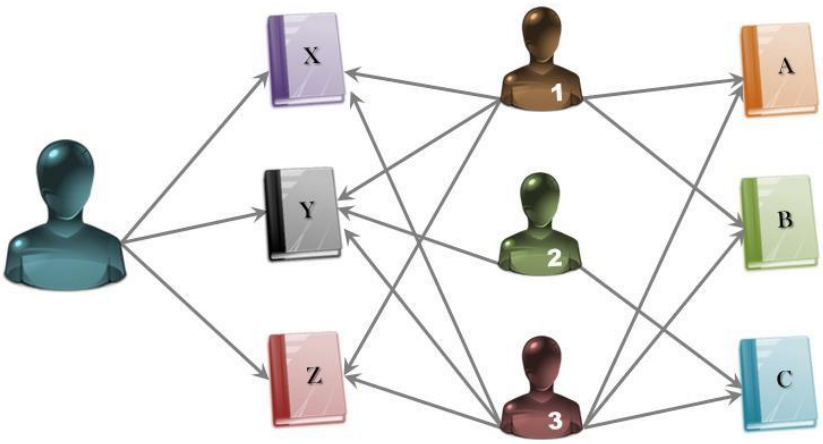
- **Problema do primeiro avaliador**: quando um novo item surge no banco de dados, não há forma deste ser recomendado para o usuário até que sejam coletadas mais informações por meio de outro usuário;
- **Problema de pontuações esparsas**: o objetivo dos sistemas de filtragem colaborativa é ajudar pessoas, focando em documentos lidos ou itens adquiridos. Caso o número de usuários seja pequeno em relação ao volume de informações no sistema existe um grande risco de as pontuações tornarem-se muito esparsas;
- **Similaridade**: caso um usuário tenha gostos que variam do normal este terá dificuldades para encontrar outros usuários com gostos similares, sendo assim suas recomendações podem se tornar pobres.

Este formato de recomendação de descoberta automática da relações entre o usuário e seus “vizinhos mais próximos” consiste em (1) realizar o cálculo da similaridade do usuário em relação aos demais usuários; (2) projetar um conjunto de usuários com maior similaridade para considerar na predição; e (3) compreender as

avaliações e realizar as predições, ponderando as avaliações dos usuários mais similares (CAZELLA et al. 2010).

Na Figura 2 é possível identificar um exemplo de aplicação da filtragem colaborativa. Com base na ilustração, considera-se que o usuário alvo aprovou os livros X, Y e Z e sabe-se que os usuários similares (vizinhos mais próximos) são 1 e 3, que também realizaram a leitura dos três livros. Além disso, ambos usuários também aprovaram os livros A e B, o processo de recomendação realizará a filtragem destes dois livros e recomendará ao usuário.

Figura 2 - Filtragem Colaborativa



Fonte: Costa et al (2013).

2.2.2 Filtragem Baseada em Conteúdo

Segundo Lops et al. (2010), os sistemas de recomendação que são baseados em filtragem de conteúdo assumem como característica predominante a utilização de metadados dos itens combinados com o histórico de interações dos utilizadores.

Esta técnica de recomendação assume a hipótese de que utilizadores possuem uma tendência a consumir itens que possuem características semelhantes ao que o próprio utilizador já consumiu no passado (THORAT et al. 2015). Estas características podem ser definidas em diferentes formatos, como textos e dados mais estruturados.

Este modelo de recomendação determina que as preferências dos usuários são aprendidas com base nas características dos itens que o mesmo classificou ou apenas acessou. Por exemplo, a partir de um artigo que o utilizador classificou como interessante é possível gerar propriedades com base nas palavras-chaves para gerar

futuras recomendações, tendo como entrada estas propriedades do conteúdo consumido pelo utilizador.

Sistemas com este modelo de recomendação apresentam algumas vantagens em relação a filtragem colaborativa, tendo em vista que determinado item não necessita interações prévias por parte dos estudantes para realizar a recomendação, entretanto, é de suma importância de que os itens possuam uma certa quantidade de metadados para que seja possível interpretar os interesses dos usuários de forma assertiva.

2.2.3 Filtragem Baseada em Conhecimento

Este modelo de recomendação busca entregar sugestões ao utilizador com base no conhecimento da necessidade de um determinado item. Alguns autores acabam denominando este modelo de filtragem como “Recomendação Utilitária” (Guttman et al. 1998).

Neste modelo de filtragem, as recomendações são geradas por meio da definição de medidas de utilidade, derivadas a partir do conhecimento que se possui das relações de um item para com algum usuário. A recomendação baseada em conhecimento pressupõe uma estrutura que guarde estas relações e que permita a sua consulta de modo a determinar a utilidade para o utilizador inferindo novas recomendações. O domínio de conhecimento considerado está associado não só às preferências do utilizador, mas também ao tipo de item a recomendar. Por exemplo, um sistema que recomenda viagens baseado em conhecimento pode tirar partido não só do que se conhece acerca da experiência do utilizador em viagens anteriores, mas também do que se sabe sobre as características dos locais que visitou e dos locais disponíveis para recomendar (REIS, 2005).

2.2.4 Filtragem Demográfica

Para Montaner et al. (2003), a filtragem demográfica utiliza a descrição de um indivíduo para aprender o relacionamento entre um item em particular e o tipo de indivíduo que poderia vir a se interessar. Este método de filtragem utiliza as definições das pessoas para conseguir entender o relacionamento entre um item e o tipo de pessoa que teria interesse neste item. O perfil do usuário é gerado a partir da classificação dos usuários em padrões que representam as características de uma classe de usuários. Dados pessoais são solicitados ao usuário, por meio de

formulários de registro, e usados como caracterização dos usuários e seus interesses. Por exemplo, Montaner menciona o método desenvolvido por Krulwich (KRULWICH apud [MONTANER et al. 2003]) em *LifeStyle Finder* utilizando um sistema demográfico denominado PRIZM. Este sistema visa dividir a população americana em 62 grupos demográficos de acordo com seus históricos de compra, características referentes ao tipo de vida e respostas a pesquisas.

2.2.5 Filtragem Híbrida

Combinando uma série de modelos de filtragem para sistemas de recomendação, a filtragem híbrida é um modelo com grande destaque devido aos bons resultados alcançados e a grande capacidade de adaptação (BOBADILLA et al. 2013).

As primeiras propostas de recomendação com a aplicação de mais de uma técnica de filtragem surgiram na década de 90, pouco tempo após a consolidação dos sistemas de recomendação como objeto de pesquisa (BALABANOVIC et al. 1997). Os métodos híbridos mostraram-se com um grande potencial para encontrar as melhores recomendações aos usuários, combinando variadas técnicas em um único modelo.

A filtragem híbrida surgiu com o objetivo de resolver eventuais desvantagens que um único método pode apresentar unindo diferentes técnicas. Dessa forma, as desvantagens de uma técnica podem ser resolvidas por meio das características de outra técnica.

2.3 SISTEMAS DE RECOMENDAÇÃO NA EDUCAÇÃO

A aplicação de tecnologias da informação à educação permitiu uma série de avanços em relação à pesquisa e inovação nesta área. Dentre estes recursos, os sistemas e ferramentas de recomendação apresentam um potencial muito grande para pesquisa e implementação. Inicialmente, os sistemas de recomendação apresentaram uma aplicação muito grande em sistemas de venda, ofertando produtos com base nos padrões de compra e pesquisa de clientes, porém cada vez mais existem estudos aplicados às finalidades educacionais, no contexto remoto ou presencial. Os sistemas de recomendação podem facilmente apoiar a tomada de decisão em relação a metodologia do professor e preencher lacunas na aprendizagem dos alunos, com base na recomendação personalizada a partir de necessidades individuais (CAMPOS et al, 2017).

Para Cazella et al. (2012), os sistemas de recomendação educacionais trabalham como filtros de informações, encaminhando o melhor conteúdo e informação que atende o aluno frente às suas necessidades. Essa filtragem e encaminhamento é possível pelo fato de existir previamente, uma análise de detecção do perfil dos alunos, o que pode ser gerado de forma explícita, por meio da aplicação de questionários, *feedbacks*, ou de forma implícita, como a análise de desempenho e *feedback* do aluno a partir de determinadas atividades ou avaliações aplicadas pelo professor.

O emprego de abordagens de aprendizagem personalizadas por meio de sistemas de recomendação visa recomendar aos utilizadores recursos educacionais apropriados ao seu perfil. Este cenário de recomendação de recursos educacionais é decorrência do modo digital de aprendizagem possibilitado atualmente. Esta oferta personalizada de aprendizado é efeito do uso crescente dos sistemas e-*learning*, que tornaram a coleta de dados mais fácil e possibilitaram fornecer aos aprendizes conteúdos educacionais individualizados (DRACHSLER et al. 2015).

Em relação ao modelo de aplicação de sistemas de recomendação na Educação:

> As abordagens híbridas vêm ganhando atenção e se mostrando consistentes para recomendação de recursos educacionais, devido a sua natureza de criar um método mais robusto e com melhores resultados. Uma vez que, o emprego de filtros isolados apresenta problemáticas, a estratégia de recomendação híbrida visa combinar técnicas para obter um processo que seja a conjunção de filtros; desta forma o resultado esperado é um sistema que combine as vantagens minimizando as desvantagens de cada um dos filtros escolhidos (MORAES, et al. 2018).

2.4 MÉTRICAS PARA AVALIAÇÃO DE SISTEMAS DE RECOMENDAÇÃO

A partir das recomendações geradas, é extremamente importante que seja realizada uma avaliação das mesmas. Esta avaliação gera uma série de dados para realização das calibragens necessárias para aderir o método de recomendação aos objetivos do sistema. Desta forma, o sistema apresentará recomendações que criarão uma experiência positiva ao usuário do sistema.

Para Gunawardana et al. (2009), as métricas para avaliação de sistemas de recomendação são baseadas na teoria dos conjuntos: precisão (*precision)*, revocação (*recall*), *f-measure* e cobertura (*coverage*). Estas métricas podem ser definidas como:

- **Precisão:** é responsável por identificar a quantidade de itens que foram recomendados pelo sistema e que são do interesse do utilizador, comparando com todos os itens que são recomendados;
- **Revocação:** determina a quantidade de itens que são do interesse do utilizador e que fazem parte da lista de itens relevantes;
- **F-*Measure*:** é a média entre a precisão e a revocação;
- **Cobertura:** é a proporção de itens que possuem alguma aptidão para recomendação em relação ao conjunto de todos os itens conhecidos pelo sistema de recomendação.

Cada uma destas métricas pode ser calculada com uma equação matemática, conforme definido por Huang et al. (2012), dadas as seguintes notações:

Dadas as notações, em que:

- R_g: conjunto de itens relevantes;
- R_r: conjunto recomendações;
- $| R_g \cap R_r |$: conjunto de recomendações corretas/relevantes;
- U_p: conjunto de todos os usuários que podem receber alguma recomendação do sistema;
- *U*: conjunto todos usuários presentes no sistema.

A Figura 3 apresenta a equação para execução do cálculo da precisão das recomendações geradas por sistemas.

Figura 3 - Equação para cálculo da precisão

$$precision = \frac{|R_g \cap R_r|}{R_r}$$

Fonte: Huang et al. (2012).

Para o cálculo da revocação, o mesmo pode ser gerado a partir da equação representada na Figura 4.

Figura 4 - Equação para cálculo da revocação

$$recall = \frac{|R_g \cap R_r|}{R_g}$$

Fonte: Huang et al. (2012).

O cálculo de F-*measure* é realizado considerando a precisão calculada anteriormente, conforme ilustrado na Figura 5.

Figura 5 - Equação para cálculo do F-measure

$$F - measure = 2\left(\frac{precision.recall}{precision + recall}\right)$$

Fonte: Huang et al. (2012).

Por fim, a cobertura das recomendações geradas pode ser calculada conforme ilustrado na Figura 6, a qual ilustra a respectiva equação matemática.

Figura 6 - Equação para cálculo da cobertura

$$cobertura = \frac{|U_p|}{U}$$

Fonte: Huang et al. (2012).

A avaliação do sistema de recomendação deve levar em consideração todas as métricas, não utilizando apenas uma como base para geração de dados para avaliação, tendo em vista que uma única métrica pode apresentar uma visão distorcida das recomendações realizadas pelo sistema.

2.5 ESTILOS DE APRENDIZAGEM

A definição mais aceita é a de Keefe (1985), que considera o Estilo de Aprendizagem (EA) como uma composição de características cognitivas, afetivas e fatores fisiológicos que balizam indicadores relativamente estáveis e que determinam como um aluno percebe, interage e responde ao ambiente de aprendizagem.

Já Schmeck (1982) define EA como uma predisposição do aluno em criar suas próprias estratégias para aprender, independentemente da característica da tarefa.

Regularmente os alunos adotam maneiras particulares e autônomas para lidar com situações de aprendizagem.

Segundo Felder e Henriques (1995), existem alunos que preferem aprender por mídias visuais como figuras e desenhos esquemáticos, outros entendem melhor modelos matemáticos e teorias. Existem ainda os alunos que possuem certa preferência em aprender as informações de forma interativa e participativa, já outros indivíduos possuem características individuais e internalizadas para adquirir conhecimentos.

Conforme Salazar (2008), muitos estudos são realizados a partir dos estilos de aprendizagem. Cada indivíduo possui formas preferenciais para assimilar da melhor forma possível um determinado conhecimento.

Ainda segundo Salazar (2008), a aprendizagem é considerada como um processo ativo direcionado para um indivíduo e não voltado para uma massa. Dessa forma, o aluno consegue contextualizar internamente o conhecimento recebido, o que acaba gerando uma interpretação individual e em alguns casos, uma interpretação única.

A diferença encontrada entre alunos durante o processo de aprendizagem e nos resultados alcançados em pesquisas no estado da arte, conduziu ao entendimento da existência de diferentes estilos de aprendizagem que tem influência no aprendizado e de como cada indivíduo recebe e compreende as informações de forma a transformá-la em conhecimento (KOLB, 1984).

2.5.1 Modelo de Felder-Silverman

Conforme visto até o momento, os estilos de aprendizagem buscam compreender melhor indivíduos com base na singularidade, não mais avaliando um conjunto grande de alunos. Com os estilos de aprendizagem personalizados, é possível compreender como é o processo de aprendizagem de cada aluno, porém vários autores abordam conceitos e estilos diferentes, cada um com base em suas pesquisas.

Com a premissa de que existem variadas vertentes relacionadas aos estilos de aprendizagem, a pesquisa em questão faz uso do modelo apresentado por Felder-Silverman para recomendação de objetos de aprendizagem aos estudantes.

Este modelo foi adotado a partir da análise de variadas pesquisas, dentre elas, uma revisão sistemática da literatura, desenvolvida por Nascimento et. al (2017), na qual foram analisados 49 trabalhos relacionados a sistemas de recomendação e o modelo de Felder-Silverman correspondeu a 70% dos trabalhos, ou seja, maior parte das pesquisas relacionadas a sistemas de recomendação fazem uso do modelo de Felder-Silverman para classificação dos estilos de aprendizagem.

Para uma breve compreensão, os autores Richard Felder e Linda Silverman publicaram em 1988 a pesquisa sobre Estilos de Aprendizagem. Segundo eles, a aprendizagem é relacionada a preparação educacional do aluno e a conexão dos seus EA com o estilo de ensino do professor (FELDER; SILVERMAN et al., 1988).

Uma das maiores contribuições de Felder e Silverman para os estudos sobre EA foi a criação de um padrão que explica como um aluno comporta-se e relaciona-se diante da informação. Com base em pesquisas de Kolb (1984), que foca na aprendizagem experimental e em estudos de Jung (1971), que apresenta como as funções psicológicas compreendem e se orientam com o mundo externo, foi criado o Modelo de Estilo de Aprendizagem Felder-Silverman (FELDER; HENRIQUES, 1995).

Em um primeiro momento, o Modelo de Felder-Silverman foi composto por cinco grandes dimensões de EA. Para Felder (1988), cada dimensão continha dois estilos de aprendizagem e que correspondiam às preferencias do aluno para aprender um conhecimento, sendo elas:

- Percepção:
    - Sensorial;
    - Intuitivo.
- Retenção:
    - Visual;
    - Auditivo.
- Organização:
    - Indutivo;
    - Dedutivo.
- Processamento:
    - Ativo;
    - Reflexivo.

- Compreensão:
    - Sequencial;
    - Global.

Após alguns anos, Felder e Henriques (1995) atualizam o modelo de estilos, reduzindo para apenas quatro dimensões, conforme representado na Tabela 1:

Tabela 1 - Modelo de Estilo de Aprendizagem Felder-Silverman

Processamento		Percepção		Entrada		Organização	
Ativo	Reflexivo	Sensorial	Intuitivo	Visual	Verbal	Sequencial	Global

Felder e Henriques (1995) publicaram a pesquisa e concluíram que o processo de aprendizagem possui duas etapas: a recepção e o processamento de informações. O objetivo do modelo dos autores é de identificar as formas de como os alunos preferem receber e processar as informações. O detalhamento de cada uma das categorias definidas pelos autores pode ser evidenciado na sequência:

- Processamento (ativo e reflexivo): determina como as informações são percebidas e convertidas em conhecimento. Os ativos preferem experimentar e participar de forma coletiva. Os reflexivos primam pela reflexão e trabalham individualmente.
- Percepção (Sensorial e Intuitivo): determina como as pessoas percebem o ambiente ao seu redor. Os sensoriais preferem fatos e dados, dando valor a experiência. Os intuitivos são menos atentos e preferem princípios teóricos.
- Entrada (Visual e Verbal): determina como os alunos preferem receber o conhecimento. Os visuais têm boa memória gravando figuras, desenhos, filmes, esquemas práticos. Os verbais preferem receber as informações ditas, escritas ou cantadas.
- Organização (Sequencial e Global): determina a forma que se acompanha um assunto. Os sequenciais aprendem melhor quando o material é apresentado de forma lógica, cronológica e sistemática. Os globais preferem escolher qual a sequência que vai aprender, sem respeitar cronologia ou ordem lógica.

Um ponto importante sobre a aplicação deste modelo, é que o FSLSM é baseado em uma experiência comprovada e baseado em respostas da aplicação do ILSQ, em amostras de diversas turmas de engenharia (FELDER; SILVERMAN et al., 1988). Esse trabalho pôs a prova a coerência entre as dimensões e seus EA vinculados.

2.6 OBJETOS DE APRENDIZAGEM

O conceito de objeto de aprendizagem (OA) não é algo fácil e nem consensual (TORRÃO, 2009). Para Aguiar (2014), a definição surge de acordo com a concepção de cada um dos autores, com base na utilidade e na importância do OA para o ensino e a aprendizagem.

Para Wiley (2022), um OA é qualquer recurso digital que pode ser reusado para apoiar a aprendizagem. Esta breve definição considera as palavras “reusado”, “digital”, “recurso” e “aprendizagem”, conforme determina o comitê de padrão de tecnologia de aprendizagem.

Wiley (2022) apresenta uma metáfora para entendimento do OA, como se fosse um átomo, ou seja, um pequeno elemento que pode combinar-se com outros elementos, a fim de formar algo maior. Cada OA pode constituir um módulo com conteúdo autoexplicativo, que faz sentido individualmente, sem a necessidade de complementos. Da mesma forma que um átomo, um OA não pode ser combinado com qualquer OA, precisam estrar em um mesmo contexto e abranger temas que são relacionados entre si.

Para Tarouco (et. al 2003):

> “Um Objeto de Aprendizagem é qualquer recurso, suplementar ao processo de aprendizagem, que pode ser reusado para apoiar a aprendizagem, termo geralmente aplicado a materiais educacionais projetados e construídos em pequenos conjuntos visando a potencializar o processo de aprendizagem onde o recurso pode ser utilizado.”

Já Koohang e Harman (2007) consideram uma definição um pouco mais abrangente para os objetos de aprendizagem, considerando-os não apenas como recursos digitais, que pode ser reutilizado e customizado para atender os objetivos específicos. Para os autores, é vantajoso tratar o OA como uma orientação instrucional ou como uma série de ferramentas que expandem o repertório de ensino.

Segundo Aguiar (2014), as definições de objetos de aprendizagem coincidem-se em alguns momentos, entretanto, deve ser adotar o conceito adequando de acordo com os objetivos que se pretende alcançar no decorrer do processo de ensino e aprendizagem.

Hodgins (2002) argumenta que os Objetos de Aprendizagem sejam considerados como blocos de LEGO de conteúdo, ou seja, o conteúdo em uma forma mínima que possa ser montado em conjunto com outros conteúdos, construindo um contexto de aprendizagem.

2.6.1 Composição de um Objeto de Aprendizagem

Os objetos de aprendizagem podem ser elaborados em variados formatos, utilizando distintas metodologias, como textos, imagens, animações, simulações, atividades, podendo ser distribuídos pela Internet, porém para Singh (2001), um OA deve ser estruturado e dividido em três partes:

- **Objetivos**: deve deixar claro os objetivos pedagógicos do uso do objeto, além de constatar uma lista de conhecimentos prévios necessários;
- **Conteúdo instrucional**: é a exibição do material didático necessário para que seja interagido com o OA proposto;
- **Prática e *feedback***: permite que o aluno utilize o OA e receba retorno sobre o atendimento dos objetivos propostos.

Para Mendes (2004), os objetos de aprendizagem também devem possuir uma composição específica para que sejam utilizados:

- **Reusabilidade**: deve ser possível utilizar o objeto de aprendizagem diversas vezes e em diferentes contextos de aprendizagem;
- **Adaptabilidade**: deve ser possível adaptar o OA em qualquer ambiente de ensino;
- **Granularidade**: determina o “tamanho” de um objeto. Um objeto de aprendizagem com maior granularidade é considerado pequeno, como uma imagem ou texto. Um OA com baixa granularidade combina vários recursos, como uma página web completa, com textos, imagens, e vídeos, por exemplo;
- **Acessibilidade**: deve ser de fácil acesso;

- **Durabilidade**: sua usabilidade deve permanecer ativa mesmo em trocas de tecnologia;
- **Interoperabilidade**: forma como o objeto opera em diversos cenários, como hardware, *browser* ou sistemas;
- **Metadados**: descrevem as propriedades de um objeto, como: título, autor, assunto, data, complexidade.

2.6.2 Padrão LOM

O padrão *Learning Objects Metadata* (LOM), foi desenvolvido pelo IEEE LTSC Working Group (IEEE/LTSC, 2002). Conforme pesquisa indicada por Nascimento (2017), o padrão LOM é uma das abordagens mais empregadas para sistemas de recomendação na educação. Para este padrão, um objeto de aprendizagem, ou objeto educacional é definido como uma entidade digital ou não digital que pode ser utilizado para aprender, educar ou treinar.

Este padrão busca identificar os objetos de aprendizagem por meio do uso de metadados. Os metadados que descrevem os objetos são definidos por propriedade e valor. Cada OA contém um conjunto de propriedades relacionadas, como: título, data de criação, autor (IEEE/ITSC, 2002).

O padrão LOM possui várias características:

1. Permite a diversidade dos OA e das instâncias de metadados que os descrevem;
2. Separação do modelo semântico e suas ligações;
3. Descrição consistente assegurados pelos valores de metadados;
4. Extensão adaptável para localização (SHEN, 2003).

O padrão LOM não determina como um sistema deve representar os OA, ou seja, não determina se a representação deve ser feita em XML, JSON, texto ou algum outro modelo. A finalidade do padrão é de facilitar o compartilhamento, troca e reusabilidade dos OA (IEEE/ITSC, 2002).

2.6.3 Categorias e Atributos do LOM

A norma determina nove categorias que agrupam inúmeros atributos para que seja realizada a descrição de um objeto de aprendizagem. Cada uma das categorias é independente e caracteriza o objeto por meio de um aspecto distinto. Uma estrutura

de metadados não precisa conter todas as categorias e atributos possíveis, tendo em vista que a utilização é opcional (IEEE/LTSC, 2002).

