LEONARDO IHARA ISHICAVA
NICHOLAS YASSUO ITO
RICARDO SEIKI MATSUDA INOUE

ANÁLISE E APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL NO DESENVOLVIMENTO DE UM ALGORITMO DE RECOMENDAÇÃO DE CONTEÚDO

São Paulo
2024

LEONARDO IHARA ISHICAVA
NICHOLAS YASSUO ITO
RICARDO SEIKI MATSUDA INOUE

ANÁLISE E APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL NO DESENVOLVIMENTO DE UM ALGORITMO DE RECOMENDAÇÃO DE CONTEÚDO

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Engenheiro Eletricista.

São Paulo
2024

Nome: ISHICAVA, Leonardo Ihara; ITO, Nicholas Yassuo; INOUE, Ricardo Seiki Matsuda

Título: Análise e Aplicação de Inteligência Artificial no Desenvolvimento de um Algoritmo de Recomendação de Conteúdo.

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Engenheiro Eletricista.

Aprovado em:

Banca Examinadora

Prof. Dr. _____

Instituição: _____

Julgamento: _____

Prof. Dr. _____

Instituição: _____

Julgamento: _____

Prof. Dr. _____

Instituição: _____

Julgamento: _____

LEONARDO IHARA ISHICAVA

NICHOLAS YASSUO ITO

RICARDO SEIKI MATSUDA INOUE

ANÁLISE E APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL NO DESENVOLVIMENTO DE UM ALGORITMO DE RECOMENDAÇÃO DE CONTEÚDO

Versão Original

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Engenheiro Eletricista.

Área de Concentração:

Engenharia Elétrica (Ênfase Computação)

Orientador:

Prof. Dr. Paulo Sérgio Cugnasca.

Co-Orientador:

Prof. Dr. Antonio Vieira da Silva Neto

São Paulo
2024

Catalogação-na-publicação

Ihara Ishicava, Leonardo
Análise e Aplicação de Inteligência Artificial no Desenvolvimento de um Algoritmo de Recomendação de Conteúdo / L. Ihara Ishicava, N. Y. Ito, R. S. Matsuda Inoue -- São Paulo, 2024.
131 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Comércio eletrônico 2.engenharia de sistemas de computação 3.inteligência artificial 4.processamento de linguagem natural 5.satisfação do consumidor I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t. III.Ito, Nicholas Yassuo IV.Matsuda Inoue, Ricardo Seiki

AGRADECIMENTOS

Inicialmente, agradecemos a todos nossos familiares mais próximos o apoio durante toda a Graduação na Escola Politécnica da Universidade de São Paulo e o constante incentivo em todo seu período. Esses agradecemos também se estendem a nossos amigos e colegas, que estiveram ao nosso lado e compartilharam desafios, conquistas e aprendizados ao longo dessa jornada.

Também agradecemos ao Prof. Dr. Paulo Sérgio Cugnasca e ao Prof. Dr. Antonio Vieira da Silva Neto uma orientação impecável, tanto em relação à disponibilidade constante quanto à clareza das instruções e ao apoio nos momentos mais desafiadores.

Por fim, agradecemos a todos os professores e profissionais que contribuíram para nossa formação acadêmica, pois este projeto não teria sido possível sem tal dedicação. A todos, nossos sinceros reconhecimentos e gratidão.

“If you do build a great experience, customers tell each other about that. Word of mouth is very powerful.”

(Jeff Bezos)

RESUMO

O uso crescente de plataformas digitais e a evolução da Inteligência Artificial nos últimos anos têm provocado mudanças significativas em setores como comércio eletrônico, entretenimento, redes sociais e serviços de streaming. Essas mudanças têm permitido uma maior personalização na entrega de produtos e conteúdo, garantindo uma experiência única e imersiva para cada usuário. A quarentena imposta pela pandemia de COVID-19 acelerou ainda mais essa transformação, ressaltando a importância dos sistemas de recomendação para atender à demanda digital em constante expansão. Dentro deste contexto, o presente projeto possui como objetivo a criação de um algoritmo de recomendação de código aberto para *e-commerce*, proporcionando sua utilização por lojas virtuais de diferentes portes mediante uso aprofundado de técnicas de IA em algoritmos de recomendação. O método de trabalho para esse fim partiu da revisão de literatura da área e da análise de um algoritmo de referência, que permitiu identificar suas limitações e implementar melhorias voltadas ao incremento da qualidade percebida das recomendações geradas. Esse processo de melhoria das recomendações foi iniciado pela coleta de uma base de dados de *e-commerce* que foi utilizada para gerar as recomendações iniciais e pelo tratamento de seus dados para aprimorar a qualidade deles para o novo sistema de recomendação. Na sequência, mecanismos de recomendação por filtragens categóricas, semânticas e morfossintáticas foram iterativamente desenvolvidos e testados de forma unitária e integrada. Ao final, o sistema de recomendação foi avaliado e comparado com o sistema de referência mediante pesquisas de satisfação com usuário. Os resultados qualitativos e quantitativos dessa pesquisa evidenciaram que as melhorias desenvolvidas foram bem-sucedidas, uma vez que se observou uma melhoria significativa da taxa de aprovação do sistema de recomendação desenvolvido em relação ao sistema de referência. Trabalhos futuros compreendem expansão para outras aplicações, análises de sensibilidade dos algoritmos utilizados, processamento de dados adicionais, robustez a variações gramaticais, versões traduzidas para outras línguas e interface visual mais atraente.

Palavras-chave: Comércio eletrônico, engenharia de sistemas de computação, inteligência artificial, processamento de linguagem natural, satisfação do consumidor.

ABSTRACT

The increasing use of digital platforms and the evolution of Artificial Intelligence (AI) in recent years have led to significant changes in areas such as e-commerce, entertainment, social networks, and streaming services. These changes have allowed for greater personalization in the delivery of products and content, ensuring a unique and immersive experience for each user. The quarantine imposed by the COVID-19 pandemic has further accelerated this transformation, highlighting the importance of recommendation systems to meet the ever-expanding digital demand. Within this context, the aim of this project is to create an open source AI-based. The method developed for this purpose started with a literature review in the area, which enabled analyzing a reference algorithm in order to identify its limitations and potential implementation improvements targeted at increasing the recommendations perceived quality. This recommendation improvement process was started by collecting an e-commerce database that was used to generate the original recommendations and by processing its data to improve their quality for the new recommendation system. Afterwards, new recommendation mechanisms by categorical, semantic and morphosyntactic filtering were iteratively developed and tested in both unitary and integrated approaches. Finally, the recommendation system was evaluated and compared with the reference system through user satisfaction surveys. The qualitative and quantitative results of this survey showed that the developed improvements were successful, since the approval rate of the new the recommendation system was significantly higher than that of the reference system. Future work includes expanding the system to other applications, performing sensitivity analyses of the algorithms, processing additional data, increasing the robustness to grammatical variations, translating data into other languages and designing a more attractive visual interface.

Keywords: E-commerce, computer systems engineering, artificial intelligence, natural language processing, customer satisfaction.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIAÇÕES

BRL	Real Brasileiro
DBSCAN	Agrupamento Espacial Baseado em Densidade para Aplicações Ruidosas (*Density-Based Spatial Clustering of Applications with Noise*)
DDR	*Double Data Rate* (Taxa de Dados Dupla)
EAP	Estrutura Analítica de Projeto
E	Expectativa (*Expectation)*
FC	Filtragem Colaborativa
GB	Gigabyte
GloVe	*Global Vectors for Word Representation* (Representação de Palavras em Vetores Globais)
GPU	Unidade de Processamento Gráfico (*Graphics Processing Unit*)
HH	Hora-Homem
IA	Inteligência Artificial
kNN	k-ésimos Vizinhos Mais Próximos (*k-Nearest Neighbors*)
M	Maximização (*Maximization*)
MAE	Erro Absoluto Médio (*Mean Absolute Error*)
ML	Aprendizado de Máquina (*Machine Learning*)
PLN	Processamento de Linguagem Natural
RAM	*Random Access Memory* (Memória de Acesso Aleatório)
RMSE	Raiz do Erro Quadrático Médio (*Root Mean-Square Error*)
RNA	Rede Neural Artificial
SQL	Linguagem de Consulta Estruturada (*Structured Query Language*)
SSD	*Solid-State Drive* (Unidade de Estado Sólido)
SSE	Soma dos Erros Quadráticos (*Sum of Squared Errors*)
SVM	Máquina de Vetores de Suporte (*Support Vector Machine*)
TB	Terabyte
TF-IDF	*Term Frequency – Inverse Document Frequency* (Frequência de Termo – Frequência Inversa no Documento)
USD	Dólares Americanos (*United States Dollar*)
V&V	Verificação e Validação

SUMÁRIO

1. INTRODUÇÃO

Apresentam-se neste capítulo da monografia as motivações, os objetivos e as justificativas do projeto.

1.1 MOTIVAÇÃO

A utilização crescente de plataformas digitais para compras e exploração de diferentes tipos de conteúdo é notável (NIELSENIQ, 2022). Isso impulsiona a necessidade de sistemas inteligentes para atender precisamente às demandas e preferências individuais dos usuários – necessidade essa que, inclusive, aumenta cada vez mais com o passar do tempo (ROBICQUET, 2023). Atualmente, porém, muitas lojas online enfrentam dificuldades em fornecerem recomendações precisas e relevantes para cada usuário, podendo resultar em uma experiência de compra insatisfatória (NIELSENIQ, 2022).

Durante a pandemia, o aumento significativo da utilização do comércio eletrônico motivou as empresas a realizar avanços e investimentos no setor de recomendação para proporcionar uma experiência de compra mais satisfatória para seus usuários (DI FANTE, 2021). O desenvolvimento e implementação dessas tecnologias são fundamentais para atender às crescentes expectativas dos consumidores (PANCINI, 2021).

No entanto, empresas novas que surgiram após a pandemia ou que não investiram nessa área apresentam uma clara desvantagem competitiva em relação às concorrentes mais estabelecidas, uma vez que os algoritmos de recomendação são informações sigilosas de cada empresa (BORBA; DE ALBUQUERQUE, 2024).

1.2 OBJETIVOS

Com o contexto prévio da seção 1.1, este projeto de formatura tem como objetivo o desenvolvimento de um algoritmo de recomendação de comércio eletrônico (*e-commerce*) de código aberto, permitindo que lojas virtuais de todos os tamanhos possam utilizá-lo para aprimorar a experiência de seus usuários e, entre outros benefícios, aumentar a competitividade delas no mercado. Esse algoritmo deve ser comparado com ao menos outra solução de código aberto já existente, seguindo

métricas estabelecidas para avaliar sua superioridade em desempenho em relação à alternativas disponíveis.

Além disso, o projeto também visa aplicar e aprofundar os conhecimentos em Inteligência Artificial (IA) e Aprendizado de Máquina (ML – *Machine Learning*) para o desenvolvimento de algoritmos de recomendação de conteúdo personalizado, levando em consideração as preferências individuais dos usuários. O projeto concentrar-se-á na análise e na implementação de técnicas de IA e ML para criar modelos de recomendação precisos e eficazes.

1.3 JUSTIFICATIVA

Dado o exposto, é possível afirmar que o projeto previamente definido, uma vez que atinja seus objetivos, consegue preencher lacunas na área de sistemas de recomendação para empresas cujo desenvolvimento da mesma esteja em desvantagem em relação às outras. Considerando os benefícios proporcionados pela IA nos sistemas de recomendação, como aprendizado automático, personalização, escalabilidade e capacidade de descoberta de padrões complexos, o projeto proposto visa não apenas suprir essas deficiências, mas também impulsionar a eficácia e a competitividade das empresas no mercado digital (D'ARC, 2021).

Ao incorporar técnicas avançadas de IA nos sistemas de recomendação, o projeto busca fornecer soluções mais adaptáveis, precisas e relevantes. Por meio do aprendizado automático, os algoritmos são capazes de ajustar-se dinamicamente às mudanças nos padrões de comportamento dos usuários, proporcionando recomendações cada vez mais personalizadas. Além disso, a escalabilidade da IA permite lidar com grandes volumes de dados e milhões de usuários simultaneamente, garantindo a eficácia contínua do sistema em um ambiente dinâmico e em constante evolução, características que não seriam obtidas sem o uso de IA, visto que seriam utilizadas abordagens mais simples que tornariam o modelo de recomendação mais previsível e limitado.

Assim, o projeto demonstra-se como viável e justificável, pois visa não apenas preencher lacunas na área de sistemas de recomendação, mas também promover a inovação e aprimorar a experiência do usuário. Ao oferecer recomendações mais

precisas e relevantes, baseadas em inteligência artificial, espera-se que o projeto contribua significativamente para aumentar a satisfação e a fidelidade dos clientes, fortalecendo assim a posição das empresas no mercado competitivo do comércio eletrônico.

1.4 Estrutura da Monografia

Esta monografia é subdividida em nove capítulos e dois apêndices.

O primeiro capítulo teve como objetivo apresentar a motivação, os objetivos e a justificativa de relevância deste trabalho de conclusão de curso, voltado à área de sistemas de recomendação de conteúdo e produtos para usuários.

O segundo capítulo versa sobre um panorama da revisão de literatura conduzida para caracterizar o estado da arte dos sistemas de recomendação.

Na sequência, o terceiro capítulo é voltado à caracterização do método que balizou a realização do projeto, com a definição de suas etapas e de tecnologias plausíveis para conduzi-las.

No quarto capítulo, apresenta-se a especificação de requisitos do sistema de recomendação de conteúdo pertencente ao escopo deste trabalho de conclusão de curso.

A arquitetura dessa solução, por sua vez, é explorada no quinto capítulo desta monografia. Essa descrição contempla aspectos estruturais e comportamentais da solução do sistema de recomendação de conteúdo.

O planejamento geral do projeto é explorado no sexto capítulo. Nele, informam-se o cronograma de atividades, os recursos utilizados e os custos do projeto. Também consta uma análise comparativa entre as expectativas de cronograma e recursos do projeto e a distribuição real das atividades executadas.

No sétimo capítulo da monografia, trata-se detalhadamente sobre o processo de desenvolvimento do projeto. Detalham-se as atividades realizadas e os seus respectivos resultados até a produção do sistema de recomendação estabelecido nos

objetivos do projeto. Parte desses resultados são expandidos no Apêndice A da monografia.

O oitavo capítulo, por sua vez, versa sobre a avaliação dos resultados do projeto. Ele contempla a descrição do método empregado para essa finalidade, a exposição dos resultados obtidos e a análise desses resultados face ao cumprimento dos requisitos e dos objetivos do projeto. Informações mais detalhadas sobre os ensaios conduzidos são relatadas no Apêndice B da monografia.

Por fim, no nono capítulo da monografia, constam as considerações finais do trabalho de conclusão de curso. Nele, abordam-se as conclusões obtidas face ao cumprimento dos objetivos à luz dos resultados atingidos, propostas de trabalhos futuros e uma consideração final que sumariza o projeto.

2. ASPECTOS CONCEITUAIS

Este capítulo do documento relata a revisão de literatura realizada com o propósito de caracterizar o estado da arte na área e servir como motivador e justificativa para o projeto e definir os aspectos conceituais que permeiam as tecnologias plausíveis para uso no projeto.

2.1 REVISÃO DA LITERATURA

A fim de obter um panorama sobre os projetos existentes no campo de sistemas de recomendação, foi realizada uma pesquisa sobre os diversos sistemas de recomendação presentes tanto no mercado como em publicações de fins acadêmicos. Esta revisão teve como objetivo compreender e analisar os métodos e técnicas mais utilizados nesse domínio, identificando modelos já existentes e ponderando quais seriam mais interessantes como base tanto para o desenvolvimento de um novo algoritmo quanto para melhorias de algoritmos preexistentes.

A pesquisa sobre sistemas de recomendação revelou uma variedade de abordagens utilizadas para recomendar itens aos usuários (GOYANI; CHAURASIYA, 2020). Uma dessas abordagens é a de sistemas de recomendação baseados em conteúdo. Nesses sistemas, itens similares aos que o usuário selecionou no passado são recomendados com base em perfis de usuário e itens. O perfil do usuário contém informações sobre gostos, preferências e necessidades, enquanto o perfil do item contém atributos dos itens. A similaridade entre o perfil do usuário e o perfil de cada item é calculada, e os itens são recomendados com base nessa similaridade (LI; CAI; LIAO, 2012).

Outra abordagem comum é a Filtragem Colaborativa (FC). Ela é uma técnica amplamente utilizada em sistemas de recomendação por sua capacidade de sugerir itens com base nas escolhas e preferências de usuários semelhantes. O processo envolve a análise da similaridade entre o usuário atual e outros, utilizando avaliações ou classificações prévias (VIJAYA KUMAR; REDDY, 2007). Ao identificar padrões entre usuários e itens, a FC gera recomendações personalizadas que enriquecem a experiência do usuário.

A FC pode ser dividida em três etapas principais: (1) cálculo da similaridade entre usuários, (2) seleção dos mais semelhantes ao usuário alvo e (3) predição ponderada com base nas preferências desses usuários (HERLOCKER, 2000).

Existem duas abordagens principais para a FC: a baseada em vizinhança, que calcula a similaridade entre usuários ou itens, sendo simples e eficiente, mas enfrentando desafios como dados esparsos e escalabilidade (VIJAYA KUMAR; REDDY, 2007), e a baseada em modelo, que utiliza algoritmos de aprendizado de máquina para descobrir padrões nos dados e melhorar a escalabilidade e a precisão das predições, embora tenha maior custo computacional (VIJAYA KUMAR; REDDY, 2007).

Por último, há a abordagem híbrida, uma combinação das técnicas de filtragem colaborativa e filtragem baseada em conteúdo. Quando apenas um dos métodos, seja a filtragem colaborativa ou a baseada em conteúdo, não consegue resolver o problema, o conceito de filtragem híbrida é utilizado. Ao utilizar a filtragem híbrida, muitos problemas da filtragem colaborativa e da filtragem baseada em conteúdo podem ser resolvidos. Por exemplo, ao aplicar inicialmente a filtragem baseada em conteúdo e, em seguida, a filtragem colaborativa, é possível oferecer uma solução para esse problema. Assim, ao torná-lo híbrido, é possível resolver esses desafios de forma mais eficaz (JAIN et al., 2018).

Existem métricas de desempenho do sistema de recomendação e indicadores relacionados à usabilidade desses sistemas. As métricas de avaliação de desempenho mais comuns são exatidão (*accuracy*), Erro Absoluto Médio (MAE – *Mean Absolute Error*), Raiz do Erro Quadrático Médio (RMSE – *Root Mean-Square Error*), revocação (*recall*) e índice f1 (*f1-score*) (GOYANI; CHAURASIYA, 2020). Já em relação aos indicadores da usabilidade e aceitação dos usuários, crucial para a adoção prática dos sistemas, existe subjetividade a respeito de suas medidas.

Os sistemas de recomendação utilizam, para indicações de usabilidade, preferências dos usuários, que podem ser tanto explícitas (diretas) quanto implícitas (indiretas). As recomendações diretas incluem e-mails ou mensagens, ao passo que as indiretas, ocorrem por meio de anúncios e redirecionamento para novos sites (MODARRESI, 2016). Essas práticas proporcionam uma personalização em tempo

real e adaptação a diferentes contextos de uso, incluindo análise do histórico do usuário. Dessa forma, promove-se uma experiência satisfatória ao usuário com o engajamento e a fidelização dos clientes.

Dessa forma, os sistemas de recomendação são fundamentais para proporcionar uma experiência personalizada aos usuários, seja por recomendações diretas, seja por recomendações indiretas. A literatura destaca a necessidade de integrar avaliações formais e métricas de usabilidade para garantir a eficácia e a confiança desses sistemas. A adoção de modelos híbridos, combinando técnicas de filtragem colaborativa e baseada em conteúdo, pode evitar limitações individuais de cada método e, também, melhorar o desempenho das predições (JAIN et al., 2018). No entanto, é importante ressaltar que essa abordagem tende a aumentar a complexidade do sistema (GHAZANFAR; PRÜGEL-BENNETT; SZEDMAK, 2012). Dessa forma, torna-se relevante uma análise, durante a implementação, a fim de garantir que os benefícios superem os desafios adicionais dessa implementação.

De forma similar, funcionalidades que permitem a adaptação em tempo real são cruciais para atender dinamicamente às preferências dos usuários. Pode-se observar isso na significativa contribuição que os sistemas de recomendação representam para muitos negócios. Por exemplo, para a Netflix, aproximadamente 80% dos conteúdos consumidos são recomendados; para a Amazon, de 30 a 40% dos itens vendidos são conteúdos recomendados, e, para a Google, as recomendações geram 38% mais cliques (MODARRESI, 2016). No entanto, há diversos desafios ao projetar e implementar um bom sistema de recomendação, incluindo questões como métricas adequadas para medir a eficácia, privacidade dos usuários e escalabilidade, sendo o maior deles na abordagem de modelagem, no sentido da necessidade de modelos, precisos, estáveis e eficientes (MODARRESI, 2016).

A revisão de literatura fornece uma base sólida para o desenvolvimento de um novo sistema de recomendação, orientando a criação de soluções inovadoras que melhorem a precisão das recomendações e a satisfação dos usuários, integrando as melhores práticas identificadas e explorando novas abordagens para superar as limitações atuais.

2.2 Análise de Bases de Dados e Sistemas de Recomendação Existentes

Com o objetivo de obter outros sistemas de recomendação preexistentes como referência inicial, projetos de código aberto foram analisados na sequência:

a) ***Customer Segmentation & Recommendation System***: utiliza algoritmo *K-means* para agrupar clientes em seus padrões de compra, o que auxilia a entender os diferentes grupos e adaptar estratégias. Também utiliza o histórico de compras dos clientes para implementar uma personalização da recomendação de produtos (NEKOUEI, 2023);

b) ***E-Commerce-Product-Recommendation-System***: utiliza a estratégia de recomendação com base em ranqueamento (*Rank-Based Product Recommendation*), filtragem colaborativa em similaridade. Dessa forma, ele consegue encontrar usuários semelhantes e recomenda produtos com base em suas interações (VAIBHAV, 2023);

c) ***E-Commerce-Recommendation-Systems***: utiliza filtragem colaborativa e filtragem baseada em conteúdo para analisar o comportamento do usuário, além de utilizar seu histórico de navegação e comportamento de compras relevantes (“E-commerce-recommendation-system”, 2018);

d) ***Amazon E-Commerce Recommendation System Using Content-Based Filtering*****:** utiliza algoritmos de filtragem colaborativa e filtragem baseada em conteúdo para analisar o comportamento do usuário e gerar recomendações relevantes (CHEAH et al., 2020);

e) ***Personalized Context-Aware Re-ranking for E-Commerce Recommendation Systems***: o objetivo principal é melhorar a eficácia dos sistemas, com foco em reclassificar e reordenar as recomendações iniciais com base no contexto do usuário (“DRR (Data-Driven Recommender)”, 2019).

Para os sistemas prévios, a abordagem de análise dos algoritmos adotados baseia-se em um modelo de caixa-branca, por meio do qual a verificação e a interpretação do código das recomendações, incluindo IA, seria viável.

A decisão de focar apenas em sistemas de recomendação de fontes abertas foi tomada devido à inacessibilidade dos algoritmos de empresas, que são considerados sigilosos. Mesmo que fosse possível acessá-los, a análise seria limitada

a uma abordagem de caixa-preta, em que apenas os resultados do sistema seriam analisados, dificultando a compreensão detalhada do funcionamento interno dos algoritmos. Portanto, optou-se por trabalhar com sistemas abertos, que oferecem transparência e permitem uma análise mais profunda das técnicas e princípios subjacentes aos sistemas de recomendação. Isso proporciona uma base sólida para análises comparativas e avaliações objetivas.

Já em relação às bases de dados empregadas nos algoritmos *Open Source* prévios, constatou-se o uso de cinco bases de dados distintas, listadas na Tabela 1.

A base de dados escolhida para o projeto foi a de amostra de *marketing* de produtos da *Amazon* em 2020 (PROMPTCLOUD, 2020), que tem como principal característica a variedade de classes presentes. Após uma análise comparativa com outras bases de dados dos sistemas de referência, optou-se por esta base devido à sua abrangência e à qualidade dos dados armazenados. Apesar de ela também conter classes menos interessantes para o sistema de recomendação, o pré-processamento previsto neste projeto filtra a base de dados para utilizar apenas os atributos que são essenciais, permitindo uma melhor gestão e análise dos dados para fornecer recomendações relevantes. Além disso, por ser uma base de dados já utilizada em um dos sistemas de recomendação de referência (CHEAH et al., 2020), a manutenção e o suporte de acesso a ela são facilitados devido à documentação existente.

Tabela 1 – Bases de Dados dos Sistemas de Recomendação de Referência

Plataforma Utilizada	Base de Dados Utilizada
Kaggle	Base de dados de vendas de presentes em site de vendas online no Reino Unido entre 01/12/2010 e 09/12/2011 (NEKOUEI, 2023).
GitHub	Base de dados de produtos de beleza da Amazon (“E-commerce-recommendation-system”, 2018).
GitHub	Base de dados de comércio genérico (“DRR (Data-Driven Recommender)”, 2019).
GitHub	Base de dados de amostra de *marketing* de produtos da *Amazon* (CHEAH et al., 2020; PROMPTCLOUD, 2020).
GitHub	Base de dados de avaliações de Produtos eletrônicos da *Amazon* (VAIBHAV, 2023).

A análise dos sistemas de referência selecionados e das respectivas bases de dados por eles empregadas foi balizada por uma ordem de prioridade de escolha dos sistemas de referência e das bases de dados, definida com base em cinco critérios: (i.) variedade de classes, (ii.) qualidade dos dados, (iii.) compatibilidade com sistemas de referência, (iv.) qualidade da documentação e (v.) facilidade de uso e manutenção. Os resultados dessa análise constam na Tabela 2.

Tabela 2 – Prioridade de Seleção dos Sistemas de Referência

Prioridade	Base utilizada
1	Base de dados de amostra de *marketing* de Produtos da *Amazon* (CHEAH et al., 2020; PROMPTCLOUD, 2020).
2	Base de dados de avaliações de Produtos eletrônicos da *Amazon* (VAIBHAV, 2023).
3	Base de dados de produtos de beleza da Amazon (“E-commerce-recommendation-system”, 2018).
4	Base de dados de vendas de presentes em site de vendas online no Reino Unido entre 01/12/2010 e 09/12/2011 (NEKOUEI, 2023).
5	Base de dados de comércio genérico (“DRR (Data-Driven Recommender)”, 2019).

A base de dados de amostra de *marketing* de produtos da Amazon (PROMPTCLOUD, 2020) foi escolhida como a mais promissora em função da variedade e da qualidade dos dados e da excelente documentação. Em segundo lugar, a base de dados de avaliações de produtos eletrônicos da Amazon (VAIBHAV, 2023) foi selecionada pela alta qualidade dos dados e especificidade em eletrônicos. A base de dados de vendas de presentes em site de vendas online no Reino Unido (NEKOUEI, 2023) ficou em terceiro lugar, sendo útil para análises de mercado específicas, mas menos abrangente. A base de dados de produtos de beleza da Amazon (“E-commerce-recommendation-system”, 2018) foi considerada adequada para nichos específicos, mas com menor variedade. Por fim, a base de dados de comércio genérico (“DRR (Data-Driven Recommender)”, 2019) foi incluída por ser aplicável amplamente, mas sem a profundidade das outras bases.

3. MÉTODO DE TRABALHO

Este capítulo da monografia documenta o método aplicado para o desenvolvimento deste projeto. Esse método inclui como etapas fundamentais (i.) coletar e preparar dados para projetar o sistema de recomendação, (ii.) projetar o sistema de recomendação selecionando-se modelos aplicáveis para essa finalidade, (iii.) realizar ajustes para a otimização do sistema e, por fim (iv.) efetuar testes com usuários para a avaliação complementar dos resultados obtidos.

3.1 COLETA E PREPARAÇÃO DE DADOS

Selecionar uma base de dados que contenha informações que se adequem ao domínio da aplicação do sistema de recomendações do presente projeto é importante desde o início de sua condução. Essa definição é relevante por balizar a definição dos modelos de IA utilizados para lidar com os dados disponíveis.

Dessa maneira, a coleta e a preparação de conjuntos de dados relevantes é necessária para que o sistema de recomendação seja projetado e executado sem problemas. Avalia-se que informações sobre usuários, suas preferências, interações passadas e características dos itens visualizados são características relevantes para as bases de dados destinadas a essa finalidade.

Tomando como exemplo uma base de dados de jogos *online*, disponível no repositório *Kaggle* (KOZYRIEV, 2023), classes interessantes para um sistema de recomendação seriam aquelas que trouxessem informações cruciais para a personalização de recomendações de acordo com as preferências individuais dos usuários – aumentando, assim, a relevância e utilidade delas. Estas seriam as seguintes:

- **Identificação**: A identificação única de cada jogo permite que o sistema de recomendação diferencie entre os diferentes títulos disponíveis na base de dados, garantindo precisão das recomendações ao se evitar confusões entre jogos com nomes semelhantes;

- **Título**: O título fornece informações básicas sobre o conteúdo do jogo, permitindo que os usuários reconheçam facilmente os jogos recomendados. Além disso, o título pode influenciar as preferências dos usuários, tornando-o uma variável importante para a personalização das recomendações;
- **Data de lançamento**: A data de lançamento é relevante porque a idade de alguns jogos pode alterar a relevância para classes distintas de usuários. Por exemplo, jogos mais recentes podem ter mais apelos junto a um público mais jovem, ao passo que jogos antigos podem ser mais atrativos a usuários mais velhos. A idade de um jogo pode influenciar sua disponibilidade em diferentes plataformas e seu preço;
- **Gênero**: O gênero é uma das características mais importantes para a personalização das recomendações, pois reflete as preferências individuais dos usuários, auxiliando o sistema de recomendação a sugerir jogos semelhantes que estejam dentro do mesmo gênero;
- **Classificação**: A classificação pode influenciar a sua atratividade para diferentes faixas etárias e públicos-alvo;
- **Plataformas**: A compatibilidade com diferentes plataformas é importante, pois os usuários podem ter preferências específicas quanto à plataforma em que desejam jogar;
- **Preço**: O preço de um jogo pode ser um fator decisivo para os usuários na hora de decidir se desejam comprá-lo. Logo, também utiliza as restrições orçamentárias dos usuários ao fazer recomendações;
- **Outros**: Outras informações, como desenvolvedor, editora, descrição do jogo, avaliações dos usuários, entre outras, também podem ser importantes para a personalização das recomendações e para fornecer aos usuários uma visão mais abrangente sobre os jogos recomendados.

3.2 Seleção e Projeto dos Modelos para Recomendação

A partir da seleção da base de dados, são previstos dois passos para conceber o sistema de recomendação. O primeiro prevê o pré-processamento das bases de dados, auxiliado por aprendizado não supervisionado se não houver indicação clara de categorias de recomendação. O segundo, por sua vez, prevê o uso de aprendizado supervisionado e Processamento de Linguagem Natural (PLN) para produzir o

sistema de recomendação propriamente dito, utilizando não só categorias geradas no passo anterior, mas também relações sintático-semânticas entre os termos textuais dos produtos.