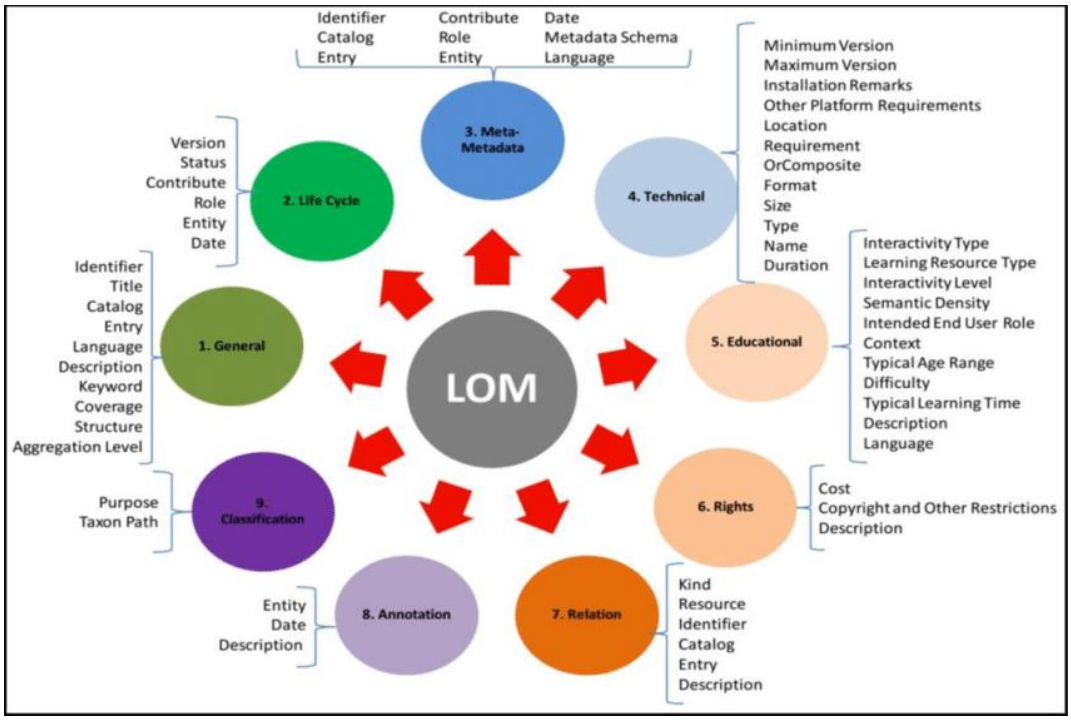
Abaixo são descritas as categorias que fazem parte do esquema básico do padrão LOM, o chamado LOMv1.0:

- **Geral**: agrupa informações gerais que descrevem o OA como um todo. Os metadados dessa categoria são: *identifier, title, catalog entry, catalog, entry, language, description, keywords, coverage, structure,* e *aggregation level;*
- **Ciclo de Vida**: agrupa informações da história e o status atual do OA, além de como ele foi afetado durante o tempo. Os metadados dessa categoria são: *version, status, contribute, role, entity* e *date;*
- **Meta-metadado**: agrupa informações sobre a própria instância do metadado. Seus metadados são: *identifier, catalog, catalog, entry, contribute, role, entitytdate, metadata squema* e *language;*
- **Técnica**: agrupa as necessidades técnicas do OA. Os metadados dessa categoria são: *format, size, location, requirements, type, name, minimum version, maximum version, instalattion remarks, other plataform requirements* e *duration;*
- **Educacional**: agrupa características educacionais do objeto. Seu conjunto de metadados é composto por: *interactivity type, learning resource type, interactivity level, semantic density, intended end user role, context, typical age range, difficulty, typical learning time, description* e *language;*
- **Direitos**: agrupa as informações em relação a propriedade intelectual e condições de uso do OA. Seus metadados são compostos por: *cost, copyright and others restrictions* e *description;*
- **Relação**: agrupa informações que vinculam objetos de aprendizagem. Seus atributos são compostos por: *kind, resource, identifier, description* e *catalog entry;*
- **Anotação**: permite que sejam inseridos comentários acerca do OA em questão. Os metadados que compõem essa categoria são: *person, description* e *date*;

- Classificação: agrupa informações que fazem com que o OA seja classificado. Os atributos possíveis são: *purpose, taxon path, source, taxon, id, entry, description* e *keywords.*

Para otimizar a representação dos metadados propostos pelo padrão LOM, a Figura 7 busca catalogar de forma gráfica, todas as categorias e os atributos possíveis para que um OA seja descrito utilizando esta norma.

Figura 7 - Padrão LOM



Fonte: PÖTTKER et al 2018.

3 TRABALHOS RELACIONADOS

Sistemas de recomendação já são implementados no âmbito educacional, cada um com sua particularidade e formato de recomendação de materiais educacionais. Cada sistema de recomendação estudado ou implementado utiliza uma série de técnicas para realização de uma melhor recomendação no cenário em que se está inserido. Considerando essa premissa, essa seção apresenta alguns trabalhos relacionados a esta pesquisa para embasamento das necessidades de melhorias e geração das lacunas dos sistemas já existentes e que poderiam ser sanadas mediante uma nova abordagem.

3.1 PLORS: A PERSONALIZED LEARNING OBJECT RECOMMENDER SYSTEM

Imran et al. (2015) propõem um Sistema de Recomendação de Objetos de Aprendizagem personalizados, integrado a um Sistema de Gerenciamento de Aprendizagem (LMS) - *Learning Management System*. O sistema de recomendação proposto, chamado de PLORS, possui algumas características para realização das melhores recomendações possíveis aos estudantes, tais como:

- Considera o perfil do estudante como um estilo de aprendizagem personalizado, levando em consideração a especialização do mesmo o tema, o conhecimento prévio e o desempenho atual para personalização das recomendações ao estudante;
- Realiza a associação de estudantes com estilo de aprendizagem semelhante, denominado como vizinhos. Com essa associação, o sistema de recomendação proposto sugere materiais que foram úteis para estudantes vizinhos, levando em consideração que possuem um estilo semelhante e a recomendação de um pode ser útil para outro;
- É gerada uma lista de recomendação possível com base no histórico de navegação dos seus vizinhos;
- O sistema gera automaticamente o entendimento se a recomendação realmente foi precisa, com base nas visitas realizadas a partir de uma recomendação feita.

O PLORS realiza a recomendação de diversos objetos de aprendizagem, tais como vídeos, atividades práticas, aplicações na vida real, e exemplos de aplicações

a partir dos objetos que o próprio usuário visitou ou que outros estudantes com perfis semelhantes visitaram. Para se obter as informações de histórico de utilização e comportamento do usuário, são verificados os logs de acesso e utilização do sistema que contém a plataforma do curso. O sistema foi utilizado dentro de um curso, mas a eficácia das recomendações não foi mencionada.

3.2 NEXT-TERM STUDANT PERFORMANCE PREDICTION

Sweeney et al. (2016) abordam que no ensino superior existe uma questão persistente, relacionada a retenção de alunos para uma graduação bem-sucedida. Segundo estatísticas, a maioria das instituições de ensino superior tem taxas de conclusão de quatro anos por volta de 50%, ou seja, cerca de metade do total de estudantes. Por mais que existam modelos de previsão que esclareçam quais fatores favorecem para o sucesso dos estudantes, a questão da seleção de disciplinas dentro do período acadêmico deve ser melhor estudadas para compreender a relação deste processo com a taxa de conclusão de um curso da graduação. Pensando neste cenário, os autores desenvolvem um Sistema de Recomendação que auxilia o aluno na seleção da grade de disciplinas a serem cursadas em um curso de graduação, a partir do desempenho atual e já conhecido deste mesmo estudante. Os autores buscam técnicas de recomendação já utilizadas em *e-commerce* para análise do progresso do aluno e para embasar as recomendações realizadas, visando um fluxo de aprovação contínua do estudante seguindo uma ordem cronológica de disciplinas viáveis a partir do seu desempenho. Este sistema realiza um processo de simulação das notas dos alunos nas disciplinas em que eles se matricularão no próximo período por meio de padrões de aprendizagem com base nos dados históricos de cada um dos estudantes. Foram exploradas várias técnicas que se mostraram eficazes para a recomendação proposta, porém o melhor resultado foi obtido a partir de uma abordagem híbrida entre o *Factorization Machines* (FM) e *Random Forests* (RF), ou seja, combinando ambos os algoritmos, foi possível gerar um índice de recomendação mais preciso para os utilizadores do sistema de recomendação. Foi realizada a validação com um conjunto de estudantes da Universidade de George Mason.

3.3 SISTEMAS DE RECOMENDAÇÃO PARA AVA MOODLE

Santos (2023) apresenta uma visão na qual alunos e professores possuem uma certa “dor” na busca de materiais didáticos na *Web* ou em plataformas privadas com uma base de conhecimento definida para questões educacionais. Muitas vezes, essa

etapa de busca por informações e materiais didáticos exige um grande esforço por parte do pesquisador. Visando este problema, o autor realizou um estudo referente à fundamentação teórica com base em uma ampla pesquisa bibliográfica buscando fundamentar conceitos importantes em relação a sistemas de recomendação, ambientes virtuais de aprendizagem (AVA) e a Taxonomia de Bloom (Bloom et al. 1956), bem como um levantamento de trabalhos e sistemas já existentes nestas áreas. Após, foi proposto o modelo arquitetural do recomendador. Segundo o autor, o AVA Moodle foi o escolhido principalmente pelo fato de ser utilizado nos cursos LC-EAD, que é exatamente o provedor dos dados necessários para a proposta. Em seguida, foram selecionadas as tecnologias adotadas para a implementação da API REST, que possui o fundamental papel de gerar as recomendações, e por último partiu-se para o desenvolvimento e análise dos resultados dos experimentos realizados. Em resumo, este trabalho propõe solucionar o problema da recomendação de atividades aos alunos do Moodle, através do uso de um conjunto de dados educacionais, tendo como finalidade que essas recomendações possam ajudar o usuário a sempre encontrar questões que possibilitem ajudar a maximizar o seu aprendizado, sempre considerando que tais questões estão no seu nível cognitivo da taxonomia revisada de Bloom.

O sistema proposto por Santos trata-se da evolução de um trabalho proposto em 2018, na qual o trabalho passou a considerar as questões que o aluno errou para realização da análise e construção das recomendações. O sistema realiza a coleta de informações de acertos e erros em questões para geração de uma análise do perfil do estudante e então a recomendação é gerada. A recomendação é criada a partir da filtragem colaborativa que no caso desse trabalho é baseada no usuário, que internamente usa o Coeficiente de Tanimoto para o cálculo das similaridades entre os alunos através de intersecções e em seguida calcula a vizinhança usando *threshold*. O algoritmo consome as informações definidas no passo anterior e realiza o processamento. Como resultado do processamento realizado pelo algoritmo, há uma resposta estruturada como um *array* contendo as questões a serem recomendadas em ordem de prioridade para o usuário alvo em relação a todos os usuários selecionados na vizinhança.

3.4 RECOMMENDER SYSTEM AND WEB 2.0 TOOLS TO ENHANCE A BLENDED LEARNING MODEL

Hoic-Bozic et al. (2016), sugere que modelos de aprendizagem mistos que combinam aprendizagem presencial e online apresentam uma grande relevância no ensino superior atual. No entanto, seu desenvolvimento deve ser paralelizado com as atuais mudanças no *e-learning*, que enfatizam uma abordagem personalizada para o aluno e com base em ferramentas disponíveis na *web* para apoiar o processo de aprendizagem de cada um dos estudantes, não apenas de uma forma geral (Como turma). Visando este cenário, os autores efetuaram uma pesquisa sobre a implementação de um modelo de *Blended Learning* no e-curso "*Hypermedia Supported Education*". O modelo *Blended* desenvolvido une um sistema de gestão de aprendizagem, ferramentas *Web* e o *E-Learning Activities Recommender System* (ELARS) para melhorar a aprendizagem online personalizada.

O sistema ELARS é um sistema de recomendação que permite a personalização de atividades virtuais, com a recomendação de quatro segmentos de itens:

- Atividades eletivas;
- Colaboradores em potencial (Colegas de classe especialistas);
- Ferramentas *Web;*
- Conselhos.

Estas recomendações são geradas para cada estudante ou para um grupo de estudantes com características semelhantes, com base em seus dados pessoais, evolução no curso, preferências, estilo de aprendizagem, nível de conhecimento e o nível das atividades executadas. Para Hoic-Bozic et al. (2016):

> As atividades do curso são agrupadas em módulos de aprendizagem e classificadas em seis categorias. Assim como e-atividades (eLA), os módulos podem conter atividades de aprendizagem de conteúdo (CA) e de teste (TA) realizadas no LMS e atividades realizadas no ELARS. Isso inclui atividades de apoio (SA) para entrega de instruções, resultados de questionários e afins, e atividades de decisão (DA) nas quais os alunos escolhem entre os itens recomendados.

Foram implementadas as técnicas de recomendação baseadas em conhecimento, baseadas em conteúdo e/ou filtragem colaborativa, adaptados ao domínio educacional, ou seja, as recomendações e filtragens incluem regras pedagógicas, que diferem um pouco em relação ao modelo de filtragem utilizado na indústria. No sistema ELARS, os professores possuem autonomia para modificar estas regras pedagógicas, que podemos chamar de regras de negócio para o ramo educacional, de acordo com as estratégias de cada professor ou com base nas características dos estudantes atuais.

O modelo implementado pelos autores foi separado em fases, ou seja, cada fase da implementação do sistema segue uma série de itens para serem cumpridos, conforme descrito abaixo:

- Fase 1: Nesta fase o aprendizado presencial foi combinado com atividades virtuais suportadas pelo LMS.
- Fase 2: Introdução de atividades colaborativas de *e-learning* e ferramentas *web*. Foram selecionadas as seguintes ferramentas da *web*: Blogger, Diigo, Flickr, Google+, Google Drive, MindMeister, SlideShare, Wikispaces e YouTube.
- Fase 3: Enriquecimento com atividades eletivas para execução conforme preferência dos estudantes do curso selecionado.

Para validação do modelo, estudantes do programa de pós-graduação do curso de Ciência de Computação participaram de algumas das fases descritas acima. Um grupo inicial realizou atividades conforme a obrigatoriedade definida pelo professor, sem o uso do sistema ELARS para recomendação de outras atividades e ferramentas que poderiam auxiliar estes estudantes. Um segundo grupo realizou as atividades obrigatórias e também contou com as recomendações do sistema de recomendação ELARS.

A partir da aplicação deste modelo, utilizando um grupo de controle e um grupo com aplicação do sistema de recomendação. A avaliação foi feita a partir da análise de desempenho dos estudantes de ambos os grupos e conforme ilustrado na Figura 8, os testes revelaram uma diferença entre as medianas dos resultados finais com uma diferença significativamente, portanto, a hipótese foi aceite e pode-se concluir que os alunos do grupo experimental que realizaram as atividades propostas pelo

sistema ELARS obtiveram resultados finais significativamente melhores, em comparação ao grupo de controle.

Figura 8 - Resultados experimentais de Hoic-Bozic

	Control group	Experimental group
N	16	21
Minimum	55	79.5
Maximum	96	96
Mean	78.6	89.2
Median	**79**	**89.5**
Standard deviation	8.8	4.5
Inter quartile range	8.5	7.38
Coefficient of quartile deviation	0.05	0.04

Fonte: Hoic-Bozic et al. (2016).

Conforme relatado pelos autores:

> Os pontos ganhos pelos alunos do grupo experimental indicam um alto nível de obtenção de resultados de aprendizagem; o resultado mínimo alcançado, mostra que os alunos mais fracos se engajaram mais. Não só os alunos do grupo experimental obtiveram resultados finais significativamente melhores, mas também resultados significativamente melhores por atividade, o que corrobora as hipóteses apesar da mudança no número de atividades incluídas na análise estatística. (“A discussão do fórum” foi substituída por “Mapeamento mental” e “E-atividade opcional 1”.) No entanto, o desenho experimental do estudo e o tamanho da amostra não são grandes o suficiente para reivindicar um grande efeito para o modelo combinado proposto.

O modelo apresentado pelos autores busca o desenvolvimento da aprendizagem combinada com recursos atuais e de conhecimento dos estudantes. Os resultados do estudo confirmam a eficácia do modelo proposto em um cenário real, juntamente com a satisfação dos estudantes com o modelo proposto. Além disto, os autores corroboram que é um modelo especificamente indicado para cursos de domínio da informática, uma vez que os alunos são imergidos a novas tecnologias e a maior parte dos estudantes já está habituada com algumas etapas do processo, porém sem um sistema realizando a etapa de análise e recomendação de materiais personalizados.

3.5 UMA ABORDAGEM PARA RECOMENDAÇÃO AUTOMÁTICA E DINÂMICA DE OBJETOS DE APRENDIZAGEM BASEADA EM ESTILOS DE APRENDIZAGEM

O trabalho proposto por Carvalho et. al (2014), traz a visão de que estudantes apresentam um maior aproveitamento quando são inseridos conteúdos personalizados conforme a necessidade de cada aluno. Com base nessa premissa, pode-se verificar que os estilos de aprendizagem devem ser levados em consideração no momento de realizar a recomendação de objetos de aprendizagem para os estudantes, entretanto, essa abordagem acarreta em um novo problema, o relacionamento entre objetos de aprendizagem x estilos de aprendizagem.

Este relacionamento é um ponto chave abordado pelos autores, onde foi construído um protótipo que busca vincular estas duas entidades conhecidas. Com o uso de padrões já estabelecidos, os autores buscaram realizar o vínculo com base no padrão LOM para organização dos objetos de aprendizagem com o modelo de Felder-Silvermann, que aborda os diferentes estilos de aprendizagem.

O objetivo do trabalho é gerar uma pontuação que aproxima ou distancia os objetos de aprendizagem para cada estilo de aprendizagem conhecido. Esta distância é medida com base na análise de uma série de metadados que cada objeto de aprendizagem possuí, dessa forma, é gerada uma lista ordenada com os objetos mais próximos de cada um dos estilos de aprendizagem.

Para realização da validação, foram criados cenários para evidenciar as recomendações realizadas considerando determinados estilos de aprendizagem, sem a necessidade de simular em um ambiente de ensino.

3.6 *OVERVIEW* DOS TRABALHOS RELACIONADOS

Conforme evidenciado na análise dos trabalhos relacionados, pode-se notar que existem trabalhos que buscam transformar a recomendação de materiais didáticos cada vez mais assertiva, cada um com um objetivo específico.

Pode-se verificar que o PLORS, realiza uma análise do perfil do estudante, seu desempenho e as necessidades para realização de simulação do desempenho do estudante e então recomendar quais as disciplinas ideais que o estudante poderia cursar no período seguinte, um modelo semelhante ao presente trabalho, porém o objetivo deste estudo é de realizar a análise de desempenho do estudante e então

recomendar materiais de aprendizagem dentro do período em que o mesmo se encontra, buscando o aumento das chances de aprovação do estudante.

O sistema de recomendação proposto por Carvalho et. al (2014), realiza a recomendação de OA com vinculo no EA de cada estudante, deixando de considerar outros fatores, como o desempenho de cada estudante com perfis semelhantes, entretanto, o trabalho em questão realiza uma validação baseada em cenários, fazendo com que seja possível evidenciar a eficácia das recomendações para cada um dos EA conhecidos, sem que seja necessário realmente implementar em uma sala de aula para coleta de dados e avaliação do algoritmo.

A Tabela 2 apresenta uma comparação entre os trabalhos relacionados, a fim de evidenciar as diferenças de cada um deles e confirmar, a não utilização do desempenho previsto dos estudantes para geração de recomendações de objetos de aprendizagem.

Tabela 2 - Comparação dos trabalhos relacionados

	Imran *et al.* (2016)	Sweeney *et al.* (2016)	Santos (2020)	Hoic-Bozic *et al.* (2016)	Carvalho *et. al* (2014)
Recomendação de OA.	Sim	Não	Sim	Sim	Sim
Considera o perfil do aluno	Sim	Sim	Sim	Sim	Sim
Origem dos dados	Logs de acesso ao sistema do curso	Desempenho conhecido de cada aluno	Erros e acertos em atividades do Moodle	Sistema de gestão educacional	Sistema de gestão educacional
Recomendação de disciplinas da grade curricular	Não	Sim	Não	Não	Não
Previsão de desempenho do aluno.	Não	Sim	Não	Não	Não

Utiliza algoritmos de *machine learning*	Não	Sim	Não	Não	Não
Taxonomia de bloom	Não	Não	Sim	Não	Não
Utiliza métodos de filtragem por conhecimento ou colaborativa ou conteúdo	Não	Não	Não	Sim	Não

Fonte: Elaborado pelo autor.

Demais sistemas de recomendação no contexto educacional realizam uma análise sobre o perfil dos estudantes e buscam a realização de recomendações baseando nos interesses de cada aluno, e então criam uma recomendação personalizada aos interesses destes estudantes.

O presente trabalho apresenta como diferenciais o fato que o processo de recomendação parte das preferências do estudante e leva em consideração as deficiências no aprendizado que cada aluno tem encontrado no decorrer da execução de uma determinada disciplina. Com base nessas deficiências de cada estudante, é criado um fluxo de recomendação vinculado ao plano de ensino do professor. Desta forma, as recomendações são diretamente atreladas aos objetivos que o estudante possui em cada uma das disciplinas. Acredita-se que com as recomendações devidas o estudante pode otimizar o seu desempenho em tempo hábil para uma aprovação no período letivo atual, para então, realizar uma evolução ao natural.

4 METODOLOGIA

Conforme abordado até o momento, o presente trabalho tem como objetivo a implementação de um *framework* capaz de realizar recomendação de objetos de aprendizagem para estudantes de uma forma geral, não apenas levando em consideração suas preferências e características, mas também as suas dificuldades de aprendizagem. Para que isto seja possível, foi necessário implementar uma série de algoritmos que possibilitam a realização de uma predição do estudante e a partir disto gerar recomendações assertivas.

Com estas recomendações, o objetivo é de que o estudante tenha suas chances de aprovação otimizadas, tendo em vista que o mesmo contará com uma gama de objetos de aprendizagem que permitirão um melhor desempenho dentro de um semestre letivo.

O capítulo em questão traz informações da metodologia aplicada para realização dos estudos necessários, da implementação do *framework* e da validação das recomendações geradas.

4.1 REVISÃO DA LITERATURA

Para avaliação dos recursos existentes para sistemas de recomendação, foi realizada uma pesquisa em diversas bases de conhecimento, justamente com o objetivo de identificar as funcionalidades existentes em sistemas de recomendação, bem como estas técnicas são aplicadas no âmbito educacional.

Nesta etapa, foi realizada uma revisão da literatura visando a busca por aplicações de sistemas de recomendação na educação e como estes podem ser integrados com um modelo preditivo para gerar dados necessários para recomendar objetos de aprendizagem com base nas necessidades de cada estudante.

Além da pesquisa relacionada a sistemas de recomendação, foi realizada uma revisão da literatura para identificar os principais conceitos que estão em volta de sistemas de recomendação, como padrões de aprendizagem, objetos de aprendizagem, organização de objetos de aprendizagem, formas de filtragem e métodos de predição.

A partir da execução desta etapa, foi possível destacar pontos importantes que são utilizados posteriormente para a implementação do *framework* proposto.

A busca por pesquisas necessárias foi realizada a partir de uma *string* de busca base, utilizando palavras chaves específicas para o objetivo do trabalho em questão. Com a busca inicial, informações foram coletadas nos trabalhos identificados, juntamente com os trabalhos que foram citados, em um formato de rede.

4.2 DEFINIÇÃO DA PROPOSTA

A partir do estudo teórico realizado, foi definida a proposta de estudo do trabalho em questão. Foi necessário avaliar como os sistemas de recomendação operam e como os mesmos podem ser aplicados no contexto educacional. Com essa visão, foi possível identificar uma necessidade que pode auxiliar os estudantes no desempenho escolar em uma determinada disciplina em que o mesmo se encontra matriculado.

Conforme abordado nos capítulos anteriores, pode-se observar que sistemas de recomendação são aplicados no contexto educacional com a recomendação de materiais didáticos que são do interesse e das necessidades dos estudantes, porém avaliando esse cenário, não foram evidenciados sistemas de recomendação que levam em consideração o perfil do estudante e um provável desempenho que possa ser otimizado mediante a recomendação de objetos de aprendizagem, avaliando não somente o cenário atual, mas um cenário futuro que pode ocorrer com base nos padrões de aprendizagem do estudante. Além disso, o estudante pode acabar tendo um desempenho inicial abaixo do esperado, logo, as recomendações futuras devem ser as mais otimizadas possíveis para que o aluno possa recuperar o seu desempenho em tempo hábil para uma aprovação.

Para mitigar este problema, o presente estudo propõe a aplicação de um sistema de recomendação que é capaz de identificar as dificuldades de aprendizagem dos estudantes, prever o seu desempenho futuro e então realizar uma série de recomendações que farão com que as chances de aprovação do estudante sejam melhoradas, não apenas respeitando um critério de interesse, mas um critério de necessidade. Este modelo de recomendação parte da análise de dados conhecidos para estabelecer a relação do estudante com outros estudantes com perfil semelhante e que já superaram uma disciplina em questão. Estes dados históricos são relevantes para prever o desempenho do estudante e então tomar uma ação de recomendação

de objetos de aprendizagem para buscar mitigar eventuais dificuldades no aprendizado.

4.3 MODELAGEM DO SISTEMA DE RECOMENDAÇÃO

Com a definição da proposta, foi realizada a modelagem dos componentes que são necessários para que o *framework* atue conforme o planejado e o destacado no decorrer da proposta. A modelagem foi realizada para facilitar a compreensão de como o algoritmo processará as informações e gerará as recomendações aos estudantes. Além disso, é possível evidenciar as entradas, o processamento e as saídas de dados que serão geradas por parte do algoritmo.

Alguns dos componentes mapeados foram os seguintes:

- Filtragem por conteúdo;
- Filtragem colaborativa;
- Filtragem por conhecimento;
- Predição do desempenho;

Com o mapeamento destes componentes, a modelagem inicial partiu do diagrama Entidade Relacionamento (ER) do Banco de Dados. Essa modelagem permitiu que fosse possível detectar mudanças nos componentes e como os dados seriam persistidos para que posteriormente seja realizado todo o processo de recomendação de materiais didáticos com base no perfil e nas necessidades do estudante. Para modelagem do modelo ER foi utilizado o sistema MySQL Workbench, que possui uma série de recursos que permite que seja gerado toda a base de dados a partir da modelagem já realizada.

Com a definição do banco de dados realizada, foi utilizado o sistema Draw.io para geração de demais diagramas que fazem parte da modelagem do sistema, como diagramas de classe, diagramas de atividades e caso de uso, conforme pode ser visto com detalhamento na seção de desenvolvimento.

4.4 FUNCIONAMENTO DO *FRAMEWORK*

Esta seção apresenta uma breve explicação em relação ao funcionamento do algoritmo. Nesta seção são apresentadas as etapas e como os dados são processados em alto nível. Um detalhamento sobre o *framework* pode ser encontrado no capítulo de desenvolvimento.

O *framework* conta com diversas classes recursivas que podem ser utilizadas em uma combinação sequencial ou individual, dependendo da forma que o utilizador deseja realizar o processo de recomendação. Para que o objetivo do trabalho fosse atingido, o *framework* faz com que estas classes interajam entre si para que a saída sejam os objetos de aprendizagem recomendados para cada estudante.

Para que as recomendações sejam realizadas, os seguintes métodos são aplicados:

1. **Predição de dados de desempenho:** Para atender o processo de previsão de desempenho do estudante, é executado um método que possui três algoritmos de aprendizagem de máquina *K-Nearest Neighbors* (KNN) (Aha et al. 1991), *Random Forest* (DINIZ et.al, 2013) e *Naive Bayes* (Lewis, 1998) com dados de treinamento. Estes dados de treinamento contêm informações sobre estudantes com perfil de aprendizagem semelhante e o seu desempenho já conhecido. Com estes dados, os algoritmos são treinados para que seja possível prever o desempenho futuro do estudante. Após o treinamento do algoritmo, são lidos dados atuais do estudante e então definido o risco de reprovação do mesmo. Com os dados obtidos da predição, é possível alimentar as filtragens descritas abaixo e realizar um processo de recomendação ainda mais refinado, aumentando a quantidade de materiais, reduzindo nível de complexidade, buscando outros tipos de materiais, com o objetivo de reforçar a aprendizagem do estudante, tendo em vista que essa previsão pode acabar evidenciando um possível risco de reprovação que ainda não é conhecido no momento.

2. **Filtragem Colaborativa:** Realizada a filtragem por conteúdo, o *framework* realiza um processo de filtragem colaborativa. Em alto nível, esta etapa é separada em alguns processos:
    a. **Identificar estudantes:** O algoritmo identifica estudantes com perfil de aprendizagem em comum;
    b. **Analisar as avaliações:** Com a lista de estudantes identificada, o algoritmo realiza um levantamento dos OA utilizados e avaliados por estes estudantes. Nesse momento, existem duas listas separadas, contendo alunos e contendo OA e a avaliação de cada estudante;

   c. **Cruzamento dos Dados:** Com as duas listas mencionadas acima, o algoritmo faz o cruzamento dos dados e gera uma matriz;
   d. **Identificar OA:** Com a matriz de dados realizada, é aplicado o algoritmo KNN para identificar o vizinho mais próximo pela distância Euclidiana. Nesse momento, o algoritmo verifica quais os potenciais OA que podem ser recomendados ao estudante com base nas avaliações dos estudantes com perfil semelhante.

3. **Filtragem baseada em conteúdo**: Esse modelo de filtragem busca os objetos de aprendizagem com base no conteúdo do plano de ensino do estudante. O *framework* parte desse modelo de recomendação para começar a interagir com outras classes.

   Nesse modelo de filtragem, os algoritmos fazem a busca a partir do Perfil de Aprendizagem do Estudante e dos OA que são relacionados ao plano de ensino da disciplina.

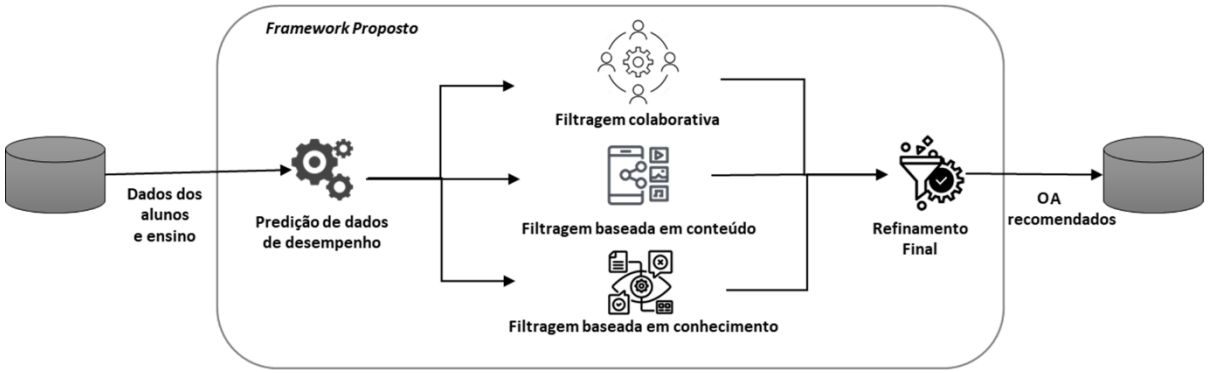
   Esse modelo de filtragem inicial faz com que o problema do primeiro avaliador deste tipo de filtragem seja superado, tendo em vista de que não existirão dados suficientes para realizar a filtragem colaborativa, portanto, esta etapa faz com que o estudante contenha uma base mínima de OA para auxiliar no ensino.

4. **Filtragem baseada em conhecimento:** Até este momento, o *framework* contém uma lista de OA que podem ser recomendados para cada estudante. Se existirem dados de desempenho do estudante na base de conhecimento, a etapa de filtragem baseada em conhecimento realiza a filtragem dos materiais identificados até o momento com base no que o estudante sabe sobre certo assunto, ou seja, nesse momento o algoritmo faz a filtragem apenas para buscar os materiais cuja complexidade está de acordo com o que o estudante conhece do assunto.

A Figura 9 representa de uma forma mais suprimida as etapas que são realizadas pelo *framework*, desde a coleta de dados até a filtragem e realização das recomendações. Conforme ilustrado pela Figura, o *framework* parte da realização de uma busca por dados básicos do estudante e da disciplina, a fim de detectar os padrões de aprendizagem. Com estes dados, é possível realizar um ponto de corte e

executar a predição de desempenho e posterior, o processo de filtragem paralela de objetos de aprendizagem.

Figura 9 - Funcionamento do Framework



Fonte: Elaborado pelo autor.

O Capítulo 5 abordará com maiores detalhes cada uma das etapas e como os dados são processados e trafegados em cada uma das etapas, considerando as entradas e como as saídas de cada uma das etapas é realizada.

4.5 VALIDAÇÃO

A partir do momento que as recomendações são geradas pelo *framework,* é necessário um trabalho para validar se estas recomendações realmente estão refletindo o objetivo do trabalho. Em tempo de desenvolvimento, foram realizadas validações individuais em cada um dos componentes, como testes funcionais exploratórios para garantir que cada componente está apresentando o resultado esperado.

A primeira etapa consiste em verificar se as filtragens estão buscando os OA apropriados ao contexto onde o aluno está aplicado. Para esta validação foram elaborados cenários de testes. Estes cenários foram desenhados pensando no modelo de aprendizagem apresentado por Felder e Silverman (Conforme detalhado no referencial teórico). A partir destas informações, foram gerados dados de alunos considerando os modelos de aprendizagem. Como objetos de aprendizagem, foram gerados OA respeitando o padrão LOM, que também se encontra no referencial teórico. Com ambas as entidades especificadas anteriormente (padrões de aprendizagem e objetos de aprendizagem), foi utilizado um padrão de plano de ensino

da Universidade Federal de Santa Maria para permitir o cruzamento entre os OA e o que será abordado ao estudante no decorrer da matéria.

Além da criação destes dados, foram gerados dados fictícios de alunos (nome, desempenho, preferências) para permitir que sejam realizadas as filtragens e as devidas recomendações. Estes dados cadastrais foram gerados com o apoio de uma API disponível na Web (https://api.invertexto.com/) e a API *Text* do Chat GPT, o que possibilitou a existência de dados suficientes para validação de cada elemento.

Com os dados simulados, foi realizada a validação unitária de cada componente:

- Filtragem baseada em conteúdo: Verificação se os OA correspondem ao que o estudante precisa aprender;
- Filtragem colaborativa: Verificação se o algoritmo identificou alunos com perfil de aprendizagem semelhante e buscou os OA que foram aprovados por estes alunos;
- Filtragem baseada em conhecimento: Verificação se o algoritmo identificou o desempenho e cruzou com os OA disponíveis na base de dados, levando em consideração o que o aluno conhece acerca do assunto;
- Predição: Verificação se o módulo preditivo foi capaz de gerar um score de risco de reprovação do estudante após o treinamento do modelo.

Tendo sucesso em cada etapa, o *framework* foi executado como um todo, combinando os métodos de filtragem e a geração de dados futuros a partir do módulo preditivo.

Para validação do *framework* estudado e implementado, foram utilizados cenários para simular diferentes ocasiões para que sejam geradas as recomendações. Conforme apresentado no capítulo de trabalhos relacionados, existem estudos que realizam uma validação com a aplicação de cenários de teste para comprovar o comportamento de algoritmos e sistemas completos.

Com a aplicação de cenários, é possível identificar as recomendações geradas e então comparar se realmente estes dados são relevantes para o cenário aplicado. Dessa forma, é possível garantir o funcionamento do *framework* antes mesmo de aplica-lo em um cenário real, como uma turma de alguma universidade e escola.

Além do mais, neste modelo de validação, é possível manter um ambiente controlado, tendo ciência da entrada de dados, simular o processamento e então comparar se a saída está de acordo com o planejado, o que seria mais complexo em um ambiente real.

O Capítulo 6 do presente trabalho apresenta dados concretos que foram coletados a partir das recomendações geradas pelo *framework*. Neste capítulo é possível identificar os cenários utilizados para validação e como os dados foram processados e recomendados aos alunos.

5 DESENVOLVIMENTO

Este capítulo apresenta informações relacionadas ao desenvolvimento do *framework* de recomendação proposto por meio da pesquisa em questão. O capítulo apresenta detalhamentos em relação a como o *framework* foi projetado em relação a requisitos funcionais e não funcionais.

A partir da leitura deste capítulo, será possível identificar a relação dos métodos implementados com os estudos realizados nos capítulos anteriores, com exemplos de implementações e visualizações gráficas por meio de diagramas.

5.1 TECNOLOGIAS

Para implementação do *framework* foram utilizadas tecnologias voltadas para processamento de dados, tendo em vista que o processo de predição e recomendação parte de análise e processamento de dados de estudantes e objetos de aprendizagem. Nesta seção é possível identificar as tecnologias empregadas para que o funcionamento do *framework* seja garantido conforme proposto pelo estudo.

5.1.1 Banco de Dados

Para armazenamento dos dados utilizados para análise e realização das recomendações foi utilizado o banco de dados PostgreSQL. Segundo Milani (2008), PostgreSQL é um Sistema Gerenciador de Banco de Dados (SGBD) Relacional, utilizado para armazenamento de dados para todas as áreas de negócios existentes, além de permitir a administração destes dados.

O PostgreSQL é uma ferramenta *open source,* ou seja, de código aberto, o que permite que usuários possam implementar melhorias e mudanças no sistema, além de não gerar custos para utilização da ferramenta.

Para implementação do *framework* proposto foi utilizado o PostgreSQL na versão 15 para armazenamento das informações necessárias para geração das recomendações, bem como as recomendações realizadas. Para gerenciamento do Banco de Dados, foi utilizado o pgAdmin versão 4, ferramenta nativa para administração de bancos de dados PostgreSQL.

5.1.2 Linguagem de Programação

Com os dados existentes na base de dados do PostgreSQL, é necessário realizar o processamento destes dados para então realizar as recomendações. O

processamento dos dados é realizado por meio do *framework* proposto que foi implementado por meio da linguagem de programação *Python*.

Para Carvalho (2023), a linguagem de programação *Python* tornou-se uma das linguagens de programação mais populares do mundo nos últimos anos. A partir de sua versatilidade, é possível implementar desde algoritmos de aprendizado de máquina até a construção de sites.

Em relatório publicado por *Python Developers Survey 2021 Results,* é possível evidenciar que a área mais utilizada com *Python* é a de análise de dados, desta forma, considerando a sua grande relevância na área de análise de dados, o *framework* proposto por este estudo foi implementado nesta linguagem de programação, aproveitando os recursos e otimizações disponíveis.

5.2 MODELAGEM

Esta seção apresenta alguns diagramas principais que representam a implementação do *framework* proposto pelo estudo em questão. Estes diagramas foram implementados em uma etapa prévia a implementação, com o objetivo de nortear os requisitos necessários para atender o que o estudo propõe. Na medida que os algoritmos foram sendo implementados, os diagramas foram atualizados para representar a real funcionalidade.

Como diagramas apresentados, pode-se visualizar o Modelo Entidade Relacionamento, um Diagrama de Classes e um Diagrama de Atividades, para que seja possível identificar todas as etapas que são realizadas pelo *framework* na geração das recomendações necessárias.

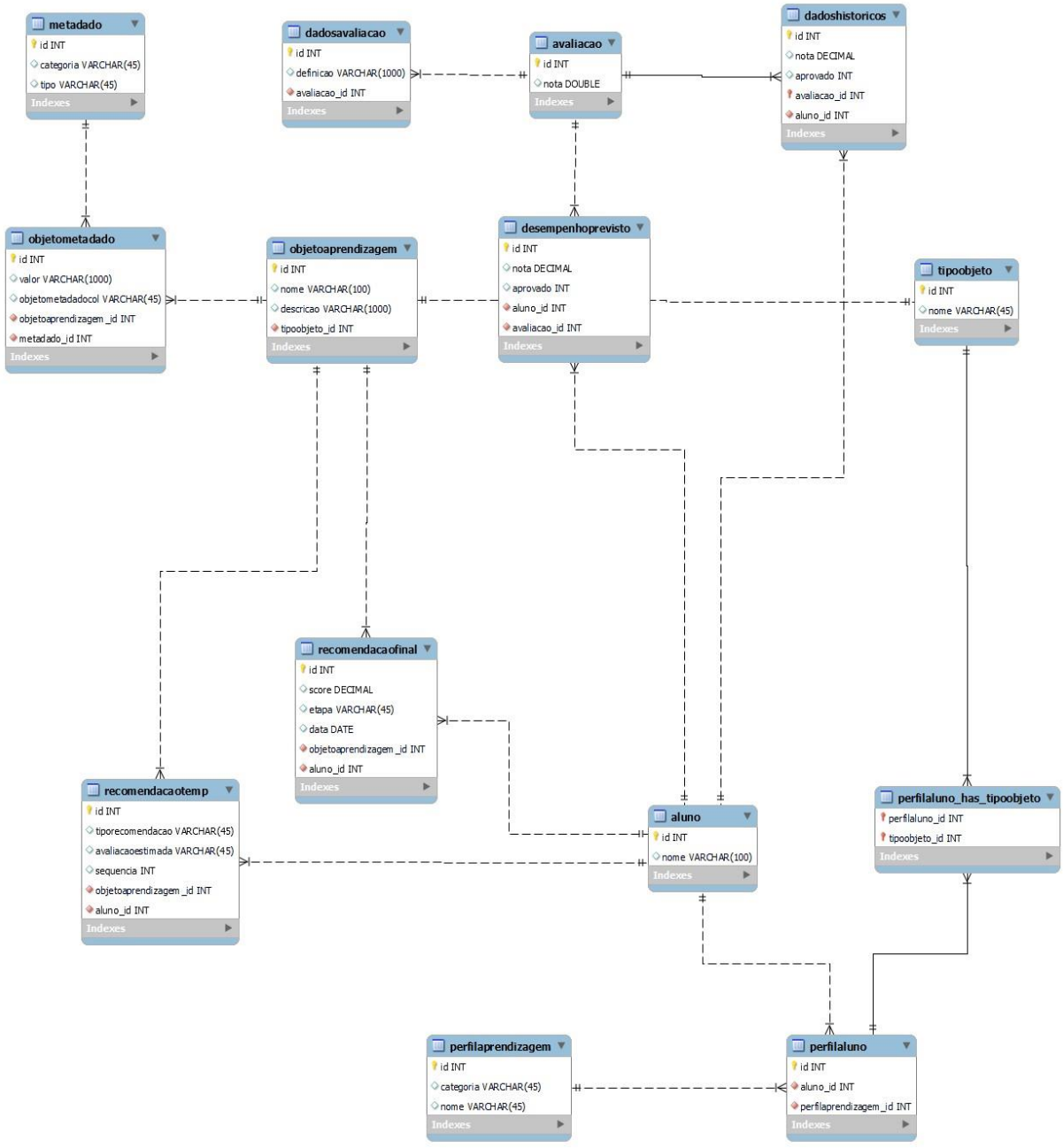
5.2.1 Modelo Entidade Relacionamento (MER)

O modelo de entidade relacionamento (MER) tem por base a descrição de algo do mundo real realizada através de um conjunto de objetos chamados “entidades” e pelo conjunto de “relacionamentos” entre esses objetos. Entidade é algo (físico ou conceitual) do mundo real. Ela pode ser forte, se não depender de outra entidade para existir ou fraca, se sua existência no modelo estiver condicionada à presença de outra entidade. (ELMASRI; NAVATHE, 2005, p. 39). Além disso, as entidades possuem “atributos”, que são propriedades particulares que as descrevem. Os valores definidos por cada atributo descrevem as entidades e formam o conjunto de dados

armazenados na base de dados (ELMASRI; NAVATHE, 2005, p. 39). Esses valores são representações da informação fornecendo pontos de acesso ao texto, ao documento, a informação completa.

Diante deste contexto, conforme ilustrado pela Figura 10, foi elaborado um MER com o objetivo de apresentar as entidades, relacionamentos e atributos da base de dados utilizada pelo *framework* implementado a partir deste estudo.

Figura 10 - Modelo ER



Fonte: Elaborado pelo autor.

Para detalhar as entidades do *framework,* a listagem abaixo apresenta cada uma delas juntamente com seus objetivos:

- metadado: Responsável por manter armazenado todos os possíveis metadados do padrão LOM para descrever objetos de aprendizagem;
- objetoMetadado: Responsável por manter o vínculo entre objetos de aprendizagem e os metadados presentes na entidade de metadado, seguindo o padrão LOM;
- dadosAvaliacao: Responsável por manter informações relacionadas as etapas de ensino que o estudante passará em uma determinada disciplina;
- objetoAprendizagem: Responsável por manter os objetos de aprendizagem;
- avaliação: Responsável por manter a avaliação realizada por alunos do passado em objetos de aprendizagem existentes na base de dados;
- recomendacaoTemp: Responsável por manter as recomendações temporárias geradas pelo *framework*.
- recomendacaoFinal: Responsável por manter as recomendações oficiais geradas a partir de todas as etapas de filtragem e refinamento do *framework*;
- aluno: Responsável por manter os alunos;
- desempenhoPrevisto: Responsável por manter o desempenho previsto do aluno avaliado na etapa de predição do *framework;*
- perfilAprendizagem: Responsável por manter os perfis de aprendizagem conforme o modelo de Felder e Silverman;
- dadosHistoricos: Responsável por manter o desempenho passado de alunos da disciplina que está sendo executada pelo aluno;
- tipoObjeto: Responsável por manter os tipos de objetos de aprendizagem possíveis;
- perfilAluno: Responsável por manter o vínculo entre aluno e o perfil de aprendizagem do mesmo, seguindo o modelo de Felder e Silverman;

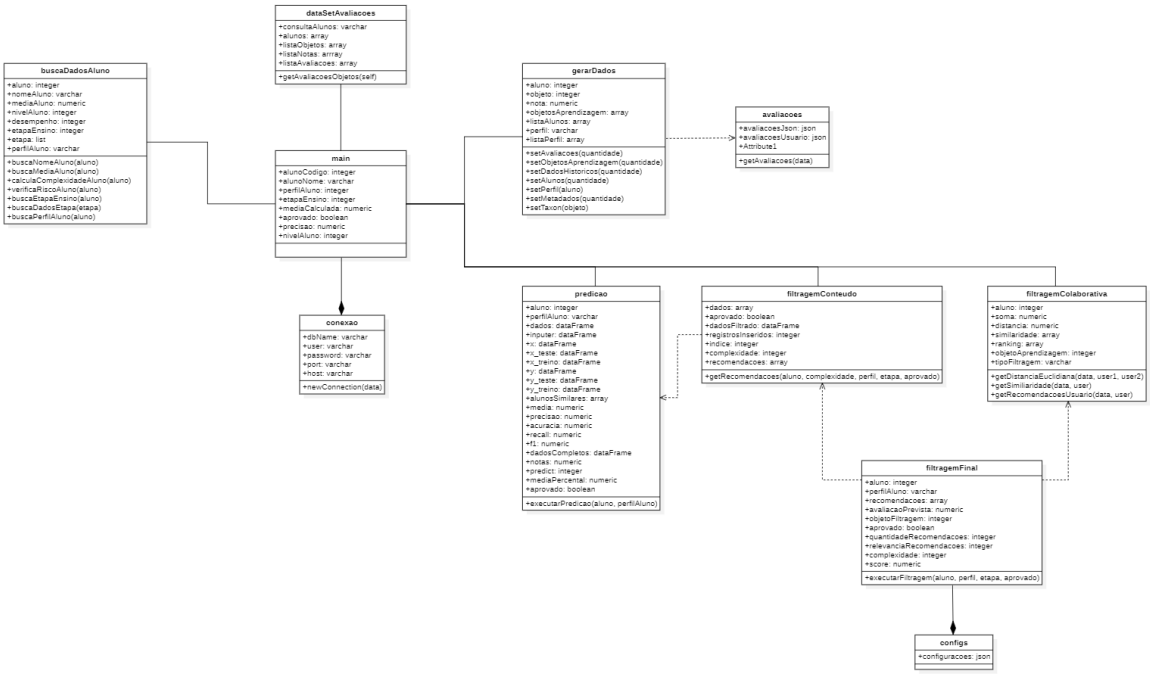
- tipoObjetoPerfil: Responsável por manter o vínculo entre o tipo de objeto de aprendizagem e a definição de quais os perfis de aprendizagem que melhor se encaixam no tipo determinado.

Com as entidades descritas anteriormente, o *framework* tem a possibilidade de coleta dos dados necessários para então processá-los e gerar as recomendações conforme o seu propósito.

5.2.2 Diagrama de Classes

Para entendimento de como as classes do *framework* interagem para geração das recomendações, foi criado um diagrama de classes. A Figura 11 ilustra todas as classes que foram desenvolvidas para que as recomendações sejam geradas. Cada classe possui um objetivo específicos, dessa forma, ao integrar estas classes o *framework* é capaz de realizar a análise dos dados para então gerar as recomendações conforme proposto pela pesquisa.

Figura 11 - Diagrama de Classes



Fonte: Elaborado pelo autor.

A visão funcional de cada uma das classes pode ser verificada conforme o detalhamento presente na Tabela 3.

Tabela 3 - Classes

Classe	Detalhamento
avaliacoes	Classe responsável por coletar as avaliações que foram realizadas por alunos em objetos de aprendizagem catalogados.
buscaDadosAluno	Classe responsável por coletar informações básicas do aluno para utilização nas classes de predição e geração de recomendações.
conexao	Classe responsável por instanciar uma conexão com a base de dados para leitura e escrita de dados.
configs	Classe responsável por manter todas as configurações possíveis para que sejam lidas pelas demais classes do *framework.*
dataSetAvaliacoes	Classe responsável por coletar as avaliações geradas sobre os objetos de aprendizagem da base de dados, vincular a cada aluno e gerar os dados para que sejam aproveitados pela classe de avaliações, em formato *json.*
filtragemColaborativa	Classe responsável por criar recomendações temporárias com base no modelo de filtragem colaborativa.
filtragemConteudo	Classe responsável por criar recomendações temporárias com base no modelo de filtragem por conteúdo.
filtragemFinal	Classe responsável por realizar o refinamento das filtragens temporárias e então popular a base de dados com as recomendações oficiais para o aluno.
gerarDados	Classe responsável pela geração de *dataset* para uso nas demais classes.
main	Classe principal do *framework,* responsável por orquestrar e realizar as requisições para cada uma das demais classes do *framework.*
predicao	Classe responsável por realização da predição dos dados para uso nas classes de filtragem.

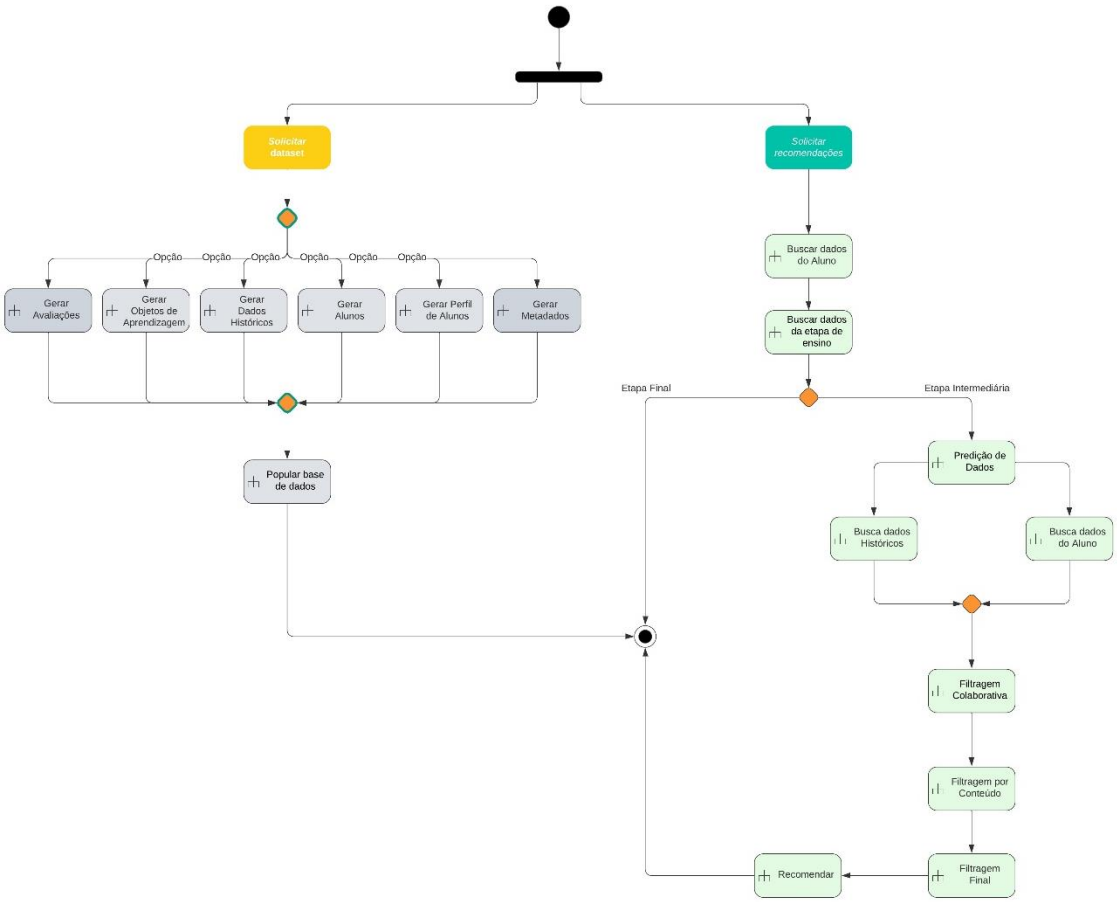
Fonte: Elaborado pelo autor.