Cada um dos passos prévios, representado no diagrama da Figura 1, é detalhado nas próximas subseções.

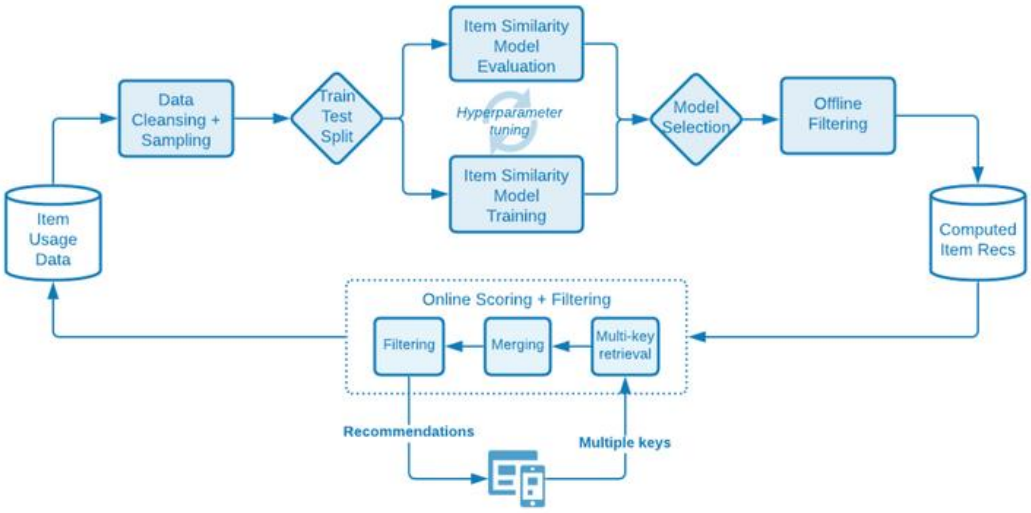


Figura 1 – Diagrama de Sistema de Recomendação

Fonte: Experience League Adobe (2024)

3.2.1 Criação de Categorias de Dados por Aprendizado Não Supervisionado (Agrupamento de Dados)

Caso as bases de dados (*datasets*) não possuam categorias explícitas acerca dos itens a serem recomendados, é necessário realizar essa categorização agrupando os dados em *clusters* que definem os grupos a serem recomendados. Para isso, avalia-se o emprego de um algoritmo de aprendizado não supervisionado para orientar a análise das bases de dados e particioná-las em grupos que possuem características semelhantes, tal como ilustrado na Figura 2.

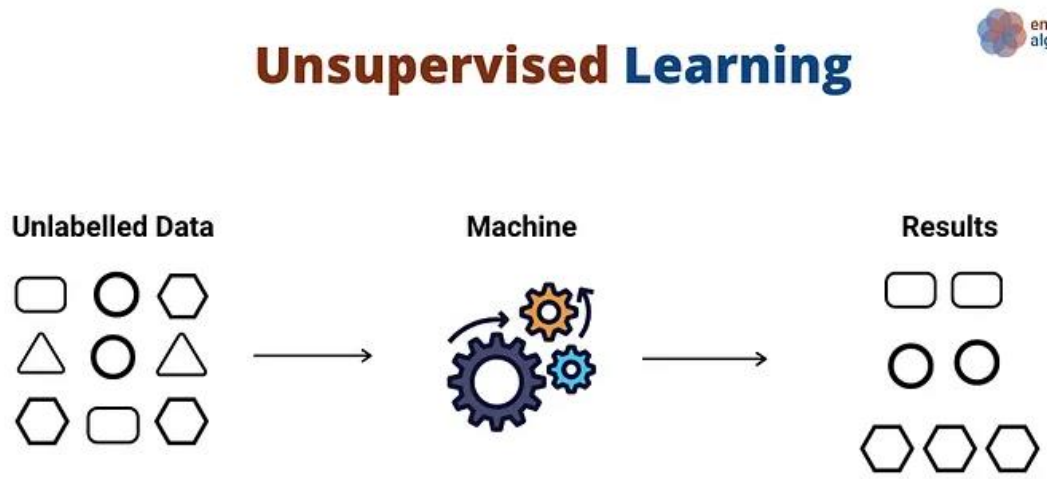


Figura 2 – Aprendizado de Máquina Não Supervisionado
Fonte: Kozan (2021)

Nessa etapa, avalia-se que conhecimento especialista dos projetistas orienta a avaliação das métricas de desempenho dos algoritmos de agrupamento de dados. Entre os algoritmos cogitados para esse fim estão os seguintes:

- ***K-Means***: Este algoritmo é uma técnica de agrupamento que divide um conjunto de dados em *k* grupos. Ele escolhe *k* pontos iniciais como centroides, atribui cada ponto de dados ao centroide mais próximo e recalcula o centroide com base na média de cada conjunto de dados. Este processo é repetido até que haja convergência do centroide de cada grupo (ARTLEY, 2022).
- **DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*):** Trata-se de um método de agrupamento eficaz para identificar *clusters* de diferentes densidades e formas, sendo útil para encontrar grupos de usuários com preferências semelhantes, mesmo em conjuntos de dados com ruído e *outliers*. Isso ocorre porque o DBSCAN considera a densidade dos pontos em torno de cada dado, agrupando pontos próximos em *clusters* e identificando pontos isolados como ruído (MONTEIRO, 2020).

Além do uso de algoritmos de agrupamento como K-Means e DBSCAN, o aprendizado não supervisionado também pode ser empregado diretamente na geração de recomendações por meio da análise de proximidade entre vetores. Nessa abordagem, o sistema de recomendação mede a similaridade entre os dados com

base na distância entre vetores. Recomendações são geradas de acordo com índices de proximidade maiores (vetores com distâncias menores), ou seja, itens ou usuários que compartilham características mais próximas têm maior probabilidade de serem recomendados. Para atributos contínuos, essa técnica utiliza métricas de distância como a distância euclidiana ou outras medidas de similaridade, como por exemplo (HRUSCHKA; CAMPELLO, 2020a):

- **Correlação de Pearson**: Mede a força e a direção da relação linear entre dois atributos contínuos, identificando relações com tendências lineares semelhantes;
- **Semelhança do Cosseno**: Avalia a similaridade entre vetores com base no ângulo entre eles, sendo útil para atributos assimétricos, para os quais a presença de uma característica é mais importante do que sua ausência.

Para atributos discretos, pode-se utilizar o **coeficiente de Jaccard**, que mede a similaridade entre conjuntos discretos considerando a interseção e a união dos conjuntos. Ele é vantajoso em cenários nos quais a presença de características compartilhadas é mais relevante do que a ausência de características não compartilhadas (HRUSCHKA; CAMPELLO, 2020a).

O uso dessas métricas de similaridade e correlação permite uma categorização mais precisa e ajustada dos dados, contribuindo para a efetividade do sistema de recomendação.

3.2.1.1 *Problemas e Alternativas para Inicialização do K-Means*

A inicialização dos centroides no K-Means é um fator crítico que pode influenciar significativamente os resultados do algoritmo. Inicializações inadequadas podem levar a convergências em mínimos locais subótimos, resultando em agrupamentos de baixa qualidade.

O K-Means tradicional frequentemente utiliza uma inicialização aleatória dos centroides, o que pode resultar em diferentes resultados dependendo da escolha inicial. No entanto, a inicialização dos centroides pode ser guiada por conhecimento especializado de domínio, em que posições iniciais supostamente boas são definidas

manualmente, reduzindo assim a variabilidade dos resultados (JAIN et al., 1988). Outros métodos também oferecem uma abordagem que pode melhorar a robustez e a consistência dos dados, tais como:

a) **K-Means++**: Este método oferece uma alternativa robusta para a inicialização, selecionando centroides iniciais que maximizam a distância entre eles. Isso geralmente resulta em uma melhor qualidade dos *clusters* formados, reduzindo a probabilidade de convergência em mínimos locais (JAIN et al., 1988).
b) **Múltiplas Inicializações Aleatórias**: Executar o K-Means várias vezes com diferentes inicializações e escolher a execução com o menor SSE (*Sum of Squared Errors*) é uma abordagem simples, mas eficaz para melhorar a qualidade dos resultados (HRUSCHKA; CAMPELLO, 2020b).
c) **Métodos Baseados em Heurísticas**: Técnicas como o Fuzzy C-Means permitem uma forma de agrupamento mais flexível, na qual os pontos de dados podem pertencer parcialmente a múltiplos clusters com diferentes graus de associação. Isso não apenas afeta a inicialização, mas também redefine o conceito de pertencimento a *clusters*, resultando em agrupamentos que refletem a natureza difusa dos dados em vez de uma atribuição rígida (HRUSCHKA; CAMPELLO, 2020c).

Essas alternativas visam melhorar a robustez do K-Means, garantindo que o algoritmo produza resultados mais consistentes e de melhor qualidade, independentemente da variabilidade na inicialização dos centroides

3.2.1.2 *Variações do K-Means*

No desenvolvimento de algoritmos de recomendação, a utilização do K-Means é uma abordagem comum para a segmentação de usuários ou itens, permitindo a personalização das recomendações. Contudo, o K-Means tradicional possui limitações que podem ser mitigadas mediante uso de suas variantes. A seguir, discutem-se algumas dessas variações, destacando suas características, benefícios e pontos de atenção.

O K-Means Paralelo/Distribuído permite a escalabilidade e a redução do tempo de processamento ao distribuir a carga de trabalho entre múltiplos núcleos ou

máquinas. No entanto, essa abordagem exige uma implementação mais complexa, com a necessidade de configurar uma infraestrutura especializada e de gerir eficientemente a sobrecarga (*overhead*) de comunicação entre os nós, o que pode impactar a eficiência do sistema (HRUSCHKA; CAMPELLO, 2020b).

O K-Means para Fluxos de Dados é adaptado para ambientes dinâmicos, em que os dados são continuamente atualizados. Essa variante permite a atualização incremental dos *clusters* em tempo real, economizando recursos computacionais e tornando o sistema mais responsivo a mudanças. Porém, sua implementação é mais desafiadora, exigindo sistemas que processam dados em tempo real, e sua aplicação pode ser menos eficiente em contextos em que os dados são mais estáticos e não requerem atualizações frequentes (HRUSCHKA; CAMPELLO, 2020b).

A K-Medianas (*K-Medoids*) oferece maior robustez contra dados atípicos (*outliers*), utilizando a mediana das distâncias entre os pontos de dados, em vez da média, para definir os centroides dos *clusters*. Isso é particularmente vantajoso em cenários com distribuições assimétricas, em que a mediana proporciona uma representação mais precisa de um conjunto de dados. Contudo, o cálculo da mediana é mais intensivo em termos computacionais, pois é necessário determinar a mediana das distâncias entre os pontos de dados dentro de cada *cluster*, o que pode aumentar o tempo de processamento (HRUSCHKA; CAMPELLO, 2020b).

O Método de Múltiplas Execuções de K-Means envolve a execução do algoritmo várias vezes com diferentes inicializações, escolhendo a melhor solução com base na minimização das distâncias intra-*cluster*. Esse método melhora a robustez e aumenta a chance de encontrar uma solução globalmente ótima, tornando os resultados mais consistentes. No entanto, o custo computacional é mais elevado devido ao aumento do número de execuções (HRUSCHKA; CAMPELLO, 2020b). Ademais, a técnica não resolve problemas estruturais do K-Means, como a dificuldade em lidar com *clusters* de formas não globulares ou com tamanhos desiguais.

3.2.1.3 *Otimização de K-Means*

O algoritmo K-Means é amplamente utilizado em algoritmos de recomendação para a segmentação de dados, permitindo a criação de grupos de usuários ou itens

com características semelhantes. A essência do K-Means reside na minimização de uma função objetivo denominada Soma dos Erros Quadráticos (SSE), que mede a variação intra-*cluster*, ou seja, a soma das distâncias quadráticas entre os pontos de dados e seus respectivos centroides dentro de cada *cluster* (HRUSCHKA; CAMPELLO, 2020b).

A função objetivo que o K-Means minimiza é dada pela equação (1):

$$J = \sum_{c=1}^{k} \sum_{\mathbf{x}_j \in \mathbf{C}_c} d\left(\mathbf{x}_j, \bar{\mathbf{x}}_c\right)^2 \quad (1)$$

Na equação (1):

- k é o número de clusters definidos;
- C_c é o conjunto de pontos de dados no c-ésimo cluster;
- x_j representa os pontos de dados pertencentes ao cluster Cc;
- $|\bar{x_c}| = \frac{1}{|C_c|}\sum_{x_j \in C_c} x_j$, em que $|\bar{x_c}|$ é o centroide do c-ésimo cluster;
- $\mathrm{d}(x_j, |\bar{x_c}|)^2$ é a distância escolhida pelo especialista de domínio ao quadrado entre o ponto x_j e o centroide $|\bar{x_c}|$.

O processo de otimização no K-Means é iterativo e consiste em dois passos principais, conhecidos como Passo E (*Expectation*) e Passo M (*Maximization*):

- Passo E (*Expectation*): Nesta etapa, com a fixação do conjunto dos centroides dos *clusters*, denotado por $\{\bar{x_c}\}$, cada ponto x_j é atribuído ao cluster cujo centroide minimiza a distância Euclidiana. Em outras palavras, cada ponto é atribuído ao *cluster* mais próximo. Essa atribuição é realizada para todos os pontos de dados até que todos estejam alocados aos seus *clusters* mais próximos, minimizando, assim, a variabilidade interna do *cluster* (HRUSCHKA; CAMPELLO, 2020b)*;*
- Passo M (*Maximization*): Após a atribuição dos pontos aos clusters, nesta etapa recalculam-se os centroides $\{\bar{x_c}\}$ como a média dos pontos de dados atribuídos a cada cluster. A média dos pontos em cada *cluster* é usada para atualizar a

posição dos centroides, ajustando-os de acordo com a nova configuração dos clusters (HRUSCHKA; CAMPELLO, 2020b).

Esses dois passos são repetidos iterativamente até que haja convergência, ou seja, até que as mudanças nas posições dos centroides e nas atribuições dos pontos sejam mínimas entre as iterações consecutivas. Este processo garante que o algoritmo converge para uma configuração que minimiza a função SSE, resultando em *clusters* que são compactos e bem definidos em termos de variância interna ao mesmo tempo em que a separação dos *clusters* é maximizada (JAIN et al., 1988).

3.2.1.4 *Validação de K-Means*

No desenvolvimento de algoritmos de recomendação, a validação dos *clusters* formados é uma etapa crucial para assegurar a qualidade e a utilidade das segmentações realizadas. Diversos métodos de validação podem ser empregados para avaliar a coesão e a separação dos clusters, além de determinar o número ideal de grupos. Entre os principais métodos de validação, destacam-se o Índice de Silhueta, a Curva do Cotovelo e o Índice de Dunn.

O Índice de Silhueta é uma métrica que avalia a qualidade dos *clusters* formados, medindo a proximidade de cada ponto de dados em relação ao seu próprio cluster em comparação com os demais clusters (HRUSCHKA; CAMPELLO, 2020d). Para cada ponto i, a largura da silhueta $s(i)$ é calculada pela equação (2):

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \quad (2)$$

Na equação (2), $a(i)$ é a distância média do ponto i a todos os outros pontos do mesmo cluster, e $b(i)$ é a menor distância média do ponto i aos pontos em qualquer outro cluster. Valores de silhueta próximos de 1 indicam que os pontos estão bem alocados em seus *clusters*, enquanto valores próximos de -1 sugerem uma possível má alocação. A média dos índices de silhueta de todos os pontos serve como uma medida global da qualidade do agrupamento.

A Curva do Cotovelo é uma técnica visual utilizada para determinar o número ideal de *clusters* ao aplicar algoritmos como o K-Means. Ela é construída plotando-se

a soma das distâncias intra-cluster (*SSE*) em função do número de clusters k. O ponto de inflexão na curva, conhecido como "cotovelo", indica o número ótimo de clusters, a partir do qual a adição de mais clusters não traz uma redução significativa na soma das distâncias (HRUSCHKA; CAMPELLO, 2020d). Esse método é intuitivo e ajuda a evitar a escolha de um número excessivo de *clusters*, o que poderia resultar em uma segmentação excessivamente detalhada.

O Índice de Dunn é outra métrica usada para avaliar a qualidade dos clusters, sendo calculado pela razão entre a menor distância inter-*cluster* e a maior distância intra-*cluster*, conforme a equação (3):

$$D = \frac{min_{1 \le i < j \le k} dist(C_i, C_j)}{max_{1 \le c \le k} dist(C_C)} \quad (3)$$

Na equação (3), $dist(Ci, Cj)$ representa a distância entre os centroides dos clusters Ci e Cj, e $diam(Cc)$ é o diâmetro do cluster Cc, ou seja, a maior distância entre quaisquer dois pontos dentro do cluster (HRUSCHKA; CAMPELLO, 2020d). Um alto índice de Dunn sugere que os clusters estão bem separados e são internamente coesos, representando características desejáveis em grupamento de dados.

3.2.2 **Recomendação por Aprendizado Supervisionado**

Uma vez que os dados para a recomendação estejam devidamente separados em grupos (*clusters*), procede-se com a criação do sistema de recomendação propriamente dito. Avalia-se que esse sistema basear-se-á em técnicas de aprendizado supervisionado, a serem instanciadas e treinadas especificamente para a aplicação alvo conforme a Figura 3.

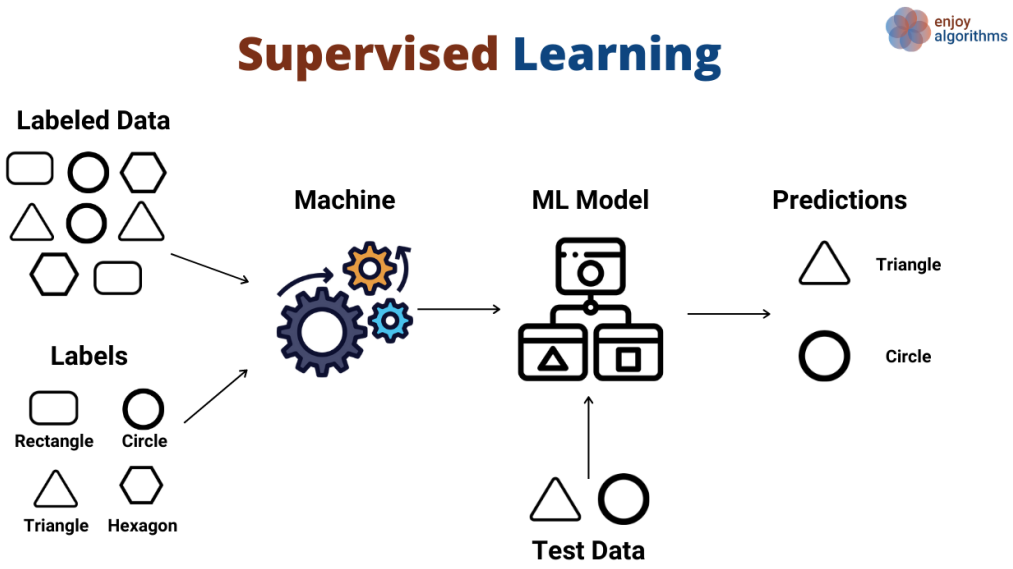


Figura 3 – Aprendizado de Máquina Supervisionado

Fonte: Kozan (2021)

Avalia-se que os seguintes algoritmos são plausíveis para esse fim:

- **Árvore de Decisão**: Consiste em um modelo de aprendizado de máquina que toma decisões baseadas em características dos dados. Ela divide recursivamente o conjunto de dados em subconjuntos, maximizando a homogeneidade das classes em cada subconjunto (IBM, 2024a).
- **Florestas Aleatórias**: Baseia-se em algoritmos de aprendizado de máquina que combinam a saída de várias árvores de decisão para obter um resultado único. Cada árvore de decisão é treinada independentemente e, quando combinadas, resultam em previsões mais precisas. Elas lidam bem com dados ruidosos e são robustas, principalmente quando as árvores individuais não estão correlacionadas entre si (IBM, 2024b);
- **KNN (*K-Nearest Neighbors*)**: Trata-se de um algoritmo que salva amostras junto com suas classes ou valores de saída. Para prever a classe ou o valor de saída de uma nova observação, a distância entre essa observação e todas as amostras de treinamento é calculada selecionando as 'K' amostras mais próximas. A classificação atribui uma nova observação à classe mais comum entre suas vizinhas (HARRISON, 2018);
- **Redes Neurais**: Consiste em sistemas computacionais inspirados na estrutura e no funcionamento do cérebro humano. Também conhecidas como redes

neurais artificiais (RNAs), elas consistem em nós interconectados, chamados de neurônios artificiais, que processam e transmitem informações. A ideia é imitar a maneira como os neurônios biológicos enviam sinais uns para os outros, formando camadas de processamento com conexões ponderadas. À medida que os dados passam por essas camadas, os pesos e limites associados a cada nó são ajustados com base nos dados de treinamento, permitindo que a rede aprenda e melhore sua precisão ao longo do tempo (IBM, 2024c);

- **SVMs (*Support Vector Machines*):** São algoritmos que analisam dados e reconhecem padrões. Elas são usadas tanto para classificação quanto para análise de regressão. O objetivo das SVMs é encontrar um hiperplano de separação que maximize as margens entre as amostras de treinamento. A margem é a distância entre o hiperplano de separação e os vetores de suporte (amostras mais próximas). A introdução da variável de folga permite que as SVMs lidem com classificações erradas e ajustem a largura da margem conforme necessário (ARAÚJO, 2020).

Além dos métodos clássicos de aprendizado supervisionado, um aspecto crucial na construção de sistemas de recomendação baseados em texto é o uso de PLN (SUMESH; ASWINI, 2023). O PLN, combinado ao aprendizado supervisionado, permite a análise automática de grandes volumes de texto, extraindo informações semânticas e sintáticas para identificar padrões e relações entre palavras e frases a partir de treinamentos prévios de modelos de aprendizado supervisionado com grandes volumes de texto (SUMESH; ASWINI, 2023). Isso possibilita recomendações mais precisas em cenários nos quais o conteúdo textual é muito importante, como recomendações de produtos com descrições detalhadas.

O PLN facilita a criação de recomendações por meio da extração de relações sintático-semânticas entre palavras (DELLOVE; KAMARALAJ, 2024). Técnicas de *word embedding*, tal como as presentes em bibliotecas como *Word2Vec* e *GloVe* (WANG, 2024), convertem palavras em vetores de alta dimensionalidade, preservando suas semelhanças de significado. Com essa abordagem, o sistema de recomendação pode considerar tanto características explícitas (como rótulos ou

categorias) quanto relações mais sutis, baseadas em similaridades semânticas (SUMESH; ASWINI, 2023).

Dessa forma, o uso combinado de PLN e aprendizado supervisionado permite, em sistemas de recomendação, identificar temas semelhantes em produtos e, a partir deles, prever quais itens são mais relevantes para o usuário (DELLOVE; KAMARALAJ, 2024). Por conseguinte, o uso de PLN para construir relações sintático-semânticas enriquece os sistemas de recomendação, pois leva em consideração tanto características textuais explícitas quanto as nuances de significado presentes nos dados textuais a partir de um conjunto de dados (*corpus*) utilizado no treinamento que antecede o processo de recomendação (DELLOVE; KAMARALAJ, 2024).

Com a finalização da elaboração do algoritmo de recomendação, a validação dos resultados dar-se-á pelo uso de produtos com recomendações já validadas para comparar com a saída do algoritmo projetado. Assim, é possível realizar eventuais ajustes no sistema para que seu desempenho possa ser iterativamente otimizado, mas mantendo-se suficientemente generalizável (i.e., evitando-se sobreajuste (*overfitting*)).

3.2.3 **Filtragem Baseada em Conteúdo**

A filtragem de conteúdo é uma abordagem amplamente utilizada em sistemas de recomendação, especialmente em contextos nos quais os itens possuem descrições ricas e detalhadas. Esses sistemas baseiam-se nas características dos itens previamente consumidos pelo usuário para recomendar novos itens com atributos semelhantes. Diferentemente da filtragem colaborativa, que considera as preferências de outros usuários, a filtragem de conteúdo foca nas preferências individuais do usuário, criando um perfil personalizado a partir dos atributos dos itens com os quais ele já interagiu.

De acordo com Wang et al. (2018), um sistema de recomendação baseado em conteúdo utiliza perfis tanto de usuários quanto de itens para identificar similaridades entre eles. O perfil do usuário inclui informações sobre suas preferências, gostos e histórico de interações, enquanto o perfil dos itens contém atributos detalhados, como palavras-chave, categorias ou descritores complexos. Esses atributos frequentemente são extraídos por meio de técnicas de PLN. Uma das técnicas fundamentais para a

representação desses atributos é o *Term Frequency – Inverse Document Frequency* (TF-IDF), que mede a relevância de uma palavra em um documento dentro de um conjunto de documentos, permitindo uma comparação eficiente de textos com base nos termos mais significativos (SALTON; MCGILL, 1986).

Essa abordagem possui a vantagem de ser independente dos dados de outros usuários, o que a torna particularmente útil em situações em que há pouca ou nenhuma informação de comportamento coletivo, como no caso de novos usuários ou quando o item é pouco popular, mitigando o problema conhecido como partida a frio ("*cold start*") (LEE; CHO, 2023). Além disso, a filtragem de conteúdo é especialmente eficaz em contextos nos quais o conteúdo dos itens pode ser representado de forma rica e estruturada, como em publicações científicas, filmes ou livros. Isso permite que o sistema ofereça recomendações mais personalizadas e precisas, ajustadas aos interesses individuais do usuário (LEE; CHO, 2023).

No entanto, a filtragem de conteúdo apresenta algumas limitações importantes. Uma das principais é a incapacidade de recomendar itens que estejam fora das preferências históricas do usuário, o que pode restringir a descoberta de novos conteúdos. Outro desafio relevante é a tendência à superespecialização do sistema, que pode gerar recomendações de itens excessivamente semelhantes aos já consumidos, reduzindo a diversidade das sugestões apresentadas (WANG et al., 2018).

A filtragem de conteúdo, portanto, é particularmente eficaz em contextos em que há uma riqueza de informações textuais associadas aos itens recomendados. Em estudos como o de Wang et al. (2018), a aplicação desse modelo a publicações científicas na área de ciência da computação mostrou-se eficiente, identificando publicações relevantes para os usuários com base em atributos textuais extraídos das publicações anteriores. Isso evidencia o potencial de sistemas de recomendação baseados em conteúdo em ambientes nos quais a semântica dos itens desempenha um papel crucial.

3.3 Otimização e Ajuste Fino

Durante a fase do desenvolvimento do trabalho, é previsto um processo detalhado de otimização e ajuste fino dos parâmetros dos modelos de IA do sistema de recomendação. Isso envolve a exploração de diferentes configurações e técnicas

de otimização, com o objetivo de melhorar tanto o desempenho quanto a capacidade de generalização dos modelos instanciados.

Por exemplo, anteveem-se experimentos sistemáticos para determinar os melhores conjuntos de hiperparâmetros que maximizem as métricas de desempenho de interesse e a relevância das recomendações geradas. Considera-se que esses experimentos se baseiam em boas práticas, como validação cruzada iterativa com realimentações e ajustes derivados dos testes pós-validação cruzada. Os indicadores utilizados para mensurar e avaliar a qualidade das recomendações são apresentados na estratégia de avaliação de resultados da seção 3.4.

3.4 Testes Reais e Avaliação de Resultados

Na fase de testes reais, os modelos desenvolvidos são submetidos a avaliações práticas com usuários reais. São conduzidos testes de usabilidade e aceitação, nos quais os usuários são convidados a interagir com as recomendações geradas e fornecer realimentações (*feedback*) sobre sua experiência.

Após a fase de desenvolvimento dos modelos, os testes reais são conduzidos seguindo um procedimento específico. Primeiramente, os modelos são implantados em um ambiente de produção, utilizando tecnologias que permitam sua escalabilidade e a confiança em seu funcionamento. Os sistemas de recomendação rastreiam as escolhas dos usuários, coletando informações sobre quais itens foram recomendados, clicados e ignorados.

Durante os testes de usabilidade, os usuários são convidados a interagir com as recomendações, avaliando aspectos como a facilidade de uso, a clareza, precisão das recomendações e a experiência geral. A realimentação dos usuários é coletada por meio de pesquisas, avaliações de qualidade ou comentários diretos, fornecendo dados qualitativos sobre a qualidade das recomendações.

Além disso, métricas quantitativas como número de iterações até chegar no produto desejado e tempo gasto no sistema são coletadas para avaliar a eficácia das recomendações. Com base nos dados coletados, torna-se possível realizar análises quantitativas das métricas de satisfação e engajamento dos usuários, fornecendo um discernimento valioso sobre a eficácia e a utilidade das recomendações geradas pelo sistema. Caso necessário, os modelos são iterativamente ajustados, otimizando

hiperparâmetros, selecionando algoritmos ou modificando as regras de filtragem para melhorar os resultados em função da realimentação dos usuários.

4. ESPECIFICAÇÃO DE REQUISITOS

Este capítulo documenta a especificação dos requisitos do sistema responsável pela criação de recomendações personalizadas para usuários de lojas virtuais. A apresentação dos requisitos inspira-se na abordagem proposta por Oberg, Probasco e Ericsson (2000). A caracterização de um requisito compreende as seguintes informações:

- Código de Identificação;
- Nome;
- Descrição;
- Prioridade;
- Estabilidade;
- *Rationale* (informações adicionais);
- Requisitos associados.