5.2.3 Diagrama de Atividade

O diagrama de atividades é um dos diversos diagramas definidos pela UML (Unified Modeling Language) [Object Management Group 2007]. É utilizado para ilustrar uma sequência de atividades que foram modeladas para representar os aspectos dinâmicos de um processo computacional. Este modelo detalha o fluxo de controle e o fluxo de dados de uma determinada atividade, mostrando as ramificações de controle e as situações onde existem processamento paralelo.

A Figura 12 ilustra as atividades que são realizadas por parte do *framework* para geração do *dataset* e para a criação das recomendações de objetos de aprendizagem, conforme a proposta do presente estudo.

Figura 12 - Diagrama de Atividades



Fonte: Elaborado pelo autor.

5.3 DATASET

Para que seja possível realizar a análise de dados e posteriormente a recomendação de objetos de aprendizagem, conforme proposto pelo estudo, são

necessários dados. Estes dados precisam estar estruturados de forma que o *framework* seja capaz de lê-los, interpretá-los e então executar os métodos implementados.

Cada uma das etapas realizadas pelo *framework* trabalha com dados específicos, como dados de alunos, avaliações, desempenhos passados, objetos de aprendizagem, etc. Para que seja possível trabalhar em um cenário de validação o mais próximo possível do mundo real, é necessário que exista uma base de dados populada com uma série de dados. Estes dados foram obtidos por meio de algoritmos implementados com o objetivo de geração de dados conforme necessidade do *framework*.

Esta seção apresenta um detalhamento sobre como os dados foram obtidos e gerados, a fim de utilização no processo de validação do *framework.*

5.3.1 Dados de Alunos

As recomendações propostas pelo *framework* são para alunos, logo é necessária uma carga de dados de alunos para que os algoritmos de recomendação sejam executados. Para obtenção destes dados, foi implementada uma classe específica para geração de dados, contento um método para geração de alunos.

Conforme ilustrado na Figura 13, este método realiza uma integração via API com o chat GPT, solicitando uma lista de nomes completos de pessoas. Com esta lista obtida, os dados são processados em um vetor de nomes, os quais são gravados na respectiva tabela de alunos da base de dados.

Figura 13 - Método de geração de alunos

```
LuccaSchrammel
def gerarAlunos(quantidadeAlunos):
    cursor = conexao.conn.cursor()
    sqlNome = f'SELECT MAX(ID) + 1 FROM ALUNO'
    cursor.execute(sqlNome)
    idMax = cursor.fetchone()
    idMax = str(idMax[0])

    openai.api_key = configuracoes.configuracoes['geracaoDados']['apiKeyChatGpt']
    completion = openai.ChatCompletion.create(
        model=configuracoes.configuracoes['geracaoDados']['modelChatGpt'],
        messages=[
            {"role": "user",
             "content": f"lista de \'{quantidadeAlunos}\' novos nomes completos de pessoas"}
        ]
    )

    nome = completion.choices[0].message.content
    listaNomes = []
    listaNomes = nome.split('\n')
    print(listaNomes)

    i = 0
    contadorInsert = 0
    while i < len(listaNomes):...

    print(str(contadorInsert) + " alunos inseridos com sucesso")
    conexao.conn.closed()
```

Fonte: Elaborado pelo autor.

Além das informações básicas relacionadas aos alunos, é necessário determinar o perfil de aprendizagem de cada um destes estudantes, tendo em vista que o processo de recomendação analisará estas informações para direcionar os objetos de aprendizagem mais assertivos. Conforme apresentado no capítulo da metodologia, foi utilizado como base o modelo de Felder e Silverman para determinar o perfil de aprendizagem dos alunos.

O modelo de Felder e Silverman determina quatro entidades que definem como os estudantes aprendem, sendo elas: Percepção, Entrada, Processamento e Entendimento. Partindo destas quatro categorias, foi criado um método que gera um valor randômico para cada uma das categorias e atribuindo a um aluno previamente cadastrado. O trecho de código abaixo apresenta a geração randômica destes valores.

```
randomPercepcao = np.random.randint(1, 3)
randomEntrada = np.random.randint(3, 5)
randomProcessamento = np.random.randint(5, 7)
randomEntendimento = np.random.randint(7, 9)
```

Os alunos gerados pelos métodos apresentados serão utilizados pelo processo de análise de dados e realização de recomendação, conforme proposto pelo estudo.

5.3.2 Dados de objetos de aprendizagem

Da mesma forma que apresentado para geração dos dados de alunos, na classe de geração de dados, foi implementado um método responsável pela geração de objetos de aprendizagem relacionado a um tema específico. Estes objetos de aprendizagem serão utilizados pelos métodos de filtragem e recomendação de dados aos alunos, dessa forma, deve existir uma relação entre os objetos e as necessidades dos alunos.

O método de geração de objetos de aprendizagem também é feito via integração com o *chat* GPT. Conforme apresentado na Figura 14, é realizada uma requisição para a API do *chat* solicitando uma lista de objetos sobre um tema específico e então persistido em banco de dados. Conforme apresentado na metodologia, o processo de validação do algoritmo será realizado com base em dados de um plano de ensino da disciplina de engenharia de *software*, logo todos os objetos de aprendizagem gerados possuí vínculo com temas relacionados a esta disciplina.

Figura 14 - Método de geração de objetos de aprendizagem

```
def gerarObjetosAprendizagem(self):
    print("Iniciando a geração de objetos de aprendizagem")
    openai.api_key = configuracoes.configuracoes['geracaoDados']['apiKeyChatGpt']
    completion = openai.ChatCompletion.create(
        model=configuracoes.configuracoes['geracaoDados']['modelChatGpt'],
        messages=[
            {"role": "user", "content": "Nome de 200 recursos educacionais sobre teste de software"}
        ]
    )

    objetosAprendizagem = completion.choices[0].message.content
    listaObjetosAprendizagem = []
    listaObjetosAprendizagem = objetosAprendizagem.split('\n')

    cursor = conexao.conn.cursor()
    sqlId = f'SELECT MAX(ID) + 1 FROM OBJETOAPRENDIZAGEM'
    cursor.execute(sqlId)
    idMax = cursor.fetchone()
    idMax = str(idMax[0])
    idMax = int(idMax)

    i = 0
    contadorInsert = 0
    while i < len(listaObjetosAprendizagem):
        linha = listaObjetosAprendizagem[i]
        linha = str(linha)
        listaObjetosAprendizagem[i] = linha[3:]
        if i > 99:
            listaObjetosAprendizagem[i] = linha[4:]
        nomeInsert = "'" + listaObjetosAprendizagem[i] + "'"
        sqlVerificacao = f'SELECT * FROM OBJETOAPRENDIZAGEM WHERE NOME = {nomeInsert}'
        cursor.execute(sqlVerificacao)
        cont = cursor.rowcount

        if cont == 0:...
        idMax = idMax + 1
        i = i + 1

    print(str(contadorInsert) + " objetos de Aprendizagem inseridos com sucesso")
```

Fonte: Elaborado pelo autor.

Com o objeto de aprendizagem cadastrado na base de dados, é necessária a geração dos metadados para identificação destes objetos de aprendizagem. Conforme apresentado na metodologia, os objetos de aprendizagem são lidos pelo *framework* desde que existam os detalhamentos necessários conforme proposto pelo padrão LOM. Este padrão determina que os objetos de aprendizagem são descritos por meio de metadados.

Neste cenário, foi desenvolvido um método exclusivo para geração dos metadados a fim de detalhar ao máximo os objetos de aprendizagem criados. Ao todo, o método insere cerca de 20 metadados para cada objeto de aprendizagem cadastrado. Na Tabela 4 é possível identificar como cada metadado foi gerado e então vinculado aos objetos de aprendizagem:

Tabela 4 - Metadados gerados

Metadado	**Origem**
Title	Título inserido a partir de solicitação de descrição ao chat GPT via API.
Description	Descrição obtida no processo de criação do objeto de aprendizagem.
Keywords	Palavras chaves obtidas a partir de solicitação ao chat GPT a partir da descrição e título do objeto de aprendizagem.
Status	Status do objeto de aprendizagem (Ativo ou Inativo).
Date	Data de criação do objeto. Foi utilizada uma data padrão.
Language	Linguagem do objeto de aprendizagem. Foi utilizado o idima PT-BR como padrão.
Format	Formato do objeto de aprendizagem. A cada iteração do método, é escolhido um valor aleatório entre: 'mp4', 'doc', 'pdf', 'mp3', 'json', 'xml', 'txt', 'ppt', 'xls', 'web', 'sql', 'exe', 'jpg', 'png', 'drawio', 'bpmn'.
Size	Tamanho do objeto de aprendizagem. Foi utilizado valor randômico entre 1 e 1024 MB.
Type	Tipo do objeto de aprendizagem. Foi utilizado um valor randômico entre 1 e 37. Este tipo está vinculado ao perfil de aprendizagem dos estudantes, sendo utilizado no processo de recomendação.
Duration	Duração do objeto de aprendizagem. Foi utilizado um valor randômico entre 1 e 60 minutos.
Difficulty	Nível de dificuldade do objeto de aprendizagem. Foi utilizado um valor randômico entre 1 e 3.

Metadado	Origem
Typical Learning Time	Tempo padrão de aprendizagem do objeto. Foi utilizado um valor randômico entre 1 e 5 aulas.
Interactivity Type	Tipo de interação com o objeto de aprendizado. Foi utilizado um valor randômico entre: 'Ativo', 'Expositivo', 'Misturado', 'Simulativo', 'Adaptativo'.
Interactivity Level	Nível de interação com o objeto de aprendizagem. Foi utilizado um valor randômico entre: ‘Baixo’, ‘Médio’ e ‘Alto’.
Taxon	Taxonomia do objeto de aprendizado. Foi gerado um valor randômico para indicar qual o perfil de aprendizagem que mais se encaixa ao objeto de aprendizagem, respeitando o modelo de Felder-Silverman.

Fonte: Elaborado pelo autor.

A Figura 15 ilustra o código fonte implementado para inclusão de alguns dos metadados citados acima.

Figura 15 - Método de geração de metadados

```
LuccaSchrammel *
def gerarMetadados(quantidadeMetadados):
    cursor = conexao.conn.cursor()
    i = 0
    while i < quantidadeMetadados:
        #General Metadados
        j = 0
        while j == 0:
            randomObjeto = np.random.randint(5237, 6694)
            sqlContObjeto = f'SELECT * FROM OBJETOMETADADO WHERE OBJETOID = {randomObjeto}'
            cursor.execute(sqlContObjeto)
            contObjetoMetadado = cursor.rowcount
            if contObjetoMetadado == 0:
                sqlNomeObjeto = f'SELECT DESCRICAO FROM OBJETOAPRENDIZAGEM WHERE ID = {randomObjeto}'
                cursor.execute(sqlNomeObjeto)
                rowNumb = cursor.rowcount
                if rowNumb == 1:
                    nomeObjeto = cursor.fetchone()
                    nomeObjeto = nomeObjeto[0]

                    sqlMetadado = f'SELECT * FROM OBJETOMETADADO'
                    cursor.execute(sqlMetadado)
                    if cursor.rowcount == 0:
                        idMax = 1
                    else:
                        sqlNome = f'SELECT MAX(ID) + 1 FROM OBJETOMETADADO'
                        cursor.execute(sqlNome)
                        idMax = cursor.fetchone()
                        idMax = idMax[0]

                    #Grava a descrição do objeto como metadado
                    cursor.execute('INSERT INTO OBJETOMETADADO VALUES (\'{}\', \'{}\', \'{}\', \'{}\')'.
                                   format(int(idMax), randomObjeto, 3, nomeObjeto))
                    conexao.conn.commit()
                    print("Description inserida com sucesso!!!")
                    idMax = idMax + 1

                    #Gerar o title do objeto a partir da descrição
                    verificaSucesso = 0
                    while verificaSucesso == 0:
                        url = configuracoes.configuracoes['geracaoDados']['urlChatGPT']
                        headers = {
                            "Content-Type": "application/json",
                            "Authorization": f"Bearer {configuracoes.configuracoes['geracaoDados']['apiKeyChatGpt']}
                        }
                        data = {
                            "model": f"{configuracoes.configuracoes['geracaoDados']['modelChatGpt']}",
                            "messages": [
                                {"role": "user", "content": f"gere um titulo para {nomeObjeto}"}
                            ]
                        }

                        # Fazer a solicitação POST para a API do ChatGPT
                        response = requests.post(url, headers=headers, json=data)
                        response_data = response.json()
                        status_code = response.status_code

                        if status_code == 200:
                            titulo = response_data['choices'][0]['message']['content']
                            titulo = str(titulo).replace('"', '').replace("'", "")

                            cursor.execute('INSERT INTO OBJETOMETADADO VALUES (\'{}\', \'{}\', \'{}\', \'{}\')'.
                                           format(int(idMax), randomObjeto, 2, str(titulo)))
                            conexao.conn.commit()
                            print("Title inserido com sucesso!!!")
                            idMax = idMax + 1
                            verificaSucesso = 1

                        if status_code != 200:
                            print("Ocorreu erro na geração do titulo. Iniciando retentativa")
                            time.sleep(2)
```

Fonte: Elaborado pelo autor.

Com os dados de objetos de aprendizagem gerados conforme apresentado, o *framework* proposto contém todas variáveis necessárias para recomendação de objetos de aprendizagem, porém ainda são necessários alguns dados para a execução do processo de predição e filtragem colaborativa.

5.3.3 Dados históricos

Um dos dados mais importante para o correto funcionamento do *framework* são os dados históricos. Estes dados históricos são utilizados pelos algoritmos de predição para analisar e determinar o risco de reprovação de alunos que estão cursando uma disciplina no momento atual. Sem os dados históricos, os algoritmos de predição não terão capacidade de informar se o aluno possuí algum tipo de chance de reprovação prévia.

Para que se tenham os dados necessários, foi implementado um método que gera cinco notas de estudantes, juntamente com a informação se estes foram aprovados no passado ou não. A Figura 16 ilustra o procedimento adotado para geração dos dados, onde são submetidas cinco notas com valores randômicos e ao final é determinado se o mesmo foi aprovado, caso a média simples seja igual ou superior à sete.

Figura 16 - Método de geração de dados históricos

```
LuccaSchrammel *
def gerarDadosHistoricos(self):
    quantidadeIteracoes = 5000
    cursor = conexao.conn.cursor()
    print("Iniciando geração de dados Históricos!!!")

    i = 0
    while i < quantidadeIteracoes:
        randomAluno = np.random.randint(1, 7000)
        randomNota1 = np.random.uniform(3, 10)
        randomNota2 = np.random.uniform(4, 10)
        randomNota3 = np.random.uniform(4, 10)
        randomNota4 = np.random.uniform(3, 10)
        randomNota5 = np.random.uniform(3, 10)

        if (randomNota1 + randomNota2 + randomNota3 + randomNota4 + randomNota5) / 5 >= 7:
            aprovado = 1
        else:
            aprovado = 0

        sqlVerificaAluno = f'SELECT * FROM ALUNO WHERE ID = {randomAluno}'
        cursor.execute(sqlVerificaAluno)
        cont1 = cursor.rowcount

        if cont1 > 0:
            sqlVerificacao = f'SELECT * FROM DADOSHISTORICOS WHERE ALUNO = {randomAluno}'
            cursor.execute(sqlVerificacao)
            cont = cursor.rowcount
            if cont == 0:
                cursor.execute('INSERT INTO DADOSHISTORICOS VALUES ((SELECT MAX(ID) + 1 FROM DADOSHISTORICOS), '
                               '\'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\' )'.
                               format(randomAluno, randomNota1, randomNota2, randomNota3, randomNota4, randomNota5,
                                      aprovado))
                conexao.conn.commit()
            else:
                print("Descontou 1")
                i = i - 1
```

Fonte: Elaborado pelo autor.

5.3.4 Avaliações

Uma das etapas do processo de recomendação realizado pelo *framework* é a filtragem colaborativa. Este modelo de filtragem consiste em identificar os estudantes com perfil semelhante e então recomendar os objetos de aprendizagem que já foram definidos no passado como importantes para os estudantes.

Para que seja possível realizar este tipo de filtragem, foi necessário implementar um método para que sejam geradas avaliações por alunos para objetos de aprendizagem, dessa forma, haverá dados suficientes para identificar objetos de aprendizagem que se encaixam nas necessidades do aluno que será utilizado para realizar recomendações por parte do *framework.* A Figura 17 representa como o método foi implementado, utilizando alunos, objetos e nota avaliativa randômica para registrar em base de dados.

Figura 17 - Método para geração de avaliação de objetos de aprendizagem

```
def gerarAvaliacoes(self):
    cursor = conexao.conn.cursor()
    i = 0
    while i < 500:
        randomAluno = np.random.randint(100, 5000)
        randomObjeto = np.random.randint(1, 1300)
        randomNota = np.random.uniform(0, 10)

        sqlVerificaObjeto = f'SELECT * FROM OBJETOAPRENDIZAGEM WHERE ID = {randomObjeto}'
        cursor.execute(sqlVerificaObjeto)
        cont1 = cursor.rowcount
        if cont1 > 0:
            sqlVerificacao = f'SELECT * FROM AVALIACAO ' \
                             f'WHERE ALUNO_ID = {randomAluno} AND OBJETO_ID = {randomObjeto} ' \
                             f'GROUP BY ALUNO_ID, OBJETO_ID'
            cursor.execute(sqlVerificacao)
            cont = cursor.rowcount
            #print(cont)

            if cont == 0:
                print(randomObjeto, randomAluno)
                cursor.execute('INSERT INTO avaliacao VALUES(\'{}\', \'{}\', \'{}\')'.
                               format(int(randomObjeto), int(randomAluno), round(randomNota, 2)))
                conexao.conn.commit()
                i += 1
        else:
            i = i
```

Fonte: Elaborado pelo autor.

Com todos os métodos apresentados nessa seção, os dados necessários para análise e recomendação dos objetos de aprendizagem por parte do *framework* estão mantidos na base de dados para serem analisados.

5.4 IMPLEMENTAÇÃO

Conforme apresentado até o momento, o *framework* foi implementado por meio da linguagem de programação Python com uso de uma base de dados em PostgreSQL. Nesta seção é apresentado em detalhes os algoritmos que foram implementados e que compõem o *framework* para realização das recomendações.

5.4.1 Coleta dos dados

O *framework* necessita de alguns dados que foram gerados para formação do *dataset*. Estes dados serão utilizados de formas distintas, dependendo do algoritmo que estará em execução. Antes de iniciar os passos para filtragem, alguns dados são necessários para alimentar os algoritmos.

O *framework* é sempre executado a partir de um aluno que é recebido via parâmetro, ou seja, deve ser realizada uma chamada ao *framework* informando qual o aluno que deve ser analisado para então realização das recomendações. Neste cenário, inicialmente é realizada a coleta das seguintes informações do aluno:

- Perfil de Aprendizagem;
- Etapa de Ensino atual – Determina em qual etapa do plano de ensino da disciplina o aluno está localizado.

A Figura 18 representa a invocação dos métodos pela classe principal do *framework* para coleta dos dados descritos acima.

Figura 18 - Busca dos dados do Aluno

```
aluno = 1

LuccaSchrammel
class main:
    nomeAluno = buscaDadosAluno.buscaNomeAluno(aluno)
    print('Iniciando processo de recomendação para o aluno: ' + str(nomeAluno))
    perfilAluno = buscaDadosAluno.buscaPerfilAluno(aluno)

    # Identificar etapa do ensino
    etapaEnsino = buscaDadosAluno.buscaEtapaEnsinoAluno(aluno)
```

Fonte: Elaborado pelo autor.

Os dados mencionados são coletados para utilização como base na filtragem por conteúdo e no refinamento das recomendações temporárias, garantindo que todas as recomendações realmente são relacionadas ao aluno que está sendo analisado. A origem dos dados é a base de dados que foi preenchida conforme detalhado na seção de *dataset.*

5.4.2 Predição

Contendo as informações básicas do aluno conforme descrito na subseção acima, a classe principal do *framework* direciona para o processo de predição de desempenho do estudante. Caso o estudante já tenha concluído todas as etapas do plano de ensino, nenhuma recomendação é realizada, encerrando o processo já na etapa acima, porém caso ainda exista no mínimo uma etapa a ser executada, o *framework* executa o método de predição.

Para execução da predição, os algoritmos esperam a identificação do aluno e o perfil de aprendizagem deste aluno, conforme representado na Figura 19 o trecho de código que invoca o método de predição do *framework.*

Figura 19 - Solicitação de Predição de Desempenho

```
#Buscando dados dos algoritmos de predição
mediaCalculada, mediaPrevista, aprovado, precisaoPrevisao = \
    predicao.predicaoAlunos.preverAlunosRisco(aluno, perfilAluno)
```

Fonte: Elaborado pelo autor.

Ainda conforme ilustrado na Figura 19, identifica-se que os seguintes dados são coletados a partir do método de predição:

- Média Calculada – A média calculada é a média simples de todas as notas existentes e previstas do aluno (Somatórios das notas / Quantidade de avaliações);
- Média Prevista – A média prevista considera a precisão da previsão para determinar um desvio padrão, ou seja, quanto menor a precisão da previsão, maior pode ser o desvio da média prevista do aluno;
- Aprovado – *Flag* que determina o resultado final da predição (1 = Aluno aprovado; 0 = Aluno reprovado);
- Precisão da Previsão – Previsão da predição dos dados, conforme cálculo dos algoritmos utilizados.

Estas informações são geradas a partir da execução dos algoritmos de predição presentes na classe de predição de dados.

Inicialmente, a classe de predição coleta algumas informações que são exclusivas para execução da predição, como os dados históricos de desempenho de estudantes que já passaram pelas etapas de ensino da disciplina. O método instancia uma conexão com a base de dados e faz uma projeção de todas as informações presentes na entidade de dados históricos, caso algum atributo esteja nulo, o próprio algoritmo preenche estes dados com a média dos vizinhos mais próximos identificados, conforme ilustrado na Figura 20.

Figura 20 - Busca dos dados históricos para predição

```
LuccaSchrammel *
class predicaoAlunos:
    LuccaSchrammel *
    def preverAlunosRisco(aluno, perfilAluno):
        cursor = conexao.conn.cursor()
        engine = sq.create_engine(
            f"postgresql+psycopg2://"
            f"{conexao.user}:"
            f"{conexao.password}@"
            f"{conexao.host}:"
            f"{conexao.port}/"
            f"{conexao.dbName}"
        )

        k = configuracoes.configuracoes['predicao']['numeroVizinhos']
        dados = pd.read_sql_query(sql='SELECT * FROM DADOSHISTORICOS', con=engine)
        nan = np.nan
        dados.fillna(nan, inplace=True)
        dados = pd.DataFrame(dados)
        imputer = KNNImputer(n_neighbors=k)

        # preenche os valores faltantes com a média dos valores não faltantes
        df_imputed = pd.DataFrame(imputer.fit_transform(dados), columns=dados.columns)
        dados = df_imputed
```

Fonte: Elaborado pelo autor.

Com os dados nulos presentes no *dataset* substituídos pela média dos vizinhos mais próximos, é realizada a separação dos dados como conjunto de treinamento e conjunto de teste. Estes dados são separados de forma randômica, sendo que 80% foram direcionados para treinamento e 20% foram direcionados para testar os algoritmos de filtragem. A Figura 21 representa como esta separação é realizada.

Figura 21 - Separação de dados de treinamento e teste

```
# Divida os dados em conjuntos de treinamento e teste
X = dados.drop("aprovado", axis=1)
y = dados["aprovado"]
X_treino, X_teste, y_treino, y_teste = train_test_split(X, y, test_size=0.2, random_state=42)

# Treine o modelo com os dados de treinamento
knn = KNeighborsClassifier(n_neighbors=k)

linha = X_treino.loc[X_treino['aluno'] == aluno]
linha = linha.reset_index(drop=True)
colunas_excluir = ['id', 'aluno', 'nota1', 'nota2', 'nota3', 'nota4', 'nota5']
linhaY = linha.drop(colunas_excluir, axis=1)
```

Fonte: Elaborado pelo autor.

Os dados do aluno que está sendo avaliado para realizar as recomendações são sempre direcionados para o conjunto de teste, tendo em vista que o mesmo está incompleto e é necessária uma recomendação do algoritmo para identificar o risco de reprovação. Neste momento, os conjuntos de treinamento e teste estão definidos e já podem ser submetidos para os algoritmos de predição escolhidos.

O método de predição conta com a integração de três tipos de algoritmos distintos para realização da predição, sendo eles:

- **KNN** – O algoritmo k-*Nearest Neighbor*, também conhecido por sua sigla (KNN), é um algoritmo de aprendizado supervisionado do tipo *lazy*, introduzido por Aha et al. (1991). O objetivo geral desse algoritmo consiste em encontrar os k exemplos rotulados mais próximos do exemplo não classificado e, com base no rótulo desses exemplos mais próximos, é tomada a decisão relativa à classe do exemplo não rotulado. Os algoritmos do pacote KNN requerem pouco esforço durante a etapa de treinamento, em contrapartida, o custo computacional para rotular um novo exemplo é relativamente alto, pois no pior dos casos, esse exemplo deverá ser comparado com todos os exemplos contidos no conjunto de exemplos de treinamento, o que pode acabar gerando um custo computacional elevado na medida que o conjunto de treinamento seja incrementado.
- **Random Forest** – Este algoritmo foi proposto por Breiman, que consiste em uma técnica de agregação de classificadores do tipo árvore de

decisão, construídos de maneira que a estrutura seja composta de diversas maneiras aleatórias. Para determinar a classe de uma instância, o método combina o resultado de várias árvores de decisão, por meio de um mecanismo de votação. Ao final cada árvore dá uma classificação, ou um voto para uma classe. A classificação final é dada pela classe que recebeu o maior número de votos entre todas as árvores da floresta (DINIZ et.al, 2013).

- **Naive Bayes** - O classificador Naive Bayes é um dos algoritmos responsáveis pela geração de predições em aprendizagem de máquina. O termo “naive” diz respeito à forma como o algoritmo realiza a analisa das características de uma base de dados: ele assume que as features são independentes entre si. O seu funcionamento pode ser descrito em termos estatísticos, sendo que para calcular a predição, o algoritmo define uma tabela de probabilidades em que consta a frequência dos preditores com relação às variáveis de saída. Com estes dados, o cálculo final leva em conta a probabilidade maior para oferecer uma solução (Lewis, 1998).

Para execução de cada um destes algoritmos, foram utilizadas bibliotecas específicas da linguagem *Python,* que torna o processo de execução o mais simplificado possível. Inicialmente, o método direciona os dados de treinamento e teste para execução do algoritmo KNN. A execução do KNN é realizado com auxílio da biblioteca *KNeighborsClassifier* do pacote *sklearn.neighbors*, responsável por instanciar os requisitos do algoritmo de predição e permite a execução a partir da entrada dos conjuntos de treinamento e teste. A Figura 22 representa o envio dos conjuntos de dados para execução do KNN

Figura 22 - Envio de dados para KNN

```
# Obter o próximo índice disponível
proximo_indice = X_teste.index.max() + 1
# Adicionar a nova linha ao DataFrame
X_teste.loc[proximo_indice] = linha.iloc[0]
proximo_indice = y_teste.index.max() + 1
y_teste.loc[proximo_indice] = 0
knn.fit(X_treino, y_treino)
# Faça previsões para os dados de teste
y_pred = knn.predict(X_teste)
acuracia = accuracy_score(y_teste, y_pred)
precisao = precision_score(y_teste, y_pred)
recall = recall_score(y_teste, y_pred)
f1 = f1_score(y_teste, y_pred)
```

Fonte: Elaborado pelo autor.

A partir do momento que o KNN foi executado e o conjunto de dados foi validado, garante-se a integridade dos dados, o que faz com que o método direcione os mesmos dados para execução do algoritmo *Random Forest*. Para execução deste algoritmo, foi utilizada a biblioteca *RandomForestClassifier* do pacote *sklearn.ensemble.* Por meio desta biblioteca, conforme ilustrado na Figura 23, foi possível submeter os dois conjuntos de dados para treinamento e teste do modelo, mantendo-se os mesmos dados da etapa anterior e reforçando que os dados do aluno processado pelo algoritmo estão presentes no conjunto de testes.

Figura 23 - Envio de dados para *Random Forest*

```
#print("Random Forest")
rf = RandomForestClassifier(n_estimators=100, random_state=42)

# Treinar o modelo usando os dados de treinamento
rf.fit(X_treino, y_treino)

# Fazer previsões usando os dados de teste
colunas_excluir = ['teste', 'predicao']
X_teste = X_teste.drop(colunas_excluir, axis=1)
y_pred = rf.predict(X_teste)

# Calcular a precisão das previsões
acuraciaRf = accuracy_score(y_teste, y_pred)
precisaoRf = precision_score(y_teste, y_pred)
recallRf = recall_score(y_teste, y_pred)
f1Rf = f1_score(y_teste, y_pred)
```

Fonte: Elaborado pelo autor.

Após a execução do *Random Forest* para predição de dados, o método direciona o mesmo conjunto de treinamento e testes para execução do algoritmo *Nayve Bayes*. Da mesma forma que no demais casos, é realizada a importação da biblioteca *GaussianNB* do pacote *sklearn.naive_bayes.* Esta biblioteca gera uma instância de todos os métodos que permite que seja realizado o processamento conforme o algoritmo correspondente. A Figura 24 representa o envio dos dados para execução do algoritmo, partindo do recebimento dos conjuntos de dados que já foram formados nas etapas anteriores. Com estes conjuntos de dados, é realizada a predição considerando as regras do algoritmo *Nayve Bayes.* Como saída do processo, haverá a média prevista e a definição de aprovação ou reprovação conforme os padrões identificados pelo algoritmo.

Figura 24 - Envio de Dados para Nayve Bayes

```
# Criar o modelo Naive Bayes
naive_bayes = GaussianNB()

# Treinar o modelo usando os dados de treinamento
naive_bayes.fit(X_treino, y_treino)

# Fazer previsões usando os dados de teste
colunas_excluir = ['teste', 'predicao']
X_teste = X_teste.drop(colunas_excluir, axis=1)
y_pred = naive_bayes.predict(X_teste)

# Calcular a precisão das previsões
acuraciaNb = accuracy_score(y_teste, y_pred)
precisaoNb = precision_score(y_teste, y_pred)
recallNb = recall_score(y_teste, y_pred)
f1Nb = f1_score(y_teste, y_pred)
```

Fonte: Elaborado pelo autor.

Ao final da execução de cada algoritmo de predição de dados, é gerado o cálculo de Acurácia, Precisão, Recall e F1, com apoio das bibliotecas *accuracy_score, precision_score, recall_score, f1_score* do pacote *sklearn.metrics.* Para garantir que a previsão foi realizada com os melhores recursos para o cenário, o método de predição verifica qual dos algoritmos apresentou uma acurácia melhor na execução do processamento dos dados do aluno. A partir dessa verificação, é definida a origem dos dados previstos para o estudante, como a média prevista e a definição se o aluno será aprovado ou não.

Os dados gerados pelo melhor algoritmo de predição são submetidos a um cálculo de margem de erro considerando o percentual de acurácia do melhor algoritmo, onde considera-se que o aluno pode ter um risco de reprovação ainda maior caso a margem de acerto do algoritmo seja mais distante de 100%. Ao final deste processamento, os dados são persistidos em uma entidade do banco de dados para registro da média calculada, juntamente com a definição de aprovação ou reprovação por parte do algoritmo preditivo.

Vale ressaltar que os dados gerados na etapa de predição geram variáveis para execução da filtragem colaborativa, filtragem por conteúdo e no refinamento das recomendações temporárias, tendo em vista que o processo de predição gerará insumos para determinar o nível de complexidade das recomendações, a quantidade de recomendações necessárias e o grau de representatividade destas recomendações para o estudante. O detalhamento de como estes dados são processados nas etapas seguintes são apresentados nas seções seguintes.

5.4.3 Definição de complexidade

Para recomendação dos objetos de aprendizagem, são aplicados os métodos de filtragem conforme definido na metodologia e apresentado com maiores detalhes nas seções abaixo. Um dos critérios usados para recomendação dos OA é o nível de complexidade destes objetos, ou seja, o nível de complexidade das recomendações deve ser relacionado ao nível de avanço nos estudos por parte do aluno.

Neste cenário, foi definida uma escala de 1 a 3, onde 1 corresponde ao nível de complexidade menor e 3 corresponde a maior complexidade. Durante a execução do *framework*, a etapa de predição faz uma projeção do desempenho do aluno e com esta projeção, é definido o nível de complexidade que mais se encaixa ao estudante no momento da execução do algoritmo.

Com os dados gerados na predição de desempenho, o nível de complexidade é definido da seguinte forma:

- Se a média prevista for inferior à 5, então a complexidade é 1;
- Se a média prevista estiver entre 5 e 8, então a complexidade é 2;
- Se a média prevista for superior à 8, então a complexidade é 3.

Esta definição de complexidade pode ser alterada nas configurações do *framework,* para que possa encaixar-se em definições distintas quanto ao nível de complexidade relacionado ao aluno.

Além da definição de complexidade com base na média de desempenho prevista para o aluno, no momento de realizar a recomendação dos objetos de aprendizagem, o *framework* considera mais dois critérios para definir uma segunda complexidade, sendo as seguintes:

- Se a previsão de desempenho definir que o aluno tem chance de reprovação, o nível de complexidade das recomendações é decrementado em 1 nível;
- Se o aluno não tiver recebido nenhuma recomendação, serão geradas recomendações para o nível de complexidade calculado, conforme apresentado anteriormente, porém também serão geradas recomendações com o nível de complexidade 1, tendo em vista que será uma introdução do tema para o aluno.

Com os processamentos e validações realizadas, o aluno poderá receber recomendação de objetos de aprendizagem com diferentes complexidades, cada uma entrando no cenário específico, conforme detalhado nesta seção.

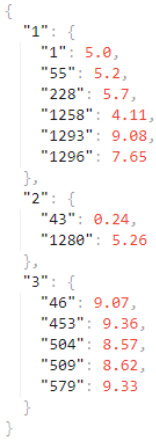
5.4.4 Filtragem Colaborativa

Com a predição do desempenho do aluno realizada, o *framework* é direcionado para as etapas de filtragem de objetos de aprendizagem relevantes para o usuário. O primeiro método de filtragem aplicado é a filtragem colaborativa, que conforme apresentado na fundamentação teórica, consiste em recomendar objetos com base na similaridade de outros alunos, ou seja, se um objeto de aprendizagem foi importante para um aluno, o mesmo pode ser considerado para outros estudantes com um perfil semelhante.

A partir das definições da filtragem colaborativa, foi implementada uma classe específica para geração destas recomendações. Para identificação dos alunos semelhantes, é realizada uma busca na base de dados a partir das avaliações realizadas por estudantes em objetos de aprendizagem já catalogados na base de dados. Estes dados foram fornecidos ao método de filtragem em um dicionário de

dados no formato *json.* A Figura 25 representa um modelo de arquivo *json* gerado, onde é gerado um formato de chave e valor, composto pelo código do aluno, o código dos objetos de aprendizagem avaliados e a nota que foi definida pelo aluno no correspondente OA.

Figura 25 - Dicionário de dados para Filtragem Colaborativa



```
{
  "1": {
    "1": 5.0,
    "55": 5.2,
    "228": 5.7,
    "1258": 4.11,
    "1293": 9.08,
    "1296": 7.65
  },
  "2": {
    "43": 0.24,
    "1280": 5.26
  },
  "3": {
    "46": 9.07,
    "453": 9.36,
    "504": 8.57,
    "509": 8.62,
    "579": 9.33
  }
}
```


Fonte: Elaborado pelo autor.

Os dados presentes no *json* apresentado acima foram gerados por meio de uma classe específica, conforme representado na Figura 26, é realizada uma coleta de cada aluno que realizou alguma avaliação em objetos de aprendizagem, e para cada objeto avaliado, é formado um objeto do tipo *json,* contendo os dados necessários para identificar a similaridade dos alunos e então, efetuar o processo de recomendação.

Figura 26 - Formação de dicionário de dados para Filtragem Colaborativa

```
def buscarAvaliacoesObjetos(self):
    cursor = conexao.conn.cursor()
    sqlAlunos = f'SELECT DISTINCT(A.ALUNO_ID), N.NOME  FROM AVALIACAO A, ALUNO N WHERE N.ID = A.ALUNO_ID'
    cursor.execute(sqlAlunos)
    alunos = cursor.fetchall()

    i = 0
    listaObjetos = []
    listaNotas = []
    listaAvaliacoes = []

    while i < len(alunos):
        aluno = (alunos[i][0])
        nomeAluno = buscaDadosAluno.buscaNomeAluno(aluno)
        alunoLista = '"' + str(aluno) + '"'
        sqlAvaliacoes = f'SELECT OBJETO_ID, NOTA FROM AVALIACAO WHERE ALUNO_ID = {aluno}'
        cursor = conexao.conn.cursor()
        cursor.execute(sqlAvaliacoes)
        avaliacoes = cursor.fetchall()

        j = 0
        while j < len(avaliacoes):
            objetoAprendizagem = str(avaliacoes[j][0])
            notaAvaliacao = str(avaliacoes[j][1])
            listaObjetos.append('"')
            listaObjetos.append(objetoAprendizagem)
            listaObjetos.append('":')
            listaObjetos.append(notaAvaliacao)
            if j < (len(avaliacoes) - 1):
                listaObjetos.append(',')
            j += 1

        join = ''.join(listaObjetos)
        json_cru_temp = {
            f'{alunoLista}' + f':' + f'''"{"'' + join
        }
        listaAvaliacoes.append(json_cru_temp)
        listaObjetos.clear()
        listaNotas.clear()
        i += 1
```

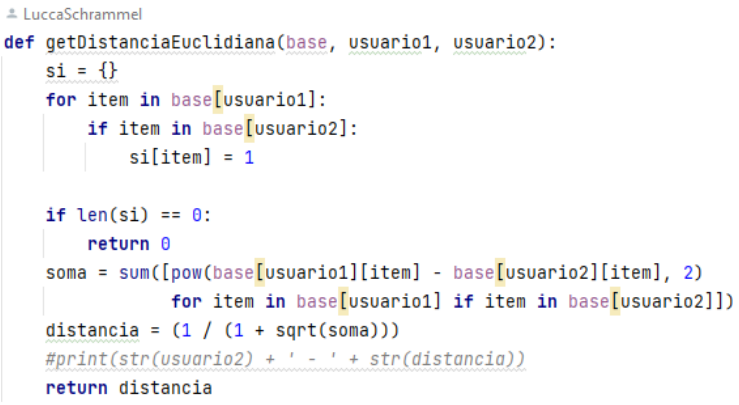
Fonte: Elaborado pelo autor.

Com o dicionário de dados, as variáveis necessárias para encontrar os alunos semelhantes já estão presentes no algoritmo, então o *framework* realiza a chamada do método responsável pela filtragem colaborativa.

Como primeiro passo da filtragem colaborativa, o algoritmo faz a identificação dos alunos similares por meio do cálculo da distância euclidiana. A partir deste valor, é possível identificar os alunos semelhantes ao aluno que está em processo de recomendação. Conforme ilustrado na Figura 27, foi implementado um método

específico para cálculo da distância euclidiana entre o aluno que receberá as recomendações com cada aluno do dicionário de dados. O método recebe como entrada o dicionário de dados formado conforme detalhado acima e então, percorre todos os usuários presentes e realiza o cálculo da distância entre o aluno da recomendação e os alunos presentes no dicionário de dados.

Figura 27 - Cálculo da distância euclidiana



Fonte: Elaborado pelo autor.

O método apresentado na Figura 27 é utilizado de forma recursiva pelo método responsável por gerar as recomendações. Conforme ilustrado na Figura 28, foi implementado um método para geração das recomendações a partir da similaridade entre os demais alunos existentes. Esse método é responsável por receber o dicionário de dados descrito anteriormente, e para cada aluno do dicionário de dados, calcular a similaridade com o aluno que receberá as recomendações. Caso este aluno esteja dentro da similaridade mínima configurada, é realizada uma inclusão de recomendação temporária para o estudante.

Figura 28 - Geração de recomendação por filtragem colaborativa

```
def getRecomendacoesUsuario(base, usuario):
    if usuario in base:
        totais = {}
        somaSimilaridade = {}
        for outro in base:
            if outro == usuario:
                continue
            similaridade = filtragemColaborativa.getDistanciaEuclidiana(base, usuario, outro)

            if similaridade <= 0:
                continue

            for item in base[outro]:
                if item not in base[usuario]:
                    totais.setdefault(item, 0)
                    totais[item] += base[outro][item] * similaridade
                    somaSimilaridade.setdefault(item, 0)
                    somaSimilaridade[item] += similaridade
        rankings = [(total / somaSimilaridade[item], item) for item, total in totais.items()]
        rankings.sort()
        rankings.reverse()
        #print(rankings)

        i = 0
        cursor = conexao.conn.cursor()
        sqlTruncate = f'TRUNCATE TABLE RECOMENDACAOTEMP'
        cursor.execute(sqlTruncate)
        sqlRow = f'SELECT * FROM RECOMENDACAOTEMP'
        cursor.execute(sqlRow)
        cont1 = cursor.rowcount

        if cont1 == 0:
            idMax = 1
        else:...

        contadorInsert = 0
        while i < len(rankings):...

        print(":) " + str(contadorInsert) + " recomendações da filtragem colaborativa registradas com sucesso!!!")
```

Fonte: Elaborado pelo autor.

Neste momento, o aluno recebeu uma lista de recomendações de objetos de aprendizagem que os estudantes similares aprovaram no passado, o que não significa necessariamente que se trata de um objeto de aprendizagem que será do interesse e de grande relevância para o estudante que está recebendo as recomendações. Para que isto ocorra, as demais etapas de filtragem devem ser aplicadas, juntando técnicas para formação das melhores recomendações possíveis.

Pode existir um cenário em que não existam dados suficientes para determinar alunos similares ao estudante que receberá as recomendações, o que acaba gerando o problema de *cold start*. Conforme descrito na fundamentação teórica deste estudo,

este problema é algo recorrente na filtragem colaborativa, especialmente nos casos em que não existam insumos suficientes para calcular a similaridade dos alunos, o que seria um caso de um aluno novo que não realizou nenhuma interação com os objetos de aprendizagem disponíveis. Caso este cenário ocorra, o *framework* não incrementa nenhuma recomendação para o aluno na etapa de filtragem colaborativa, o que fará com que apenas objetos de aprendizagem considerados na filtragem por conhecimento e conteúdo serão recomendados para o aluno.

5.4.5 Filtragem por Conteúdo

A filtragem por conteúdo é uma das etapas mais importantes presente no *framework* para recomendação de objetos de aprendizagem. Foi implementado um método exclusivo para realizar este tipo de filtragem, dado que será realizada a recomendação de objetos de aprendizagem que se encaixam nas necessidades do aluno.

O método implementado recebe os seguintes dados:

- Aluno – Identificador do aluno que receberá as recomendações;
- Perfil do aluno – Perfil de aprendizagem do aluno que receberá as recomendações;
- Nível de dificuldade – Nível de dificuldade específico para o aluno, gerado na etapa de predição de dados;
- Etapa de ensino – Etapa de ensino em que o aluno se encontra em relação ao plano de ensino da disciplina;
- Detalhes da etapa de ensino – Detalhamento da etapa de ensino do aluno, coletado na etapa da busca de dados;
- Definição de aprovado – Definição se o aluno será aprovado ou não, gerado na etapa de predição de dados.

Os dados recebidos via parâmetro foram gerados nas etapas executadas até este momento, e servirão como variáveis para geração das recomendações pelo modelo de filtragem por conteúdo e filtragem por conhecimento.

Em um primeiro momento, conforme ilustrado na Figura 29, é realizada a projeção de todos os objetos de aprendizagem presentes na base de dados, juntamente com alguns de seus metadados que serão utilizados no processo de

filtragem. Essa projeção parte de uma consulta na base de dados e então o preenchimento de um *data frame* com os dados coletados.

Figura 29 - Projeção de dados para filtragem por conteúdo

```
dados = pd.read_sql_query(sql=
                          '''
SELECT
    MAIN.OBJETOID AS OBJETOAPRENDIZAGEM,
    MAX(CASE
            WHEN MAIN.METADADOID = 2
            THEN MAIN.VALOR
        END) AS TITLE,
    MAX(CASE
            WHEN MAIN.METADADOID = 3
            THEN MAIN.VALOR
        END) AS DESCRIPTION,
    STRING_AGG((CASE
            WHEN MAIN.METADADOID = 4
            THEN MAIN.VALOR
        END), ',') AS KEYWORDS,

    MAX(CASE
            WHEN MAIN.METADADOID = 10
            THEN MAIN.VALOR
        END) AS TYPE,
    MAX(CASE
            WHEN MAIN.METADADOID = 12
            THEN MAIN.VALOR
        END) AS DIFFICULTY,
    MAX(CASE
            WHEN MAIN.METADADOID = 25
            THEN MAIN.VALOR
        END) AS TAXON
FROM
    OBJETOMETADADO MAIN
GROUP BY
    1
''', con=engine)
```

Fonte: Elaborado pelo autor.

Neste momento, o *data frame* apresentado possui os dados de todos os objetos de aprendizagem existentes na base de dados, o que permite que sejam aplicados filtros para então coletar e recomendar os objetos de aprendizagem que mais encaixam-se ao aluno.

Anterior ao processo de geração das recomendações, algumas regras são aplicadas:

- Caso o aluno tenha uma previsão com risco de reprovação, o nível de dificuldade dos objetos de aprendizagem que serão recomendados será decrementado em 1 nível;
- Caso o aluno ainda não tenha recebido nenhuma recomendação para o tema, o nível de complexidade dos objetos de aprendizagem e definido em “1”, o que corresponde ao nível de complexidade mais baixo.

Com as regras de complexidade aplicadas e o *data frame* preenchido com todos os objetos de aprendizagem, o método de filtragem inicia o processo de busca pelos objetos de aprendizagem. São efetuadas três etapas de filtragem, sendo que os seguintes dados são filtrados em variados momentos:

- Nível de complexidade do objeto de aprendizagem;
- Título do objeto de aprendizagem;
- Descrição do objeto de aprendizagem;
- Palavras chaves do objeto de aprendizagem.

Em todas as filtragens, são considerados apenas os objetos de aprendizagem que estão dentro da complexidade sugerida ao aluno, além do fato de que os objetos devem ser relacionados a etapa de ensino atual do estudante. A Figura 30 representa uma das etapas de filtragem aplicadas sobre os objetos de aprendizagem contidos no *data frame.* Após cada uma das filtragens, é inserido um registro na entidade de recomendações temporárias, contendo os dados do aluno e do objeto de aprendizagem recomendado.

Figura 30 - Etapa da filtragem por conteúdo

```
dados_filtrado = \
        dados[
                ((dados['difficulty'] == str(nivelDificuldadeAluno))            |
                (dados['difficulty'] == str(nivelDificuldadeAlunoInicial)))     &
                (dados['taxon'] == str(perfilAluno))                            &
                (dados['description'].str.contains(keysEtapaEnsino))
        ]
registrosInseridos = 0
# Percorre todas as linhas do DataFrame
for indice, linha in dados_filtrado.iterrows():
        objetoAprendizagem = linha['objetoaprendizagem']
        tipoFiltragem = "Filtragem por Conteúdo"
        cursor.execute('INSERT INTO RECOMENDACAOTEMP VALUES(\'{}\', \'{}\', \'{}\', \'{}\', \'{}\')'
                       .format(int(idMax), int(aluno), int(objetoAprendizagem), tipoFiltragem, 1))
        conexao.conn.commit()
        registrosInseridos = registrosInseridos + 1
        idMax = idMax + 1
```

Fonte: Elaborado pelo autor.

Todos os objetos de aprendizagem que se encaixam nos filtros aplicados são inseridos na entidade de recomendações temporárias. Neste cenário, um objeto de aprendizagem pode ser inserido mais de uma vez, desde que em momentos distintos do processo de filtragem de dados. Para otimizar as recomendações geradas pelos métodos de filtragem colaborativa, filtragem por conteúdo e filtragem por conhecimento são refinadas na etapa seguinte, ou seja, existe um método seguinte que determina com maior precisão o que deve ou não ser considerado como recomendação válida ao aluno.

5.4.6 Refinamento

Até este momento, foram geradas recomendações pelos métodos de filtragem colaborativa, conteúdo e conhecimento. Estas recomendações estão mantidas em uma entidade de recomendações temporárias, pois necessitam de um refinamento para então gerar as recomendações oficiais.

Para execução deste refinamento, foi implementada uma classe com um método responsável por realizar essa filtragem final. Ao final do processo de recomendação, a classe principal direciona os dados para o método de refinamento. Neste método, inicialmente, conforme ilustrado pela Figura 31, as recomendações temporárias são carregadas em uma lista de recomendações que será refinada pelo método em questão.

Figura 31 - Busca dos dados para refinamento

```
sqlRecomendacoes = f'''
                    SELECT
                        R.OBJETOAPRENDIZAGEM_ID,
                        COUNT(*),
                        O.VALOR
                    FROM
                        RECOMENDACAOTEMP R,
                        OBJETOMETADADO O
                    WHERE
                        R.OBJETOAPRENDIZAGEM_ID = O.OBJETOID AND
                        O. METADADOID = 12 AND
                        R.SEQUENCIA = 2 AND
                        R.OBJETOAPRENDIZAGEM_ID IN
                            (SELECT OBJETOAPRENDIZAGEM_ID FROM RECOMENDACAOTEMP WHERE
                                TIPORECOMENDACAO = 'Filtragem por Conteúdo')
                    GROUP BY R.OBJETOAPRENDIZAGEM_ID, O.VALOR
                    ORDER BY 2 DESC
                    '''

cursor.execute(sqlRecomendacoes)
recomendacoes = cursor.fetchall()
```

Fonte: Elaborado pelo autor.