Os requisitos especificados para o projeto são apresentados entre a Tabela 3 e a Tabela 11

Tabela 3 – Especificação do Requisito 00

Código: 00	☒ Funcional ☐ Não Funcional
Requisito: Base de dados.	
Descrição: O sistema deve receber uma base de dados como entrada para realizar sua análise e processamento.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☒ Alta ☐ Média ☐ Baixa	
***Rationale*:** A entrada de uma base de dados é essencial para o funcionamento do sistema. Fonte de base de dados: “Amazon Product Dataset 2020” (PROMPTCLOUD, 2020).	
Requisitos associados: #01 e #02	

Tabela 4 – Especificação do Requisito 01

Código: 01	☒ Funcional ☐ Não Funcional
Requisito: Manipulação de dados.	
Descrição: O sistema deve realizar a limpeza e o pré-processamento dos dados para viabilizar sua análise por especialistas de domínio.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☒ Alta ☐ Média ☐ Baixa	
***Rationale*:** A limpeza e o pré-processamento dos dados são essenciais para garantir a qualidade das análises e o funcionamento adequado do sistema, evitando erros e inconsistências nas informações e, consequentemente, resultados.	
Requisitos associados: #00 e #02	

Tabela 5 – Especificação do Requisito 02

Código: 02	☒ Funcional ☐ Não Funcional
Requisito: Divisão dos dados (treino e teste) para testes reais com usuários.	
Descrição: Para os testes reais com usuários, o sistema deve permitir realizar a devida divisão dos dados em conjuntos para treinamento e teste para aprimorar as recomendações do sistema.	
Prioridade: ☐ Alta ☒ Média ☐ Baixa	
Estabilidade: ☐ Alta ☒ Média ☐ Baixa	
***Rationale*:** A divisão dos dados é relevante para promover treinamentos iterativos do sistema com base em realimentações de usuários. O objetivo dessa realimentação é promover o aumento da confiança e da eficiência dos modelos no projeto, segundo respostas de usuários. Referência para implementação: biblioteca Python "*sklearn*" (SCIKIT-LEARN TEAM, 2024), biblioteca *gensim* (ŘEHŮŘEK, 2024).	
Requisitos associados: #07	

Tabela 6 – Especificação do Requisito 03

Código: 03	☒ Funcional ☐ Não Funcional
Requisito: Seleção e criação de algoritmo de recomendação.	
Descrição: O sistema deve ser provido de um algoritmo de recomendação que permita, a partir de dados de entrada, gerar sugestões de itens correlatos que possam ser comparáveis com recomendações de outros sistemas.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☒ Alta ☐ Média ☐ Baixa	
***Rationale*:** Implementar um algoritmo de recomendação é o núcleo do projeto. Referências para implementação: biblioteca *RapidFuzz* (PYTHON SOFTWARE FOUNDATION, 2024a), biblioteca *nltk* (THE NLTK TEAM, 2024) e biblioteca *gensim* (ŘEHŮŘEK, 2024).	
Requisitos associados: #04	

Tabela 7 – Especificação do Requisito 04

Código: 04	☒ Funcional ☐ Não Funcional
Requisito: Seleção de sistema de recomendação de referência para comparação.	
Descrição: Deve existir um sistema de recomendação de referência para que se possa comparar os resultados e o desempenho do sistema projetado com a referência preexistente.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☒ Alta ☐ Média ☐ Baixa	
***Rationale*:** Comparar o sistema a referências preexistente é crucial para analisar o cumprimento dos objetivos do projeto. Referência para comparação: Sistema preexistente na plataforma GitHub *"Amazon E-Commerce Recommendation System Using Content-Based Filtering"* (CHEAH et al., 2020).	
Requisitos associados: #03 e #05	

Tabela 8 – Especificação do Requisito 05

Código: 05	☒ Funcional ☐ Não Funcional
Requisito: Comparação dos resultados.	
Descrição: Deve ser possível comparar o sistema projetado com o sistema de referência e analisar as melhorias do sistema desenvolvido em relação ao sistema de referência.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☐ Alta ☒ Média ☐ Baixa	
***Rationale*:** Validar o projeto requer a comparação e avaliação da presunção de melhoria de desempenho em relação ao sistema de referência. As possíveis métricas a serem analisadas compreendem os índices descritos na seção 3.2.1.4.	
Requisitos associados: #04	

Tabela 9 – Especificação do Requisito 06

Código: 06	☐ Funcional ☒ Não Funcional
Requisito: Tempo de resposta para as recomendações.	
Descrição: É necessário que o algoritmo forneça recomendações em tempo hábil < 90 segundos para garantir uma boa experiência do usuário e a eficiência do sistema.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☒ Alta ☐ Média ☐ Baixa	
***Rationale*:** Restrição importante para a viabilidade técnica do desenvolvimento do projeto, uma vez que tempos de resposta elevados comprometem a usabilidade e a aceitação do sistema por parte dos usuários.	
Requisitos associados: #03, #04, #05, #07 e #08	

Tabela 10 – Especificação do Requisito 07

Código: 07	☒ Funcional ☐ Não Funcional
Requisito: Testes reais com usuários.	
Descrição: O sistema deve permitir a execução de testes reais com usuários, em que eles possam interagir com as recomendações geradas e fornecer realimentações sobre sua experiência.	
Prioridade: ☐ Alta ☒ Média ☐ Baixa	
Estabilidade: ☐ Alta ☒ Média ☐ Baixa	
***Rationale*:** Realização de testes com usuários reais é relevante para avaliar a usabilidade e a aceitação das recomendações do algoritmo, garantindo que o produto final atenda às necessidades e expectativas dos usuários.	
Requisitos associados: #05	

Tabela 11 – Especificação do Requisito 08

Código: 08	☐ Funcional ☒ Não Funcional
Requisito: Recursos computacionais mínimos	
Descrição: O sistema deve, no mínimo, ser passível de execução em um computador com as seguintes especificações: processador Intel Core i3-2350M, 8GB de RAM, armazenamento de 128GB.	
Prioridade: ☒ Alta ☐ Média ☐ Baixa	
Estabilidade: ☐ Alta ☒ Média ☐ Baixa	
***Rationale*:** Recursos computacionais mínimos são cruciais para garantir que o sistema opere de forma eficiente e estável, evitando problemas de desempenho computacional e garantindo uma boa experiência do usuário.	
Requisitos associados: #06	

5. ARQUITETURA DO SISTEMA DE RECOMENDAÇÃO

Este capítulo da monografia é destinado ao detalhamento da arquitetura do sistema de recomendação. Ela é descrita tanto sob o ponto de vista estrutural, com a identificação de seus principais componentes, quanto sob a ótica funcional, com detalhes da dinâmica operacional salientados por diagramas de sequência.

5.1 DIAGRAMAS FUNCIONAL E ESTRUTURAL

A arquitetura do sistema foi definida por meio de uma representação funcional, que descreve a dinâmica do comportamento do sistema, e de uma representação estrutural, que define os componentes utilizados pelo sistema para seu funcionamento. O diagrama da arquitetura funcional do sistema é representado na Figura 4.

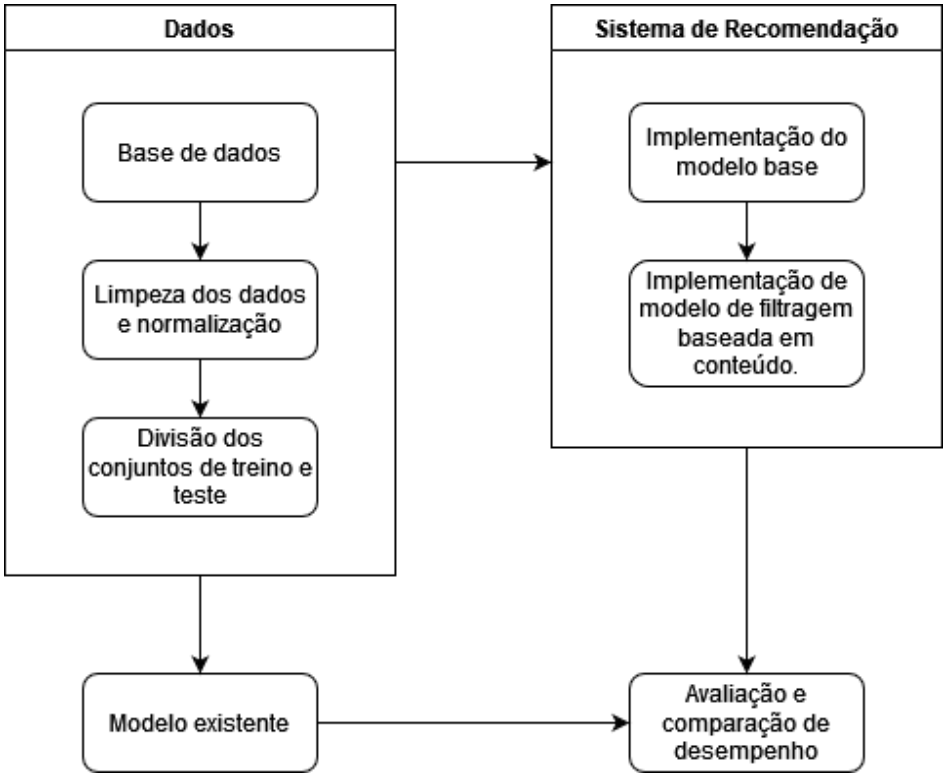


Figura 4 – Diagrama da Arquitetura Funcional do Sistema

O diagrama da arquitetura funcional do sistema apresenta uma mesma entrada para dois modelos distintos, um desenvolvido com o auxílio dos diversos métodos exemplificados, denominado “Sistema de Recomendação”, e outro para comparação

de resultados e análise de métricas de desempenho, indicado como “Modelo existente”.

A primeira etapa envolve o pré-processamento da base de dados, incluindo a limpeza para remover valores desnecessários, eventual agrupamento semiautomatizado via aprendizado não supervisionado e conhecimento especialista, e a separação dos dados em conjuntos de treino e teste. Em seguida, ambos os algoritmos recebem o mesmo conjunto de dados e geram resultados para comparação na etapa seguinte.

O diagrama da arquitetura estrutural do sistema, por sua vez, é representado na Figura 5. Nele são representados os pacotes de software do sistema utilizadas para o desenvolvimento de cada etapa, seguindo a dinâmica descrita pela Figura 4. As relações hierárquicas entre cada bloco são ilustradas pelas setas e indicam que os dados tratados pelo pacote de origem de uma seta são carregados e utilizados no pacote de destino da mesma seta. Bibliotecas de terceiros utilizadas dentro dos pacotes sinalizados na Figura 4 são indicadas e descritas na Tabela 12 e ao longo da seção 5.2.

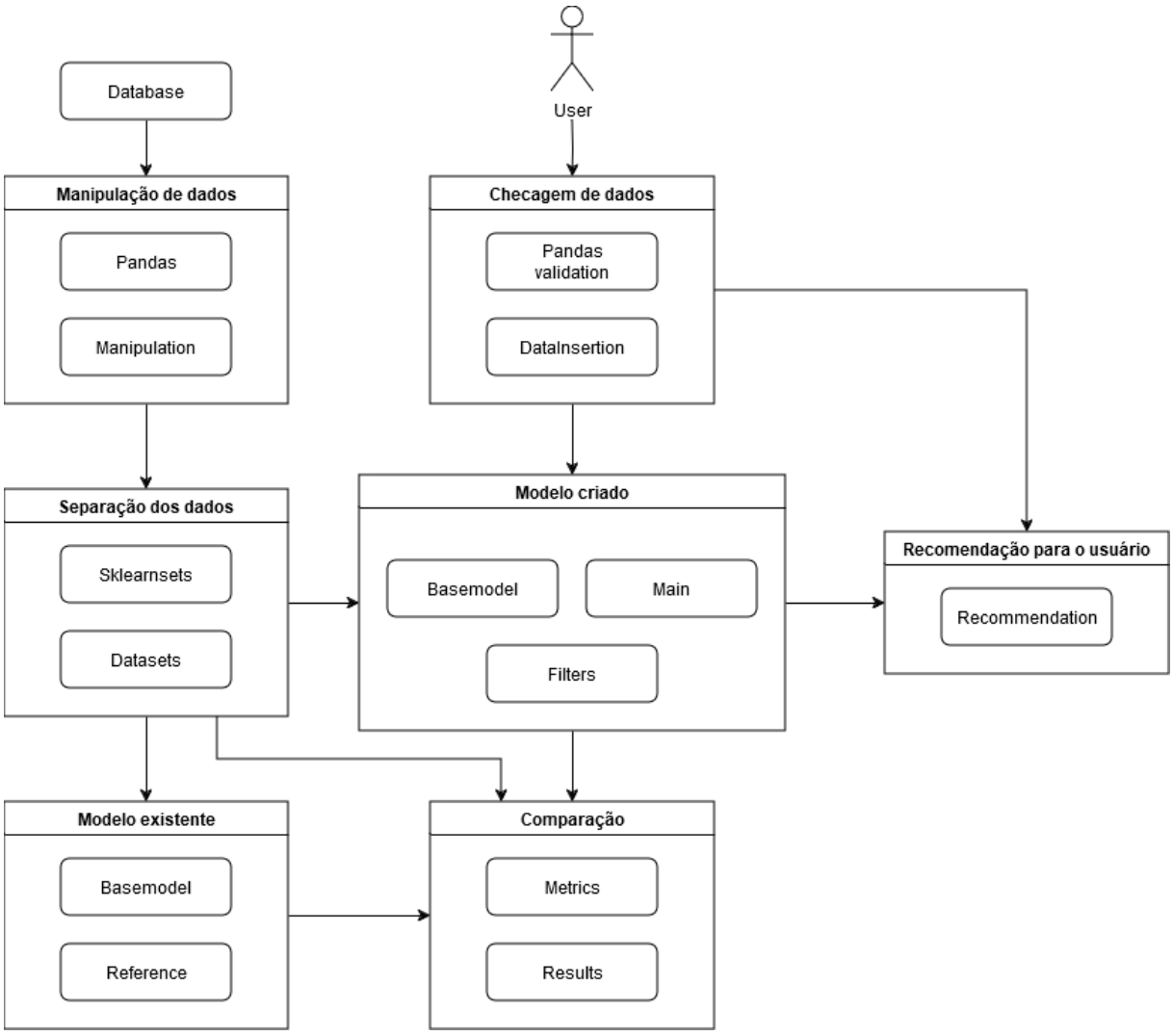


Figura 5 – Diagrama da Arquitetura Estrutural do Sistema

Os pacotes de software referenciados na Figura 5 são descritos na Tabela 12.

Tabela 12 – Descrição dos Pacotes de Software da Figura 5

Pacote	Descrição
Database	Pacote que contém a base de dados do sistema (existente ou criado).
Pandas	Pacote que realiza a conversão dos dados da base de dados em *dataframes* para processamento pelo sistema.
Manipulation	Pacote que realiza a manipulação dos dados com o auxílio das ferramentas disponíveis no pacote *pandas*.
Datasets	Pacote que realiza a carga dos dados manipulados para separação em conjuntos de treinamento e teste. Também armazena os conjuntos de dados de treinamento e teste gerados com o auxílio das ferramentas disponíveis no pacote *sklearn.*

Pacote	Descrição
Sklearnsets	Pacote que separa os dados em conjuntos de treinamento e teste com o auxílio das ferramentas disponíveis no *pacote sklearn.*
Basemodel	Pacote que realiza a carga dos conjuntos de treinamento e teste para criação do modelo base de um sistema (existente ou criado).
Reference	Pacote que realiza a criação do modelo de referência.
Pandasvalidation	Pacote que realiza a validação dos dados fornecidos para o usuário com o auxílio das ferramentas disponíveis no pacote *pandas*.
DataInsertion	Pacote que realiza a carga de dados fornecidos pelo usuário ao sistema (existente ou criado) e armazena seu valor validado.
Main	Pacote que realiza a criação do modelo desenvolvido com o auxílio do pacote de filtragem (*Filters*).
Filters	Pacote que realiza a filtragem dos dados de entrada com base nos atributos de categoria existentes e com o auxílio de recursos e ferramentas disponíveis nas bibliotecas *RapidFuzz, gensim* e *nltk*, utilizadas dentro do próprio pacote.
Results	Pacote que realiza a carga dos resultados de ambos os modelos (existente e criado) e armazena o resultado da comparação entre os mesmos.
Metrics	Pacote que executa as métricas para avaliação da comparação entre os modelos base (existente e criado).
Recommendation	Pacote que executa a geração de uma recomendação para o usuário.

5.2 DIAGRAMAS DE SEQUÊNCIA

O diagrama de sequência apresentado na Figura 6 representa a manipulação dos dados da base de dados escolhida, prevista nos requisitos #00 e #01. Os dados são lidos com auxílio da biblioteca *pandas* (THE PANDAS TEAM, 2024) e manipulados para que apenas os dados relevantes para o projeto sejam selecionados. Após a manipulação, o subconjunto de dados gerado (*“dataframe pandas*”) é salvo em outro arquivo.

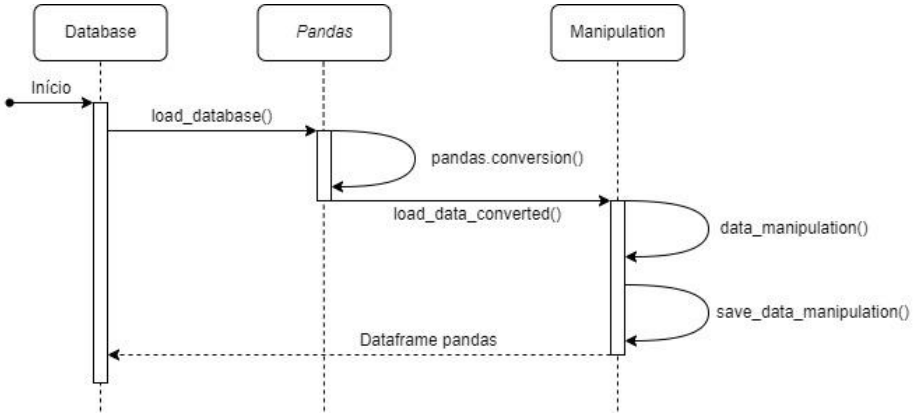


Figura 6 – Diagrama de Sequência para os Requisitos #00 e #01

O diagrama de sequência do requisito #02 é ilustrado na Figura 7. Nele, é demonstrada a divisão dos dados em subconjuntos de treinamento e de teste utilizando a biblioteca *sklearn* (SCIKIT-LEARN TEAM, 2024). Após a separação desses subconjuntos, esses dados são salvos em um modelo projetado para a manipulação de conjuntos de dados.

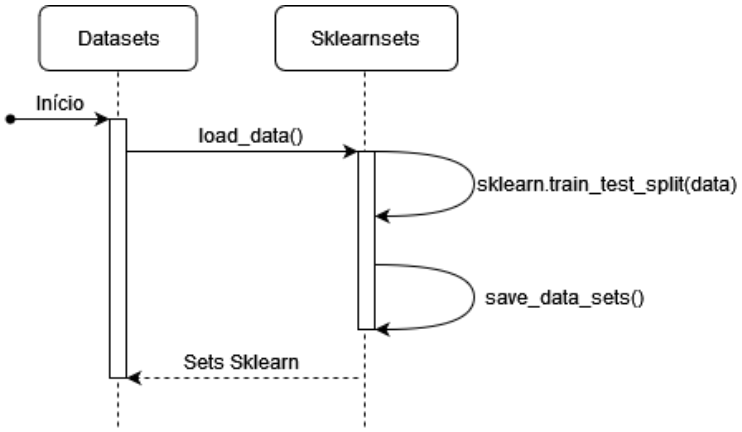


Figura 7 – Diagrama de Sequência para o Requisito #02

O diagrama de sequência do requisito #03, ilustrado na Figura 8, destaca as principais funções para a criação e execução do modelo de IA desenvolvido. O processo começa com a carga dos parâmetros necessários, seguido pelo pré-processamento que organiza e transforma os dados em vetores. Em seguida, os filtros refinam as informações, permitindo que o modelo seja treinado e gere as recomendações. Por fim, o modelo completo é salvo.

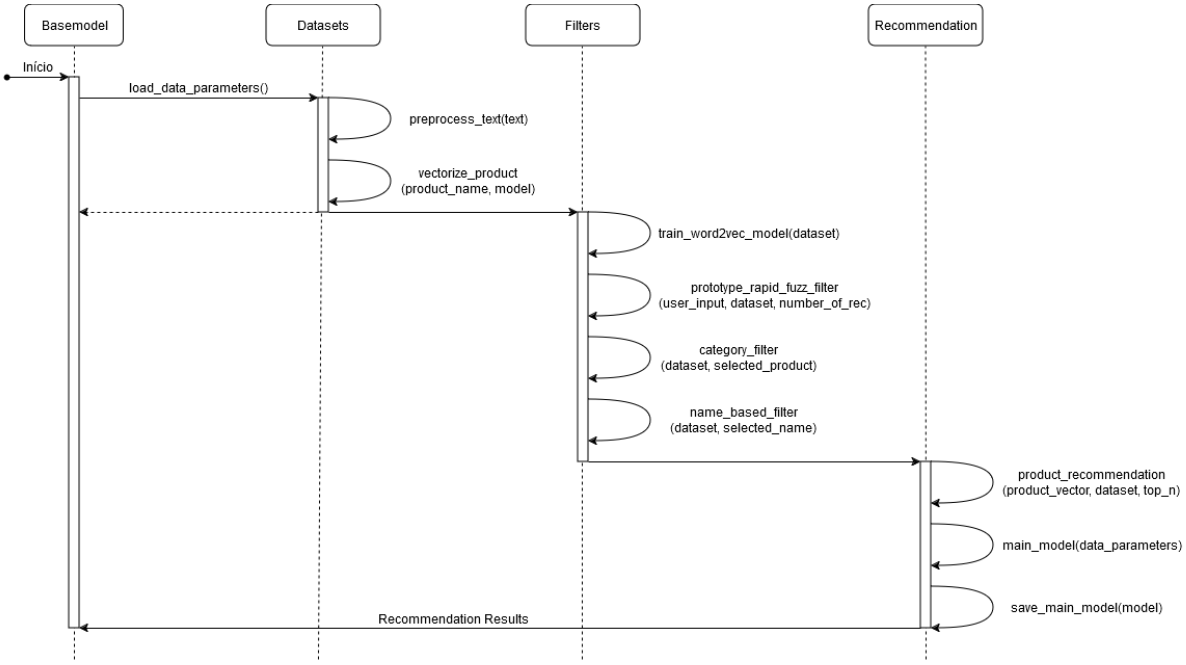


Figura 8 – Diagrama de Sequência para o Requisito #03

O diagrama de sequência do requisito #04, ilustrado na Figura 9, representa o fluxo de operações necessário para a manipulação e a preparação dos dados de referência. O processo inicia-se com a carga dos dados brutos, que são convertidos para o formato adequado e manipulados, incluindo a criação de variáveis auxiliares (*dummy*) e a normalização dos dados. Após tal pré-processamento, os dados manipulados são particionados e carregados no sistema de referência. Por fim, o sistema de referência é treinado com esses dados e salvo para ser utilizado, juntamente com as recomendações produzidas por ele, no Requisito #05.

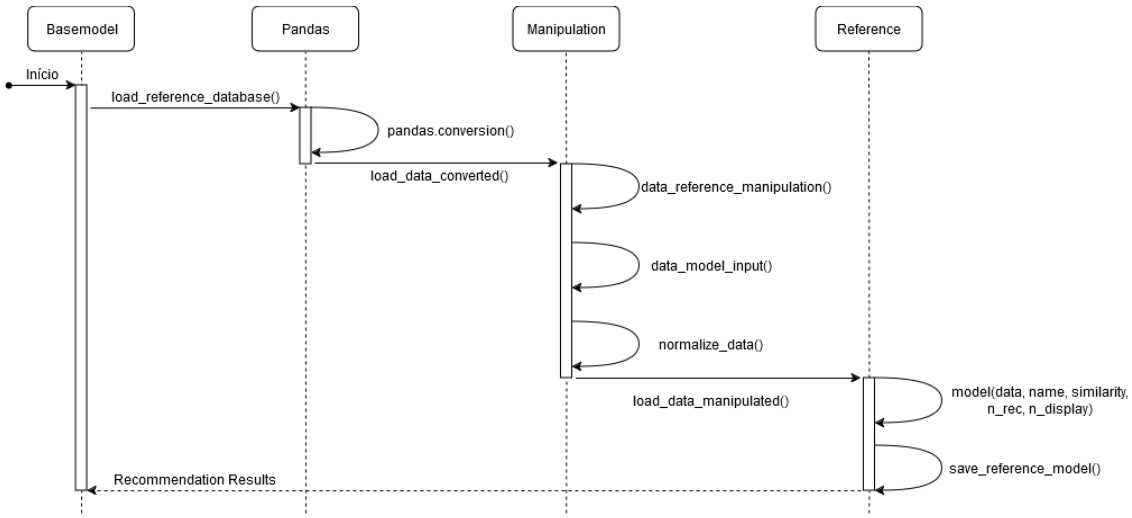


Figura 9 – Diagrama de Sequência para o Requisito #04

O diagrama de sequência do requisito #05, mostrado na Figura 10, detalha as etapas de avaliação e comparação dos modelos. O processo começa com a aquisição dos modelos do sistema de referência e do sistema desenvolvido neste trabalho, seguido pela obtenção dos resultados. Em seguida, as métricas de desempenho são calculadas e os dados são comparados para avaliar a eficácia de cada modelo. O processo é finalizado com o armazenamento dos resultados de comparação, permitindo análises posteriores sobre o desempenho dos modelos treinados.

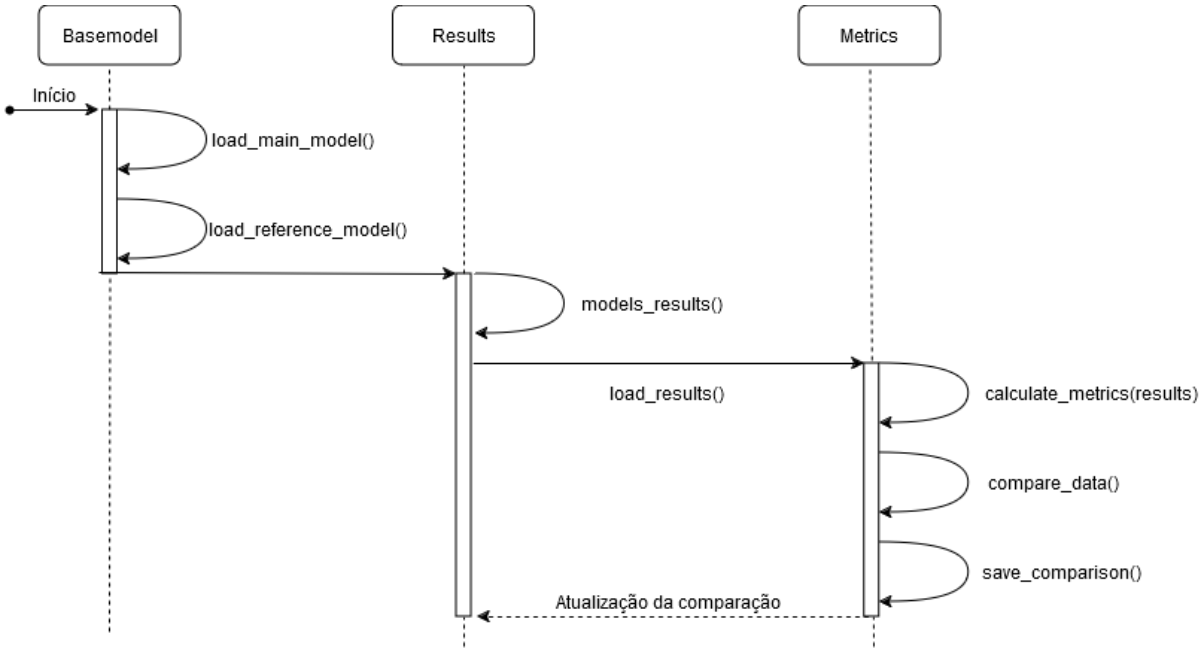


Figura 10 – Diagrama de Sequência para o Requisito #05

O diagrama de sequência do requisito #07 é representado na Figura 11. Nele é representada a execução de interação do sistema com usuários reais, utilizando o sistema implementado para realizar uma recomendação para um usuário de acordo com os seus dados fornecidos.

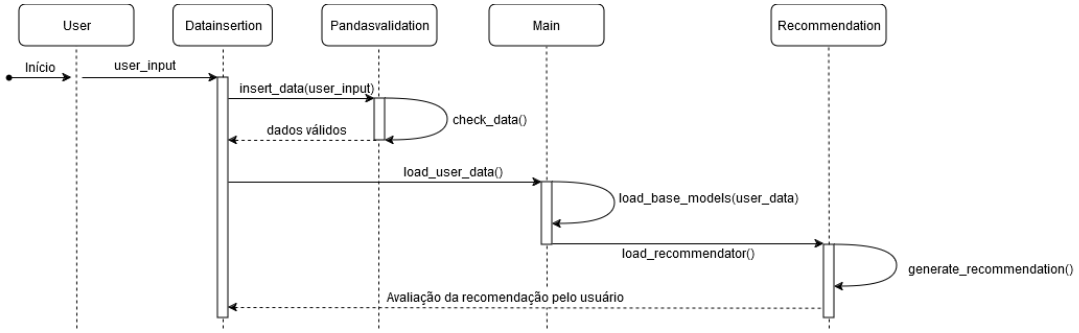


Figura 11 – Diagrama de Sequência para o Requisito #07

Os métodos referenciados nos diagramas de sequência da Figura 6 até a Figura 11 são descritos na Tabela 13. Os métodos foram ordenados de acordo com a ordem de suas primeiras referências entre a Figura 6 e a Figura 11.

Tabela 13 – Descrição dos Métodos Apresentados nos Diagramas de Sequência

Método	Descrição
load_database()	Elemento do pacote *Database* que realiza a carga da base de dados.
pandas_conversion()	Elemento do pacote *pandas* que converte os dados em *dataframes*.
load_data_converted()	Elemento do pacote *pandas* que realiza a carga dos dados convertidos em *dataframes.*
data_manipulation()	Elemento do pacote *Manipulation* que realiza a manipulação dos dados com o auxílio das ferramentas disponíveis no pacote *pandas*.
save_data_manipulation()	Elemento do pacote *Manipulation* que salva as manipulações feitas sobre os dados.
load_data()	Elemento do pacote *Datasets* que realiza a carga dos dados tratados.
sklearn.train_test_split(data)	Elemento do pacote *Sklearnsets* que separa os dados tratados em conjuntos de treino e teste com o auxílio das ferramentas disponíveis no pacote *sklearn*.
save_data_sets()	Elemento do pacote *Sklearnsets* que salva os conjuntos de dados de treinamento e teste gerados com o auxílio das ferramentas disponíveis no pacote *sklearn*.
load_data_parameters()	Elemento do pacote *Basemodel* que realiza a carga dos dados e parâmetros utilizados para criação do modelo base (existente ou criado).
preprocess_text(text)	Elemento do pacote *Datasets* que realiza o pré-processamento do texto, incluindo a remoção de stopwords e lematização. Utiliza a biblioteca *nltk* para essa finalidade.

Método	Descrição
vectorize_product(product_name, model)	Elemento do pacote *Datasets* que vetoriza o nome do produto usando a *biblioteca gensim* (*Word2Vec*) para transformar as palavras em vetores numéricos.
train_word2vec_model(model)	Elemento do pacote *Filters* que treina um modelo da biblioteca *gensim* (*Word2Vec*) com base no conjunto de dados fornecido
prototype_rapid_fuzz_filter(user_input, dataset, number_of_rec)	Elemento do pacote *Filters* que filtra produtos usando uma proximidade morfossintática para encontrar recomendações que correspondam ao input do usuário.
category_filter(dataset, selected_product)	Elemento do pacote *Filters* que filtra o conjunto de dados para selecionar produtos que compartilhem categorias semelhantes ao produto selecionado.
name_based_filter(dataset,product_name)	Elemento do pacote *Filters* que filtra o conjunto de dados baseado no nome do produto, pré-processando e vetorização do nome para encontrar itens semelhantes.
product_recommendation(product_vector, dataset, top_n)	Elemento do pacote *Recommendation* que determina as recomendações, usando os filtros implementados, dos *top_n* produtos mais semelhantes ao vetor do produto fornecido.
main_model(data_parameters)	Elemento do pacote *Recommendation* que cria uma instância do modelo desenvolvido.
save_main_model(model)	Elemento do pacote *Recommendation* que salva o modelo desenvolvido em um arquivo.
load_reference_database()	Elemento do pacote *Basemodel* que carrega a base de dados de referência para ser utilizada nas manipulações de dados.
data_reference_manipulation()	Elemento do pacote *Manipulation* que realiza a manipulação de dados no sistema de referência

Método	Descrição
data_model_input()	Elemento do pacote *Manipulation* que prepara os dados para entrada no sistema de referência, organizando e estruturando-os.
normalize_data()	Elemento do pacote *Manipulation* que normaliza os dados para padronização de uso no sistema de referência.
load_data_manipulated()	Elemento do pacote *Manipulation* que realiza a carga dos dados tratados em *dataframes* para serem processados pelo sistema de referência.
model (data, name, similarity, n_rec, n_display)	Elemento do pacote *Reference* que cria e treina um modelo de recomendação com os dados de treino e parâmetros do sistema de referência.
save_reference_model()	Elemento do pacote *Reference* que salva o modelo do sistema de referência em um arquivo.
load_main_model()	Elemento do pacote *Basemodel* que carrega o modelo do sistema de recomendação desenvolvido previamente salvo.
load_reference_model()	Elemento do pacote *Basemodel* que carrega o modelo do sistema de referência previamente salvo
load_models()	Elemento do pacote *Results* que arrega ambos os modelos dos sistemas (desenvolvido e de referência) para comparações e análise de desempenho.
models_results()	Elemento do pacote *Results* que gera e exibe os resultados da execução de ambos os modelos de sistemas (desenvolvido e de referência), permitindo a avaliação das recomendações.
load_results()	Elemento do pacote *Results* que realiza a carga dos resultados de ambos os modelos de sistemas (desenvolvido e de referência).