Com os dados presentes na lista de recomendações temporárias, antes de iniciar o processo de geração das recomendações oficiais, são definidas algumas variáveis:

- A quantidade de recomendações gravada será correspondente com a respectiva configuração do aluno, avaliando a quantidade de recomendações se aprovado ou reprovado;
- O nível de relevância das recomendações deve atender a configuração relacionada se aluno aprovado ou reprovado;
- Cálculo de *score* para cada recomendação.

Conforme apresentado, a quantidade de recomendações que serão gravadas e a relevância destas recomendações são originadas de uma configuração presente no *framework,* ou seja, é possível aplicar esta configuração para que a recomendação atenda casos. Além disto, o método de filtragem final aplica alguns cálculos para definição do *score* de cada recomendação, o que determina a relevância desta recomendação para o aluno.

O cálculo da relevância da recomendação atende os seguintes critérios, conforme ilustrado na Figura 32:

- O *framework* realiza 4 etapas de filtragem. A cada etapa atendida pelo objeto de aprendizagem, o seu *score* é incrementado, pois atende mais requisitos de filtragem;
- Conforme o nível de complexidade do objeto de aprendizagem, é realizado um acréscimo percentual na recomendação, desta forma, as recomendações com nível mais simples terão uma relevância inicial maior;
- Se o objeto de aprendizagem foi recomendado pelos algoritmos de filtragem colaborativa e filtragem por conteúdo, é realizado um novo incremento sobre o *score* calculado;

Figura 32 - Cálculo de relevância da recomendação

```
score = ((100 / 4) * contRecomendacoes)
if score < 100:
    if nivelComplexidade == 1:
        score = (score + (score * configuracoes.
                          configuracoes['filtragemFinal']['adicionalScoreNivelUm'] / 100))
    if nivelComplexidade == 2:
        score = (score + (score * configuracoes.
                          configuracoes['filtragemFinal']['adicionalScoreNivelDois'] / 100))
    if nivelComplexidade == 3:
        score = (score + (score * configuracoes.
                          configuracoes['filtragemFinal']['adicionalScoreNivelTres'] / 100))

if (score < 100 and cont > 0):
    score = (score + (score * configuracoes.
                      configuracoes['filtragemColaborativa']['adicionalFiltagemColaborativa'] / 100))
    if score > 100:
        score = 100
```

Fonte: Elaborado pelo autor.

Após a realização do cálculo de relevância, é aplicado o filtro para garantir que as recomendações inseridas estejam dentro da relevância configurada. As recomendações que atenderem todos os filtros aplicados serão inseridas em uma entidade de recomendações oficiais, com os dados do aluno, dados do objeto de aprendizagem e o nível de relevância desta recomendação.

A Figura 33 apresenta a validação das configurações para inclusão das recomendações oficiais. Caso a recomendação já tenha sido feita em algum momento passado, o *framework* toma a decisão a partir de uma configuração se repetirá a

recomendação ou se a mesma será ignorada, tendo em vista que a mesma já foi realizada no passado.

Figura 33 - Inclusão de recomendações oficiais

```
if score >= nivelRelevanciaRecomendacoes:
    sqlVerificaRecomendacoesExistentes = f'SELECT * FROM RECOMENDACAOFINAL ' \
                                         f'WHERE ALUNO = {aluno} ' \
                                         f'AND OBJETOAPRENDIZAGEM = {objetoAprendizagem} ' \
                                         f'AND ETAPAENSINO = {etapaEnsino}'
    cursor.execute(sqlVerificaRecomendacoesExistentes)
    contRecomendacoesExistentes = cursor.rowcount
    if contRecomendacoesExistentes == 0:
        cursor.execute('INSERT INTO RECOMENDACAOFINAL VALUES(\'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\')'
                       .format(int(idMax), aluno, int(objetoAprendizagem), score, int(etapaEnsino), dataAtual))
        conexao.conn.commit()
        contadorInserts = contadorInserts + 1
        idMax = idMax + 1

    if contRecomendacoesExistentes > 0 and \
            configuracoes.configuracoes['filtragemFinal']['permiteRecomendacaoExistente'] == "true":
        cursor.execute('INSERT INTO RECOMENDACAOFINAL VALUES(\'{}\', \'{}\', \'{}\', \'{}\', \'{}\', \'{}\')'
                       .format(int(idMax), aluno, int(objetoAprendizagem), score, int(etapaEnsino), dataAtual))
        conexao.conn.commit()
        contadorInserts = contadorInserts + 1
        idMax = idMax + 1

    if contRecomendacoesExistentes > 0 and \
            configuracoes.configuracoes['filtragemFinal']['permiteRecomendacaoExistente'] == "false":
        recomendacoesNaoInseridas = recomendacoesNaoInseridas + 1

    if i >= len(recomendacoes):
        exit()

i = i + 1
```

Fonte: Elaborado pelo autor.

Com as recomendações oficiais inseridas na base de dados, o aluno possuirá os dados necessários para uso dos objetos de aprendizagem recomendados. Cabe a aplicação que está integrada ao *framework* apresentar estas recomendações para uso por parte dos alunos.

O processo de recomendação pode ser executado diversas vezes para o mesmo aluno, pois o estudante avançará as etapas conforme o plano de ensino, apresentará novas interações e desempenho conhecido, o que acaba gerando novas variáveis para realizar recomendações atualizadas para o aluno.

5.4.7 Configurações para Recomendações

Para geração das recomendações oficiais e execução dos métodos propostos pelo *framework,* existem algumas configurações que podem ser aplicadas para personalizar a funcionalidade dos algoritmos.

Para aplicação e leitura destas configurações, foi criada uma classe específica para manter estes dados. Conforme ilustrado na Figura 34, as configurações são mantidas em um dicionário de dados no formato *json*, dessa forma, cada um dos métodos faz a leitura das configurações para então aplicar sobre os métodos de filtragem, busca de dados e geração de dados.

Figura 34 - Dicionário de dados de configurações

```
configuracoes = \
  {
    "filtragemFinal": {
      "adicionalScoreNivelUm": 10,
      "adicionalScoreNivelDois": 5,
      "adicionalScoreNivelTres": 2,
      "permiteRecomendacaoExistente": "false"
    },
    "filtragemColaborativa": {
      "proximidadeMinima": 5,
      "adicionalFiltagemColaborativa": 15
    },
    "predicao": {
      "numeroVizinhos": 4,
      "algoritmoPadrao": "Naive Bayes"
    },
    "geral": {
      "nivelRelevanciaRecomendacoes": 50,
      "quantidadeRecomendacoes": 5
    },
    "regrasReprovacao": {
      "nivelRelevanciaRecomendacoes": 70,
      "quantidadeRecomendacoes": 10
    },
```

Fonte: Elaborado pelo autor.

As configurações possíveis e suas aplicações podem ser identificadas na Tabela 5:

Tabela 5 - Lista de configurações utilizadas pelo *framework*

Configuração	**Aplicação**
filtragemFinal.adicionalScoreNivelUm	Determina o percentual que será adicionado ao *score* das recomendações oficiais quando objeto com nível de complexidade igual à um.
filtragemFinal.adicionalScoreNivelDois	Determina o percentual que será adicionado ao *score* das recomendações oficiais quando objeto com nível de complexidade igual à dois.
filtragemFinal.adicionalScoreNivelTres	Determina o percentual que será adicionado ao *score* das recomendações oficiais quando objeto com nível de complexidade igual à três.
filtragemColaborativa.proximidadeMinima	Determina a proximidade mínima dos alunos similares contidos no dicionário de dados para realização da filtragem colaborativa.
filtragemColaborativa. adicionaFiltragemColaborativa	Determina o percentual adicional que será aplicado em recomendações que estiverem inseridas no processo de filtragem colaborativa + filtragem por conteúdo.
predicao.numeroVizinhos	Determina o número de vizinhos mais próximos que serão considerados para predição de dados.
predicao.algoritmoPadrao	Determina o algoritmo de predição padrão que será utilizado caso a validação do melhor algoritmo falhar.

Configuração	Aplicação
geral.nivelRelevanciaRecomendacoes	Determina o nível de relevância mínima das recomendações para inserir na entidade de recomendações oficiais, quando aluno não tiver risco de reprovação.
geral.quantidadeRecomendacoes	Determina a quantidade de recomendações oficiais que serão inseridas na respectiva entidade, quando aluno não tiver risco de reprovação.
regrasReprovacao. nivelRelavanciaRecomendacoes	Determina o nível de relevância mínima das recomendações para inserir na entidade de recomendações oficiais, quando aluno tiver risco de reprovação.
regrasReprovacao. quantidadeRecomendacoes	Determina a quantidade de recomendações oficiais que serão inseridas na respectiva entidade, quando aluno tiver risco de reprovação.
database.dbName	Nome da base de dados integrada pelo *framework.*
database.user	Nome do usuário da base de dados integrada pelo *framework.*
database.password	Senha de conexão com a base de dados integrada pelo *framework.*
database.port	Porta de conexão com o servidor da base de dados integrada pelo *framework.*
database.host	IP do servidor para conexão da base de dados integrada pelo *framework.*
geracaoDados.apiKeyChatGpt	API Key padrão para integração com *chat* GPT pelos métodos responsáveis pela geração da massa de dados.

Configuração	Aplicação
geracaoDados.modelChatGpt	Modelo de conexão com *chat* GPT pelos métodos responsáveis pela geração da massa de dados.
geracaoDados.urlChatGpt	URL padrão para conexão com *chat* GPT para geração da massa de dados.

Fonte: Elaborado pelo autor.

Com as configurações listadas, é possível personalizar alguns comportamentos do *framework* para realização das recomendações de objetos de aprendizagem aos alunos, sem que seja necessário manipular as regras de negócio implementadas em cada método de recomendação e busca de dados.

5.5 REPOSITÓRIO

A implementação do *framework* juntamente com a base de dados, conforme apresentado pelo estudo, pode ser encontrado no repositório *GitHub* do autor, por meio da seguinte URL: https://github.com/LuccaSchrammel/recommendation.

6 VALIDAÇÃO

Este capítulo tem como objetivo apresentar e discutir a metodologia e os resultados obtidos na execução de experimentos com o *framework* proposto. Busca-se demonstrar a real aplicação dos algoritmos implementados, bem como o processo de predição pode ser um diferencial para antecipar recomendações de objetos de aprendizagem para estudantes em diversos contextos.

O processo de validação de dados faz uso do *dataset* gerado para testes dos algoritmos implementados. O *dataset* conta com 10.271 alunos cadastrados, 6.144 objetos de aprendizagem, 116.879 metadados, vinculados a cada um dos objetos de aprendizagem existentes. Para realização da predição do desempenho, foram gerados dados históricos de 5.005 alunos. Este número de dados formam a base inicial para execução dos métodos de filtragem e recomendação propostos pelo estudo.

Na Seção 6.1 é apresentada a metodologia utilizada para realizar a validação do *framework* estudado e implementado. As demais seções deste capítulo apresentam maiores detalhes sobre como a validação foi realizada considerando a metodologia proposta, bem como os resultados obtidos.

6.1 PARÂMETROS DO *FRAMEWORK*

Para iniciar o processo de validação, foram definidas as parametrizações do *framework.* Estes parâmetros afetam os resultados obtidos, dessa forma, é importante que sejam considerados e apresentados para garantir que as recomendações estão respeitando as parametrizações aplicadas.

A Tabela 6 apresenta as parametrizações utilizadas para validação do estudo, juntamente com o valor atribuído para cada parâmetro. Nas seções abaixo é possível confirmar se estas parametrizações realmente estão sendo consideradas pelo *framework* na realização das recomendações.

Tabela 6 - Parâmetros do *framework*

Parâmetro	**Valor do Parâmetro**
adicionalFiltragemColaborativa	15
adicionalScoreNivelDois	5
adicionalScoreNivelTres	2

Parâmetro	Valor do Parâmetro
adicionalScoreNivelUm	10
algoritmoPadrao	Naive Bayes
geral.nivelRelevanciaRecomendacoes	50
geral.quantidadeRecomendacoes	5
numeroVizinhos	4
permiteRecomendacoesExistentes	False
proximidadeMinima	5
reprovacao.nivelRelevanciaRecomendacoes	70
reprovacao.quantidadeRecomendacoes	10

Fonte: Elaborado pelo autor.

Os parâmetros apresentados na Tabela acima permitem que sejam realizadas alterações no comportamento dos algoritmos, e consequentemente, alterar as recomendações que são geradas pelo *framework,* conforme a necessidade em cada aplicação.

Para a validação deste estudo, os parâmetros não serão alterados em nenhum cenário de teste, a fim de não impactar os cenários e as combinações para geração das recomendações. Na verificação dos resultados é possível confirmar se as parametrizações foram respeitadas.

6.2 CENÁRIOS DE TESTE

Conforme citado até este momento, o processo de validação deste estudo consiste na elaboração de cenários de teste, aplicar estes cenários sob o *framework,* coletar as recomendações geradas e então realizar um teste de mesa para garantir que as recomendações realmente atendem os cenários aplicados.

A Tabela 7 apresenta a lista de cenários gerados e aplicados sob o *framework* para coleta das recomendações geradas. Os resultados obtidos em cada um dos cenários podem ser identificados na Seção 6.4 para verificação do comportamento do *framework* em cada um dos cenários aplicados.

Tabela 7 - Cenários para validação

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
1	Sensorial, Visual, Ativo e Sequencial	2	False	2
2	Sensorial, Visual, Ativo e Sequencial	2	False	1
3	Intuitivo, Visual, Ativo e Sequencial	3	True	2
4	Intuitivo, Verbal, Ativo e Sequencial	4	True	2
5	Intuitivo, Verbal, Reflexivo e Sequencial	5	True	3
6	Intuitivo, Visual, Reflexivo e Global	2	False	1
7	Sensorial, Visual, Reflexivo e Global	3	True	2
8	Intuitivo, Visual, Ativo e Global	4	False	1
9	Sensorial, Visual, Ativo e Global	4	True	2
10	Intuitivo, Verbal, Reflexivo e Global	Final	True	2

Fonte: Elaborado pelo autor.

Cada etapa de ensino se refere a um momento do plano de ensino que o aluno se encontra no momento. Conforme apresentado na Tabela 7, as etapas dos cenários estão descritas com um código sequencial. Para melhor compreensão na leitura dos resultados e das recomendações geradas, abaixo segue o detalhe dos assuntos abordados em cada uma das etapas do plano de ensino usado como base para o estudo em questão:

- **Etapa 1**: Esta etapa determina que sejam abordados temas relacionados a engenharia de software, sendo uma introdução para a disciplina abordada. Além disso, devem ser apresentados conceitos iniciais e algumas ferramentas case para utilização na engenharia de software;
- **Etapa 2**: Esta etapa determina que sejam abordados temas mais relacionados ao processo de desenvolvimento de software, tais como metodologias ágeis e modelos tradicionais. Além disto, são esperados temas relacionados ao ciclo de vida de um processo de desenvolvimento de software;
- **Etapa 3**: Esta etapa determina que sejam abordados os diagramas da engenharia de software, como caso de uso, sequência e classes. Devem

ser apresentadas definições destes diagramas, além da modelagem e criação destes;

- **Etapa 4**: Esta etapa determina que sejam abordados temas relacionados a requisitos de software, bem como todos tópicos que envolvem este tema;
- **Etapa 5**: Por fim, a etapa 5 determina que sejam apresentados assuntos relacionados a teste de software e arquitetura de sistemas computacionais.

Os dados das etapas de ensino são considerados para realizar a filtragem dos objetos de aprendizagem necessários em um determinado momento da disciplina que está em execução, fazendo com que não sejam realizadas recomendações gerais ou genéricas.

6.3 RESULTADOS E DISCUSSÕES

Após a execução do *framework* para cada cenário apresentado na seção anterior, foram coletadas as recomendações para que seja feita a verificação se as recomendações geradas realmente atendem aquilo que estava proposto pelo estudo. Os resultados de cada cenários são apresentados em tabelas especificadas para cada um destes cenários definidos, tendo em vista que são apresentadas e comparadas uma série de informações para tornar a visão mais clara possível para o leitor.

Na sequência, são apresentados os resultados agrupados em cada um dos cenários modelados e apresentados na seção 6.3.

Cenário 1:

A Tabela 8 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 8 - Validação | cenário 1

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
1	Sensorial, Visual, Ativo e Sequencial	2	False	2

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 35 apresenta o log de execução do *framework* para o aluno 172 da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 35 - Validação | cenário 1

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\lu
Iniciando processo de recomendação para o aluno: 172 - Srta. Fabiana Martines Sobrinho
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 2
Média Calculada: 6.59
Média Prevista: 6.26
Aprovado: False
Precisão: 95.01
Nível Complexidade: 2
:) 22 recomendações da filtragem colaborativa registradas com sucesso!!!
:) 121 recomendações da filtragem por conteúdo registradas com sucesso !!!
:) 9 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor

A Tabela 9 apresenta as variáveis que foram usadas no teste do cenário 1, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 9 - Variáveis | cenário 1

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
172	7.82	2	7.82, 7.71, 6.92, 5.15, 5.34	6.59	False

Fonte: Elaborado pelo autor.

A Tabela 10 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 10 - Recomendações | cenário 1

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
3813	Os 5 principais benefícios do modelo de desenvolvimento em cascata.	Benefícios do processo cascata.	Redução de custos, Foco no processo, Documentação completa e Controle de riscos	1	82.50
3824	Por que o modelo cascata é a escolha superior para o desenvolvimento de software.	Vantagens do modelo cascata em relação a outros modelos de desenvolvimento.	Controle de qualidade, Fácil gerenciamento, Previsibilidade do processo, Linearidade, Documentação	1	82.50
3884	Lidando com atrasos no processo cascata: estratégias eficazes para manter o projeto no prazo.	Como lidar com atrasos no processo cascata.	Qualidade Priorização, Flexibilidade, Comunicação.	1	82.50
3898	Conduzindo a gestão de processos de negócios de forma eficiente no modelo cascata.	Como lidar com a gestão do processo de negócios no processo cascata.	Monitoramento, Controle, Implementação, Design, Análise.	1	82.50
3934	A chave para o sucesso da entrega de projetos: a análise de requisitos de interoperabilidade no modelo cascata.	A importância da análise de requisitos de interoperabilidade no processo cascata.	Eficiência, Compatibilidade, Entrega.	2	78.75
3967	A importância da análise de requisitos de qualidade na efetividade do processo cascata.	A importância da análise de requisitos de qualidade no processo cascata.	Satisfação do cliente, Comunicação, Riscos.	1	82.50
5693	A importância da estimativa de prazos na metodologia tradicional de gerenciamento de projetos.	Estimativa de prazos na metodologia tradicional.	Cronograma, Praz, Análise de riscos, Recursos.	1	82.50
15885	Gerenciamento eficiente do ciclo de vida de software: maximizando o retorno do investimento.	Planejamento do ciclo de vida de software.	Requisitos, Design, Desenvolvimento, Manutenção.	1	82.50

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
26031	Comunicando-se em equipe: a chave para um ciclo de vida de desenvolvimento de software bem-sucedido.	Comunicação em equipe no ciclo de vida de desenvolvimento de software.	Colaboração, Transparência, Comunicação constante, Gestão de conflitos.	2	78.75

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. Este teste foi realizado manualmente para cada um dos cenários, avaliando o cenário aplicado e se as recomendações geradas realmente estão vinculadas ao cenário de teste. Todas as recomendações geradas e os objetos com possibilidade de recomendação foram avaliadas pelo autor a fim de identificar a real relação com os cenários de teste utilizados. A Tabela 11 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 11 - Critérios de aceite | cenário 1

Tema	Nível	Perfil	Relevância	Qtd	Média
100%	100% ***	100%	100%	90% ***	98%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 8,57%
- F-measure: 15,78%
- Cobertura: 0,14%

Observações:

1. Foram geradas recomendações com nível de complexidade inferior ao esperado, pois os algoritmos de recomendação reduzem o nível de complexidade caso o aluno tenha risco de reprovação. Este comportamento pôde ser confirmado neste cenário.
2. A quantidade de recomendações ficou abaixo do esperado pela falta de objetos de aprendizagem que se encaixam nas variáveis do cenário de teste.

Cenário 2:

A Tabela 12 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 12 - Validação | cenário 2

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
2	Sensorial, Visual, Ativo e Sequencial	2	False	1

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 36 apresenta o log de execução do *framework* para o aluno 654 da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 36 - Validação | cenário 2

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\User
Iniciando processo de recomendação para o aluno: 654 -  Gabriel Felipe Pereira Rocha
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 2
Média Calculada: 4.87
Média Prevista: 4.58
Aprovado: False
Precisão: 94.11
Nível Complexidade: 1
:) 19 recomendações da filtragem colaborativa registradas com sucesso!!!
:) 75 recomendações da filtragem por conteúdo registradas com sucesso !!!
:) 7 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor

A Tabela 13 apresenta as variáveis que foram usadas no teste do cenário 2, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 13 - Variáveis | cenário 2

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
654	1.50	2	1.50, 6.29, 6.20, 5.97, 4.37	4.87	False

Fonte: Elaborado pelo autor.

A Tabela 14 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 14 - Recomendações | cenário 2

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
3813	Os 5 principais benefícios do modelo de desenvolvimento em cascata.	Benefícios do processo cascata.	Redução de custos, Foco no processo, Documentação, Controle de riscos, Previsibilidade.	1	82.50
3824	Por que a modelo cascata é a escolha superior para o desenvolvimento de software.	Vantagens do modelo cascata em relação a outros modelos de desenvolvimento.	Controle de qualidade, Fácil gerenciamento, Previsibilidade do processo., Documentação, Linearidade.	1	82.50
3884	Lidando com atrasos no processo cascata: estratégias eficazes para manter o projeto no prazo.	Como lidar com atrasos no processo cascata.	Qualidade Priorização, Flexibilidade, Comunicação.	1	82.50
3898	Conduzindo a gestão de processos de negócios de forma eficiente no modelo cascata.	Como lidar com a gestão do processo de negócios no processo cascata.	Monitoramento, Controle, Implementação, Design, Análise.	1	82.50
3967	A importância da análise de requisitos de qualidade na efetividade do processo cascata.	A importância da análise de requisitos de qualidade no processo cascata.	Satisfação do cliente, Comunicação, Riscos.	1	82.50
5693	A importância da estimativa de prazos na metodologia tradicional de gerenciamento de projetos.	Estimativa de prazos na metodologia tradicional.	Cronograma, Praz, Análise de riscos, Recursos.	1	82.50
5885	Gerenciamento eficiente do ciclo de vida de software: maximizando o retorno do investimento.	Planejamento do ciclo de vida de software.	Requisitos, Design, Desenvolvimento, Manutenção, Aposentadoria.	1	80.50

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 15 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 15 - Critérios de aceite | cenário 2

Tema	Nível	Perfil	Relevância	Qtd	Média
85% ***	100%	100%	100%	70% ***	91%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 85%
- Revocação: 10,29%
- F-measure: 18,35%
- Cobertura: 0,11%

Observações:

1. A quantidade de recomendações foi inferior ao previsto pela ausência de objetos de aprendizagem relacionados às variáveis aplicadas.
2. Das recomendações realizadas, uma delas não atingiu todos os critérios na filtragem de conteúdo em relação ao tema/assunto do objeto de aprendizagem.

Cenário 3:

A Tabela 16 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 16 - Validação | cenário 3

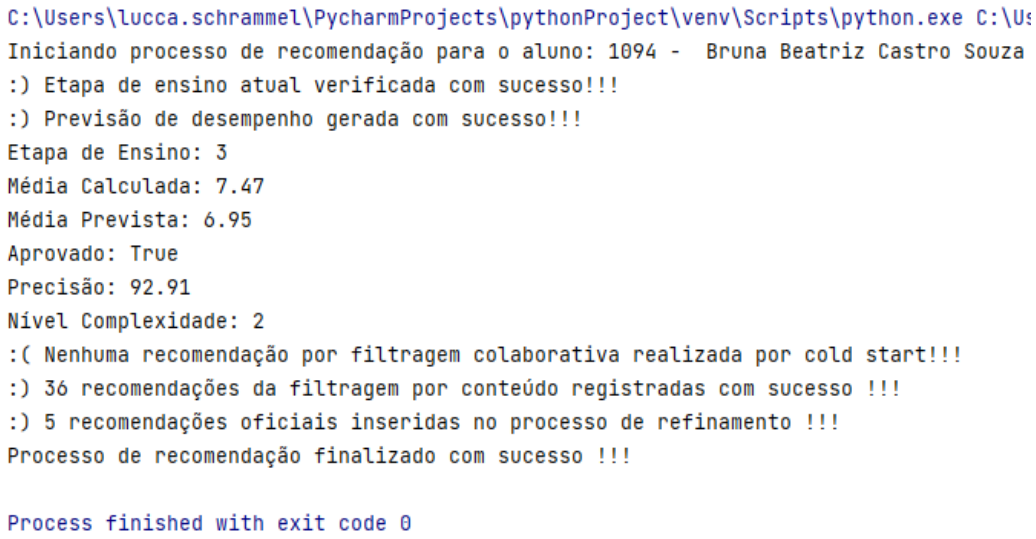
Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
3	Intuitivo, Visual, Ativo e Sequencial	3	True	2

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 37 apresenta o log de execução do *framework* para o aluno 654

da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 37 - Validação | cenário 3



Fonte: Elaborado pelo autor.