Método	Descrição
calculate_metrics(results)	Elemento do pacote Metrics que realiza o cálculo das métricas de ambos os modelos de sistemas (desenvolvido e de referência).
compare_data()	Elemento do pacote *Metrics* que compara os resultados de ambos os modelos de sistemas (desenvolvido e de referência).
save_comparison()	Elemento do pacote *Metrics* que salva o resultado da comparação dos modelos de sistemas (desenvolvido e de referência) em um arquivo.
insert_data(user_input)	Elemento do pacote *Datainsertion* que recebe os dados inseridos pelo usuário (*user_input*) e realiza a sua carga em um *dataframe pandas*.
check_data()	Elemento do pacote *Pandasvalidation* que verifica se os dados fornecidos pelo usuário são válidos. Essa verificação é realizado com o auxílio das ferramentas disponíveis na biblioteca *pandas*.
load_user_data()	Elemento do pacote *Datainsertion* que realiza a carga dos dados validados fornecidos pelo usuário.
load_base_models(user_data)	Elemento do pacote *Main* que carrega o modelo do sistema de recomendação desenvolvido e o executa com o conjunto de dados validados fornecidos pelo usuário.
load_recommendator()	Elemento do pacote *Main* que carrega o resultado da execução do modelo do sistema de recomendação desenvolvido com o conjunto de dados validados fornecidos pelo usuário.
generate_recommendation()	Elemento do pacote *Recommendation* que gera recomendações para o usuário.

6. PLANEJAMENTO DO PROJETO

Este capítulo do documento relata o planejamento do desenvolvimento do projeto e inclui (i.) o cronograma das atividades do projeto e (ii.) os recursos utilizados em seu desenvolvimento.

6.1 CRONOGRAMA DE ATIVIDADES DO PROJETO

O cronograma da Tabela 14 identifica a distribuição mensal das atividades executadas no projeto. As células marcadas com um "**X"** indicam que a atividade de uma linha foi realizada no mês da coluna correspondente (2 – Fevereiro até 12 – Dezembro).

Tabela 14 – Cronograma de Atividades do Projeto

Atividade	2024										
	2	3	4	5	6	7	8	9	10	11	12
I. Familiarização com Temas Introdutórios e Referências do Projeto	X	X									
II. Revisão de Literatura		X	X	X	X	X	X	X	X	X	
III. Especificação do Projeto			X	X	X	X	X	X	X	X	
IV. Desenvolvimento do Sistema					X	X	X	X	X	X	
IV-a. Implementação do Requisito #00						X					
IV-b. V&V do Requisito #00						X					
IV-c. Implementação do Requisito #01						X	X	X			
IV-d. V&V do Requisito #01						X	X	X			
IV-e. Implementação do Requisito #02									X	X	
IV-f. V&V do Requisito #02										X	
IV-g. Implementação do Requisito #03							X	X	X	X	
IV-h. V&V do Requisito #03							X	X	X	X	
IV-i. Implementação do Requisito #04						X	X	X			
IV-j. V&V do Requisito #04							X	X			
IV-k. Implementação do Requisito #05								X	X	X	
IV-l. V&V do Requisito #05								X	X	X	
IV-m. Implementação do Requisito #06								X	X	X	
IV-n. V&V do Requisito #06								X	X	X	

Atividade	2024										
	2	3	4	5	6	7	8	9	10	11	12
IV-o. Implementação do Requisito #07									X	X	
IV-p. V&V do Requisito #07										X	
IV-q. Implementação do Requisito #08					X	X	X	X	X	X	
IV-r. V&V do Requisito #08					X	X	X	X	X	X	
V. Documentação Técnica do Projeto	X	X	X	X	X	X	X	X	X	X	X
VI. Apresentação Final do Projeto											X

A estratégia de implementação consistiu em desenvolver os requisitos seguindo os diagramas de sequência apresentados. Dado que algumas entradas de partes do sistema dependem de saídas de outras, o desenvolvimento do sistema foi ordenado de maneira que os requisitos dependentes só fossem implementados após a validação correta das suas respectivas dependências. Após todos os requisitos terem sido implementados, também houve uma etapa de verificação e validação de todos os requisitos em conjunto.

Avalia-se que o cronograma inicialmente idealizado para o projeto requereu ajustes pontuais apenas nas durações dos requisitos #02 e #03, que foram estendidas em um mês. Tais ajustes relacionaram-se ao prolongamento de estudos e testes de utilização das bibliotecas de software envolvidas na implementação desses requisitos até que resultados satisfatórios fossem atingidos.

Dessa forma, com exceção das prorrogações dos requisitos #02 e #03, o restante do planejamento foi seguido sem alterações durante todo o desenvolvimento do projeto. Ao final, culminou-se no cumprimento dos objetivos estabelecidos dentro dos prazos estabelecidos no projeto.

6.2 Recursos Técnicos Utilizados no Projeto

Os recursos técnicos utilizados no projeto são listados a seguir. Nessa lista, são especificados itens como hardware, bases de dados, documentos técnicos e artefatos de software, bibliotecas, linguagens de programação e ambientes de desenvolvimento.

- Biblioteca *sklearn* (SCIKIT-LEARN TEAM, 2024);

- Biblioteca *pandas* (THE PANDAS TEAM, 2024);
- Biblioteca *numpy* (THE NUMPY TEAM, 2022);
- Biblioteca *gensim* (ŘEHŮŘEK, 2024);
- Biblioteca *nltk* (THE NLTK TEAM, 2024);
- Biblioteca *fuzzywuzzy* (PYTHON SOFTWARE FOUNDATION, 2020);
- Biblioteca *RapidFuzz* (PYTHON SOFTWARE FOUNDATION, 2024a)
- Python 3.12.3 (PYTHON SOFTWARE FOUNDATION, 2024b);
- Visual Studio Code (MICROSOFT CORPORATION, 2024a);
- Repositório no GitHub (ISHICAVA; ITO; INOUE, 2024);
- Hardware computacional pessoal do aluno:
    - Processador: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz;
    - RAM DDR4 8GB;
    - Sistema Operacional Windows 11 Home 64 bits;
    - Armazenamento 475GB SSD.
- Base de dados "*Dataset de produtos da Amazon*" (PROMPTCLOUD, 2020);
- Sistema de recomendação de referência "*Amazon E-Commerce Recommendation System Using Content-Based Filtering*" (CHEAH et al., 2020);
- *Microsoft Hyper V* (MICROSOFT CORPORATION, 2024b).

6.3 CUSTOS DO PROJETO

A Tabela 15 apresenta a descrição dos custos do desenvolvimento do projeto, que inclui a quantificação dos recursos estimadas e as respectivas justificativas.

Tabela 15 – Descrição dos Custos do Projeto

Parâmetro	Valor	Justificativa
Esforço de Trabalho (Hora-Homem – HH)	30HH semanais (10HH por membro) por 11 meses Total: 1350HH	Custo de tempo de desenvolvimento do sistema para os três membros.

Parâmetro	Valor	Justificativa
Custo Financeiro para Recursos do Projeto (Reais Brasileiros – BRL)	Nulo	As ferramentas de software utilizadas são *Open Source* e não apresentam nenhum custo adicional.

Avalia-se que todas as estimativas de custo planejadas no projeto foram cumpridas ao longo de seu desenvolvimento, sem a necessidade de quaisquer alterações. Isso se deve ao fato de que, apesar de a implementação dos requisitos #02 e #03 ter requerido mais esforço, culminando nas prorrogações de um mês descritas na seção 6.1, outros requisitos, como #00, #04, #06 e #07, demandaram menos esforço do que a previsão inicial. Por conseguinte, o esforço global de dedicação ao projeto manteve-se dentro do patamar planejado.

7. DESENVOLVIMENTO DO SISTEMA DE RECOMENDAÇÃO

Este capítulo da monografia destina-se a apresentar o desenvolvimento do projeto e inclui a criação do ambiente de desenvolvimento, o tratamento da base de dados, a análise detalhada do sistema de referência, a análise exploratória dos dados e a implementação do sistema de recomendação.

A base de dados utilizada, juntamente com os códigos desenvolvidos no relato deste capítulo, estão presentes em um repositório no GitHub criado para o presente projeto (ISHICAVA; ITO; INOUE, 2024).

7.1 AMBIENTE DE DESENVOLVIMENTO

Com o intuito de facilitar e uniformizar o desenvolvimento do projeto para todos os integrantes do grupo, decidiu-se criar um ambiente de desenvolvimento comum para todos os membros para as atividades de implementação, verificação e validação do projeto.

Para essa finalidade, optou-se por utilizar uma máquina virtual, uma vez que nela podem ser instalados e configurados os pacotes, bibliotecas e ferramentas necessários, isolando-os do ambiente físico das máquinas físicas e facilitando a detecção e o tratamento de eventuais problemas. Além disso, o compartilhamento da máquina virtual é relativamente simples, uma vez que, ao compartilhar a máquina virtual inteira (configurações e arquivo do disco rígido virtual) em si, todas as configurações feitas por um usuário podem ser facilmente aderidas pelos demais.

Avaliou-se, em função das características e bibliotecas usadas no projeto, que o uso do sistema operacional *Ubuntu* (CANONICAL LTD, 2024) mostrou-se adequado para a configuração e a satisfação dos requisitos técnicos para a máquina virtual. Primeiramente, foi escolhido o software *VirtualBox* (ORACLE, 2024), cujas funcionalidade e reputação são altas na área de máquinas virtuais. Porém, após alguns testes, verificou-se que o *VirtualBox*, ao ser um hipervisor “Tipo 2”, tem limitações de uso de aceleração de hardware e processamento paralelo por placas gráficas (GPU – *Graphics Processing Unit*) (AMAZON WEB SERVICES, 2023; ORACLE, 2024), o que levou a uma lentidão que inviabilizaria a realização deste projeto nesse ambiente virtualizado.

Dessa maneira, abandonou-se o *VirtualBox* e procedeu-se ao uso do hipervisor Tipo 1 *HyperV* (MICROSOFT CORPORATION, 2024b), nativo do Microsoft Windows e com suporte à aceleração de hardware e processamento paralelo por GPUs. O desempenho do *Hyper-V* mostrou-se significativamente melhor e, portanto, suficiente para os propósitos do projeto.

7.2 Tratamento da Base de Dados

Para realizar o desenvolvimento inicial do projeto, foi utilizada a base de dados de produtos da Amazon (PROMPTCLOUD, 2020), que contém informações detalhadas sobre diversos produtos e, portanto, está alinhada aos objetivos do projeto descritos na seção 2.2. O objetivo desse tratamento de dados foi garantir a qualidade e a consistência das informações, removendo-se registros inválidos ou inconsistentes. Além disso, esse tratamento tem como foco a realização de uma análise exploratória dos dados, que permite examinar e compreender a relação existente entre eles e viabilizar a construção do sistema de recomendação após eventual pré-processamento.

Inicialmente, foram seguidos os passos descritos pelo sistema de referência adotado (CHEAH et al., 2020). O primeiro passo consistiu em remover colunas que apresentavam dados com valores nulos ou irrelevantes para a análise, além de linhas com valores nulos em campos considerados essenciais para o projeto. A presença de registros com valores nulos nos campos considerados para o projeto representa uma fonte de inconsistências para o uso dos dados em sua aplicação final e, portanto, os registros com essas características devem ser tratados *a priori*. Essa primeira operação está representada na Figura 12.

```
[29]
    cols = [0,2,3,5,6,8,9,12,14,15,16,17,18,19,20,21,22,23,24,25,26,27]
    dataset.drop(dataset.columns[cols], axis =1, inplace=True)

[30]
    dataset.dropna(inplace = True)
    dataset.info()

...
<class 'pandas.core.frame.DataFrame'>
Index: 7216 entries, 0 to 10001
Data columns (total 6 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Product Name           7216 non-null   object
 1   Category               7216 non-null   object
 2   Selling Price          7216 non-null   object
 3   About Product          7216 non-null   object
 4   Product Specification  7216 non-null   object
 5   Shipping Weight        7216 non-null   object
dtypes: object(6)
memory usage: 394.6+ KB
```

Figura 12 – Remoção das Colunas com Valores Nulos ou Irrelevantes e Remoção de Linhas com Valores Nulos

Apesar de uma quantidade significativa de dados ter sido excluída, cerca de 28% do número total de registros, esse tratamento é utilizado apenas para a análise exploratória inicial, identificando a distribuição dos produtos dentro das categorias existentes. Como, após a exclusão dos registros falhos, mantêm-se representatividade significativa de mais de 7000 produtos e de suas respectivas categorias, considera-se que os registros removidos não prejudicam o projeto em operações posteriores.

A coluna que continha os valores de categorias foi, na sequência, dividida em três colunas distintas, permitindo que um produto seja analisado em três níveis de profundidade de categorias diferentes. Dessa forma, torna-se possível realizar uma análise das categorias existentes e de sua distribuição entre os produtos da base de dados. Esse passo, que foi copiado do sistema de referência, é representado na Figura 13.

```
31]
    new = dataset["Category"].str.split("|", n = 3, expand = True)

    # making the first category called Main Category
    dataset["Main Category"]= new[0]

    # making the second category called sub_category
    dataset["Sub-Category"]= new[1]

    # making the third category called side_category
    dataset["Side Category"]= new[2]

    # making the last column consist of the remaining categories
    dataset["Other Categories"]= new[3]

    # Dropping old category columns and the remaining categories
    dataset.drop(columns =["Category", "Other Categories"], inplace = True)
```

Figura 13 – Redistribuição da Coluna de Categorias

Para os valores de preço e peso, também se seguiu a técnica do sistema de referência de excluir as unidades de medida de preço e peso dos produtos para facilitar a conversão dos dados de valores textuais para valores numéricos. As operações descritas estão representadas na Figura 14.

```
dataset['Shipping Weight(Pounds)'] = dataset['Shipping Weight(Pounds)'].str.strip('ounces')
dataset['Shipping Weight(Pounds)'] = dataset['Shipping Weight(Pounds)'].str.strip('pounds')
dataset['Selling Price($)'] = dataset['Selling Price($)'].str.replace('$', '')

dataset.head()
```

Figura 14 – Remoção das Unidades de Medida das Colunas de Preço e Peso

Antes da exclusão das referidas unidades, no entanto, é importante assegurar que todos os registros da base de dados possuem unidades coerentes e consistentes entre si, de tal forma a evitar interpretações errôneas sobre os dados numéricos. Pelo fragmento de código da Figura 14 e pela inspeção da base de dados antes da remoção das referidas unidades de medida, notou-se que os valores de preço de todos os registros estão, de fato, expressos na mesma unidade (USD – Dólar Americano).

Por outro lado, o mesmo não se aplica para os valores de peso, uma vez que é evidente a utilização de duas unidades distintas (onças e libras). Para contornar esse problema, já como processamento particularizado no presente projeto, foram armazenados os índices dos produtos que apresentavam seu valor de peso em onças

para que as medidas de peso fossem convertidas e uniformizadas para libras. Esse processamento é ilustrado na Figura 15.

```
indexes_ounces = dataset[dataset['Shipping Weight(Pounds)'].str.contains('ounces', na=False)].index
print(dataset.loc[indexes_ounces, 'Shipping Weight(Pounds)'])
```

```
2       12.8 ounces
3       13.4 ounces
4       13.4 ounces
6        9.8 ounces
8       12.2 ounces
           ...
9988     7.7 ounces
9992    0.16 ounces
9993     5.1 ounces
9994     0.8 ounces
9998    0.96 ounces
Name: Shipping Weight(Pounds), Length: 4082, dtype: object
```

Figura 15 – Armazenamento dos Índices dos Produtos com Peso em Onças

```
print(indexes_ounces)
indexes_ounces_updated = []

indexes_ounces_updated = dataset.index.intersection(indexes_ounces)
print(indexes_ounces_updated)

dataset.loc[indexes_ounces_updated, 'Shipping Weight(Pounds)'] = dataset.loc[indexes_ounces_updated, 'Shipping Weight(Pounds)'] / 16
print(dataset.loc[indexes_ounces_updated, 'Shipping Weight(Pounds)'])
```

```
Index([   2,    3,    4,    6,    8,   10,   13,   14,   18,   19,
       ...
       9978, 9979, 9980, 9983, 9986, 9988, 9992, 9993, 9994, 9998],
      dtype='int64', length=4082)
Index([   2,    3,    4,    6,    8,   10,   13,   14,   18,   19,
       ...
       9978, 9979, 9980, 9983, 9986, 9988, 9992, 9993, 9994, 9998],
      dtype='int64', length=4069)
Index([   2,    3,    4,    6,    8,   10,   13,   14,   18,   19,
       ...
       9978, 9979, 9980, 9983, 9986, 9988, 9992, 9993, 9994, 9998],
      dtype='int64', length=4069)
2       0.80000
3       0.83750
4       0.83750
...
9993    0.31875
9994    0.05000
9998    0.06000
Name: Shipping Weight(Pounds), Length: 4069, dtype: float64
```

Figura 16 – Conversão do Valor de Peso em Onças para Libras

Durante a conversão de unidades relatada anteriormente, também foram constatadas outras inconsistências que precisaram ser resolvidas para garantir a correção dos dados. O detalhamento e o tratamento dessas inconsistências, construídos e executados de forma independente, sem seguir o sistema de referência, são descritos na sequência.

Utilizando um notebook Jupyter e operações SQL (*Structured Query Language*), foi realizada uma inspeção dos dados para identificar valores fora do padrão esperado. A primeira inconsistência dessa natureza relaciona-se ao uso de vírgula como separador de milhares nas colunas de preço e peso. Essas vírgulas foram removidas em conformidade com o tratamento indicado na Figura 17 e na Figura 18 para viabilizar o processamento dos dados numéricos como tipos numéricos *(float).*

```
    indexes = dataset[                                    .str.contains(',', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Selling Price($)'])
    dataset['Selling Price($)'] = dataset['Selling Price($)'].str.replace(',', '', regex=False)
```

(variable) indexes: Index

```
Index([ 237,  296,  839,  916, 1292, 1655, 1833, 2091, 2297, 3121, 3122, 4080,
       4451, 6931, 7204, 7325, 7376, 7612, 7947, 8452, 8551, 8638, 8905, 8943,
       9299, 9462, 9580, 9949],
      dtype='int64')
237              1,179.99
296              1,265.00
839              1,899.00
916     895.00 - 2,497.50
1292             1,734.00
1655     32.43 - 9,999.99
1833             1,099.99
2091     42.89 - 9,999.99
2297             2,599.00
3121             1,209.94
3122             1,079.37
```

Figura 17 – Remoção da Vírgula como Separador de Milhares da Coluna de Preço

```
    indexes = dataset[dataset['Shipping Weight(Pounds)'].str.contains(',', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Shipping Weight(Pounds)'])
    dataset['Shipping Weight(Pounds)'] = dataset['Shipping Weight(Pounds)'].str.replace(',', '', regex=False)
```

```
Index([2297], dtype='int64')
2297    1,070
Name: Shipping Weight(Pounds), dtype: object
```

Figura 18 – Remoção da Vírgula como Separador de Milhares da Coluna de Peso

Na sequência, também foram excluídas as linhas contendo valores de preço e peso inválidos, como é indicado da Figura 19 até a Figura 21, bem como as linhas cuja coluna de preço não contém valores exatos, mas sim faixas de preço. Os dados com essa informação são indicados pela presença do caractere “-” no referido campo. A Figura 22 mostra a remoção das linhas que apresentam faixas de preço.

```
    indexes = dataset[dataset['Selling Price($)'] == 'Total price:'].index
    print(indexes)
    print(dataset.loc[indexes, 'Selling Price($)'])
    dataset.drop(indexes, inplace=True)
```

```
Index([ 225,  286, 1033, 1819, 3059, 3284, 3893, 3910, 4035, 4042, 4173, 5148,
       5152, 5409, 5508, 5878, 6236, 6350, 6445, 6638, 6639, 6779, 7370, 7523,
       7722, 7728, 8222, 9024],
      dtype='int64')
225     Total price:
286     Total price:
1033    Total price:
1819    Total price:
3059    Total price:
3284    Total price:
3893    Total price:
3910    Total price:
4035    Total price:
4042    Total price:
4173    Total price:
...
7728    Total price:
8222    Total price:
9024    Total price:
Name: Selling Price($), dtype: object
```

Figura 19 – Remoção das Linhas Contendo o Valor de Preço Inválido "*Total price:*"

```
    indexes = dataset[dataset['Selling Price($)'].str.contains('&', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Selling Price($)'])
    dataset.drop(indexes, inplace=True)
```

```
Index([2572], dtype='int64')
2572    & FREE Shipping. Details
Name: Selling Price($), dtype: object
```

```
    indexes = dataset[dataset['Selling Price($)'].str.contains('Currently', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Selling Price($)'])
    dataset.drop(indexes, inplace=True)
```

```
Index([2639, 5335], dtype='int64')
2639    Currently unavailable.
5335    Currently unavailable.
Name: Selling Price($), dtype: object
```

```
    indexes = dataset[dataset['Selling Price($)'].str.contains('from', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Selling Price($)'])
    dataset.drop(indexes, inplace=True)
```

```
Index([2977, 3248, 4557, 5137, 5736, 6140, 6673, 6674], dtype='int64')
2977    from 2 sellers
3248     from 1 seller
4557    from 4 sellers
5137    from 2 sellers
5736    from 4 sellers
6140     from 1 seller
6673    from 7 sellers
6674    from 8 sellers
Name: Selling Price($), dtype: object
```

Figura 20 – Remoção das Linhas Contendo Valores de Preço Inválidos com Conteúdo “*&*”, “*Currently*” e “*From*”

```
    indexes = dataset[dataset['Shipping Weight(Pounds)'].str.contains(r'\. ', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Shipping Weight(Pounds)'])
    dataset.drop(indexes, inplace=True)
```

```
Index([1619], dtype='int64')
1619    .
Name: Shipping Weight(Pounds), dtype: object
```

Figura 21 – Remoção da Linha Contendo Valor de Peso Inválido “.”

```
indexes = dataset[dataset['Selling Price($)'].str.contains('-', na=False)].index
print(indexes)
print(dataset.loc[indexes, 'Selling Price($)'])
dataset.drop(indexes, inplace=True)
```

```
Index([ 480,  804,  916, 1655, 1821, 1868, 2091, 2200, 2387, 2912, 3283, 3580,
       4254, 5104, 5389, 5781, 5911, 5949, 6199, 6405, 6542, 6589, 6938, 7136,
       7189, 7233, 7321, 7461, 7670, 7770, 8296, 8943, 9062, 9261, 9299, 9319,
       9431, 9462, 9715, 9883],
      dtype='int64')
480        47.19 - 47.99
804         8.25 - 31.95
916     895.00 - 2497.50
1655     32.43 - 9999.99
1821      94.95 - 159.95
1868       6.99 - 155.38
2091     42.89 - 9999.99
2200       53.00 - 71.75
2387       34.03 - 56.41
2912        9.86 - 11.86
...
9462    748.00 - 2024.91
9715       55.41 - 91.49
9883     193.69 - 199.99
Name: Selling Price($), dtype: object
```

Figura 22 – Remoção das Linhas Contendo Faixas de Valores como Preço

Por fim, os valores de preço com formatação errada foram ajustados para garantir a consistência dos dados. O erro de formatação indicado apresentava o valor real do produto, seguido do mesmo valor com espaços em branco, em todos os valores de preço que continham algum espaço em branco. Para contornar esse problema, foi extraído o valor real do produto, utilizando o espaço em branco como separador entre o valor real e a parte duplicada, que foi excluída. Esse ajuste de formatação é mostrado na Figura 23.

```
    indexes = dataset[dataset['Selling Price($)'].str.contains(' ', na=False)].index
    print(indexes)
    print(dataset.loc[indexes, 'Selling Price($)'])
    dataset.loc[indexes, 'Selling Price($)'] = dataset.loc[indexes, 'Selling Price($)'].str.split(' ').str[0]
```

```
Index([  39,  282, 1029, 1303, 1437, 1556, 1644, 1709, 1908, 2498, 3204, 3322,
       3673, 3742, 3766, 4497, 4968, 4973, 5105, 5119, 5882, 6105, 6233, 6416,
       6836, 6926, 7273, 7309, 7374, 7452, 7460, 7494, 7502, 7582, 7583, 7589,
       7605, 7723, 7776, 7810, 7855, 7893, 7909, 7933, 7986, 8149, 8257, 8341,
       8429, 8487, 8557, 8568, 8571, 8595, 8643, 8671, 8680, 8829, 8910, 8925,
       8999, 9027, 9131, 9182, 9195, 9352, 9664, 9757, 9890, 9952],
      dtype='int64')
39        6.94  6 . 94
282     10.44  10 . 44
1029    13.46  13 . 46
1303    15.63  15 . 63
1437      4.23  4 . 23
            ...
9352       54.99 54.99
9664    16.45  16 . 45
9757       21.38 21.38
9890      4.38  4 . 38
9952      5.70  5 . 70
Name: Selling Price($), Length: 70, dtype: object
```

Figura 23 – Ajuste de Valores de Preço Contendo Formatação Errada

Após a limpeza realizada com os passos ilustrados entre a Figura 17 e a Figura 23, específicos para este projeto, foram eliminados 80 registros de produtos da base de dados em comparação com os dados do sistema de referência adotado (CHEAH et al., 2020). Apesar de ser possível aproveitar metade desses registros realizando um tratamento secundário[1], sua quantidade representa menos de 1% do total de dados, o que é considerado de baixa relevância para o projeto.

As informações finais da base de dados, referentes aos dados tratados para o projeto, estão representadas na Figura 24. Os arquivos referentes à base de entrada (“AmazonData.csv”), o processamento realizado e a base de dados processada (“DataCleaningEDA.ipynb”) estão presentes nos diretórios “*data*” e “*initial dataset processing*”, respectivamente, do repositório do GitHub do projeto (ISHICAVA; ITO; INOUE, 2024).

[1] Sobre os dados que contêm faixas de preço, seria possível adotar algum valor dentro do intervalo. Porém, seria necessário avaliar cada um dos produtos individualmente para realizar a adoção de um valor coerente. Tendo em vista o esforço para esse ajuste e a baixa relevância dessa pequena amostra de dados, optou-se pela não realização desse tratamento secundário.

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 7136 entries, 0 to 10001
Data columns (total 8 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Product Name             7136 non-null   object 
 1   Selling Price($)         7136 non-null   float64
 2   About Product            7136 non-null   object 
 3   Product Specification    7136 non-null   object 
 4   Shipping Weight(Pounds)  7136 non-null   float64
 5   Main Category            7136 non-null   object 
 6   Sub-Category             7136 non-null   object 
 7   Side Category            6155 non-null   object 
dtypes: float64(2), object(6)
memory usage: 501.8+ KB
```

Figura 24 – Informações Gerais da Base de Dados após a Realização do Tratamento

Observando as informações gerais da base de dados após o tratamento, é perceptível a ausência de valores referentes à coluna “*Side Category*” para diversos produtos registrados. Essa diferença ocorre porque nem todo registro apresenta mais de duas categorias diferentes, e as duas categorias principais de um registro são representadas pelas colunas “*Main Category*” e “*Sub-Category*”.

O uso do Jupyter Notebook e de SQL foi essencial para a detecção e a correção de inconsistências, melhorando a qualidade da base de dados para o projeto. Este processo foi fundamental para assegurar a precisão e a confiabilidade das análises subsequentes, estabelecendo uma base sólida para as próximas etapas do projeto.

7.3 Análise Detalhada do Sistema de Referência

O sistema de referência escolhido (CHEAH et al., 2020) foi analisado e testado para que se pudesse comparar a utilização de seus métodos de recomendação e as métricas calculadas com o sistema de recomendação desenvolvido neste projeto. O objetivo dessa comparação é indicar forças e fraquezas do sistema de referência, visando a um eventual reaproveitamento das partes bem executadas e à substituição das deficiências por aprimoramentos específicos do projeto relatado nesta monografia.

Durante a análise do sistema de referência, foram percebidos problemas que, no período preliminar de escolha de um sistema de referência, não haviam sido constatados avaliando-se apenas a descrição do sistema de referência. A não detecção desses problemas na etapa de seleção do sistema de referência deve-se, principalmente, ao fato de as fraquezas encontradas serem perceptíveis somente executando-se o código do sistema de referência ou analisando-o minuciosamente – o que não fora realizado antes da fase de desenvolvimento deste projeto.

As principais falhas encontradas foram as seguintes:

a) **Tratamento de dados simples e ineficiente:** A base de dados utilizada (PROMPTCLOUD, 2020) apresenta uma quantidade significativa de dados, e é necessário realizar um processamento objetivo para que ela possa ser analisada. No sistema de referência (CHEAH et al., 2020), esse processamento é não só superficial, dada a necessidade de tratamento adicional descrita na seção 7.2, mas também ineficaz, uma vez que boa parte do tratamento realizado sobre os dados não é, de fato, utilizado para gerar as recomendações.
b) **Cálculo ineficaz de métricas:** O cálculo das métricas é realizado utilizando a técnica TF-IDF (SALTON; MCGILL, 1986). Porém, esse cálculo é feito com uma quantidade reduzida de termos-chave, baseada apenas no nome e na categoria do produto e sem considerar descrições detalhadas já existentes na base de dados de referência (PROMPTCLOUD, 2020) ou outros dados sintático-semânticos de referência (*corpus*). Em decorrência desse processamento simplificado, a relevância dos resultados é negativamente

afetada. Além disso, apesar de funções de distância diferentes serem utilizadas para mensurar a similaridade dos produtos, não há uma análise comparativa delas. Inclusive, devido à dimensionalidade dos dados, algumas das diferentes funções de distância utilizadas resultam em matrizes idênticas, tornando o processo de uso de múltiplas funções inócuo.

c) **Inutilização de partes do código:** Assim como a não utilização de certas etapas do tratamento de dados mencionado no item “a)”, outros segmentos do código são executados e não são utilizados depois. Entre elas, situam-se parte das matrizes do item “b)”, que são apenas calculadas e construídas.
d) **Utilização de método sem relação com o restante do código:** A não utilização das matrizes mencionadas anteriormente é consequência do uso de uma biblioteca adicional, denominada *fuzzywuzzy* (PYTHON SOFTWARE FOUNDATION, 2020), que realiza parte do processo de metrificação mediante uso de uma quarta função de distância especificamente para *strings*. Além de o método utilizado gerar resultados questionáveis, identificados por meio de ensaios particulares, essa forma de recomendação não foi previamente introduzida nem explicada na documentação do sistema de referência (CHEAH et al., 2020).

Dessa forma, é possível perceber que o sistema de referência (CHEAH et al., 2020) possui falhas não triviais de projeto. Por conseguinte, elas servem de base para serem tratadas a fim de que o sistema de recomendação desenvolvido neste projeto possa aprimorar as recomendações providas e, assim, atender aos requisitos especificados no capítulo 4.

7.4 Desenvolvimento e Testes de Técnicas para Resolver Problemas do Sistema de Referência e Projetar o Sistema de Recomendação

Visto que a análise realizada na seção 7.3 levou à identificação de deficiências no sistema de referência (CHEAH et al., 2020), foi realizada uma pesquisa para definir soluções e melhorias para tais problemas. A partir desse levantamento, projetou-se um esboço do tratamento necessário para incrementar a qualidade das recomendações fornecidas a partir da base de dados de referência (PROMPTCLOUD, 2020) com os devidos ajustes já explorados na seção 7.2.

Os passos seguidos no desenvolvimento, até o presente momento, foram os seguintes:

a) Inicialmente, foram realizados testes preliminares de viabilidade de uso de bibliotecas de semelhança semântica de palavras para gerar recomendações;
b) Na sequência, implantou-se um primeiro nível de filtragem de recomendação baseado nas informações de categorias dos produtos;
c) Realizou-se a implementação de um segundo nível de filtragem que emprega as bibliotecas de semelhança semântica exercitadas inicialmente para aprimorar recomendações baseadas nos dados dos nomes dos produtos;
d) Explorou-se o funcionamento de bibliotecas de semelhança morfossintática para atuar, em conjunto do filtro de semelhança semântica, na geração de recomendações;
e) Por fim, desenvolveu-se um terceiro nível de filtragem que utiliza bibliotecas de análise morfossintática para atuar sobre os dados dos nomes dos produtos.

Cada um dos passos prévios é documentado em uma das próximas subseções da monografia.

7.4.1 **Testes com Bibliotecas de Semelhança Semântica de Palavras**

Para uma base de dados como a utilizada, o tratamento por TF-IDF (SALTON; MCGILL, 1986) é útil, mas mais bem aproveitado em situações com mais texto, como a descrição dos produtos e não suas categorias. Além disso, considerou-se adequado empregar métodos com uma abordagem mais voltada para a codificação das categorias e descrições dos produtos, tal como os presentes nas bibliotecas *Word2Vec* (MIKOLOV et al., 2013) e *GloVe* (PENNINGTON; SOCHER; MANNING, 2014). A biblioteca *Word2Vec*, por exemplo, é parte integrante de uma biblioteca maior, denominada *gensim* (ŘEHŮŘEK, 2024), e é disponibilizada com modelos previamente já treinados com textos (*corpus*) de bilhões de palavras (MIKOLOV et al., 2013). Essa biblioteca codifica em vetores as palavras presentes nos objetos tratados, de maneira a aproximar palavras semanticamente semelhantes – o que permite capturar padrões que são naturais para seres humanos, mas não necessariamente triviais para máquinas.

Devido a essa característica de treinamento prévio, a biblioteca *Word2Vec* foi escolhida para uso no sistema de recomendação em detrimento da biblioteca *GloVe*. Por intermédio dele, a biblioteca *Word2Vec* permite capturar relações semânticas entre os termos-chave dentro do contexto em que estão inseridos (MIKOLOV et al., 2013). Ademais, possui flexibilidade para atualizações dinâmicas e melhor desempenho em domínios específicos de aplicação, como as descrições de produtos do sistema de recomendação pertencente ao escopo deste trabalho (MIKOLOV et al., 2013).

A fim de testar os métodos mencionados, um tratamento de dados específico precisou ser realizado. Nessa abordagem, as palavras presentes na coluna "*Category*" da base foram vetorizadas e reduzidas a apenas palavras-chave, em letras minúsculas e sem pontuação. Esse tratamento está representado na Figura 25.

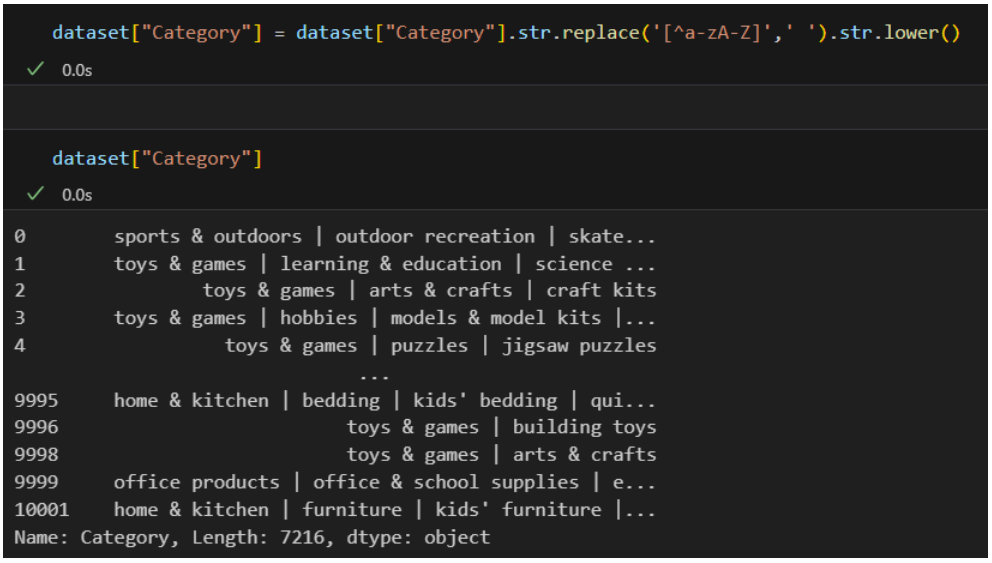


Figura 25 – Vetorização da Coluna "*Category*" em Palavras-Chave Minúsculas e sem Pontuação

Após a vetorização, uma outra biblioteca, denominada *nltk* (THE NLTK TEAM, 2024), foi utilizada para remoção de *stop words*, conforme ilustrado na Figura 26. Esse processo elimina palavras comuns da língua inglesa, como "*to*", "*the*", "*a*", "*an*", "*and*", reduzindo o ruído no texto e melhorando a eficiência computacional. Ademais, também aumenta o foco do resto da solução em palavras mais informativas da aplicação fim, potencialmente resultando em modelos mais adequados e análises textuais mais claras.

```
stop_re = '\\b'+'\\b|\\b'.join(nltk.corpus.stopwords.words('english'))+'\\b'
dataset["Category"] = dataset["Category"].str.replace(stop_re, '')
```
✓ 0.0s

```
dataset["Category"]
```
✓ 0.0s

```
0        sports & outdoors | outdoor recreation | skate...
1        toys & games | learning & education | science ...
2                toys & games | arts & crafts | craft kits
3        toys & games | hobbies | models & model kits |...
4                  toys & games | puzzles | jigsaw puzzles
                                ...
9995     home & kitchen | bedding | kids' bedding | qui...
9996                           toys & games | building toys
9998                           toys & games | arts & crafts
9999     office products | office & school supplies | e...
10001    home & kitchen | furniture | kids' furniture |...
Name: Category, Length: 7216, dtype: object
```

Figura 26 – Remoção de *Stop Words* da coluna "*Category*"

Com as *stop words* removidas, o próximo passo consiste em processar o texto e dividi-lo em vetores de palavras, convertendo cada *string* em uma lista de palavras, sendo cada elemento dessa lista uma palavra individual. Esse processamento é ilustrado na Figura 27.

```
dataset["Category"] = dataset["Category"].str.split()
dataset["Category"]
```
✓ 0.0s

```
0        [sports, &, outdoors, |, outdoor, recreation, ...
1        [toys, &, games, |, learning, &, education, |,...
2        [toys, &, games, |, arts, &, crafts, |, craft,...
3        [toys, &, games, |, hobbies, |, models, &, mod...
4         [toys, &, games, |, puzzles, |, jigsaw, puzzles]
                                ...
9995     [home, &, kitchen, |, bedding, |, kids', beddi...
9996                    [toys, &, games, |, building, toys]
9998                   [toys, &, games, |, arts, &, crafts]
9999     [office, products, |, office, &, school, suppl...
10001    [home, &, kitchen, |, furniture, |, kids', fur...
Name: Category, Length: 7216, dtype: object
```

Figura 27 – Separação dos Valores da Coluna em Vetores

Essa transformação prepara os dados para a biblioteca *Word2Vec*, que converte palavras em vetores numéricos para capturar as relações semânticas entre

os termos-chave dentro do contexto em que estão inseridos. A Figura 28 ilustra esse processo, mostrando o uso da biblioteca *Word2Vec* na coluna "*Category*" dividida em vetores e inserida no modelo. Utilizando a palavra "*art*" como parâmetro de teste, foram obtidas recomendações de palavras como "*paints*", "*watercolor*", "*painting*" e "*cutting*", que são elementos com maior proximidade semântica e contextual em relação à palavra "*art*".

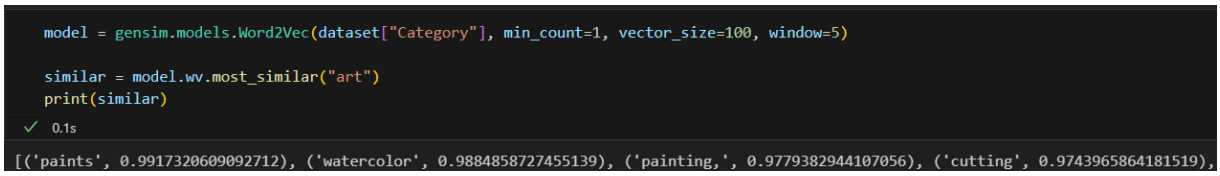


Figura 28 – Exemplo de Utilização da Biblioteca *Word2Vec*

7.4.2 Filtragem das Categorias

Outra abordagem realizada para otimizar as recomendações do sistema foi a filtragem prévia da base de dados para limitar a recomendação dentro da mesma categoria do produto dado como entrada. Dessa forma, mesmo sem nenhum tratamento por aprendizado de máquina, a base de dados passa por um pré-processamento que a ordena de maneira mais relevante.

Dado que a o tratamento realizado anteriormente e descrito na seção 7.2 já separa a coluna “*Category*” nas colunas “*Main Category*”, “*Sub Category*”, “*Side Category*” e “*Other Category*”, um processamento com base nessas quatro colunas foi realizado. Para que produtos com categorias semelhantes fossem ordenados primeiro, um sistema de pontuação foi elaborado para que produtos que compartilhem as mesmas categorias da entrada recebam mais pontos, e produtos com categorias diferentes não recebam pontos.

Para isso, uma coluna de suporte chamada “*score*” foi criada, e nela se adicionou a pontuação conforme a coincidência de cada um dos níveis de granularidade categórica do produto dado como entrada. Esse processo, ilustrado na Figura 29, indica que cada nível de categoria coincidente acrescenta “1” ao valor de “*score*”.

```python
# Calcula uma pontuação para cada produto com base na similaridade das categorias
dataset['score'] = 0

# Aumenta a pontuação se a "Main Category" ou "Sub Category" corresponder
dataset.loc[dataset['Main Category'] == main_category_input, 'score'] += 1
dataset.loc[dataset['Sub Category'] == sub_category_input, 'score'] += 1
dataset.loc[dataset['Side Category'] == side_category_input, 'score'] += 1
dataset.loc[dataset['Other Category'] == other_category_input, 'score'] += 1
```

Figura 29 – Código de Criação da Coluna "*score*" e Atribuição de Pontos conforme as Categorias do Produto Dado como Entrada

Após tal atribuição de pontos, a base de dados é reordenada em ordem decrescente de pontuação em "*score*" e apenas a pontuação máxima é mantida, garantindo uma similaridade maior dos produtos filtrados. Na sequência, a coluna de suporte "*score*" é removida e o resultado é impresso, conforme estabelecido no trecho de código da Figura 30. O resultado impresso corresponde à lista de produtos recomendados em ordem decrescente de relevância.

```python
# Ordena os produtos com base na pontuação, do mais alto para o mais baixo
produtos_recomendados = dataset.sort_values(by='score', ascending=False)

# Remove produtos com pontuação 0 (sem correspondência)
produtos_recomendados = produtos_recomendados[produtos_recomendados['score'] > 0]

# Remove a coluna de pontuação antes de retornar
produtos_recomendados = produtos_recomendados.drop(columns='score')

# Retorna as recomendações
if produtos_recomendados.empty:
    print('Nenhuma recomendação encontrada para essas categorias.')

print(produtos_recomendados)
```

Figura 30 – Código de Ordenação e Impressão da Base de Dados Filtrada pela Semelhança das Categorias

Quando o produto de entrada não possuir informações categóricas, ou quando não há outros produtos que compartilhem nenhuma das categorias do dado de entrada, o esquema de filtragem por categorias não gera nenhuma recomendação. Esses casos excepcionais são tratados por outras etapas do processo de recomendação, exploradas da seção 7.4.3 até a seção 7.5.

7.4.3 **Recomendações de Nomes dos Produtos com a Biblioteca de Filtragem Semântica**

Os testes da biblioteca *Word2Vec*, realizados no item 7.4.1, foram utilizados como referência para gerar recomendações a partir de similaridades semânticas das palavras que compõem o nome dos produtos. Para isso, é necessário que os nomes sejam vetorizados no modelo da biblioteca, de tal forma que a comparação de similaridade entre as palavras seja realizada de maneira correta.

Inicialmente, duas funções são definidas. A função “*preprocess_text*” realiza um tratamento prévio do texto, no qual as *strings* dos nomes são processadas e padronizadas com as seguintes características: (i.) texto apenas com letras minúsculas e (ii.) sem caracteres especiais e palavras conectivas (*stop words*), que são removidas nessa fase.

Além disso, a função “*preprocess_text*” também contempla o tratamento de número (singular-plural) e tempo verbal das palavras, considerado crucial no processo de normalização do texto para facilita a análise de similaridade entre termos e reduzir a duplicidade de informações. Essa tarefa, realizada convertendo-se palavras no plural em sua forma singular, baseia-se em técnicas como *Stemming* e *Lemmatization* (JURAFSKY; MARTIN, 2024). O *Stemming* consiste em reduzir as palavras ao seu radical básico sem considerar o contexto linguístico, o que pode gerar resultados imprecisos, pois palavras diferentes podem ser reduzidas a uma mesma forma simplificada. Por exemplo, as palavras "*playing*" e "*played*" podem ser reduzidas a "*play*", mas o processo ignora diferenças gramaticais e de significado. Por outro lado, a *Lemmatization* analisa o contexto e devolve a forma base ou canônica da palavra (*lemma*), levando em conta aspectos gramaticais como o tempo verbal e o número (singular/plural). Por essa razão*, Lemmatization* foi escolhida para esse tratamento, visto que mantém melhor a integridade semântica e o sentido do texto.

A função “*vectorize_product*”, por sua vez, vetoriza as *strings* processadas no modelo *Word2Vec*, criando uma matriz de valores para comparação.

A Figura 31 mostra o código das duas funções previamente descritas.

```python
# Processing text function
def preprocess_text(text):
    text = text.replace('[^a-zA-Z]',' ').lower()
    stop_re = '\\b'+'\\b|\\b'.join(nltk.corpus.stopwords.words('english'))+'\\b'
    text = text.replace(stop_re, '')
    text = text.split()

    # Add lemmatization using WordNetLemmatizer
    lemmatizer = WordNetLemmatizer()
    lemmatized_text = [lemmatizer.lemmatize(word) for word in text]
    return lemmatized_text
```

```python
# vectorizing text function
def vectorize_product(product_name, model):

    words = [word for word in product_name if word in model.wv]
    if len(words) > 0:
        return np.mean([model.wv[word] for word in words], axis=0)
    else:
        return np.zeros(model.wv.vector_size)
```

Figura 31 – Funções de Pré-Processamento e de Vetorização de Texto

Na sequência, um modelo *Word2Vec* é treinado com base nos nomes dos produtos, aprendendo as representações desses nomes na base de dados e considerando que cada nome de produto já foi transformado em uma lista de palavras. Nesse treinamento, o modelo usa as palavras dos nomes dos produtos como entrada, processa cada lista de palavras (produto) e aprende representações vetoriais para cada palavra individualmente com base nas palavras que a cercam (i.e., o contexto). Este processo é ilustrado na Figura 32 – mais precisamente com a atribuição do valor "*dataset["Processed Product Name"]*" ao parâmetro "*sentences*" do modelo *Word2Vec.* Estes argumentos podem ser modificados da seguinte maneira para alterar o tipo de modelo criado:

- *sentences*: define as frases a serem usadas para treinar o modelo;
- *vector_size:* define a dimensão dos vetores criados para cada palavra. No caso, o valor 100 define vetores com 100 dimensões para o treinamento do modelo;

- *window:* especifica o tamanho da janela de contexto ao redor de uma palavra para o aprendizado. No caso do valor 5, o modelo considera cinco palavras antes e cinco palavras depois para aprender as relações de contexto;
- *min_count:* define a frequência mínima que uma palavra deve ter para ser incluída no treinamento. No caso do valor 1, qualquer palavra que apareça será incluída no modelo;
- *workers*: especifica o número de processos usados para treinar o modelo.

```
modelo = Word2Vec(sentences=dataset["Processed Product Name"], vector_size=100, window=5, min_count=1, workers=4)
```

Figura 32 – Criação do Modelo *Word2Vec* das Palavras nos Nomes dos Produtos

Com isso, colunas de suporte para o processamento da base de dados são criadas. Essas colunas, nomeadas "*Processed Product Name*" e "*Product Vector*", alocam, respectivamente, as *strings* processadas a partir dos nomes dos produtos e a vetorização dessas *strings*. A Figura 33 mostra a criação das colunas "*Processed Product Name*" e "*Product Vector*" e a utilização das funções "preprocess_text" e "*vectorize_product*", descritas anteriormente, para esse fim.

```
# Applying text preprocess in dataset
dataset["Processed Product Name"] = dataset["Product Name"].apply(preprocess_text)

model = train_word2vec_model(dataset)

# Applying vectorizing function in dataset
dataset["Product Vector"] = dataset["Processed Product Name"].apply(lambda x: vectorize_product(x, model))

# Pré-processando o nome do produto fornecido pelo usuário
processed_product_name = preprocess_text(product_name)
```

Figura 33 – Criação das Colunas de Suporte e Aplicação das Funções

Por fim, foi criada a função "*product_recommendation*", que realiza a recomendação a partir do nome do produto. Ela recebe a entrada "*input_text*", que corresponde a uma *string* com o nome de um produto base para recomendações, e a entrada "*top_n*", que indica a quantidade de produtos a serem recomendados (valor padrão igual a 5).

A função “*product_recommendation*” utiliza todos os métodos descritos anteriormente para a criação do modelo do *Word2Vec* e da vetorização dos nomes dos produtos, para finalmente compará-los com a *string* de entrada (que também é processada e vetorizada) com base no índice de similaridade do cosseno. Essa comparação cria uma matriz de valores em que os valores mais altos e próximos de um representam uma similaridade maior. Após a ordenação dessa matriz em ordem decrescente de similaridade, são obtidos os produtos com maior semelhança semântica de seus nomes. Os produtos que ocupam as primeiras “*top_n*” posições são, por fim, impressos em uma lista que consiste nas recomendações geradas.

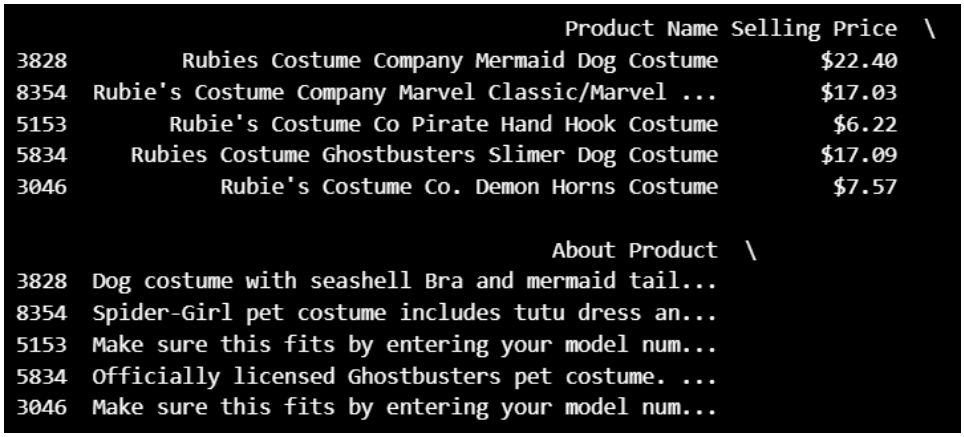
A Figura 34 ilustra a função de recomendação descrita anteriormente.

```
# Product Recommendation function
def product_recommendation(product_vector, dataset, top_n=5):

    # Calcular similaridades cosseno
    similarities = dataset["Product Vector"].apply(lambda x: cosine_similarity([product_vector], [x])[0][0])

    # Ordenar por similaridade e pegar os top_n produtos
    top_indices = similarities.nlargest(top_n).index

    # Retornar o DataFrame com os produtos recomendados, mas mantendo os nomes originais
    return dataset.loc[top_indices, dataset.columns != 'Product Vector']
```

Figura 34 – Função de Recomendação com base na Semelhança dos Nomes dos Produtos

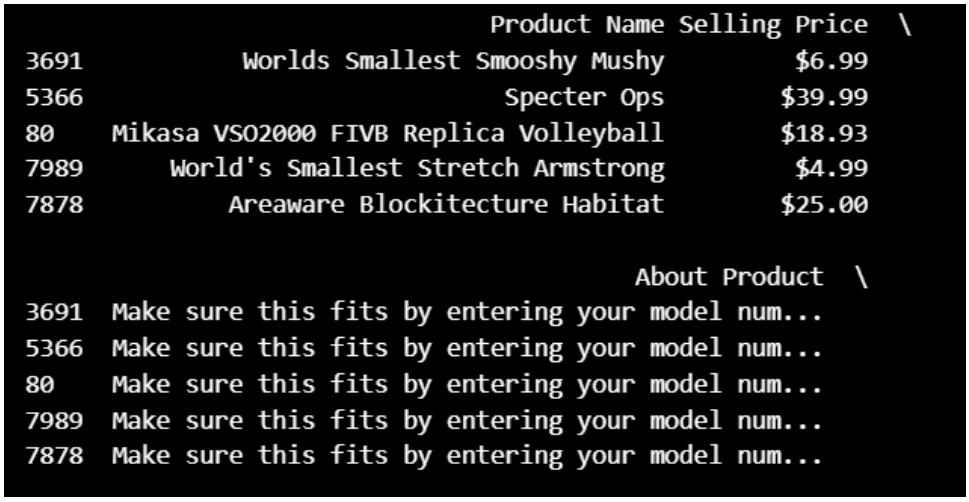
É possível notar que, caso o nome do produto não apresente palavras próximas ao produto desejado (mesmo sendo da mesma categoria), a recomendação pode não ser tão precisa, como ilustrada nos testes realizados na Figura 35 e na Figura 36.

No exemplo da Figura 35, voltada à recomendação de itens relacionados a fantasias de super-heróis, é possível notar recomendações plausíveis, com a presença das palavras dadas como entrada nos nomes dos produtos sugeridos. Já na Figura 36, apesar de sabidamente existirem fones-de-ouvido na base de dados de referência, sob o rótulo “*wireless headphones*”, as recomendações para “*wireless headsets*” não se relacionam ao assunto.



	Product Name	Selling Price	About Product
3828	Rubies Costume Company Mermaid Dog Costume	$22.40	Dog costume with seashell Bra and mermaid tail...
8354	Rubie's Costume Company Marvel Classic/Marvel ...	$17.03	Spider-Girl pet costume includes tutu dress an...
5153	Rubie's Costume Co Pirate Hand Hook Costume	$6.22	Make sure this fits by entering your model num...
5834	Rubies Costume Ghostbusters Slimer Dog Costume	$17.09	Officially licensed Ghostbusters pet costume. ...
3046	Rubie's Costume Co. Demon Horns Costume	$7.57	Make sure this fits by entering your model num...

Figura 35 – Resultado da Recomendação Utilizando "*hero costume*" como Entrada



	Product Name	Selling Price	About Product
3691	Worlds Smallest Smooshy Mushy	$6.99	Make sure this fits by entering your model num...
5366	Specter Ops	$39.99	Make sure this fits by entering your model num...
80	Mikasa VSO2000 FIVB Replica Volleyball	$18.93	Make sure this fits by entering your model num...
7989	World's Smallest Stretch Armstrong	$4.99	Make sure this fits by entering your model num...
7878	Areaware Blockitecture Habitat	$25.00	Make sure this fits by entering your model num...

Figura 36 – Resultado da Recomendação Utilizando "*wireless headset*" como Entrada

7.4.4 **Testes com Bibliotecas de Semelhança Morfossintática de Palavras**

A fim de complementar o filtro semântico desenvolvido e melhorar a qualidade das recomendações geradas pelo sistema, optou-se por analisar a contribuição de ferramentas voltadas à comparação morfossintática de palavras para esse fim. A biblioteca *RapidFuzz* (PYTHON SOFTWARE FOUNDATION, 2024a), sucessora da biblioteca *fuzzywuzzy* (PYTHON SOFTWARE FOUNDATION, 2020), apresenta funcionalidades voltadas para a medição de similaridade entre *strings*, além de apresentar melhorias de desempenho em comparação com sua versão antecessora.

Entre os recursos disponíveis na biblioteca, existem oito métodos de pontuação diferentes para a realização do cálculo de similaridade de *strings*. As descrições de

cada um dos métodos de pontuação e as métricas em que se baseiam estão indicadas na Tabela 16.

Tabela 16 – Descrição dos Métodos Apresentados no Filtro Morfossintático (PYTHON SOFTWARE FOUNDATION, 2024a)

Método de Pontuação	**Métrica Principal**	**Descrição**
Ratio	Levenshtein	Mede a similaridade usando a distância Levenshtein direta, sensível à ordem e à totalidade dos caracteres.
Partial_ratio	Levenshtein (parcial)	Calcula a distância Levenshtein na maior subsequência correspondente entre as strings comparadas.
Token_sort_ratio	Levenshtein e Tokenização[2]	Separa e ordena os *tokens* (palavras), em seguida, aplica a distância Levenshtein.
Partial_token_sort_ratio	Levenshtein (parcial) e Tokenização	Aplica Levenshtein parcial após separar e ordenar *tokens*, permitindo maior flexibilidade de ordem e comprimento.
Token_set_ratio	Levenshtein e Conjunto de Tokens	Remove duplicatas, criando um conjunto (*set*) de *tokens*, e calcula Levenshtein entre os conjuntos resultantes.
Partial_token_set_ratio	Levenshtein (parcial) e Conjunto de Tokens	Permite correspondência parcial com conjuntos de *tokens*, aplicando Levenshtein parcial sobre conjuntos.
Q_ratio	Combinação de métricas	Combina *ratio* com *token_sort_ratio*.
W_ratio	Combinação de métricas	Combina *ratio*, *partial_ratio*, *token_sort_ratio* e *partial_token_set_ratio*.

Os testes da biblioteca *RapidFuzz* como mecanismo de recomendação por filtragem morfossintática foram realizados tendo como foco principal a avaliação dos

[2] O conceito de Tokenização refere-se à divisão de uma *string* em palavras.

métodos de pontuação e uma análise iterativa que sustente sua adoção ou seu descarte do sistema. Para tanto, cada iteração de teste baseia-se em exercitar cenários de teste, formados por combinações de termos pertencentes aos nomes de determinados produtos da base de dados de referência (PROMPTCLOUD, 2020), e avaliar os resultados de cada um dos métodos de pontuação da Tabela 16.

Para o primeiro teste, utilizou-se como base um produto denominado "*Space Base*". Com o intuito de verificar a viabilidade do resultado da recomendação, dois cenários de teste foram exercitados com variações distintas do nome do produto: uma com o nome original e outra com as palavras do nome do produto em ordem invertida (i.e., "*Base Space*"). Para cada uma dessas variações, geraram-se duas recomendações utilizando todos os métodos de pontuação.

A codificação do cenário de teste "*Space Base*" é representada na Figura 37.

```
products_list = dataset["Product Name"]
input = "Space Base"
limit = 2
ratio = process.extract(input, products_list, scorer=fuzz.ratio, limit=limit, processor=utils.default_process)
p_ratio = process.extract(input, products_list, scorer=fuzz.partial_ratio, limit=limit, processor=utils.default_process)
tsort_ratio = process.extract(input, products_list, scorer=fuzz.token_sort_ratio, limit=limit, processor=utils.default_process)
ptsort_ratio = process.extract(input, products_list, scorer=fuzz.partial_token_sort_ratio, limit=limit, processor=utils.default_process)
tset_ratio = process.extract(input, products_list, scorer=fuzz.token_set_ratio, limit=limit, processor=utils.default_process)
ptset_ratio = process.extract(input, products_list, scorer=fuzz.partial_token_set_ratio, limit=limit, processor=utils.default_process)
W_ratio = process.extract(input, products_list, scorer=fuzz.WRatio, limit=limit, processor=utils.default_process)
Q_ratio = process.extract(input, products_list, scorer=fuzz.QRatio, limit=limit, processor=utils.default_process)

Scorer_list = {
    'ratio':ratio,
    'partial ratio':p_ratio,
    'token sort ratio':tsort_ratio,
    'partial token sort ratio':ptsort_ratio,
    'token set ratio':tset_ratio,
    'partial token set ratio':ptset_ratio,
    'Wratio':W_ratio,
    'Qratio':Q_ratio,
}

pd.set_option('display.width', 1000)
pd.set_option('display.max_colwidth', 60)
pd.set_option('display.float_format', '{:.2f}'.format)

df = pd.DataFrame(columns=['Scorer', 'Product Name', 'Value'])
for key,value in Scorer_list.items():
    for i in range(limit):
        new_row = pd.DataFrame([[key, value[i][0], value[i][1]]], columns=['Scorer', 'Product Name', 'Value'])
        df = pd.concat([df, new_row], ignore_index=True)

print(df)
```

Figura 37 – Testes dos Métodos da Biblioteca *RapidFuzz* Utilizando "*Space Base*" como Entrada

Os resultados do cenário de teste "*Space Base*" estão indicados na Figura 38. Por intermédio deles, foi possível averiguar que o método de pontuação *partial token set ratio* é inviável, uma vez que mesmo a primeira recomendação não é o produto original. Todos os outros métodos apresentaram resultados condizentes com o produto esperado; contudo, avaliou-se que os pontuadores *ratio, token sort ratio* e *Q*

ratio não apresentaram um resultado otimizado, uma vez que a segunda recomendação dos três apresentou valores baixos em comparação com os outros métodos.

	Scorer	Product Name	Value
0	ratio	Space Base	100.00
1	ratio	Specter Ops	57.14
2	partial ratio	Space Base: The Emergence of Shy Pluto	100.00
3	partial ratio	Space Base	100.00
4	token sort ratio	Space Base	100.00
5	token sort ratio	Clank!, Base	60.00
6	partial token sort ratio	Space Base	100.00
7	partial token sort ratio	Lego Space & Airport Set	84.21
8	token set ratio	Space Base: The Emergence of Shy Pluto	100.00
9	token set ratio	Space Base	100.00
10	partial token set ratio	Mrs. Grossman's Outer Space Reusable Sticker Activity Se...	100.00
11	partial token set ratio	Osmo - Super Studio Disney Mickey Mouse & Friends Game -...	100.00
12	Wratio	Space Base	100.00
13	Wratio	Space Base: The Emergence of Shy Pluto	90.00
14	Qratio	Space Base	100.00
15	Qratio	Specter Ops	57.14

Figura 38 – Resultado da Recomendação Morfossintática Utilizando "*Space Base*" como Entrada

A codificação do segundo cenário do primeiro teste, com o nome do produto invertido (i.e., "*Base Space*"), é representada na Figura 39.