A Tabela 17 apresenta as variáveis que foram usadas no teste do cenário 3, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 17 - Variáveis | cenário 3

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
1094	9.10, 8.90	3	9.10, 8.90, 7.28, 5.85, 5.63	7.47	True

Fonte: Elaborado pelo autor.

A Tabela 18 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 18 - Recomendações | cenário 3

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
1460	Diagrama de caso de uso para sistema de acompanhamento de progresso acadêmico.	Exemplo de diagrama de caso de uso para sistema de acompanhamento de progresso acadêmico.	Academia, Progresso acadêmico, Estudantes, Professores, Sistemas de Informação Acadêmica.	2	52.50
1502	Validação efetiva do diagrama de caso de uso: técnicas e estratégias.	Técnicas de validação de diagrama de caso de uso.	Revisão por pares, Análise de cenário, Modelagem formal, Teste de aceitação., Teste de funcionalidade.	1	55.00
1649	Desenvolvendo soluções com eficiência: Implementação de sistemas através de diagramas de caso de uso.	Implementação de sistemas com base em diagramas de caso de uso.	Requisitos, Funcionalidades, Atores, Fluxo de eventos, Validadores.	1	82.50
3571	Explorando as melhores ferramentas de modelagem para otimizar o desenvolvimento de software.	Ferramentas de modelagem de dados de software.	Diagrama de Entidade-Relacionamento, Modelagem conceitual de dados, Linguagem de modelagem de dados (ex: UML), Normalização de dados.	2	78.75
6423	Desvendando os segredos do Diagrama de Caso de Uso e sua integração com os Testes de Software.	Webinar: diagrama de caso de uso e testes de software.	Diagrama de caso de uso, Testes de software, Análise de requisitos, Desenvolvimento de software, Qualidade de software.	2	52.50

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 19 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 19 - Critérios de aceite | cenário 3

Tema	Nível	Perfil	Relevância	Qtd	Média
100%	60% ***	100%	100%	100%	92%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 22,72%
- F-measure: 37,02%
- Cobertura: 0,08%

Observações:

1. O nível de complexidade de algumas recomendações foi inferior ao definido no cenário, pois o aluno ainda não possuía nenhuma recomendação registrada em base de dados, neste caso, o *framework* realiza a recomendação de alguns objetos de aprendizagem em um nível inferior ao relacionado com o desempenho do estudante.

Cenário 4:

A Tabela 20 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 20 - Validação | cenário 4

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
4	Intuitivo, Verbal, Ativo e Sequencial	4	True	2

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 38 apresenta o log de execução do *framework* para o aluno 701

da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 38 - Validação | cenário 4

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe
Iniciando processo de recomendação para o aluno: 701 -  Sophia Martins Alves
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 4
Média Calculada: 7.68
Média Prevista: 7.3
Aprovado: True
Precisão: 94.51
Nível Complexidade: 2
:( Nenhuma recomendação por filtragem colaborativa realizada por cold start!!!
:) 73 recomendações da filtragem por conteúdo registradas com sucesso !!!
:) 5 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor.

A Tabela 21 apresenta as variáveis que foram usadas no teste do cenário 4, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 21 - Variáveis | cenário 4

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
701	8.50, 8.87, 8.00	4	8.50, 8.87, 8.00, 8.00, 5.05	7.68	True

Fonte: Elaborado pelo autor.

A Tabela 22 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 22 - Recomendações | cenário 4

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
1648	A importância dos diagramas de caso de uso na análise de requisitos: Um guia passo a passo.	Análise de requisitos com diagramas de caso de uso.	Ator, Fluxo principal, Fluxo alternativo, Cenário, Requisito funcional.	2	78.75
3873	Garantindo a excelência da documentação no processo cascata.	Como garantir a qualidade da documentação no processo cascata.	Revisão, Padronização, Monitoramento, Participação, Testes.	2	78.75
3975	A análise de requisitos de gestão da comunicação: a chave para um processo cascata bem-sucedido.	A importância da análise de requisitos de gestão da comunicação no processo cascata.	Eficiência, Melhoria contínua, Comunicação, Tempo e orçamento.	1	82.50
6405	Garantindo o sucesso do projeto: validação do diagrama de caso de uso com os stakeholders.	Como validar um diagrama de caso de uso com stakeholders.	Compreensão, Relevância, Consistência, Testabilidade, Compartilhamento.	2	78.75
6421	Diagrama de caso de uso como ferramenta para identificar funcionalidades essenciais do sistema.	Como utilizar o diagrama de caso de uso para identificar as funcionalidades cruciais do sistema.	"Diagrama de caso de uso, Identificação de funcionalidades, Requisitos do sistema, Análise de requisitos, Interface do usuário.	2	78.75

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 23 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 23 - Critérios de aceite | cenário 4

Tema	Nível	Perfil	Relevância	Qtd	Média
100%	80%	100%	100%	100%	96%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 9,80%
- F-measure: 17,85%
- Cobertura: 0,08%

Observações:

1. Foi realizada a recomendação de objetos de aprendizagem com complexidade nível 1 pelo fato de não existirem recomendações anteriores para este aluno, conforme definido pelas regras de complexidade.

Cenário 5:

A Tabela 24 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 24 - Validação | cenário 5

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
5	Intuitivo, Verbal, Reflexivo e Sequencial	5	True	3

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 39 apresenta o log de execução do *framework* para o aluno 4881 da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado. Neste cenário, foi realizado o processo de recomendação para um aluno que já possuía recomendações na base de dados, dessa forma, é possível verificar as regras de complexidade sendo aplicadas, juntamente com a não repetição de recomendações, conforme definido nas configurações do *framework.*

Figura 39 - Validação | cenário 5

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\
Iniciando processo de recomendação para o aluno: 4881 -  Marcos da Silva Carvalho
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 5
Média Calculada: 9.09
Média Prevista: 8.64
Aprovado: True
Precisão: 94.51
Nível Complexidade: 3
:) 26 recomendações da filtragem colaborativa registradas com sucesso!!!
:) 17 recomendações da filtragem por conteúdo registradas com sucesso !!!
:( 1 recomendações não inseridas por já estarem recomendadas ao aluno !!!
:) 5 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor.

A Tabela 25 apresenta as variáveis que foram usadas no teste do cenário 5, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 25 - Variáveis | cenário 5

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
4881	10, 9.50, 9.87, 9	5	10, 9.50, 9.87, 9, 7.06	9.09	True

Fonte: Elaborado pelo autor.

A Tabela 26 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 26 - Recomendações | cenário 5

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
2743	SocialAdManager: Gerenciando suas campanhas de publicidade em redes sociais de forma eficiente.	Ferramenta de gerenciamento de anúncios em redes sociais.	Segmentação de público-alvo, otimização de desempenho, análise de métricas, personalização de criativos, automação de campanhas.	3	51.00
3710	Avaliando a eficiência: Testes de desempenho em destaque.	Testes de desempenho.	Carga de trabalho, Escalabilidade, Concorrência, Tempo de resposta, Utilização de recursos (CPU, memória, rede)	3	51.00
5397	Automatizando a Qualidade: Ferramentas CASE para Geração Automática de Testes.	Ferramentas case para geração automática de testes.	Automação de testes, Teste de software, Ferramentas de teste, Geração automática de testes, Teste de interface do usuário.	3	51.00
6183	Análise de requisitos para sistemas eficientes de gerenciamento de projetos de infraestrutura: Identificando as demandas e otimizando os processos.	Análise de requisitos para sistemas de gerenciamento de projetos de infraestrutura.	Riscos, Escopo Cronograma, Recursos, Comunicação	3	76.50
6603	Descobrindo insights na Inteligência Artificial através de testes exploratórios.	Testes exploratórios em inteligência artificial.	Dados, Algoritmo, Interpretabilidade, Adversarialidade, Generalização.	3	51.00

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 27 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 27 - Critérios de aceite | cenário 5

Tema	Nível	Perfil	Relevância	Qtd	Média
80%	100%	100%	100%	100%	96%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 80%
- Revocação: 10,41%
- F-measure: 18,40%
- Cobertura: 0,08%

Observações:

1. Como já existiam recomendações para este aluno, o *framework* optou por recomendar apenas temas com o nível de complexidade relacionado ao estudante, respeitando as regras de complexidade apresentadas anteriormente.

Cenário 6:

A Tabela 28 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 28 - Validação | cenário 6

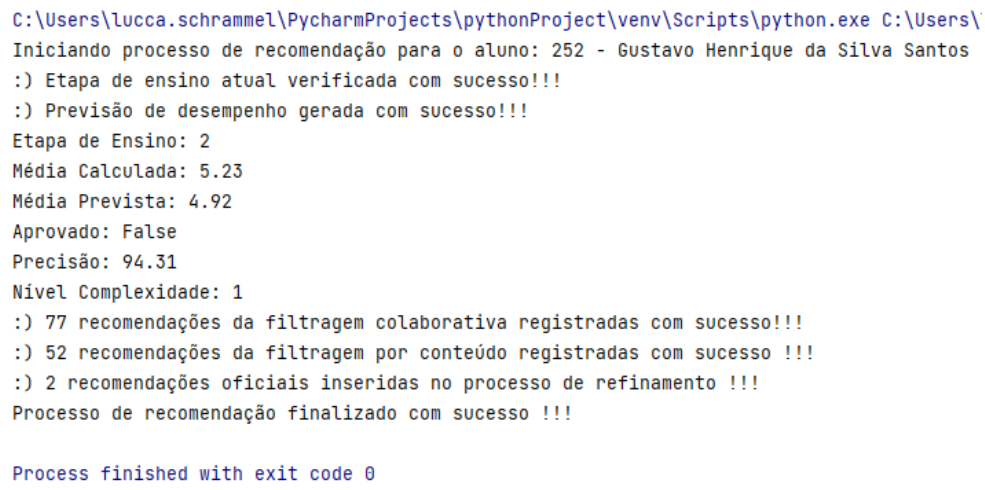
Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
6	Intuitivo, Visual, Reflexivo e Global	2	False	1

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 40 apresenta o log de execução do *framework* para o aluno 252

da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 40 - Validação | cenário 6



Fonte: Elaborado pelo autor.

A Tabela 29 apresenta as variáveis que foram usadas no teste do cenário 6, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 29 - Variáveis | cenário 6

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
252	0.5	2	0.5, 6.48, 8.26, 5.75, 5.17	5.23	False

Fonte: Elaborado pelo autor.

A Tabela 30 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 30 - Recomendações | cenário 6

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
4039	Capacitação para o sucesso: Treinando sua equipe para o modelo cascata de desenvolvimento de software.	Como treinar a equipe para trabalhar com o modelo cascata de desenvolvimento de software.	Treinamento técnico, Habilidade de comunicação, Organização, Gestão de riscos, Trabalho em equipe.	1	82.50
5425	Ferramentas CASE para análise de impacto: Simplificando mudanças de processos.	Ferramentas case para análise de impacto de mudanças de processos.	Modelagem de processos, Planejamento da mudança, Análise de impacto, Gerenciamento de mudança, Automação de fluxo de trabalho.	1	82.50

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 31 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 31 - Critérios de aceite | cenário 6

Tema	Nível	Perfil	Relevância	Qtd	Média
100%	100%	100%	100%	20%	84%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 1,76%

- F-measure: 3,45%
- Cobertura: 0,03%

Observações:

1. A quantidade de recomendações foi inferior ao previsto, tendo em vista que não foram encontrados mais itens na base de conhecimento que se encaixam no cenário aplicado.

Cenário 7:

A Tabela 32 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 32 - Validação | cenário 7

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
7	Sensorial, Visual, Reflexivo e Global	3	True	2

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 41 apresenta o log de execução do *framework* para o aluno 4178 da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 41 - Validação | cenário 7

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\lu
Iniciando processo de recomendação para o aluno: 4178 -  Maria Eduarda Oliveira da Costa
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 3
Média Calculada: 7.84
Média Prevista: 7.29
Aprovado: True
Precisão: 92.81
Nível Complexidade: 2
:) 13 recomendações da filtragem colaborativa registradas com sucesso!!!
:) 52 recomendações da filtragem por conteúdo registradas com sucesso !!!
:) 5 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor.

A Tabela 33 apresenta as variáveis que foram usadas no teste do cenário 7, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 33 - Variáveis | cenário 7

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
4178	9.00, 8.5	3	9.00, 8.50, 8.04, 6.85, 6.79	7.84	True

Fonte: Elaborado pelo autor.

A Tabela 34 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 34 - Recomendações | cenário 7

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
1419	Guia prático para criar um diagrama de caso de uso com ferramentas de modelagem.	Como criar um diagrama de caso de uso utilizando ferramentas de modelagem.	Modelagem de casos de uso, Ferramentas de modelagem, UML (Unified Modeling Language), Requisitos de software, Diagrama de casos de uso.	1	82.50
1504	Identificação de atores no diagrama de caso de uso: uma abordagem prática.	Como identificar atores no diagrama de caso de uso.	Atividades, Papel, Comportamento, Função, Interação.	1	82.50
1646	Diagrama de Caso de Uso: Uma ferramenta indispensável para modelagem de sistemas eficientes.	Modelagem de sistemas com diagrama de caso de uso.	Ator, caso de uso, fluxo básico, fluxo alternativo, relacionamento entre atores e casos de uso.	2	78.75

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
5365	Ferramentas CASE para modelagem de diagramas de sequência: Uma análise comparativa.	Ferramentas case para modelagem de diagramas de sequência.	UML (Unified Modeling Language), Visualização de sequência, Diagrama de sequência, Modelagem de software, Fluxo de processo.	1	82.50
6485	Modelando a interação do sistema com outras soluções usando o diagrama de caso de uso.	Como utilizar o diagrama de caso de uso para modelar a interação do sistema com outras soluções.	Interação, Diagrama de caso de uso, Modelagem, Sistema, Soluções externas.	1	82.50

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 35 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 35 - Critérios de aceite | cenário 7

Tema	Nível	Perfil	Relevância	Qtd	Média
100%	100%***	100%	100%	100%	100%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 10%
- F-measure: 18,18%
- Cobertura: 0,08%

Observações:

1. Foram geradas recomendações com o nível de complexidade igual à 1 pois, não existiam recomendações anteriores para o aluno.

Cenário 8:

A Tabela 36 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 36 - Validação | cenário 8

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
8	Intuitivo, Visual, Ativo e Global	4	False	1

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 42 apresenta o log de execução do *framework* para o aluno 922 da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 42 - Validação | cenário 8

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\l
Iniciando processo de recomendação para o aluno: 922 -  Cristian David Garcia Rodriguez
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 4
Média Calculada: 3.81
Média Prevista: 3.54
Aprovado: False
Precisão: 93.21
Nível Complexidade: 1
:) 13 recomendações da filtragem colaborativa registradas com sucesso!!!
:) 47 recomendações da filtragem por conteúdo registradas com sucesso !!!
:) 8 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor.

A Tabela 37 apresenta as variáveis que foram usadas no teste do cenário 8, juntamente com os dados gerados pela etapa de predição de dados. Com estes

dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 37 - Variáveis | cenário 8

Dados existentes			Dados gerados na previsão		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
922	0.50, 4.50, 2.30	4	0.50, 4.50, 2.30, 7.13, 4.63	3.81	False

Fonte: Elaborado pelo autor.

A Tabela 38 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 38 - Recomendações | cenário 8

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
1972	Elicitando requisitos com eficiência: Técnicas avançadas para o sucesso do projeto.	Técnicas de elicitação de requisitos.	Grupos focais., Questionários estruturados, Prototipagem rápida, Observação de usuários, Entrevistas com stakeholders.	1	82.50
3960	Análise de requisitos de integração: a chave para o sucesso do processo cascata.	A importância da análise de requisitos de integração no processo cascata.	Compatibilidade, Eficiência, Confiabilidade, Reutilização, Coerência.	1	82.50
3968	A importância da análise de requisitos de comunicação no sucesso do processo cascata.	A importância da análise de requisitos de comunicação no processo cascata.	Compreensão, Planejamento, Controle, Identificação de riscos, Qualidade.	1	82.50
5418	Ferramentas CASE: Análise de impacto de mudanças de requisitos em tempo real.	Ferramentas case para análise de impacto de mudanças de requisitos.	Controle de versão., Gestão de projetos, Ferramentas CASE, Análise de impacto, Mudança de requisitos.	1	82.50

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
5482	UsaCase: Ferramenta CASE para análise de requisitos de usabilidade de sistemas.	Ferramentas case para análise de requisitos de usabilidade de sistemas.	Usabilidade, Ferramentas CASE, Análise de requisitos, Testes de usabilidade, Design centrado no usuário.	1	82.50
5514	Ferramentas CASE para garantir a manutenibilidade dos sistemas médicos: Uma análise de requisitos.	Ferramentas case para análise de requisitos de manutenibilidade de sistemas médicos.	Manutenibilidade, Sistemas médicos, Análise de requisitos, Ferramentas CASE, Gestão de mudanças.	1	82.50
5530	Case Tools: análise de requisitos de segurança em interfaces móveis.	Ferramentas case para análise de requisitos de segurança de sistemas de interface do usuário para dispositivos móveis.	Segurança, Dispositivos móveis, Análise de requisitos, Interface do usuário, Ferramentas CASE.	1	82.50
5541	PerformanceAnalyzerUI: Uma ferramenta CASE para análise de requisitos de performance de sistemas de interface do usuário em tempo real.	Ferramentas case para análise de requisitos de performance de sistemas de interface do usuário em tempo real.	Ferramentas CASE, Análise de requisitos de performance, Sistemas de interface do usuário, Tempo real, Testes de desempenho.	1	82.50

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário de teste aplicado. A Tabela 39 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 39 - Critérios de aceite | cenário 8

Tema	Nível	Perfil	Relevância	Qtd	Média

100%	100%	100%	100%	80%	96%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 21,62%
- F-measure: 35,55%
- Cobertura: 0,12%

Observações:

1. A quantidade de recomendações não foi atingida pela falta de objetos de aprendizagem que se encaixem nos parâmetros para recomendação.

Cenário 9:

A Tabela 40 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 40 - Validação | cenário 9

Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
9	Sensorial, Visual, Ativo e Global	4	True	2

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 43 apresenta o log de execução do *framework* para o aluno 2846 da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 43 - Validação | cenário 9

```
C:\Users\lucca.schrammel\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users
Iniciando processo de recomendação para o aluno: 2846 - Pedro Henrique Oliveira Souza
:) Etapa de ensino atual verificada com sucesso!!!
:) Previsão de desempenho gerada com sucesso!!!
Etapa de Ensino: 4
Média Calculada: 7.55
Média Prevista: 7.1
Aprovado: True
Precisão: 93.71
Nível Complexidade: 2
:( Nenhuma recomendação por filtragem colaborativa realizada por cold start!!!
Dificuldade Inicial: 2
Dificuldade calculada: 2
:) 33 recomendações da filtragem por conteúdo registradas com sucesso !!!
:( 1 recomendações não inseridas por já estarem recomendadas ao aluno !!!
:) 5 recomendações oficiais inseridas no processo de refinamento !!!
Processo de recomendação finalizado com sucesso !!!

Process finished with exit code 0
```

Fonte: Elaborado pelo autor.

A Tabela 41 apresenta as variáveis que foram usadas no teste do cenário 9, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 41 - Variáveis | cenário 9

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
2846	8.75, 8.50, 9.40	4	8.75, 8.50, 9.40, 4.88, 6.22	7.55	True

Fonte: Elaborado pelo autor.

A Tabela 42 apresenta os objetos de aprendizagem que foram gerados como saída do *framework*. Estes objetos de aprendizagem devem atender os critérios de aceite para confirmar a real aplicação para o cenário de teste aplicado.

Tabela 42 - Recomendações | cenário 9

Objetos de Aprendizagem Recomendados					
ID	**Título**	**Descrição**	**Palavras Chave**	**Nv**	***Score***
3816	Análise de Requisitos na Abordagem Cascata: Etapa Crítica para o Sucesso do Desenvolvimento de Software.	Análise de requisitos no processo cascata.	Especificação de requisitos, Documentação de requisitos, Análise de requisitos, Verificação de requisitos, Validação de requisitos.	2	52.50
3931	A relevância da análise de requisitos de manutenibilidade no modelo cascata de desenvolvimento de software.	A importância da análise de requisitos de manutenibilidade no processo cascata.	Eficiência, qualidade, economia, segurança, produtividade.	2	52.50
5505	Usabilidade em E-commerce: Analisando Requisitos com Ferramentas CASE.	Ferramentas case para análise de requisitos de usabilidade de sistemas para e-commerce.	Usabilidade, e-commerce, requisitos, análise, ferramentas case.	2	52.50
5875	Análise de requisitos para gerenciamento de projetos internacionais: uma abordagem tradicional.	Análise de requisitos de gestão de projetos internacionais na metodologia tradicional.	Comunicação intercultural, Cooperação internacional, Gerenciamento de riscos globais, Sustentabilidade global, Adaptação cultural e regulatória.	2	52.50
6452	Utilizando o Diagrama de Caso de Uso para Garantir a Segurança do Sistema: Dicas e Orientações.	Artigo: como utilizar o diagrama de caso de uso para identificar os requisitos de segurança do sistema.	Diagrama de caso de uso, Requisitos de segurança, Identificação de requisitos, Sistema, Uso do diagrama.	2	52.50

Fonte: Elaborado pelo autor.

Com as recomendações geradas, é necessário realizar um teste de mesa para confirmar que realmente os objetos de aprendizagem serão relevantes para o cenário

de teste aplicado. A Tabela 43 apresenta os critérios que foram atendidos pelas recomendações geradas.

Tabela 43 - Critérios de aceite | cenário 9

Tema	Nível	Perfil	Relevância	Qtd	Média
100%	100%	100%	100%	100%	100%

Fonte: Elaborado pelo autor.

Após a análise das recomendações geradas pelo *framework*, foram realizados os cálculos de métricas de avaliação de recomendações geradas, conforme apresentado no referencial teórico desta pesquisa. Os resultados obtidos na execução do cenário em questão podem ser verificados abaixo:

- Precisão: 100%
- Revocação: 11,36%
- F-measure: 20,40%
- Cobertura: 0,08%

Observações:

1. Não foram geradas recomendações com nível de complexidade 1 pelo fato de já existirem recomendações para este aluno na base de dados.

Cenário 10:

A Tabela 44 apresenta o cenário específico que foi considerado para realização das recomendações. Estas recomendações devem ser associadas às características do cenário que está sendo aplicado.

Tabela 44 - Validação | cenário 10

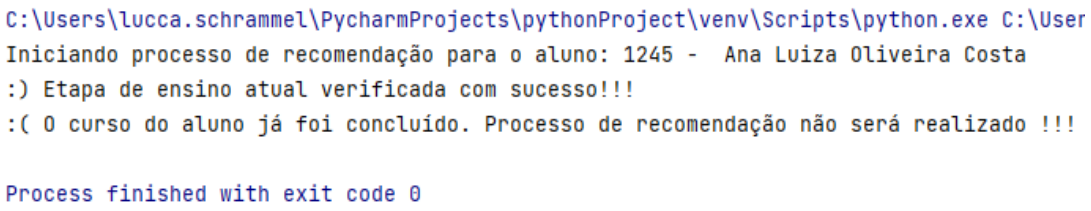
Seq.	Perfil de Aprendizagem	Etapa	Aprovado	Nível
10	Intuitivo, Verbal, Reflexivo e Global	Final	True	2

Fonte: Elaborado pelo autor.

Com o cenário definido, foi realizada a execução do *framework* para as variáveis propostas. A Figura 44 apresenta o log de execução do *framework* para o aluno 1245

da base de dados, registrando o vínculo dos dados entre o cenário proposto e o cenário executado.

Figura 44 - Validação | cenário 10



Fonte: Elaborado pelo autor.

A Tabela 45 apresenta as variáveis que foram usadas no teste do cenário 10, juntamente com os dados gerados pela etapa de predição de dados. Com estes dados, é possível verificar como ocorreu o processamento dos dados nas etapas anteriores a geração das recomendações.

Tabela 45 - Variáveis | cenário 10

Dados existentes			**Dados gerados na previsão**		
Aluno	**Notas**	**Etapa**	**Notas**	**Média**	**Aprovado**
1245	4.86, 9.46, 8.13, 8.48, 8.92	Final	4.86, 9.46, 8.13, 8.48, 8.92	7.97	True

Fonte: Elaborado pelo autor.