```
products_list = dataset["Product Name"]
input = "Base Space"
limit = 2
ratio = process.extract(input, products_list, scorer=fuzz.ratio, limit=limit, processor=utils.default_process)
p_ratio = process.extract(input, products_list, scorer=fuzz.partial_ratio, limit=limit, processor=utils.default_process)
tsort_ratio = process.extract(input, products_list, scorer=fuzz.token_sort_ratio, limit=limit, processor=utils.default_process)
ptsort_ratio = process.extract(input, products_list, scorer=fuzz.partial_token_sort_ratio, limit=limit, processor=utils.default_process)
tset_ratio = process.extract(input, products_list, scorer=fuzz.token_set_ratio, limit=limit, processor=utils.default_process)
ptset_ratio = process.extract(input, products_list, scorer=fuzz.partial_token_set_ratio, limit=limit, processor=utils.default_process)
W_ratio = process.extract(input, products_list, scorer=fuzz.WRatio, limit=limit, processor=utils.default_process)
Q_ratio = process.extract(input, products_list, scorer=fuzz.QRatio, limit=limit, processor=utils.default_process)

Scorer_list = {
    'ratio':ratio,
    'partial ratio':p_ratio,
    'token sort ratio':tsort_ratio,
    'partial token sort ratio':ptsort_ratio,
    'token set ratio':tset_ratio,
    'partial token set ratio':ptset_ratio,
    'Wratio':W_ratio,
    'Qratio':Q_ratio,
}

pd.set_option('display.width', 1000)
pd.set_option('display.max_colwidth', 60)
pd.set_option('display.float_format', '{:.2f}'.format)

df = pd.DataFrame(columns=['Scorer', 'Product Name', 'Value'])
for key,value in Scorer_list.items():
    for i in range(limit):
        new_row = pd.DataFrame([[key, value[i][0], value[i][1]]], columns=['Scorer', 'Product Name', 'Value'])
        df = pd.concat([df, new_row], ignore_index=True)

print(df)
```

Figura 39 – Testes dos Métodos da Biblioteca *RapidFuzz* Utilizando "*Base Space*" como Entrada

Por intermédio dos resultados correlatos, presentes na Figura 40, considerou-se que as análises realizadas na etapa anterior se mantêm a menos do método de

quantificação *partial ratio*, que levou a uma pontuação menor em comparação com o primeiro cenário e, consequentemente, a uma piora na qualidade das recomendações. Contudo, dado que esse método considera principalmente a ocorrência de subsequências de *strings*, e seus resultados, ainda que menores do que no primeiro cenário, foram suficientemente altos, com aproximadamente 80% de similaridade, seu uso permaneceu viável para inspeções em outros testes.

	Scorer	Product Name	Value
0	ratio	Space Base	50.00
1	ratio	Barbie Car Wash Playset	48.48
2	partial ratio	RoyalBaby Space Shuttle Kids Bike for Boys and Girls, 14...	80.00
3	partial ratio	Chic Home Spaceship 5 Piece Comforter Set Space Explorer...	80.00
4	token sort ratio	Space Base	100.00
5	token sort ratio	Clank!, Base	60.00
6	partial token sort ratio	Space Base	100.00
7	partial token sort ratio	Lego Space & Airport Set	84.21
8	token set ratio	Space Base: The Emergence of Shy Pluto	100.00
9	token set ratio	Space Base	100.00
10	partial token set ratio	Mrs. Grossman's Outer Space Reusable Sticker Activity Se...	100.00
11	partial token set ratio	Osmo - Super Studio Disney Mickey Mouse & Friends Game -...	100.00
12	Wratio	Space Base	95.00
13	Wratio	Mrs. Grossman's Outer Space Reusable Sticker Activity Se...	85.50
14	Qratio	Space Base	50.00
15	Qratio	Barbie Car Wash Playset	48.48

Figura 40 – Resultado da Recomendação Morfossintática Utilizando "*Base Space*" como Entrada

Outro aspecto a ser salientado é o de que os métodos *ratio* e *Q ratio*, embora tenham identificado corretamente o produto esperado na primeira recomendação, fizeram-no com uma taxa de sucesso de apenas 50%. Esse resultado decorre da inversão das palavras do nome do produto, tal como previsto no cenário de teste. Ainda assim, apesar do resultado quantitativo desfavorável, ambas as métricas são mantidas para a avaliação posterior devido ao acerto do produto a ser recomendado.

Logo, por intermédio dos dois cenários do primeiro teste, foi possível identificar a inviabilidade do método de pontuação *partial token set ratio* e que todos os demais métodos da Tabela 16, mesmo com as variações observadas, ainda se mostram promissores para serem utilizados no sistema de recomendação. Dessa forma, para refinar os métodos de pontuação potencialmente úteis, foi desenvolvida uma segunda série de testes, composta por quatro cenários de teste distintos.

Cada cenário considera uma quantidade de palavras e erros de digitação presentes na *string* de entrada. Esse teste foi escolhido para verificar a robustez dos

métodos de pontuação e sua possibilidade de uso apesar da presença de erros de ortografia. Para esses cenários, utilizou-se como base um produto denominado "*Junior Learning Fantail Books Turquoise Non Fiction Educational Action Games*", em que cada palavra considerada para a *string* de entrada está presente no nome do produto. Os cenários considerados estão indicados na Tabela 17.

Tabela 17 – Cenários Considerados para a Segunda Série de Testes

Valores de Entrada	Número de Palavras	Número de Erros de Digitação
"*Learning*", "*Turquoise*", "*Fiction*"	1	0
"*Learninc*", "*Turqooise*", "*Fection*"	1	1
"*Learning Fantail Books*"	3	0
"*Learninc Fentail Bools*"	3	3

Após a realização da segunda série de testes, cujos resultados são apresentados no Apêndice A devido à sua longa extensão, foi possível descartar os métodos de pontuação *ratio*, *token sort ratio* e *Q ratio.* Eles foram avaliados como inapropriados para o sistema de recomendação porque apresentaram resultados com pontuações e qualidades baixas para todos os cenários de teste avaliados.

Cabe salientar, também, que o método *token set ratio*, apesar de gerar recomendações de baixa qualidade nos cenários com erros de digitação, apresentou ótimos resultados nos testes sem esses erros. Assim, ele ainda pode ser útil em casos específicos nos quais a entrada de dados seja isenta de inconsistências.

Uma terceira série de testes foi realizada com o intuito de refinar a escolha dos métodos ainda considerados como viáveis até tal ponto – isto é, *partial token sort ratio*, *W ratio, partial ratio e token set ratio*. Para essa série de testes, foi adotado outro conjunto de cenários envolvendo múltiplos valores de entrada. Cada cenário considera uma combinação de quantidades de *strings* de entrada e quantidades de palavras por *string*. Esse teste foi escolhido para verificar o desempenho dos métodos de pontuação viáveis, avaliar sua adequação para múltiplos valores simultâneos de

entrada e selecionar o método que apresentasse um melhor resultado geral para ser efetivamente adotado no sistema de recomendação.

Para o terceiro conjunto de testes, utilizou-se como base um produto denominado "*Nanoblock Birthday Cake Building Kit*", em que cada palavra considerada para as *strings* de entrada está presente no nome do produto. Os cenários considerados estão indicados na Tabela 18.

Tabela 18 – Cenários Considerados para a Terceira Série de Testes

Valores de Entrada	Número de Palavras	Número de *Strings*
"*Nanoblock*"	1	1
"*Nanoblock*", "*Birthday*", "*Building*"	1	3
"*Nanoblock Building Kit*"	3	1
"*Nanoblock Birthday Cake*", "*Birthday Cake Kit*", "*Nanoblock Building Kit*"	3	3

Após a realização da terceira série de testes, foi possível descartar mais dois métodos de pontuação: *partial token sort ratio* e *W ratio.* Apesar de indicarem recomendações boas, esses métodos apresentaram os piores resultados de pontuação quantitativa entre os métodos considerados como viáveis. Os resultados da terceira série de testes estão indicados na Tabela 19.

Tabela 19 – Resultados da Terceira Série de Testes

Valor de Entrada	Método de Pontuação	Pontuação
Nanoblock	`partial_ratio`	100
Nanoblock	`partial_ratio`	100
Nanoblock	`partial_token_sort_ratio`	100
Nanoblock	`partial_token_sort_ratio`	100
Nanoblock	`token_set_ratio`	100
Nanoblock	`token_set_ratio`	100
Nanoblock	`W_ratio`	90
Nanoblock	`W_ratio`	90

Valor de Entrada	Método de Pontuação	Pontuação
Birthday	*partial_ratio*	100
Birthday	*partial_ratio*	100
Birthday	*partial_token_sort_ratio*	100
Birthday	*partial_token_sort_ratio*	100
Birthday	*token_set_ratio*	100
Birthday	*token_set_ratio*	100
Birthday	*W_ratio*	90
Birthday	*W_ratio*	90
Building	*partial_ratio*	100
Building	*partial_ratio*	100
Building	*partial_token_sort_ratio*	100
Building	*partial_token_sort_ratio*	100
Building	*token_set_ratio*	100
Building	*token_set_ratio*	100
Building	*W_ratio*	90
Building	*W_ratio*	90
Nanoblock Birthday Cake	*partial_ratio*	100
Nanoblock Birthday Cake	*partial_ratio*	69,56521739
Nanoblock Birthday Cake	*partial_token_sort_ratio*	73,91304348
Nanoblock Birthday Cake	*partial_token_sort_ratio*	73,91304348
Nanoblock Birthday Cake	*token_set_ratio*	100
Nanoblock Birthday Cake	*token_set_ratio*	72,22222222
Nanoblock Birthday Cake	*W_ratio*	90
Nanoblock Birthday Cake	*W_ratio*	85,5
Birthday Cake Kit	*partial_ratio*	94,11764706
Birthday Cake Kit	*partial_ratio*	88,23529412
Birthday Cake Kit	*partial_token_sort_ratio*	100
Birthday Cake Kit	*partial_token_sort_ratio*	82,35294118
Birthday Cake Kit	*token_set_ratio*	100
Birthday Cake Kit	*token_set_ratio*	100
Birthday Cake Kit	*W_ratio*	85,5
Birthday Cake Kit	*W_ratio*	85,5
Nanoblock Building Kit	*partial_ratio*	81,08108108

Valor de Entrada	Método de Pontuação	Pontuação
Nanoblock Building Kit	*partial_ratio*	78,94736842
Nanoblock Building Kit	*partial_token_sort_ratio*	81,81818182
Nanoblock Building Kit	*partial_token_sort_ratio*	77,77777778
Nanoblock Building Kit	*token_set_ratio*	100
Nanoblock Building Kit	*token_set_ratio*	90
Nanoblock Building Kit	*W_ratio*	85,5
Nanoblock Building Kit	*W_ratio*	85,5

O método *token set ratio* apresentou ótimos resultados para os cenários da terceira série de testes. Para cada teste, a primeira recomendação gerada por ele apresentou a pontuação máxima e correspondeu ao produto utilizado como base. Entretanto, as recomendações subsequentes apresentam uma piora considerável em relação à primeira. Apenas para situações em que ele consegue identificar um produto específico, sua primeira recomendação é excelente. Com isso, apesar de esse método não apresentar um desempenho bom para uma ampla gama de cenários, ele ainda pode ser considerado útil para realizar recomendações pontuais.

Para situações gerais, o método *partial ratio* apresentou o melhor desempenho dentre os outros pontuadores considerados viáveis. Dessa maneira, ele foi escolhido como o método de pontuação principal a ser utilizado na parte morfossintática do sistema de recomendação.

Em suma, os testes de refinamento documentados nesta seção culminaram na seleção dos métodos *partial ratio* e *token set ratio* como os mais viáveis para o sistema de recomendação, visto que os dois métodos apresentaram bons resultados sob cenários variados apesar de algumas vantagens e desvantagens. Dessa forma, como alternativa à escolha de um único pontuador ideal no sistema, considerou-se o uso de mais de um ao mesmo tempo, realizando-se apenas um ajuste entre os dois métodos escolhidos. Para exatamente uma recomendação pontual, que devolva uma pontuação próxima do valor máximo, o método *token set ratio* é utilizado, enquanto que para recomendações em situações diversas, o método *partial ratio* é empregado. As implementações realizadas estão descritas detalhadamente na próxima seção da monografia.

7.4.5 Recomendações de Nomes dos Produtos com a Biblioteca de Filtragem Morfossintática

Os testes da biblioteca *RapidFuzz*, realizados na seção 7.4.4, foram utilizados como referência para a criação do filtro morfossintático do sistema de recomendação. Para gerar as recomendações, foram considerados dois métodos de pontuação de similaridade, cada um para casos distintos: *token set ratio*, para uma única recomendação pontual, e *partial ratio*, para recomendações em cenários gerais. As justificativas que permeiam essas escolhas decorrem dos resultados detalhados na seção 7.4.4.

A função desenvolvida para gerar as recomendações sintáticas recebe como parâmetros o valor de entrada fornecido pelo usuário (geralmente o nome do produto desejado), a lista contendo o nome de todos os produtos da base de dados de referência (PROMPTCLOUD, 2020) e o número de recomendações a serem geradas. Inicialmente, a função de recomendação por filtragem morfossintática utiliza o método *token set ratio* para gerar a primeira recomendação caso essa técnica apresente a pontuação máxima. Tanto as recomendações subsequentes quanto a primeira recomendação no caso de *token set ratio* não ter pontuação máxima são obtidas com o método *partial ratio*. Ao final do processamento, a função devolve uma lista com as recomendações geradas e suas respectivas pontuações.

A função projetada para realizar as recomendações foi desenvolvida de maneira modular, de tal forma a facilitar a posterior integração com os outros filtros desenvolvidos. A Figura 41 e Figura 42 ilustram, respectivamente, a função descrita anteriormente e um código de teste para sua utilização com a *string* de busca “*Action Toys*”, gerando cinco recomendações.

```
def prototype_rapid_fuzz_filter(user_input, products, number_of_rec):
    list_of_rec = []
    token_set_ratio_match = process.extract(user_input, products, scorer=fuzz.token_set_ratio, limit=1, processor=utils.default_process)
    partial_ratio_matches = process.extract(user_input, products, scorer=fuzz.partial_ratio, limit=number_of_rec, processor=utils.default_process)

    if token_set_ratio_match[0][1] == 100:
        list_of_rec.extend(token_set_ratio_match)
        for match in partial_ratio_matches:
            if match[0] != token_set_ratio_match[0][0]:
                list_of_rec.append(match)

    else:
        for match in partial_ratio_matches:
            list_of_rec.append(match)

    return list_of_rec
```

Figura 41 – Código do Filtro Morfossintático

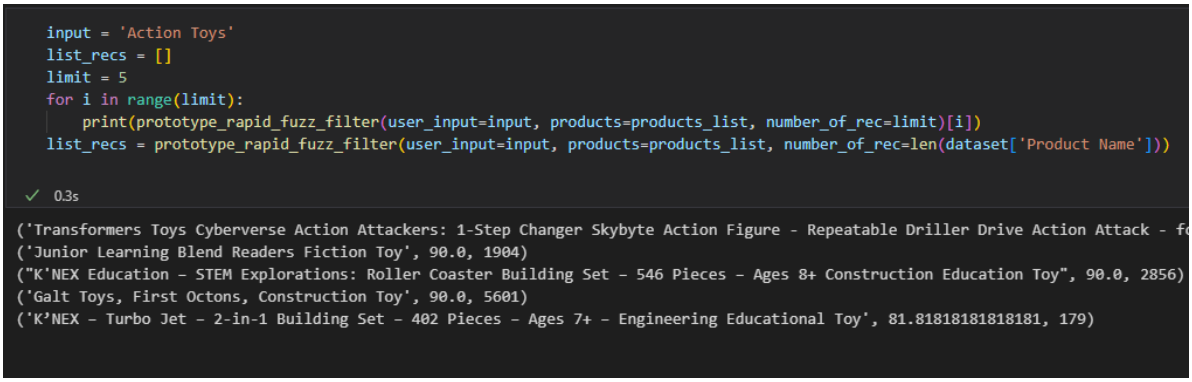


Figura 42 – Teste do Filtro Morfossintático

7.5 Integração das Recomendações Morfossintática e Semântica para o Desenvolvimento do Sistema de Recomendação

O Sistema de Recomendação desenvolvido neste projeto levou em consideração a unificação das técnicas de filtragem e recomendação desenvolvidas e testadas em resposta aos problemas encontrados no sistema de referência realizados anteriormente. Dessa forma, o Sistema de Recomendação combina os filtros por categorias de produtos (seção 7.4.2), por características semânticas de seus nomes (seção 7.4.3) e por características morfossintáticas dessas mesmas denominações (seção 7.4.5).

O código-fonte do Sistema de Recomendação foi estruturado de maneira modular, de tal forma que cada teste realizado antes da integração final fosse organizado em funções que são acionadas ao longo da execução do programa. Essa abordagem permite que a manutenção do sistema seja rápida e fácil, uma vez que cada parte está isolada. Logo, a abordagem simplifica o processo de integração entre os diferentes filtros e favorece a depuração e a construção de um Sistema de Recomendação efetivamente funcional.

Ademais, o maior esforço para o projeto do Sistema de Recomendação é direcionado à lógica de concatenação de uso dos filtros criados e na interação com o usuário. Inicialmente, o sistema verifica se a recomendação deve ser realizada de acordo com dois cenários:

- Cenário 1 – A entrada para a recomendação é um produto da base de dados de referência (PROMPTCLOUD, 2020): Este cenário explora o caso em que o

Sistema de Recomendação está inserido em um sistema de comércio eletrônico para prover aos usuários recomendações baseadas em produtos já disponíveis para venda. Nesse caso, portanto, há um produto da base de dados que é utilizado como base para a recomendação – como, por exemplo, na recomendação de produtos após uma compra *online*. Nesse caso, o sistema possui todas as informações necessárias para direcionar a recomendação a ter parâmetros semelhantes ao da base, como categorias e preço.

- Cenário 2 – A base para a recomendação é uma *string* arbitrária: Este cenário explora o caso em que o usuário ativamente procura um produto de seu interesse por palavras-chave ou utilizando um produto que não está registrado na base de dados do sistema. A situação assemelha-se à realização de uma busca em uma barra de pesquisa, por exemplo, e cabe ao Sistema de Recomendação buscar itens similares na base de dados com a qual ele foi construído (PROMPTCLOUD, 2020).

Assim, a lógica de recomendação altera-se dependendo de qual cenário é tratado:

- No caso do Cenário 1, como as informações do produto estão integralmente disponíveis, utiliza-se, primeiramente, a filtragem por categoria (seção 7.4.2), direcionando a recomendação para produtos de um mesmo grupo. Após a aplicação deste filtro, utilizam-se paralelamente os filtros semântico (seção 7.4.3) e morfossintático (seção 7.4.5), que geram duas listas independentes de recomendações. Essas listas são, então, entrelaçadas realizando-se uma ponderação entre as posições de cada produto nas duas listas. O resultado da recomendação solicitada corresponde aos produtos cujas somas das posições nas listas dos filtros semântico e morfossintático são as menores (isto é, tendem a ser as melhores recomendações dos filtros).
- Já no Cenário 2, como a entrada é uma *string* arbitrária fornecida pelo usuário, não há como utilizar a filtragem por categoria porque não há uma referência para extrair esse dado da base de dados. Dessa forma, essa etapa é ignorada, e a filtragem baseia-se na aplicação dos filtros semântico e morfossintático da mesma forma como no Cenário 1.

Como o Sistema de Recomendação inclui todas as filtragens mencionadas anteriormente, considera-se que, em tese, ele resolve os problemas encontrados no sistema de referência estudados na seção 7.3. Por meio de testes realizados com o Sistema de Recomendação, avaliou-se que ele realiza recomendações adequadas em cenários cujos termos de busca são mais comuns dentro dos produtos da base de dados (PROMPTCLOUD, 2020) e tendem a divergir de recomendações úteis quando os termos pesquisados são abstratos ou escassos na base de dados. Esse padrão de comportamento é esperado em função dos recursos de processamento utilizados.

A Figura 43 e a Figura 44 ilustram exemplos de recomendações do sistema para os cenários 1 e 2, respectivamente. Em ambas as figuras, o resultado indicado após a palavra “Valor” quantifica a qualidade da recomendação calculada pelo sistema, com menores valores representando sugestões potencialmente melhores.

```
Selected Product:
Disney Mickey Golf Ball Spinner Pewter Key Ring Key Accessory


Word2Vec with Category Filter:
36      Disney Mickey Golf Ball Spinner Pewter Key Rin...
689     Disney Tangled Princess Rapunzel Figure Soft T...
7129    Disney Cinderella Carriage Pewter Key Ring Col...
7245              Disney Lilo & Stitch Fan Buddy Key Ring
2292              Disney Baymax Soft Touch PVC Key Holder
Name: Product Name, dtype: object


RapidFuzz with no Category Filter:
('Disney Mickey Golf Ball Spinner Pewter Key Ring Key Accessory', 100.0, 36)
('Disney Dopey Pewter Key Ring', 88.0, 2095)
('Disney Eeyore Brass Key Ring', 69.76744186046511, 3216)
('Marvel Hulk Fist Pewter Key Ring', 63.829787234042556, 370)
('Disney Cinderella Carriage Pewter Key Ring Collectible', 63.06306306306306, 7129)