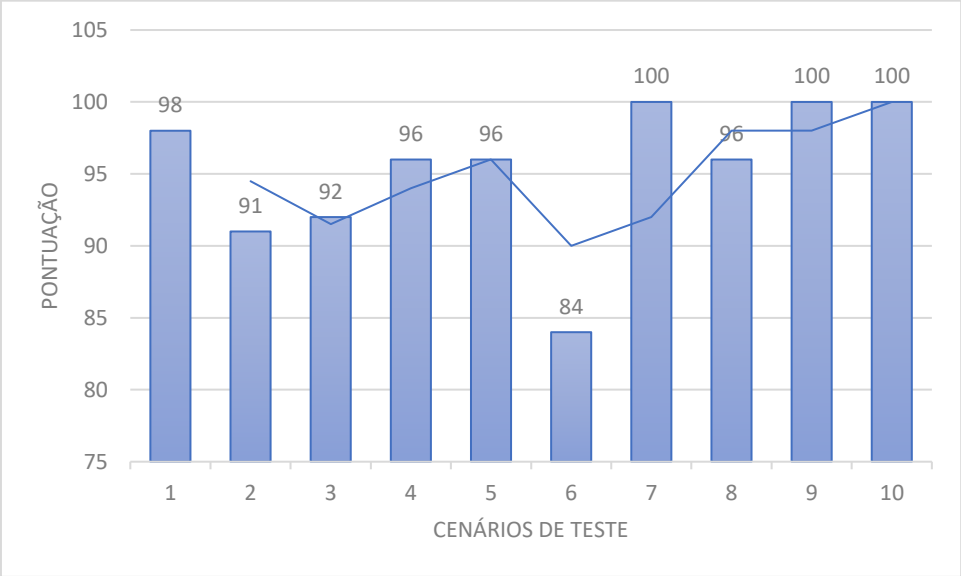
Como o aluno já havia finalizado a matéria, o *framework* não realizou nenhuma recomendação, pois todas as etapas já foram superadas. Neste caso, as métricas das recomendações não foram calculadas, pelo fato de não existir recomendações para serem consideradas nos cálculos.

Com o cenário 10 executado, todos os cenários propostos para validação foram finalizados, dessa forma, todos os resultados obtidos já podem ser avaliados para identificar o comportamento do *framework* em diferentes casos de aplicação.

Para facilitar a visualização dos dados, na sequência serão apresentadas algumas informações que fazem menção a todos os cenários aplicados, ou seja, será possível visualizar de uma forma mais resumida os resultados obtidos com as recomendações geradas em cada um dos casos aplicados.

Com base nos testes em cada um dos cenários, foi feito um cálculo para verificar quantos critérios foram totalmente atendidos nas etapas de filtragem. A Figura 45 apresenta a pontuação obtida em cada um dos cenários aplicados na validação. Pode-se evidenciar que a maior parte dos cenários superou os 95% de acerto em relação às métricas avaliadas.

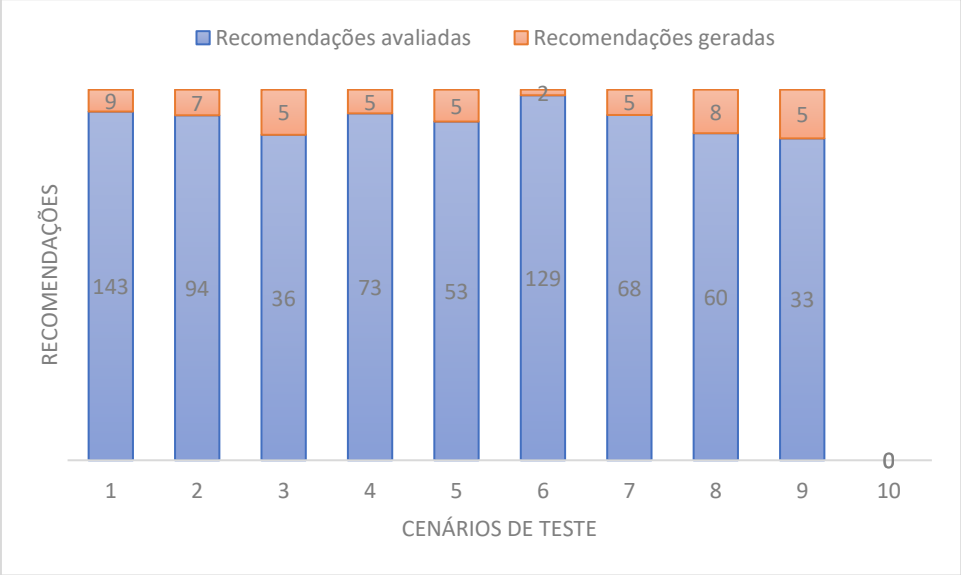
Figura 45 - Gráfico de pontuação dos cenários



Fonte: Elaborado pelo autor.

A Figura 46 apresenta as combinações de recomendações possíveis e as recomendações oficialmente geradas após a execução de todos os algoritmos do *framework.* É possível identificar que em cada um dos cenários, uma quantidade considerável de objetos de aprendizagem foi, em algum momento da execução dos algoritmos, contabilizada como possíveis recomendações para o aluno, entretanto, ao final do processo, apenas a quantidade destacada em laranja (Figura 46) realmente foi recomendada para o aluno. Por meio deste gráfico, é possível evidenciar o papel do *framework* em busca dos objetos de aprendizagem que mais se encaixam nas necessidades do aluno, seguindo todas as combinações e parametrizações aplicadas para geração das recomendações.

Figura 46 - Gráfico de recomendações

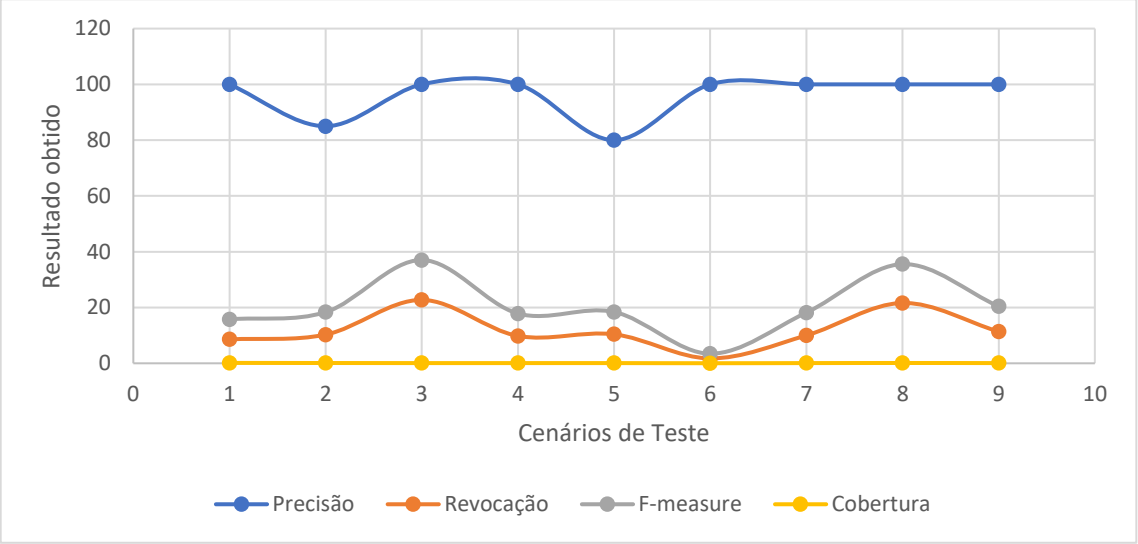


Fonte: Elaborado pelo autor.

Conforme apresentado em cada cenário, após cada teste aplicado foram calculadas as métricas de análise destas recomendações. Estas métricas atendem o padrão apresentado no referencial teórico desta pesquisa. A Figura 47 apresenta de forma gráfica os dados obtidos em cada um dos cenários de teste aplicados. A partir da análise realizada da Figura 47, é possível evidenciar as seguintes situações:

1. A precisão das recomendações manteve-se acima de 80% em todos os cenários de teste.
2. A cobertura de recomendações geradas manteve-se muito baixa em todos os cenários, o que apresenta um comportamento de filtragem restritivo, gerando um número baixo de recomendações com base em todo o conjunto de dados possíveis para recomendação.
3. A revocação também se manteve com um valor baixo, o que destaca que por mais que foram consideradas várias possibilidades de recomendações, apenas os objetos de aprendizagem com maior probabilidade de aceitação pelos estudantes foram recomendados.

Figura 47 - Métricas das recomendações



7 CONCLUSÃO

A partir do estudo, desenvolvimento e validação do *framework*, foi possível evidenciar que existem alternativas para geração de recomendações antecipadas para alunos ou usuários em outros cenários.

Anterior a implementação, foram identificados casos em que as recomendações são geradas após o conhecimento dos padrões de cada estudante, porém esta pesquisa apresenta uma alternativa que busca antecipar eventuais dificuldades no aprendizado de cada aluno.

Os algoritmos implementados fazem trabalham com dados passados para projetar o futuro e então realizar recomendações, que conforme a validação, são alternativas que poderão ser utilizadas no processo de aprendizagem, não se limitando a um nível específico de ensino, sendo possível implementar em vários cenários que permitem o uso de recursos digitais.

Como contribuição, o estudo apresenta o conceito de predição que pode ser incrementado ao processo de recomendação de objetos de aprendizagem. Dessa forma, estudantes poderão ter acesso a OA para uso no processo de aprendizagem, tentando antecipar eventuais necessidades específicas. Os algoritmos de filtragem podem ser acoplados ao processo de predição para garantir que as recomendações geradas realmente apresentem um impacto positivo no processo de aprendizagem do estudante. Espera-se que as recomendações possam ser aproveitas pelos estudantes, tendo em vista que, na validação dos algoritmos, os cenários apresentaram uma precisão satisfatória nas recomendações geradas.

Como limitação da pesquisa, fica a ausência de uma validação em um cenário real para aplicação de um estudo ou análise que possam comprovar o benefício ao receber as recomendações geradas pelo *framework* proposto. Além desta limitação, o *framework* proposto necessita de dados para que as recomendações sejam geradas, como alunos, objetos de aprendizagem e dados históricos. Estes dados devem existir previamente em uma base de dados para uso por parte dos algoritmos de recomendação, não sendo de responsabilidade do framework a geração destes dados na base de dados, caso contrário, o funcionamento dos algoritmos de filtragem e recomendação serão impactados, não sendo aplicados em sua totalidade.

No segundo semestre de 2023, a pesquisa apresentada por meio deste texto foi publicada em formato de artigo científico no Congresso Brasileiro de Informática na Educação (SBIE), um evento anual da Sociedade Brasileira de Computação (SBC) que promove e incentiva as trocas de experiências entre as comunidades científica, profissional, governamental e empresarial na área de Informática na Educação

7.1 TRABALHOS FUTUROS

Existem várias possibilidades para implementações de trabalhos futuros, desde a melhoria dos algoritmos como a adição de novas funcionalidades do *framework.* Neste momento, é possível destacar as seguintes possibilidades de trabalhos futuros:

- Adição de novas métricas na predição de dados: Em casos de poucos dados para gerar a predição, seria interessante possuir outros dados sobre os alunos para realizar a recomendação de objetos de aprendizagem, como idade, sexo, escolaridade, região, frequência, interações com objetos de aprendizagem, entre outros. Com estes dados, seria possível encontrar ainda mais padrões e alunos com semelhanças para então realizar a predição e as recomendações;
- Aplicação do *framework* em um ambiente real: Conforme apresentado, o *framework* foi implementado e validado em um ambiente controlado. Para evidenciar o funcionamento em um cenário real, seria importante formar uma base de conhecimento e então gerar as recomendações para alunos de uma instituição de ensino. Com essa aplicação, pode ser realizado um estudo para comparar o desempenho de estudantes que tiveram auxílio das recomendações e alunos que não fizeram parte do estudo;
- Implementar uma API *REST* para integrar ao *backend* e então gerar as recomendações para outras aplicações web, app ou *desktop*.

As opções de trabalhos futuros mencionadas podem colaborar com o presente estudo para confirmar ainda mais a real importância de realizar as recomendações específicas para cada aluno, tendo como base o perfil de aprendizagem.

REFERÊNCIAS

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. **IEEE Transactions on Knowledge and Data Engineering**, v. 17, n. 6, p. 734–749, jun. 2005.

AGUIAR, Eliane Vigneron Barreto; FLÔRES, Maria Lucia Pozzatti. **Objetos de aprendizagem: Teoria e prática**. Porto Alegre: Evangraf, 2014.

AHA, D. W.; KIBLER, D.; ALBERT, M. K. **Machine Learning**, v. 6, n. 1, p. 37–66, 1991.

AHA, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. Machine learning, 6, 37-66.

AGUIAR, Janderson, FECHINE, Joseana, COSTA, Evandro (2019). Experimentando a Influência dos Traços de Personalidade do Modelo Big Five na Recomendação de Recursos Educacionais. Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE), [S.I.], p. 1711.

AHAD, A. D.; ANSHARI, M.; RAZZAQ, A. Domestication of Smartphones Among Adolescents in Brunei Darussalam. **International Journal of Cyber Behavior, Psychology and Learning**, v. 7, n. 4, p. 26–39, out. 2017.

ANSHARI, M., ALAS, Y., GUAN, L. S. Developing online learning resources: Big data, social networks, and cloud computing to support pervasive knowledge. **Education and Information Technologies**, v. 21, n. 6, p. 1663–1677, 21 maio 2015.

B. THORAT, P.; M. 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, 16 jan. 2015.

BALABANOVIĆ, M.; SHOHAM, Y. Fab: content-based, collaborative recommendation. **Communications of the ACM**, v. 40, n. 3, p. 66–72, 1 mar. 1997.

BOBADILLA, J. et al. Recommender systems survey. **Knowledge-Based Systems**, v. 46, p. 109–132, jul. 2013.

BLOOM, B. S. et al. **Taxonomy of educational objectives**. New York: David Mckay, 1956. 262 p. (v. 1).

CAMPOS, Aline de; HOLLERWEGER, Leonéia; SANTOS, Gabriela; FARIAS, Adriano F.; BEHAR, Patricia A. Mapeamento de Soluções tecnológicas em Sistemas de Recomendação Educacionais em âmbito brasileiro. **Informática na Educação: teoria& prática**, Porto Alegre, v. 20, n. 3, p. 79-96, set./dez. 2017.

CARVALHO, Carolina**. O que é python? História, Sintaxe e um Guia para iniciar na Linguagem**. Disponível em: https://www.alura.com.br/artigos/python?gclid=Cj0KCQjw4NujBhC5ARIsAF4Iv6davw2Ulo-B-3j_efgV6TR8smR3UOhOV6HCGd0wvaXnBdtHYM7-VfEaAppjEALw_wcB. Acesso em 31 mai. 2023.

CARVALHO, Vitor; DORÇA, Fabiano. Uma Abordagem para Recomendação Automática e Dinâmica de Objetos de Aprendizagem Baseada em Estilos de Aprendizagem e em Metadados no padrão IEEE LOM. In: WORKSHOP DE DESAFIOS DA COMPUTAÇÃO APLICADA À EDUCAÇÃO (DESAFIE!), 3., 2014, Brasília. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2014. p. 47-56

CAZELLA, S. C., Nunes, M. A. S. N., Reategui, E. B. **A Ciência da Opinião: Estado da arte em Sistemas de Recomendação**. In **XXX Congresso da Sociedade Brasileira de Computação — Jornada de Atualização em Informática (JAI)**, 2010, p. 161-210.

CAZELLA, S.; BEHAR, P.; SCHNEIDER, D.; SILVA, K.; FREITAS, R. (2012). Desenvolvendo um Sistema de Recomendação de Objetos de Aprendizagem baseado em Competências para a Educação: relato de experiências. In: **Anais do XXIII Simpósio Brasileiro de Informática na Educação**.

CERVI, Emerson U. **Manual de métodos quantitativos para iniciantes em Ciência Política – Volume 1 / Emerson Urizzi Cervi.** 1ª edição. Curitiba: CPOP-UFPR, 2017. 256 p.

COSTA, E.; AGUIAR, J.; JONATHAS MAGALHÃES. **Sistemas de Recomendação de Recursos Educacionais**: conceitos, técnicas e aplicações. 19 dez. 2013.

DINIZ, F. A. et al. RedFace: um sistema de reconhecimento facial baseado em técnicas de análise de componentes principais e autofaces. **Revista Brasileira de Computação Aplicada**, v. 5, n. 1, 27 maio 2013.

DRACHSLER, H. et al. Panorama of Recommender Systems to Support Learning. **Recommender Systems Handbook**, p. 421–451, 2015.

EASTERBROOK, S. et al. Selecting Empirical Methods for Software Engineering Research. **Guide to Advanced Empirical Software Engineering**, p. 285–311, 2008.

ELMASRI Ramez; NAVATHE Shamkant B. **Sistemas de Banco de Dados**. 4. ed. São Paulo: Pearson, 2005.

FABRE, M.-C. J. M.; TAMUSIUNAS, F.; TAROUCO, L. M. R. Reusabilidade de objetos educacionais. **RENOTE**, v. 1, n. 1, 28 fev. 2003.

FELDER, R. Learning and teaching styles in engineering education In **Engineering education**, v. 78, n. 7, p. 674–681, 1988.

FELDER, R. M.; HENRIQUES, E. R. Learning and Teaching Styles In Foreign and Second Language Education. **Foreign Language Annals**, v. 28, n. 1, p. 21–31, mar. 1995.

FISK, P. **Education 4.0 ... the future of learning will be dramatically different, in school and throughout life**. Disponível em: <https://www.peterfisk.com/2017/01/future-education-young-everyone-taught-together/>.

GOLDBERG, D. et al. Using collaborative filtering to weave an information tapestry. **Communications of the ACM**, v. 35, n. 12, p. 61–70, 1 dez. 1992.

GOLDBERG, Y. **Neural Network Methods for Natural Language Processing**. Cham: Springer International Publishing, 2017.

GOMEZ-URIBE, C. A.; HUNT, N. The Netflix recommender system: Algorithms, business value, and innovation. **ACM Transactions on Management Information Systems**, v. 6, n. 4, p. 1–19, 28 dez. 2015.

GRBOVIC, M.; CHENG, H. Real-time Personalization using Embeddings for Search Ranking at Airbnb. **Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**, 19 jul. 2018.

Gulzar, Z., Leema, A. A., and Deepak, G. (2018). PCRS: **Personalized Course Recom-mender System Based on Hybrid Approach**. Procedia Computer Science, 125:518–524

GUNAWARDANAASELA; SHANIGUY. **A Survey of Accuracy Evaluation Metrics of Recommendation Tasks**. 1 dez. 2009.

GUTTMAN, R. H.; MOUKAS, A. G.; MAES, P. Agent-mediated electronic commerce: a survey. **The Knowledge Engineering Review**, v. 13, n. 2, p. 147–159, jul. 1998.

HAMDAN, Mahani, AHMAD, Norainie, JAIDIN, Jainatul Halida, ANSHARI, Muhammad. Internationalisation **in Brunei's Higher Education and Its Policy Implications**: Case Study of Universiti Brunei Darussalam, 2020.

HODGINS, W.H. **The future of learning objects.** *In* **Engineering education: Learning outcomes providing future possibilities**, 2002.

HOIC-BOZIC, N.; HOLENKO DLAB, M.; MORNAR, V. Recommender System and Web 2.0 Tools to Enhance a Blended Learning Model. **IEEE Transactions on Education**, v. 59, n. 1, p. 39–44, fev. 2016.

HUANG, W. et al. **Recommending Citations**: Translating Papers into References. 2012. p. 1910-1914.

IEEE Standard for Learning Object Metadata. [s.d.].

IMRAN, H. et al. PLORS: a personalized learning object recommender system. **Vietnam Journal of Computer Science**, v. 3, n. 1, p. 3–13, 27 ago. 2015.

JUNG, C. G. **Tipos psicológicos**. [S.l.]: Editora Vozes Limitada, 1971.

KEEFE, J.W. Learning Style: An Overview in **NASSP's Student Learning Styles: Diagnosing and Prescribing Programs,** p. 1-17, 1985.

KENNETH, Reeder, MACFADYEN, Leah, ROCHE, Jorg, CHASE, Mackie. Negotiating culture in cyberspace**:** participation patterns and problematics Language, **Learning and Technology**, vol. 8, no. 2, p. 88-105, 2004.

KOLB, D. A. **Experimental Learning: experience as the source of learning and development**. London: Prentice-Hall, 1984.

KOOHANG, A.; HARMAN, **K. Learning Objects**: theory, praxis, issues and trends. Santa Rosa, CA: Informing Science Press, p.1- 44, 2007.

LEWIS, D. D. Naive (Bayes) at forty: The independence assumption in information retrieval. **Machine Learning**: **ECML-98**, p. 4–15, 1998.

LINDEN, G.; SMITH, B.; YORK, J. Amazon.com recommendations: item-to-item collaborative filtering. **IEEE Internet Computing**, v. 7, n. 1, p. 76–80, jan. 2003.

LOPS, P.; DE GEMMIS, M.; SEMERARO, G. Content-based Recommender Systems: State of the Art and Trends. **Recommender Systems Handbook**, p. 73–105, 5 out. 2010.

LUDKE, Menga, ANDRÉ, Marli E.D.A. **Pesquisa em educação**: abordagens qualitativas. São Paulo, Editora Pedagógica e Universitária, 1986. 99p.

LUIZ HENRIQUE SALAZAR. Detecção de Estilos de Aprendizagem em Ambientes Virtuais de Aprendizagem Utilizando Redes Bayesianas. 27 out. 2017.

MILANI, André. **PostgreSQL**: guia do programador, São Paulo, Novatec Editora, 2008.

MARQUES, Gerso Adriano. **Um modelo de sistema de recomendação de atividades complementares para capacitação profissional do aluno de graduação**. UNISINOS, São Leopoldo, 2022.

MONTANER, M.; LÓPEZ, B.; DE LA ROSA, J. L. **Artificial Intelligence Review**, v. 19, n. 4, p. 285–330, 2003.

MORAES, Thayron Crystian Hortences de Moraes, STIUBIENER, Itana. **Uma abordagem híbrida baseada no estilo de aprendizagem para recomendação de objetos de aprendizagem**, 2018.

Object Management Group. **UML 2.1.2 Superstructure Specification**, 2007.

P.C.R. Pinho et al. (2019). Developments in **Educational Recommendations Systems: a systematic review.** IEEE Frontiers in Educational Conference (FIE), p. 1-7.

PÖTTKER, L. M. V.; FERNEDA, E.; MOREIRO-GONZÁLEZ, J. A. Mapeamento relacional entre padrões de metadados educacionais. **Perspectivas em Ciência da Informação**, v. 23, n. 3, p. 25–38, set. 2018.

PRISCILLA DO NASCIMENTO et al. **Recomendação de Objetos de Aprendizagem baseada em Modelos de Estilos de Aprendizagem**: Uma Revisão Sistemática da Literatura. 27 out. 2017.

REATEGUI, Eliseo Berni; CAZELLA, Sílvio César. Sistemas de Recomendação, 2005.

REGINALDO GOTARDO; MASSA, R.; HRUSCHKA, E. R. **Predição do Desempenho do Aluno usando Sistemas de Recomendação e Acoplamento de Classificadores**. 22 nov. 2013.

REIS, Luís Felipe Marçal. **Sistema de Recomendação Baseado em Conhecimento**. 2013. Dissertação (Mestrado em Engenharia Informática). Universidade de Coimbra, 2005.

RESNICK, P.; VARIAN, H. R. Recommender systems. **Communications of the ACM**, v. 40, n. 3, p. 56–58, 1 mar. 1997.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to Recommender Systems Handbook. **Recommender Systems Handbook**, p. 1–35, 5 out. 2010.

S. A. A. Maria, S. C.Cazella and P. A. Behar (2019). Sistemas de recomendação: conceitos e técnicas de aplicação. in **Recomendação pedagógica em educação a distância**, P. A. Behar, Org., Porto Alegre: Penso.

SANTOS, T. G. L. DOS. **Sistemas de recomendação para o AVA moodle: uma abordagem baseada em filtragem colaborativa e na taxonomia revisada de bloom**. Disponível em: <https://repositorio.ufpb.br/jspui/handle/123456789/18269>. Acesso em: 3 jul. 2023.

SCHMECK, R. R. Inventory of learning processes. Student learning styles and brain behavior, Reston, **VA: Reston Publishing Company**, v. 11, 1982.

SHEN, Z.; SHI, Y.; XU, G. A learning resource metadata management system based on LOM specification. 25 jun. 2003.

Silva, F. L., da Silva, K. K. A., Slodkowski, B. K., & Cazella, S. C. (2022). A Aplicação de Sistemas de Recomendação no Contexto Educacional: **uma Revisão Sistemática da Literatura**. Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología, (32), e1-e1.

SINGH, H. Introduction to Learning Objects. 2001. Disponível em www.imsproject. org/content/packing/ims-cp-bestv1p1.html. Acesso em 08 jan. 2023.

SÔNIA ELISA CAREGNATO; ROZI MARA MENDES; RIOS, V**. A propriedade intelectual na elaboração de objetos de aprendizagem**. 1 jan. 2004.

SWEENEY, M. et al. N**ext-Term Student Performance Prediction: A Recommender Systems Approach**. [s.l: s.n.]. Disponível em: <https://files.eric.ed.gov/fulltext/EJ1120275.pdf>. Acesso em: 3 jul. 2023.

SYWELEM, Mohamed, AL-HARBI, Qassem, FATHEMA, Nafsania, WITTE, James. Learning style preferences of student teachers: A cross-cultural perspective. **Institute for Learning Styles Journal**, Spring, vol. 1, p. 10-24, 2012.

TORRÃO, S. **Produção de objetos de aprendizagem para e-learning**. 2009. Disponível em:

http://www.scribd.com/doc/10691731/produAodeObjectosdeAprendizagemParaeLearning>. Acesso em 10 dez. 2022.

XU, S. Bayesian. (2016). Naïve Bayes classifiers to text classification. Journal of Information Science, v. 44, n. 1, p. 48–59.

WILEY, D. A. LEARNING OBJECT DESIGN AND SEQUENCING THEORY. [s.l: s.n.]. Disponível em: <https://opencontent.org/docs/dissertation.pdf>. Acesso em: 3 jul. 2022.

YIN, R. K. **Estudo de Caso**: Planejamento e Métodos. [S.l.]: Bookman editora, 2010.