Final Recommendation:
1 - Produto: Disney Mickey Golf Ball Spinner Pewter Key Ring Key Accessory, Valor: 0
2 - Produto: Disney Cinderella Carriage Pewter Key Ring Collectible, Valor: 6
3 - Produto: Disney Tangled Princess Rapunzel Figure Soft Touch PVC Key Ring Key Accessory, Valor: 7
4 - Produto: Disney Lilo & Stitch Fan Buddy Key Ring, Valor: 11
5 - Produto: Rubber Bracelets | Disney Princess Dream Big Collection | Party Accessory, Valor: 15
6 - Produto: Mickey Through The Years 3D Foam Key Ring Blind Bag, Valor: 16
7 - Produto: amscan Disney Mickey Roadster Square Plates, 9", Party Favor One Size, Multicolor, Valor: 33
8 - Produto: DC Suicide Squad The Joker Icon Pewter Key Chain, Valor: 77
9 - Produto: AMSCAN Skeleton Tank Top Halloween Costume Accessory for Women, One Size, Valor: 164
10 - Produto: Disney Baymax Soft Touch PVC Key Holder, Valor: 211
```

Figura 43 – Recomendações do Sistema para o Cenário 1 (Produto Preexistente na Base de Dados)

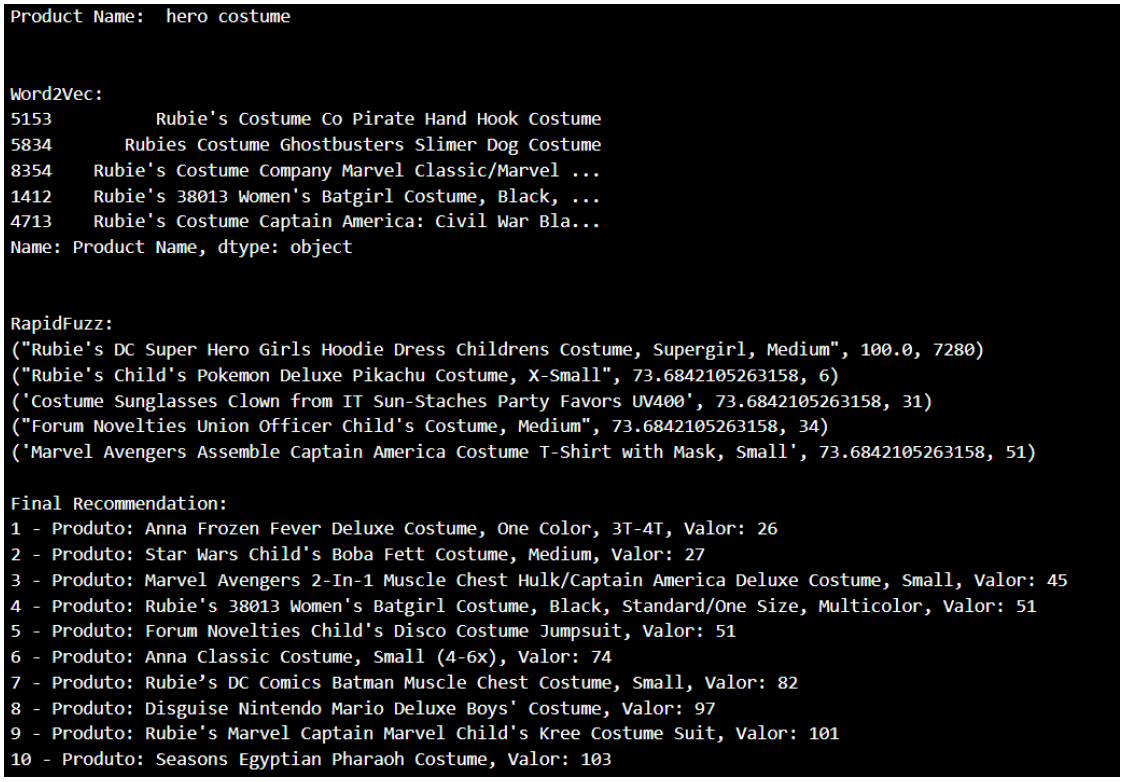


Figura 44 – Recomendações do Sistema para o Cenário 2 (Busca por *Strings*)

8. TESTES E VALIDAÇÃO DO SISTEMA DE RECOMENDAÇÃO

Este capítulo tem como objetivo apresentar os testes e a validação do algoritmo desenvolvido, utilizando uma abordagem de métodos mistos que combina análises quantitativas e qualitativas para avaliar a qualidade das recomendações.

São abordados (i.) a verificação da precisão e da relevância das recomendações geradas pelo sistema em comparação a um sistema de referência, (ii.) a avaliação da experiência dos usuários em relação à usabilidade e à satisfação com o protótipo, e (iii.) a identificação de oportunidades de aprimoramento a partir das percepções e da realimentação dos usuários. Neste capítulo, portanto, exploram-se tanto testes quantitativos, focados em métricas objetivas de desempenho e qualidade, quanto testes qualitativos, voltados para a experiência subjetiva dos usuários, oferecendo uma base sólida para a interpretação dos resultados e para ajustes futuros no sistema de recomendação.

8.1 INTRODUÇÃO AOS TESTES HÍBRIDOS

A escolha de uma abordagem de métodos mistos para validar o sistema de recomendação se fundamenta na necessidade de capturar uma visão ampla e profunda sobre o desempenho e a aceitação das recomendações. A pesquisa mista oferece uma visão mais abrangente ao combinar os pontos fortes das abordagens quantitativa e qualitativa, sendo especialmente indicada para estudos que envolvem interação e percepção do usuário (CRESWELL; CRESWELL, 2022). Os objetivos dos testes são:

- Quantificar a precisão e a satisfação dos usuários com as recomendações, facilitando uma comparação direta entre o sistema desenvolvido e um sistema de referência.
- Obter discernimentos (*insights*) que revelam semelhanças na experiência do usuário e identificam aspectos específicos que contribuem para recomendações adequadas.

8.2 Estrutura e Planejamento dos Testes Quantitativos

Para os testes quantitativos, estruturou-se um método que mensura a qualidade das recomendações por meio de uma **escala de Likert** de 1 a 5, na qual o extremo inferior 1 indica "Péssimo" e o extremo superior 5, "Ótimo". Essa escala possibilita uma análise comparativa de satisfação, capturando a percepção do usuário de maneira objetiva. As principais etapas e componentes desse teste são detalhados na sequência.

8.2.1 Seleção da Amostra

Foram selecionados dez casos de teste, compreendendo produtos e expressões (*strings*) selecionados como entradas, e as respectivas recomendações geradas pelo sistema de referência (CHEAH et al., 2020) e pelo sistema desenvolvido no projeto. Um grupo de quinze voluntários que não participaram do projeto, incluindo familiares, amigos e colegas, foi escolhido para avaliar as recomendações. Essa escolha garante um nível de variabilidade nas preferências e nas percepções dos participantes, fornecendo dados que refletem uma gama diversificada de usuários.

Os detalhes do questionário aplicado, incluindo os casos de teste e as perguntas consideradas, estão presentes no Apêndice B.

8.2.2 Procedimento de Avaliação

Cada participante avaliou a qualidade das recomendações para os produtos designados em ambos os sistemas, atribuindo uma nota em escala de Likert. Esse procedimento permitiu a identificação de padrões de preferência e a avaliação da percepção de qualidade de cada sistema. As avaliações foram conduzidas de maneira anônima para reduzir a influência externa e assegurar a veracidade das respostas.

8.2.3 Métricas Utilizadas

A **média das avaliações** fornecidas pelos participantes foi utilizada para obter uma visão geral da qualidade percebida das recomendações em termos de satisfação do usuário. Esse valor reflete a opinião coletiva dos participantes sobre a recomendação específica. A **taxa de aprovação**, calculada como o percentual de

avaliações entre 4 e 5, serviu como um indicador de aceitação positiva. Por último, a **comparação entre sistemas** foi realizada confrontando diretamente as médias das notas para o sistema de referência (CHEAH et al., 2020) e o sistema desenvolvido no projeto, permitindo identificar, objetivamente, as áreas em que o novo sistema potencialmente se destacou e/ou necessita de melhorias adicionais.

8.3 Estrutura e Procedimentos dos Testes Qualitativos

A pesquisa qualitativa visa compreender mais profundamente as percepções dos usuários sobre a experiência com as recomendações, fornecendo discernimentos (*insights*) subjetivos que complementam os dados quantitativos. Essa análise qualitativa baseia-se em entrevistas semiestruturadas, que permitem tanto a liberdade de expressão dos participantes quanto a consistência dos temas investigados (CRESWELL; CRESWELL, 2022). Os passos e instrumentos utilizados são descritos nas próximas seções:

8.3.1 Seleção e Caracterização dos Participantes

O mesmo grupo de participantes da análise quantitativa foi selecionado para garantir diferentes perfis de uso e preferências. Cada participante relatou suas percepções sobre as recomendações, fornecendo dados variados. Essa diversidade maximiza a relevância dos *insights* e melhora a generalização das conclusões para diferentes tipos de usuários.

8.3.2 Instrumento de Coleta de Dados – Entrevistas Semiestruturadas

As entrevistas seguiram um roteiro de perguntas abertas, organizadas por temas centrais e permitindo ao entrevistador flexibilidade para explorar respostas mais aprofundadas. As principais questões abordadas foram:

- **Precisão das Recomendações:** “*Quão precisas você considera que foram as recomendações para os produtos apresentados?*”;
- **Facilidade de Uso:** “*Quais aspectos da experiência de uso você achou mais intuitivos ou, ao contrário, difíceis de entender?*”;

- **Aspectos de Melhoria:** “*Existem pontos específicos que você acredita que poderiam ser melhorados nas recomendações?*”.

Todas as respostas foram transcritas integralmente para garantir a precisão e a autenticidade dos dados qualitativos coletados. A íntegra dos resultados está disponível no diretório “*results*” do repositório GitHub (ISHICAVA; ITO; INOUE, 2024) do projeto.

8.3.3 Processo de Análise Qualitativa dos Dados

Para a análise qualitativa, os dados foram submetidos a um processo de codificação e categorização temática. Esse processo seguiu as seguintes etapas:

- **Codificação Inicial:** Identificação de palavras e frases-chave nas respostas, que refletem as impressões sobre a qualidade, precisão e utilidade das recomendações;
- **Agrupamento por Temas:** As respostas foram organizadas em categorias frequentes, como precisão das recomendações, usabilidade e sugestões de melhoria, conforme identificado nas entrevistas;
- **Análise Temática:** Os temas extraídos foram, então, analisados para identificar padrões que indicam tanto aspectos satisfatórios quanto áreas de insatisfação, permitindo entender como as recomendações podem ser aprimoradas.

Essa análise qualitativa detalhada proporciona uma visão dos pontos de vista dos usuários, que não seriam capturados apenas pelas métricas quantitativas.

8.4 MÉTRICAS DE AVALIAÇÃO

As métricas de avaliação dos testes foram definidas para capturar tanto a eficiência objetiva quanto a aceitação subjetiva do sistema. Essa abordagem múltipla considera:

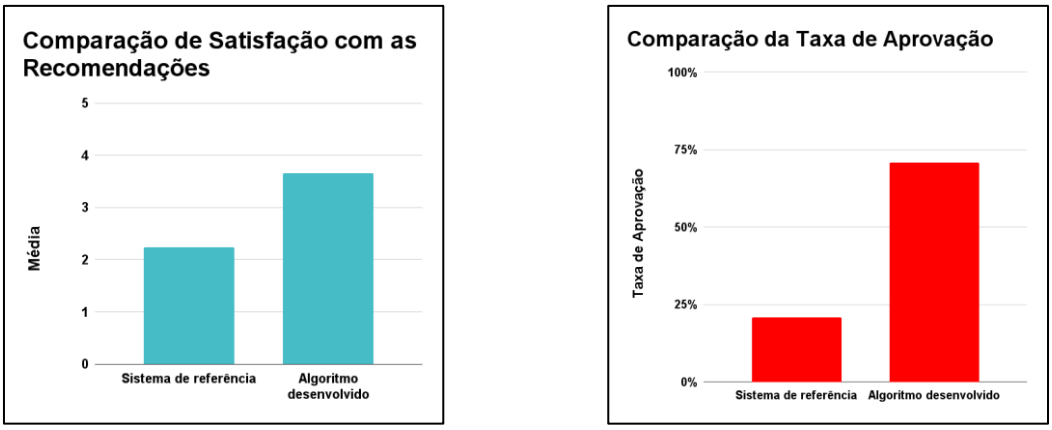
1. **Qualidade Percebida**: A média das notas nas avaliações quantitativas reflete a percepção geral dos usuários sobre a relevância das recomendações;

2. **Indicadores de Realimentação Positiva e Negativa**: A análise qualitativa permite quantificar o percentual de comentários positivos em relação aos negativos, fornecendo uma visão do sentimento geral dos usuários;
3. **Taxa de Aprovação Geral**: Percentual de avaliações com notas 4 ou 5 em todas as recomendações, indicando uma aceitação global do sistema.
4. **Comparação entre Sistemas**: A média das avaliações e os comentários qualitativos para o sistema de referência e o sistema desenvolvido são comparados, destacando os pontos fortes e as áreas de melhoria.

Essas métricas proporcionam uma análise completa da eficácia e da recepção do sistema, permitindo ajustes fundamentados nas preferências e impressões dos usuários.

8.5 Compilação e Análise dos Dados

A etapa de compilação e análise dos dados foi essencial para sintetizar os resultados obtidos nos testes quantitativos e qualitativos, permitindo uma avaliação precisa do desempenho do algoritmo desenvolvido. Os dados quantitativos, apresentados nos gráficos da Figura 45, demonstraram uma melhoria significativa em relação ao sistema de referência. No gráfico (a) da Figura 45, observa-se que a média das avaliações do algoritmo desenvolvido foi de **3,65** na escala de Likert, representando um aumento de **63,6%** em relação ao sistema de referência, cuja média foi de apenas **2,23**. Além disso, a taxa de aprovação, medida como o percentual de notas entre 4 e 5, foi de **71%** para o novo algoritmo, enquanto o sistema de referência obteve apenas **21%**, tal como evidenciado no gráfico (b) da Figura 45. Esses números evidenciam um progresso significativo em termos de precisão e satisfação do usuário.



(a) (b)

Figura 45 – Compilação dos Resultados Quantitativos com Usuários Voluntários: Satisfação com as Recomendações (a) e Taxa de Aprovação (b)

Enquanto os dados quantitativos evidenciam avanços objetivos, os relatos qualitativos fornecem *insights* sobre a experiência dos usuários. Os participantes relataram maior relevância nas recomendações geradas pelo novo sistema, especialmente quando os produtos apresentados refletiam diretamente o termo pesquisado ou estavam em contextos mais específicos. No entanto, em buscas que continham erros de escrita e genéricas, os participantes identificaram uma menor precisão e relevância nos resultados.

Também foram apontadas oportunidades de melhoria. Entre as sugestões, destacam-se a inclusão de imagens dos produtos para facilitar a visualização, traduções para português e a priorização de itens populares ou mais diretamente associados à marca pesquisada. Essas observações, aliadas à análise integrada dos resultados, destacam não apenas o impacto positivo das alterações já implementadas, mas também oferecem direções claras para otimizar ainda mais a experiência do usuário e a eficácia das recomendações.

8.6 INTERPRETAÇÃO DOS RESULTADOS

Os resultados obtidos demonstram que o sistema de recomendação desenvolvido representa uma evolução significativa em relação ao sistema de referência, tanto em termos de precisão quanto de aceitação pelos usuários. O aumento substancial na média das avaliações e na taxa de aprovação reflete um

avanço na qualidade das recomendações, indicando um novo sistema mais alinhado às expectativas e preferências dos usuários.

A análise qualitativa forneceu *insights* que complementam os resultados quantitativos, reforçando os avanços alcançados pelo sistema. Os participantes elogiaram a precisão das recomendações, frequentemente alinhadas às suas necessidades, e a organização aprimorada por categorias, que facilitou a identificação de produtos relevantes, em especial com buscas de produtos específicos.

Contudo, foram destacados aspectos que ainda necessitam de atenção para melhorar a experiência do usuário, como a necessidade da inclusão do sistema em português, para torná-lo mais acessível, melhorias na personalização das recomendações para entradas amplas ou mal escritas e incluir imagens e preços dos produtos, o que faria com que a interface se tornasse mais visual e detalhada. Essas realimentações são fundamentais para orientar ajustes futuros, garantindo uma experiência mais satisfatória, atrativa e alinhada às expectativas dos usuários.

Em suma, as interpretações prévias reforçam que o processo de desenvolvimento do sistema de recomendação seguiu um caminho adequado e também destacam prioridades claras para o desenvolvimento futuro.

8.7 Análise dos Requisitos Não Funcionais

Além da análise das métricas de desempenho, também é importante tecer considerações sobre os requisitos não funcionais #06 e #08, definidos no capítulo 4 e respectivamente relacionados ao **tempo de resposta** e aos **recursos computacionais** do sistema de recomendação.

No quesito **desempenho**, o sistema apresentou tempos de resposta adequados nas condições testadas, atendendo ao patamar de até 90 segundos especificado no Requisito #06. Foi identificado que, no hardware especificado na seção 6.2, as recomendações são geradas em intervalos de 5 a 50 segundos. Também se observou que, à medida que as entradas se tornam menos triviais (por exemplo com erros de digitação ou múltiplas palavras), o tempo de processamento tende a aumentar. Esse comportamento destaca a necessidade de otimizar a

escalabilidade do sistema para que o tempo de resposta seja aceitável mesmo em situações mais exigentes.

Também vale salientar que, considerando que o processador mínimo especificado no Requisito #08 possui em torno de 30% menos capacidade de processamento do que o processador utilizado nos ensaios, o Requisito #06 também é atendido pelo hardware do Requisito #08. Ademais, os recursos de RAM e armazenamento do Requisito #08 também são atendidos pelo Sistema de Recomendação.

Apesar de o Requisito #07 ser funcional e não mencionar aspectos relacionados à usabilidade, os ensaios com usuários voluntários permitem tecer considerações preliminares sobre esse tema. A realimentação dos participantes foi geralmente positiva, destacando a clareza das recomendações e a organização por categorias. Contudo, foram relatadas dificuldades pontuais com a interface, que poderia ser mais intuitiva e visualmente atraente. Sugestões frequentes compreendem a inclusão de imagens e preços nos resultados, bem como a possibilidade de traduções para português, o que aumentaria a acessibilidade para um público mais amplo. Esses elementos foram considerados prioritários para melhorar a experiência do usuário, especialmente para aqueles menos familiarizados com sistemas de recomendação.

Dessa forma, embora o sistema tenha atendido satisfatoriamente aos requisitos não funcionais, esta análise destaca áreas de aprimoramento. Investir na otimização de desempenho, aprimorar a interface visual e funcional, e garantir maior personalização são passos fundamentais para atender às demandas de usuários mais diversos e exigentes. Com esses ajustes, o sistema deve estar mais bem preparado para suportar uma adoção em larga escala e oferecer uma experiência de alta qualidade.

9. CONSIDERAÇÕES FINAIS

Este capítulo tem como objetivo apresentar conclusões obtidas até o presente estágio de desenvolvimento do projeto. São abordados (i.) a avaliação do grau de cumprimento dos objetivos do projeto, (ii.) os seus próximos passos e (iii.) possíveis melhorias e trabalhos correlatos que podem ser futuramente desenvolvidos a partir deste projeto.

9.1 CONCLUSÃO

O projeto “Análise e Aplicação de Inteligência Artificial no Desenvolvimento de um Algoritmo de Recomendação de Conteúdo” apresentou avanços significativos ao longo de seu desenvolvimento, consolidando um sistema de recomendação robusto, funcional e superior à referência considerada (CHEAH et al., 2020). Entre os principais marcos alcançados, destacam-se a implementação integrada dos filtros categóricos, semânticos e morfossintáticos, que permitiram a geração de recomendações relevantes e contextualizadas, mesmo em cenários com dados textuais mais complexos.

Os resultados obtidos evidenciam uma evolução expressiva em relação ao sistema de referência, com melhorias claras na precisão das recomendações e na satisfação dos usuários. A validação híbrida, combinando análises quantitativas e qualitativas, foi essencial para aferir o desempenho do algoritmo e identificar oportunidades de aprimoramento. Métricas como o aumento de 63,6% na média das avaliações e a taxa de aprovação de 71% pelos usuários que participaram dos ensaios conduzidos demonstram que o sistema está alinhado com os objetivos do projeto, traduzidos em requisitos funcionais e não funcionais.

Além disso, o projeto permitiu exercitar o uso de técnicas de inteligência artificial aplicadas a sistemas de recomendação, que compreendem desde o uso de filtros avançados e tratamento de linguagem natural até a estruturação e a execução de métodos unitários e integrados de projeto, verificação e validação. Esses aprimoramentos reforçam a capacidade de adaptação do sistema para o cumprimento dos requisitos correlatos.

Assim, o trabalho atingiu seus objetivos principais, proporcionando o desenvolvimento e a entrega de um sistema de recomendação funcional, validado e com potencial para expansões e aplicações em novos contextos.

9.2 SUGESTÕES DE TRABALHOS FUTUROS

Como trabalho futuro, este projeto apresenta potencial para se expandir o domínio de suas aplicações, podendo ser utilizado para recomendar novos produtos direcionados a vendedores. Um dos principais benefícios dessa aplicação seria a possibilidade de basear as recomendações de acordo com o histórico de busca e de compras de um consumidor, permitindo uma análise comparativa e um aprendizado das preferências de consumo em diferentes contextos. Dessa forma, vendedores poderiam identificar tanto tendências quanto lacunas no mercado, adaptando-se de acordo com as necessidades e perfis de clientes.

Outra possível aplicação é a flexibilidade de o sistema ser aplicado em outras bases de dados – com a ressalva de que as novas bases estejam no mesmo formato da base de dados utilizada. Conceitualmente, o algoritmo pode ser reaproveitado em diferentes contextos, indo além do varejo tradicional para recomendações em áreas como a indústria de serviços e os conteúdos digitais.

Sob a ótica técnica, avalia-se que a realização de análises de sensibilidade dos parâmetros utilizados no algoritmo, incluindo testes de ablação dos algoritmos de IA, é uma área relevante a ser explorada, pois poderia identificar ajustes que aumentem ainda mais a precisão e a relevância das recomendações. A incorporação de informações adicionais aos nomes e às categorias dos produtos, como suas descrições de uso, também pode conduzir a modelos mais sofisticados e com potencial de melhorar as recomendações fornecidas.

Os ensaios realizados com usuários voluntários também evidenciaram outra gama de trabalhos futuros decorrentes do desenvolvimento realizado no projeto. Entre elas, destacam-se os seguintes itens:

a) Para aprimorar a eficiência e a robustez do sistema, é fundamental aprimorar os mecanismos que tratam e interpretam entradas com escrita errada,

garantindo que mesmo consultas mal formatadas resultem em boas recomendações;

b) Disponibilizar uma versão do sistema em português ampliaria sua acessibilidade ao público nacional, tornando-o mais intuitivo para um público diversificado e facilitando sua usabilidade em contextos locais;
c) A análise do impacto visual de elementos como imagens e preços também representa uma oportunidade de melhoria, pois esses componentes podem tornar a interface mais atraente e funcional.

Todas as sugestões de trabalhos futuros prévias destacam o potencial do sistema para evoluir progressiva e continuamente, tornando-se uma ferramenta ainda mais robusta e versátil do que a versão desenvolvida neste trabalho. Com essas melhorias, ele poderá atender a um espectro mais amplo de demandas, ampliando sua eficácia e utilidade em diversas aplicações.

9.3 Consideração Final

Esta monografia documenta o Trabalho de Conclusão de Curso “Análise e Aplicação de Inteligência Artificial no Desenvolvimento de um Algoritmo de Recomendação de Conteúdo”. O projeto demonstrou progresso significativo ao longo de todas as suas etapas, com foco no desenvolvimento de um algoritmo funcional e que superasse o sistema de referência por meio da integração de filtros de diferentes naturezas (categórico, morfossintático e semântico).

As análises quantitativas e qualitativas realizadas durante o desenvolvimento e por meio de pesquisas com usuários voluntários demonstraram que o sistema está alinhado aos objetivos estabelecidos, mostrando avanços importantes em relação ao sistema de referência e atendendo aos objetivos estabelecidos por meio dos requisitos funcionais e não funcionais do projeto. A modularização do código, a clareza na documentação e a definição de processos de desenvolvimento, verificação e validação estruturados foram diferenciais que favorecem a escalabilidade e a longevidade do projeto.

Com uma base sólida, o sistema de recomendação apresenta-se como uma solução viável e com potencial para novas aplicações e melhorias. A continuidade do

trabalho, conforme indicado nas sugestões de trabalhos futuros, permite explorar ao máximo o potencial da tecnologia desenvolvida, alinhando inovação e funcionalidade para atender a demandas diversificadas de mercado e usuários.

REFERÊNCIAS

ADOBE EXPERIENCE LEAGUE. **The Science Behind Target's Recommendations Algorithms**. Disponível em: <https://experienceleague.adobe.com/en/docs/target/using/recommendations/criteria/recommendations-algorithms>. Acesso em: 20 maio. 2024.

AMAZON WEB SERVICES. **Qual é a diferença entre hipervisores tipo 1 e tipo 2?** Disponível em: <https://aws.amazon.com/pt/compare/the-difference-between-type-1-and-type-2-hypervisors/v>. Acesso em: 29 ago. 2024.

ARAÚJO, R. **Máquinas de Vetor de Suporte**. Disponível em: <https://ricardomatsumura.medium.com/máquinas-de-vetor-de-suporte-c2293f68d02d>. Acesso em: 20 maio. 2024.

ARTLEY, B. **Unsupervised Learning: K-Means Clustering**. Disponível em: <https://towardsdatascience.com/unsupervised-learning-k-means-clustering-27416b95af27>. Acesso em: 20 maio. 2024.

BORBA, H. H. P. DA S.; DE ALBUQUERQUE, W. L. L. **Tendências E Desafios da Implementação da Inteligência Artificial nas Empresas**. Disponível em: <https://revistaft.com.br/tendencias-e-desafios-da-implementacao-da-inteligencia-artificial-nas-empresas/>. Acesso em: 20 maio. 2024.

CANONICAL LTD. **Ubuntu**. Disponível em: <https://ubuntu.com>. Acesso em: 29 ago. 2024.

CHEAH, S. et al. **Amazon E-Commerce Recommendation System Using Content-Based Filtering**. Disponível em: <https://github.com/sherincheah/amz-ecom-recommender/tree/main>. Acesso em: 20 maio. 2024.

CRESWELL, J. W.; CRESWELL, J. D. **Research Design: Qualitative, Quantitative, and Mixed Methods Approaches**. 5. ed. Thousand Oaks: SAGE Publications Ltd., 2022.

D'ARC, T. **Por que Usar um Sistema de Recomendação com Inteligência Artificial?** Disponível em: <https://www.smarthint.co/sistema-de-recomendacao-inteligencia-artificial/>. Acesso em: 20 maio. 2024.

DELLOVE, O. A. A. G.; KAMALARAJ, R. Natural Language Processing (NLP) in Recommendation Systems. **International Journal of Innovative Research in Computer and Communication Engineering**, 2024.

DI FANTE, A. L. **Entendendo Sistemas de Recomendação**. Disponível em: <https://arturlunardi.medium.com/entendendo-sistemas-de-recomendação-c50a20856394>. Acesso em: 20 maio. 2024.

DRR (Data-Driven Recommender). Disponível em: <https://github.com/rec-agent/drr>. Acesso em: 20 maio. 2024.

E-commerce-recommendation-system. Disponível em: <https://github.com/Xinyi6/E-commerce-recommendation-system>. Acesso em: 20 maio. 2024.

GHAZANFAR, M. A.; PRÜGEL-BENNETT, A.; SZEDMAK, S. Kernel-Mapping Recommender system algorithms. **Information Sciences**, v. 208, p. 81–104, 2012.

GOYANI, M.; CHAURASIYA, N. A Review of Movie Recommendation System: Limitations, Survey and Challenges. **Electronic Letters on Computer Vision and Image Analysis**, v. 19, n. 3, p. 18–37, 2020.

HARRISON, O. **Machine Learning Basics with the K-Nearest Neighbors Algorithm**. Disponível em: <https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>. Acesso em: 20 maio. 2024.

HERLOCKER, J. L. **Understanding and improving automated collaborative filtering systems**. University of Minnesota, 2000.

HRUSCHKA, E. R.; CAMPELLO, R. **Representações de Dados e Proximidade**. São Paulo: 2024a.

HRUSCHKA, E. R.; CAMPELLO, R. **Agrupamento de Dados e Aplicações Algoritmos Particionais Parte I**. São Paulo: 2024b.

HRUSCHKA, E. R.; CAMPELLO, R. **Agrupamento de Dados e Aplicações Algoritmos Particionais Parte II**. São Paulo: 2024c.

HRUSCHKA, E. R.; CAMPELLO, R. **Validação**. São Paulo: 2024d.

IBM. **What is a Decision Tree?** Disponível em: <https://www.ibm.com/topics/decision-trees>. Acesso em: 20 maio. 2024a.

IBM. **What is Random Forest?** Disponível em: <https://www.ibm.com/topics/random-forest>. Acesso em: 20 maio. 2024b.

IBM. **What is a Neural Network?** Disponível em: <https://www.ibm.com/topics/neural-networks>. Acesso em: 20 maio. 2024c.

ISHICAVA, L.; ITO, N.; INOUE, R. **RecommendationSystem**. Disponível em: <https://github.com/Nyito/RecommendationSystem>. Acesso em: 22 ago. 2024.

JAIN et al. **Algorithms for Clustering Data**. 1. ed. Englewood Cliffs: Prentice Hall, 1988.

JAIN, K. N. et al. **Movie recommendation system: Hybrid information Filtering System**. Intelligent Computing and Information and Communication. **Anais**...Singapore: Springer, 2018.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models**. 3(Draft) ed. Palo Alto, California: University of Stanford, 2024. v. 3 (Draft)

KOZAN, M. **Supervised and Unsupervised Learning (an Intuitive Approach)**. Disponível em: <https://medium.com/@metehankozan/supervised-and-unsupervised-learning-an-intuitive-approach-cd8f8f64b644>. Acesso em: 20 maio. 2024.

KOZYRIEV, A. **Game Recommendations on Steam**. Disponível em: <https://www.kaggle.com/datasets/antonkozyriev/game-recommendations-on-steam>. Acesso em: 20 maio. 2024.

LEE, W.-M.; CHO, Y.-S. A Flexible Two-Tower Model for Item Cold-Start Recommendation. **IEEE Access**, v. 11, p. 146194–146207, 2023.

LI, H.; CAI, F.; LIAO, Z. Content-based filtering recommendation algorithm using HMM. **Proceedings - 4th International Conference on Computational and**

Information Sciences, ICCIS 2012, p. 275–277, 2012.

MICROSOFT CORPORATION. **Visual Studio Code - Code Editing. Redefined**. Disponível em: <https://code.visualstudio.com/>. Acesso em: 22 maio. 2024a.

MICROSOFT CORPORATION. **Introdução ao Hyper-V no Windows 10**. Disponível em: <https://learn.microsoft.com/pt-br/virtualization/hyper-v-on-windows/about/>. Acesso em: 29 ago. 2024b.

MIKOLOV, T. et al. Efficient Estimation of Word Representations in Vector Space (word2vec). **Computer Science**, n. January 2013, p. 1–12, 2013.

MODARRESI, K. Recommendation system based on complete personalization. **Procedia Computer Science**, v. 80, p. 2190–2204, 2016.

MONTEIRO, G. **Entendendo DBSCAN**. Disponível em: <https://gabriellm.medium.com/entendendo-dbscan-770f680d9160>. Acesso em: 20 maio. 2024.

NEKOUEI, F. **Customer Segmentation & Recommendation System**. Disponível em: <https://www.kaggle.com/code/farzadnekouei/customer-segmentation-recommendation-system>. Acesso em: 20 maio. 2024.

NIELSENIQ. **The evolution of e-commerce globally**. Disponível em: <https://nielseniq.com/global/en/insights/analysis/2022/the-evolution-of-e-commerce-globally>. Acesso em: 21 maio. 2024.

OBERG, R.; PROBASCO, L.; ERICSSON, M. Applying requirements management with use cases. **Rational Software Corporation**, v. 5, p. 24, 2000.

ORACLE. **Oracle Virtual Box**. Disponível em: <https://www.virtualbox.org>. Acesso em: 29 ago. 2024.

PANCINI, L. **Home Tecnologia Inteligência Artificial na hora da compra é preferência entre brasileiros**. Disponível em: <https://exame.com/tecnologia/inteligencia-artificial-na-hora-da-compra-e-preferencia-entre-brasileiros/>. Acesso em: 20 maio. 2024.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. **GloVe: Global Vectors for Word Representation**. 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). **Anais**...Doha: Association for Computational Linguistics, 2014.

PROMPTCLOUD. **Amazon Product Dataset 2020**. Disponível em: <https://data.world/promptcloud/amazon-product-dataset-2020>. Acesso em: 30 maio. 2024.

PYTHON SOFTWARE FOUNDATION. **fuzzywuzzy - PyPI**. Disponível em: <https://pypi.org/project/fuzzywuzzy/>. Acesso em: 3 out. 2024.

PYTHON SOFTWARE FOUNDATION. **RapidFuzz - PyPI**. Disponível em: <https://pypi.org/project/RapidFuzz/>. Acesso em: 3 out. 2024a.

PYTHON SOFTWARE FOUNDATION. **Python Release Python 3.12.3 | Python.org**. Disponível em: <https://www.python.org/downloads/release/python-3123/>. Acesso em: 22 maio. 2024b.

ŘEHŮŘEK, R. **Gensim Documentation - Gensim 4.3.3 documentation**. Disponível em: <https://radimrehurek.com/gensim/auto_examples/index.html#%23documentation>.

Acesso em: 28 ago. 2024.

ROBICQUET, A. **What’s Next For Personalization In Digital Commerce**. Disponível em: <https://www.forbes.com/sites/forbestechcouncil/2023/03/14/whats-next-for-personalization-in-digital-commerce>. Acesso em: 21 maio. 2024.

SALTON, G.; MCGILL, M. J. **Introduction to Modern Information Retrieval (TF-IDF)**. 1. ed. New York: McGraw-Hill, 1986.

SCIKIT-LEARN TEAM. **Scikit-Learn**. Disponível em: <https://scikit-learn.org/stable/whats_new/v1.4.html#version-1-4-2>. Acesso em: 31 maio. 2024.

SUMESH, S.; ASWINI, S. H. **Natural Language Processing based Recommendation System for Courses**. 2023 International Conference on Inventive Computation Technologies (ICICT). **Anais**...2023.

THE NLTK TEAM. **NLTK documentation - NLTK 3.9.1 documentation**. Disponível em: <https://www.nltk.org>. Acesso em: 28 ago. 2024.

THE NUMPY TEAM. **NumPy documentation — NumPy v1.26 Manual**. Disponível em: <https://numpy.org/doc/stable/>. Acesso em: 22 maio. 2024.

THE PANDAS TEAM. **pandas documentation — pandas 2.2.2 documentation**. Disponível em: <https://pandas.pydata.org/docs/>. Acesso em: 22 maio. 2024.

VAIBHAV, P. **Ecommerce-product-recommendation-system**. Disponível em: <https://github.com/Vaibhav67979/Ecommerce-product-recommendation-system>. Acesso em: 20 maio. 2024.

VIJAYA KUMAR, P. N.; REDDY, V. R. A Survey on Recommender Systems (RSS) and Its Applications. **International Journal of Innovative Research in Computer and Communication Engineering**, v. 3297, n. 8, p. 5254–5260, 2007.

WANG, D. et al. A Content-Based Recommender System for Computer Science Publications. **Knowledge-Based Systems**, v. 157, p. 1–9, 2018.

WANG, R. **Revisiting GloVe, Word2Vec and BERT: On the Homogeneity of Word Vectors**. Disponível em: <https://www.cs.toronto.edu/~rwang/files/embeddings.pdf>. Acesso em: 10 out. 2024.

APÊNDICE

APÊNDICE A – Resultado da Segunda Série de Testes do Filtro Morfossintático

O presente apêndice tem como objetivo apresentar os resultados da segunda série de testes do filtro morfossintático, explorada e analisada na seção 7.4.4. Esses resultados constam na Tabela 20.

Tabela 20 – Resultados da Segunda Série de Testes do Filtro Morfossintático

Valor de Entrada	Método de Pontuação	Pontuação
Learning	ratio	52,63158
Learning	ratio	47,05882
Learning	partial_ratio	100
Learning	partial_ratio	100
Learning	token_sort_ratio	53,84615
Learning	token_sort_ratio	50
Learning	partial_token_sort_ratio	100
Learning	partial_token_sort_ratio	100
Learning	token_set_ratio	100
Learning	token_set_ratio	100
Learning	partial_token_set_ratio	100
Learning	partial_token_set_ratio	100
Learning	Q_ratio	52,63158
Learning	Q_ratio	47,05882
Learning	W_ratio	90
Learning	W_ratio	90
Turquoise	ratio	47,05882
Turquoise	ratio	44,44444
Turquoise	partial_ratio	100
Turquoise	partial_ratio	100
Turquoise	token_sort_ratio	48
Turquoise	token_sort_ratio	47,05882
Turquoise	partial_token_sort_ratio	100
Turquoise	partial_token_sort_ratio	100
Turquoise	token_set_ratio	100
Turquoise	token_set_ratio	100

Valor de Entrada	Método de Pontuação	Pontuação
Turquoise	partial_token_set_ratio	100
Turquoise	partial_token_set_ratio	100
Turquoise	Q_ratio	47,05882
Turquoise	Q_ratio	44,44444
Turquoise	W_ratio	90
Turquoise	W_ratio	90
Fiction	ratio	46,15385
Fiction	ratio	44,44444
Fiction	partial_ratio	100
Fiction	partial_ratio	100
Fiction	token_sort_ratio	46,15385
Fiction	token_sort_ratio	44,44444
Fiction	partial_token_sort_ratio	100
Fiction	partial_token_sort_ratio	100
Fiction	token_set_ratio	100
Fiction	token_set_ratio	100
Fiction	partial_token_set_ratio	100
Fiction	partial_token_set_ratio	100
Fiction	Q_ratio	46,15385
Fiction	Q_ratio	44,44444
Fiction	W_ratio	90
Fiction	W_ratio	90
Learninc	ratio	50
Learninc	ratio	42,10526
Learninc	partial_ratio	93,33333
Learninc	partial_ratio	93,33333
Learninc	token_sort_ratio	50
Learninc	token_sort_ratio	47,05882
Learninc	partial_token_sort_ratio	93,33333
Learninc	partial_token_sort_ratio	87,5
Learninc	token_set_ratio	50
Learninc	token_set_ratio	47,05882
Learninc	partial_token_set_ratio	93,33333

Valor de Entrada	Método de Pontuação	Pontuação
Learninc	partial_token_set_ratio	87,5
Learninc	Q_ratio	50
Learninc	Q_ratio	42,10526
Learninc	W_ratio	84
Learninc	W_ratio	84
Turqooise	ratio	46,15385
Turqooise	ratio	44,44444
Turqooise	partial_ratio	88,88889
Turqooise	partial_ratio	88,88889
Turqooise	token_sort_ratio	48
Turqooise	token_sort_ratio	44,44444
Turqooise	partial_token_sort_ratio	88,88889
Turqooise	partial_token_sort_ratio	88,88889
Turqooise	token_set_ratio	48
Turqooise	token_set_ratio	44,44444
Turqooise	partial_token_set_ratio	88,88889
Turqooise	partial_token_set_ratio	88,88889
Turqooise	Q_ratio	46,15385
Turqooise	Q_ratio	44,44444
Turqooise	W_ratio	80
Turqooise	W_ratio	80
Fection	ratio	44,44444
Fection	ratio	44,44444
Fection	partial_ratio	100
Fection	partial_ratio	92,30769
Fection	token_sort_ratio	47,05882
Fection	token_sort_ratio	44,44444
Fection	partial_token_sort_ratio	100
Fection	partial_token_sort_ratio	85,71429
Fection	token_set_ratio	100
Fection	token_set_ratio	47,05882
Fection	partial_token_set_ratio	100
Fection	partial_token_set_ratio	85,71429

Valor de Entrada	Método de Pontuação	Pontuação
Fection	Q_ratio	44,44444
Fection	Q_ratio	44,44444
Fection	W_ratio	90
Fection	W_ratio	83,07692
Learning Fantail Books	ratio	60
Learning Fantail Books	ratio	52,63158
Learning Fantail Books	partial_ratio	100
Learning Fantail Books	partial_ratio	72,72727
Learning Fantail Books	token_sort_ratio	55
Learning Fantail Books	token_sort_ratio	52,83019
Learning Fantail Books	partial_token_sort_ratio	68,18182
Learning Fantail Books	partial_token_sort_ratio	68,18182
Learning Fantail Books	token_set_ratio	100
Learning Fantail Books	token_set_ratio	58,33333
Learning Fantail Books	partial_token_set_ratio	100
Learning Fantail Books	partial_token_set_ratio	100
Learning Fantail Books	Q_ratio	60
Learning Fantail Books	Q_ratio	52,63158
Learning Fantail Books	W_ratio	90
Learning Fantail Books	W_ratio	85,5
Learninc Fentail Bools	ratio	56,14035
Learninc Fentail Bools	ratio	53,84615
Learninc Fentail Bools	partial_ratio	86,36364
Learninc Fentail Bools	partial_ratio	68,18182
Learninc Fentail Bools	token_sort_ratio	55
Learninc Fentail Bools	token_sort_ratio	52
Learninc Fentail Bools	partial_token_sort_ratio	68,18182
Learninc Fentail Bools	partial_token_sort_ratio	63,63636
Learninc Fentail Bools	token_set_ratio	55
Learninc Fentail Bools	token_set_ratio	52
Learninc Fentail Bools	partial_token_set_ratio	68,18182
Learninc Fentail Bools	partial_token_set_ratio	63,63636
Learninc Fentail Bools	Q_ratio	56,14035

Valor de Entrada	Método de Pontuação	Pontuação
Learninc Fentail Bools	Q_ratio	53,84615
Learninc Fentail Bools	W_ratio	77,72727
Learninc Fentail Bools	W_ratio	61,36364

APÊNDICE B – Questionário de Testes e Validação do Sistema de Recomendação

O presente apêndice tem como objetivo apresentar o questionário respondido pelos usuários voluntários que participaram dos testes do sistema de recomendação. Esses testes foram relatados no capítulo 8 da monografia.

Ao todo, o questionário compreende dez cenários de teste com cinco perguntas cada. As perguntas respondidas pelos usuários voluntários estão na Figura 46, ao passo que os cenários de teste considerados são apresentados entre a Figura 47 e a Figura 56. Em todas as figuras, o “algoritmo 1” corresponde ao sistema de referência (CHEAH et al., 2020), ao passo que o “algoritmo 2” é o sistema de recomendação desenvolvido neste projeto.

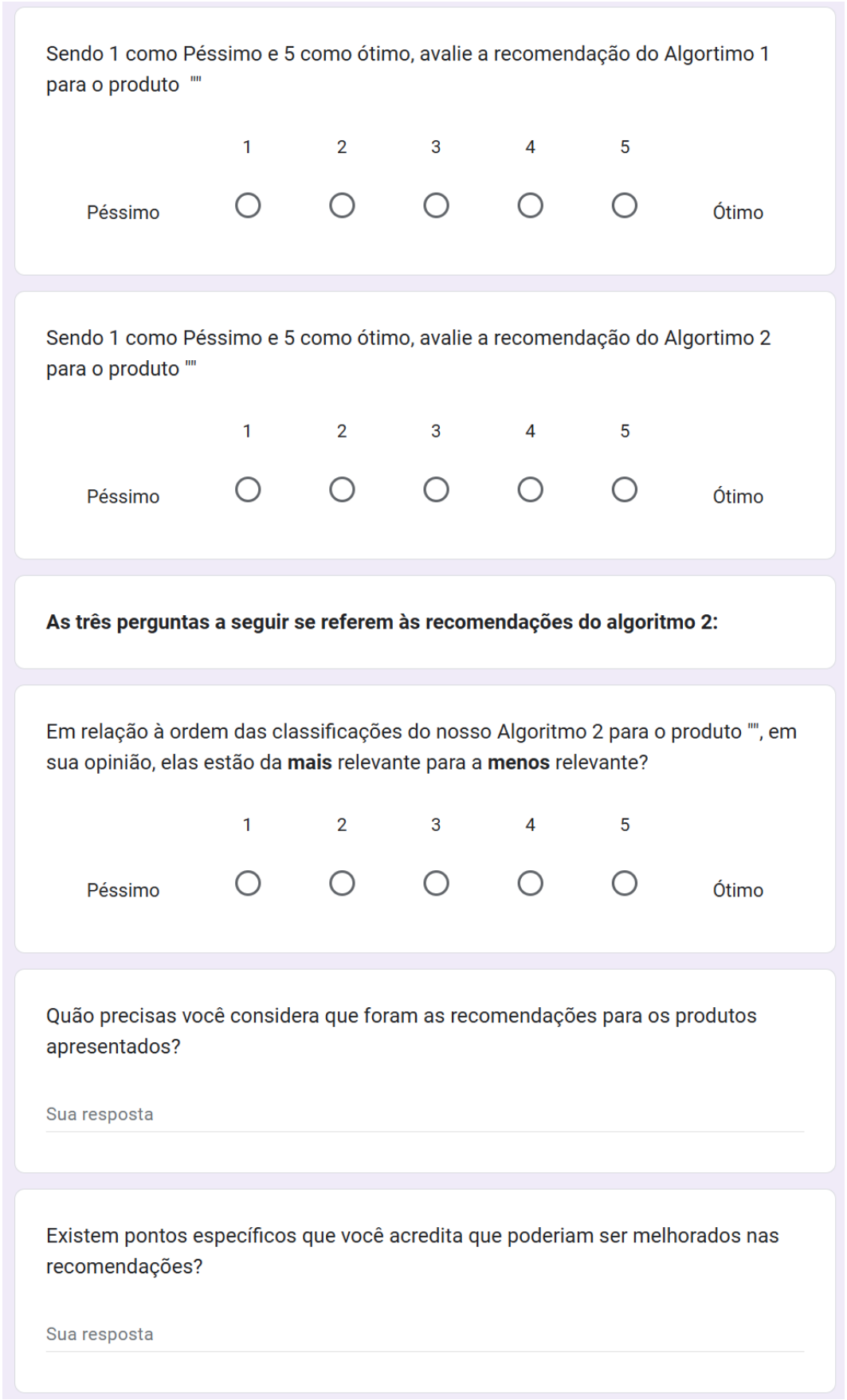


Figura 46 – Perguntas Aplicáveis aos Cenários de Teste

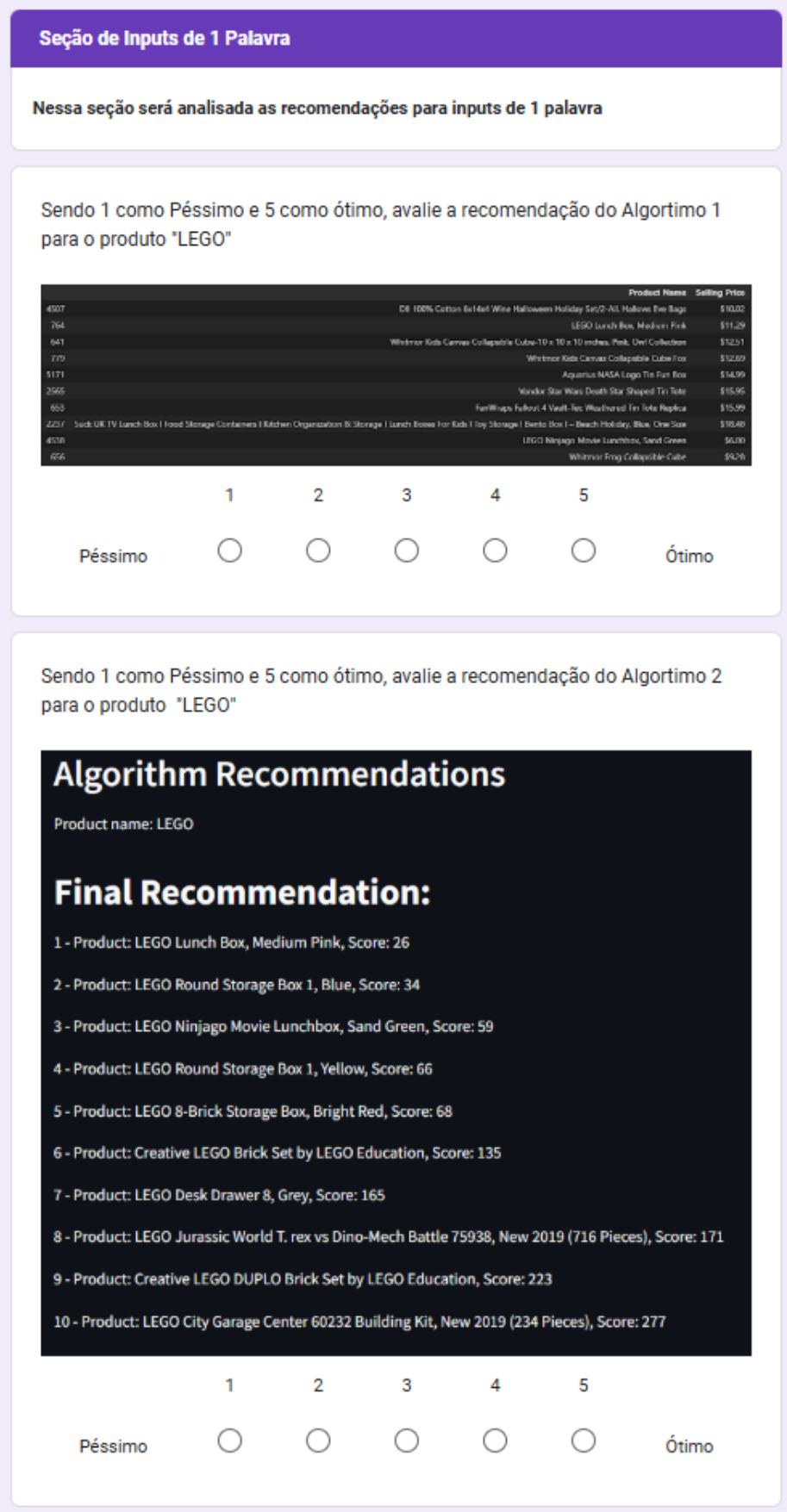


Figura 47 – Cenário de Teste 1 do Sistema de Recomendação

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Costume"

	Product Name	Selling Price
455	amscan Cape Hooded Black Child	$10.16
147	Star Wars: The Force Awakens Child's Stormtrooper Costume, Medium	$12.88
699	Forum Novelties I'm Invisible Costume Stretch Body Suit, Pink, Child Large	$16.94
696	Nella The Knight Classic Child Girl Costume, M (3T-4T)	$17.88
129	Nick Jr. Dora the Explorer Child's Dora Costume with Backpack, Small	$23.74
234	Juniors Go Go Girl Black Costume (Standard;Child Large)	$24.76
568	Anna Classic Costume, Small (4-6x)	$25.89
81	Great Pretenders Fairy Blooms Deluxe Dress with Wings, Green, Medium	$34.99
218	Anna Frozen Fever Deluxe Costume, One Color, 3T-4T	$34.99
409	Forum Novelties Party Supplies 80407 Plush Monkey Child's Mascot Costume, Medium	$9.48

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 2 para o produto "Costume"

Algorithm Recommendations

Product name: Costume

Final Recommendation:

1 - Product: Forum Novelties Child's Disco Costume Jumpsuit, Score: 43

2 - Product: Disguise Yellow Ranger Beast Morpher Deluxe Girls' Costume, Score: 52

3 - Product: Rubie's Rag Doll Children's Costume, Score: 70

4 - Product: Rubie's Marvel Captain Marvel Child's Kree Costume Suit, Score: 77

5 - Product: Seasons Egyptian Pharaoh Costume, Score: 79

6 - Product: Branch Classic Trolls Costume, Multicolor, Small (4-6), Score: 80

7 - Product: Forum Novelties I'm Invisible Costume Stretch Body Suit, Pink, Child Large, Score: 81

8 - Product: Disguise Nintendo Mario Deluxe Boys' Costume, Score: 82

9 - Product: Marvel Avengers 2-In-1 Muscle Chest Hulk/Captain America Deluxe Costume, Small, Score: 85

10 - Product: Rubie's Costume Indian Maiden Value Child Costume, Small, Score: 93

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Figura 48 – Cenário de Teste 2 do Sistema de Recomendação

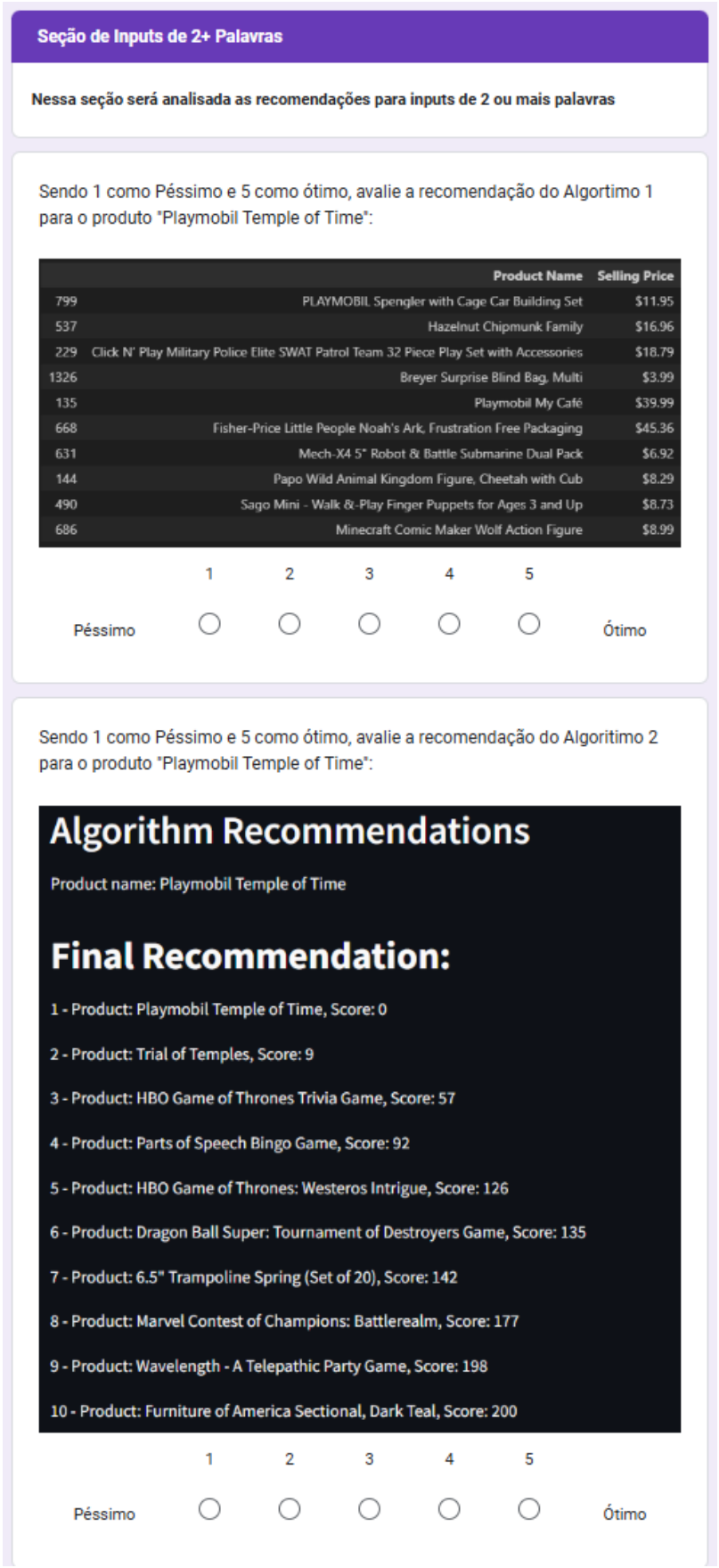
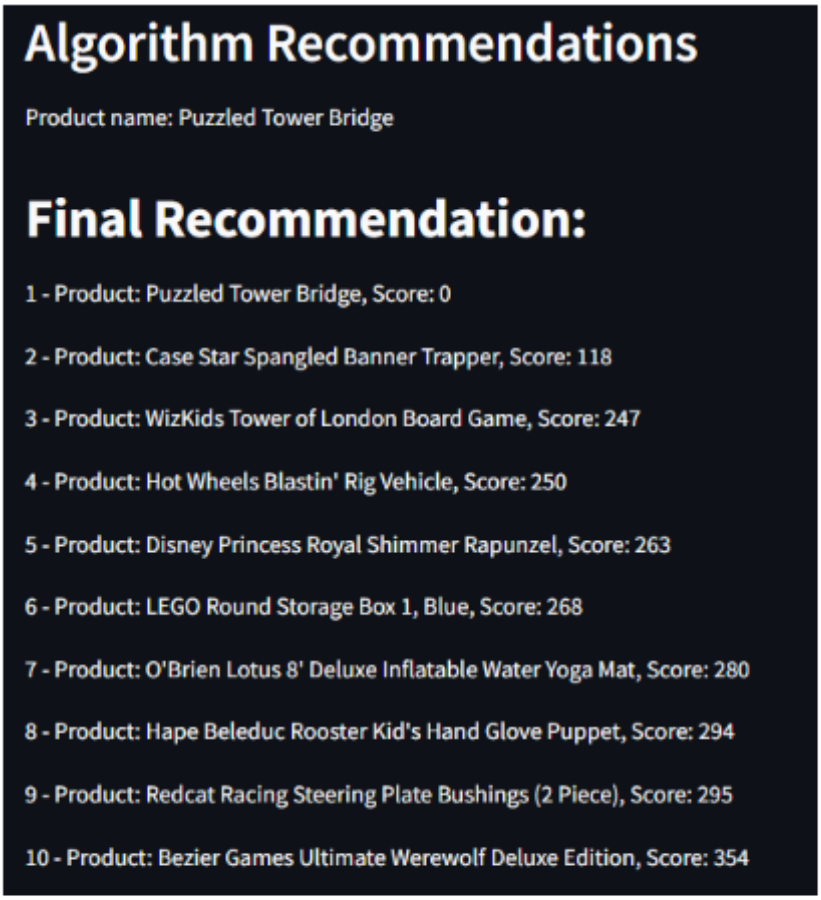


Figura 49 – Cenário de Teste 3 do Sistema de Recomendação

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Puzzled Tower Bridge":

	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algoritimo 2 para o produto "Puzzled Tower Bridge":



	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Figura 50 – Cenário de Teste 4 do Sistema de Recomendação

Seção de Inputs de 2+ Palavras em sequência

Nessa seção será analisada as recomendações para inputs de 2 ou mais palavras em sequência

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Learning Fantail Books":

	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algoritimo 2 para o produto "Learning Fantail Books":

	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Figura 51 – Cenário de Teste 5 do Sistema de Recomendação

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Base Space":

	Product Name	Selling Price
88	Firefly: The Game - Esmeralda Game Expansion	$12.50
291	Pressman Toys Giant Snakes & Ladders Game (4 Player)	$14.90
118	Indie Boards and Cards Flash Point Fire Rescue 2nd Story	$14.99
278	Smart Play Ingenio Colors & Shapes Memory Match Game	$15.20
252	Toysmith Get Outside GO! Neon Dart Ball Set, Packaging may vary	$16.78
223	Schylling Shuffle Shot	$19.99
250	Carson-Dellosa Publishing Language Arts Learning Games, Grade 2	$24.99
302	Ninja Division NAS Howl & Yip Board Game	$3.07
193	GreenBrier Games Yashima Legends from Fairytale Board Game	$35.00
299	Inlaid Cribbage Box with Cards	$43.00

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algoritimo 2 para o produto "Base Space":

Algorithm Recommendations

Product name: Base Space

Final Recommendation:

1 - Product: Space Base, Score: 1

2 - Product: Power Trains Car Pack: Space, Score: 7

3 - Product: Daron Space Adventure Saturn V Rocket Model Playset, Score: 9

4 - Product: Daron Space Adventure Lunar Rover Playset, Score: 23

5 - Product: Disney Cars Table Lamp, Score: 72

6 - Product: ETA hand2mind Blue Plastic Base Ten Rods, Set of 50, Score: 106

7 - Product: Eureka Mickey Teacher Cards, Score: 185

8 - Product: Key Education Sight Word Space Station Board Game Early Learning Game105 Pcs. Carson-Dellosa 840001, Score: 186

9 - Product: Creative Converting BB375533 Space Blast 9oz Cups -8 Pack, Score: 210

10 - Product: 3B Scientific A18/21 Mini Human Spinal Column Model - Flexible, On Base, 17.3" Height, Score: 230

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Figura 52 – Cenário de Teste 6 do Sistema de Recomendação

Seção de Inputs de 2+ Palavras em sequência fora do padrão

Nessa seção será analisada as recomendações para inputs de 2 ou mais palavras em sequência fora do padrão

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Fantail Books Learning":

	Product Name	Selling Price
3769	PlayGo Baby Rock Star Guitar	$14.99
15	Yellind Fritz; Voice-Activated Spider Pet; Ages 5 & Up	$17.85
1222	Playgo Musical Spinning Wheel	$19.95
5313	B. toys by Battat – Under the Sea. Iverboone – B. Softies – Musical Octopus Toy – Soft Octopus Plush with 8 Instruments – Sensory Toys for babies 10 months +	$19.99
4617	Stephen Joseph Xylophone, Unicorn	$21.80
2582	Hape Toddler Beat Box Set, Wooden Music Toy Set	$32.00
2570	Singing Machine Kids Candy House Portable Bluetooth Sing-Along Speaker with LED Lit Microphone and Rooftop (SMK470)	$35.74
574	Basic Fun Fisher-Price Play Tape Recorder	$49.99
5565	Foamnasium Toddler Step, Red	$52.89
807	Manhattan Toy Musical Shapes Maraca Wooden Toddler Instrument Toy	$7.99

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 2 para o produto "Fantail Books Learning":

Algorithm Recommendations

Product name: Fantail Books Learning

Final Recommendation:

1 - Product: Carson-Dellosa Publishing Language Arts Learning Games, Grade 2, Score: 11

2 - Product: LeapFrog Learning Friends 100 Words Book, Green, Score: 12

3 - Product: Carson Dellosa Math Windows Learning Cards (140069), Score: 50

4 - Product: Key Education Sight Word Space Station Board Game Early Learning Game105 Pcs. Carson-Dellosa 840001, Score: 79

5 - Product: Learning Resources 3 Realistic-Looking Baskets of Nutritious Mealtime Food, Score: 96

6 - Product: Didax Educational Resources Volume Measurement Dominoes Children's Mathematical Learning Aids, Score: 106

7 - Product: Junior Learning Fantail Books Turquoise Non Fiction Educational Action Games, Score: 110

8 - Product: Carson Dellosa Early Learning Skills/Learning Cards (D44046), Score: 112

9 - Product: Learning Advantage Wood Pattern Blocks, Score: 117

10 - Product: Smithsonian Rug US Map Learning Carpets Bedding Play Mat Classroom Decorations Blue Area Rugs 8x10, Navy, Score: 119

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Figura 53 – Cenário de Teste 7 do Sistema de Recomendação

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Space Base":

	Product Name	Selling Price
88	Firefly: The Game - Esmeralda Game Expansion	$12.50
291	Pressman Toys Giant Snakes & Ladders Game (4 Player)	$14.90
118	Indie Boards and Cards Flash Point Fire Rescue 2nd Story	$14.99
278	Smart Play Ingenio Colors & Shapes Memory Match Game	$15.20
252	Toysmith Get Outside GO! Neon Dart Ball Set, Packaging may vary	$16.78
223	Schylling Shuffle Shot	$19.99
250	Carson-Dellosa Publishing Language Arts Learning Games, Grade 2	$24.99
302	Ninja Division NAS Howl & Yip Board Game	$3.07
193	GreenBrier Games Yashima Legends from Fairytale Board Game	$35.00
299	Inlaid Cribbage Box with Cards	$43.00

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algoritimo 2 para o produto "Space Base":

Algorithm Recommendations

Product name: Space Base

Final Recommendation:

1 - Product: Space Base, Score: 1

2 - Product: Daron Space Adventure Lunar Rover Playset, Score: 23

3 - Product: urb SPACE 6 Pack, Black, Score: 34

4 - Product: Posterazzi PSTRFF201146S Space Journey. Astronauts on Another Planet. Vivid Universe Photo Print, 11 x 17, Multi, Score: 57

5 - Product: ETA hand2mind Blue Plastic Base Ten Rods, Set of 50, Score: 78

6 - Product: BLAST BALL Kick Ball (2 Pack), Score: 175

7 - Product: 9" SOCCER BALL INFLATE, Score: 175

8 - Product: Aquabeads Jewel Bead Refill Pack, Blue, Score: 199

9 - Product: amscan Metallic Blue Shred, Score: 200

10 - Product: Power Trains Car Pack: Space, Score: 210

	1	2	3	4	5	
Péssimo	○	○	○	○	○	Ótimo

Figura 54 – Cenário de Teste 8 do Sistema de Recomendação

Seção de Input de 1 Palavra escritas incorretamente

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Naonblock" (a escrita está errada, deveria ser "Nanoblock"):

	Product Name	Selling Price
1673	LEGO Classic World Fun 10403 Building Kit (295 Piece)	$14.59
1474	Clicformers Basic Set (110 Piece) Educational Building Blocks Kit, Construction STEM Toy, Creative Building Bricks	$19.18
332	BathBlocks STEM Floating Construction Set	$24.99
1014	LEGO City Dirt Road Pursuit 60172 Building Kit (297 Pieces)	$29.00
1505	Tileblox Inspire 60 Piece Set Magnetic Building Blocks, Educational Magnetic Tiles Kit , Magnetic Construction STEM Toy Set	$39.76
1523	K'NEX Education – Kid K'NEX Group Building Set – 131 Pieces – Ages 3+ – Preschool Educational Toy	$45.99
99	COBI Small Army DW Twin-Turret Tank	$51.39
619	PLAYMOBIL How to Train Your Dragon III Astrid & Hiccup	$9.70
847	Clicformers Basic Set (90 Piece) Educational Building Blocks Kit, Construction STEM Toy, Creative Building Bricks	$9.72
1368	Tegu Magnetron Magnetic Wooden Block Set	$98.88

	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algoritimo 2 para o produto "Naonblock":

Algorithm Recommendations

Product name: Naonblock

Final Recommendation:

1 - Product: Ludonova Ceylon, Score: 120

2 - Product: Galison David Hicks Backgammon Set, Score: 140

3 - Product: FOLDOVER STICKER, Score: 141

4 - Product: Artstraws Class Pack, Score: 268

5 - Product: Celebrity 24DNB 24" Latex Balloons, Navy Blue, Score: 271

6 - Product: Amscan Crazy Party Wig Costume, Blue, Score: 273

7 - Product: Hauck Batmobile Pedal Go Kart, Score: 290

8 - Product: urb SPACE 6 Pack, Black, Score: 306

9 - Product: Sloth Animal Club Patch, Score: 313

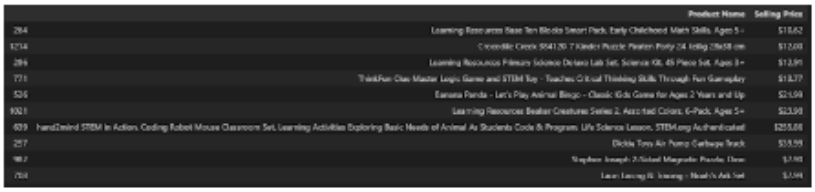
10 - Product: Anagram Balloons 2827901 Foil Balloon, 34", Blue, Score: 356

	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Figura 55 – Cenário de Teste 9 do Sistema de Recomendação

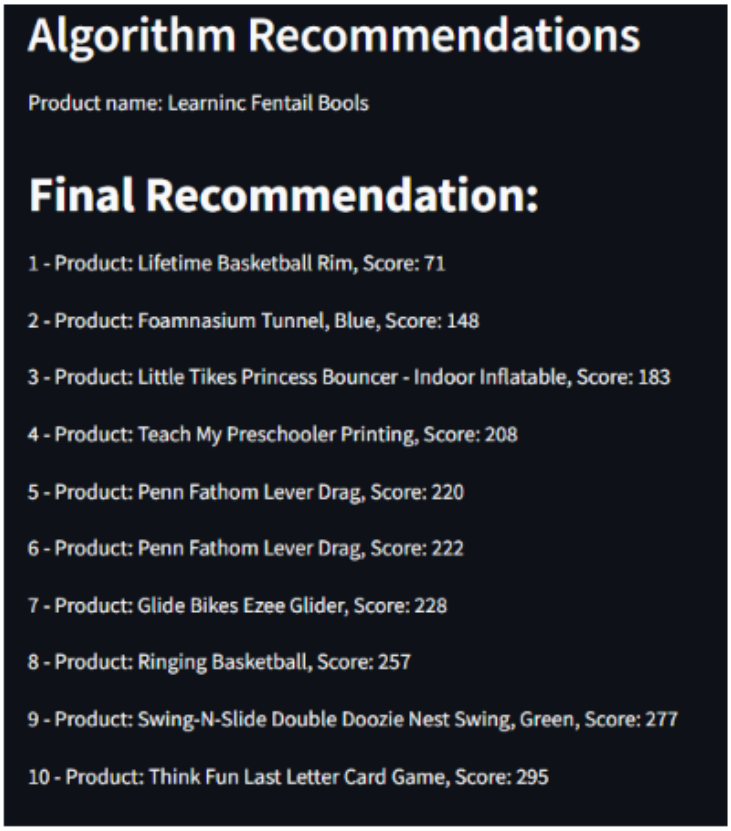
Seção de Input de 2+ Palavra escritas incorretamente

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algortimo 1 para o produto "Learninc Fentail Bools"



	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Sendo 1 como Péssimo e 5 como ótimo, avalie a recomendação do Algoritimo 2 para o produto "Learninc Fentail Bools" (a escrita está errada, deveria ser "Learning Fantail Books"):



	1	2	3	4	5	
Péssimo	◯	◯	◯	◯	◯	Ótimo

Figura 56 – Cenário de Teste 10 do Sistema de Recomendação