UNIVERSIDADE FEDERAL DE JUIZ DE FORA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

MARIA LUIZA DANTAS COTRIM

ANÁLISE COMPARATIVA DA APLICAÇÃO DOS ALGORITMOS DE ASSOCIAÇÃO APRIORI E FP-GROWTH EM BASE DE DADOS DO SETOR COSMÉTICO

JUIZ DE FORA
2024
MARIA LUIZA DANTAS COTRIM

ANÁLISE COMPARATIVA DA APLICAÇÃO DOS ALGORITMOS DE ASSOCIAÇÃO APRIORI E FP-GROWTH EM BASE DE DADOS DO SETOR COSMÉTICO

Trabalho de Conclusão de Curso apresentado a Faculdade de Engenharia da Universidade Federal de Juiz de Fora, como requisito parcial para a obtenção do título de Engenheiro de Produção.

Orientador: Dsc., Mariana Paes da Fonseca

JUIZ DE FORA

2024

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Cotrim, Maria Luiza Dantas.
ANÁLISE COMPARATIVA DA APLICAÇÃO DOS ALGORITMOS DE ASSOCIAÇÃO APRIORI E FP-GROWTH EM BASE DE DADOS DO SETOR COSMÉTICO / Maria Luiza Dantas Cotrim. -- 2024.
55 p.

Orientadora: Mariana Paes da Fonseca
Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Juiz de Fora, Faculdade de Engenharia, 2024.

1. Ciência de Dados. 2. Algoritmos. 3. Associação. I. Fonseca, Mariana Paes da, orient. II. Título.

MARIA LUIZA DANTAS COTRIM

ANÁLISE COMPARATIVA DA APLICAÇÃO DOS ALGORITMOS DE ASSOCIAÇÃO APRIORI E FP-GROWTH EM BASE DE DADOS DO SETOR COSMÉTICO

Trabalho de Conclusão de Curso apresentado a Faculdade de Engenharia da Universidade Federal de Juiz de Fora, como requisito parcial para a obtenção do título de Engenheiro de Produção.

Aprovada em 2 de setembro de 2024.

BANCA EXAMINADORA

Fonseca

D. Sc. (Mariana Paes da Fonseca) (Orientadora)
Universidade Federal de Juiz de Fora

D. Sc., (Roberto Malheiros Moreira Filho)
Universidade Federal de Juiz de Fora

D. Sc., (Antonio Angelo Missiaggia Picorone)
Universidade Federal de Juiz de Fora

AGRADECIMENTOS

A realização desse trabalho de conclusão de curso não teria sido possível sem o apoio de pessoas essenciais. Agradeço, primeiramente, à minha família, que tornou alcançável o meu sonho de me formar como engenheira de produção e me apoiou durante toda a trajetória de graduação. Aos meus pais, pelo apoio incondicional e aos meus irmãos pela compreensão nos momentos de ausência.

Também agradeço a todos os professores que contribuíram na minha formação e que compartilharam tanto conhecimento comigo ao longo desses cinco anos. Em especial à minha professora e orientadora Mariana, por toda a sua paciência, dedicação e orientação.

Aos meus amigos da universidade que estiveram comigo nos momentos desafiadores e também nas comemorações e alegrias. Agradeço, especialmente, ao Lucas Araújo, por toda a parceria que tivemos desde o início do curso, nós nos apoiamos em cada prova, trabalho e atividade extracurricular que desempenhamos nesses anos. As minhas amigas da cidade onde eu nasci e que tanto me deram suporte nessa trajetória mesmo com a distância.

Por fim, à empresa júnior Mais consultoria, que foi para mim uma verdadeira escola sobre união, liderança e resiliência. Agradeço por todas as amizades que fiz lá e guardarei com carinho, Ramon e Rodrigo, por toda a conexão que temos e por todos os desafios que vivemos juntos, Lavínia por ter me ensinado tanto, Mariana por ter sido uma mentora e todos os demais membros que eu tive contato lá.

A todos vocês, meu profundo agradecimento.

RESUMO

A era digital e a quantidade massiva de dados que a humanidade produz faz com que estes sejam considerados ativos estratégicos. Sob essa ótica, a Ciência de Dados, disciplina que gera *insights* a partir desse ativo, apoia a transformação de indústrias, empresas e organizações com sua capacidade de gerar vantagem competitiva e aumentar produtividade. É nesse contexto que o presente trabalho se propõe a estudar a ciência de dados à luz de técnicas de associação, por meio dos algoritmos Apriori e *FP-Growth.* Dessa forma, a pesquisa tem como objetivo comparar o desempenho dos dois algoritmos em um caso específico e também gerar *insights* relevantes sobre os dados. Esse estudo foi feito por meio da extração de uma base de dados de venda do setor cosmético e manipulação em linguagem Python, aplicando o algoritmo Apriori e *FP-Growth* no *Jupyter Notebook*, uma aplicação *web* de código aberto. Foram descobertos padrões relevantes, como a preferência dos clientes pela compra de *Face Serums* seguido de *Eye Creams and Treatments.* Em relação ao desempenho dos algoritmos, o Apriori teve um desempenho superior em tempo de execução quando comparado ao *FP-Growth,* enquanto que as demais métricas (número de regras de associação geradas e desempenho quando dividimos a base) os dois tiveram desempenho similar. Este trabalho não só evidencia nuances importantes sobre desempenho e aplicabilidade das técnicas, mas também oferece *insights* práticos para empresas melhorarem suas estratégias de *marketing* e vendas, promovendo uma abordagem mais personalizada.

Palavras-chave: Ciência de Dados. Algoritmos. Associação.

ABSTRACT

The digital age and the massive amount of data humanity produces make data a strategic asset. From this perspective, Data Science, a discipline that generates insights from this asset, supports the transformation of industries, companies, and organizations with its ability to create competitive advantage and increase productivity. In this context, the present work aims to study data science through the lens of association techniques, using the Apriori and FP-Growth algorithms. Thus, the research aims to compare the performance of the two algorithms in a specific case and also to generate relevant insights from the data. This study was conducted by extracting a sales database from the cosmetics sector and manipulating it in Python, applying the Apriori and FP-Growth algorithms in Jupyter Notebook, an open-source web application. Relevant patterns were discovered, such as customers' preference for buying Face Serums followed by Eye Creams and Treatments. Regarding the algorithms' performance, Apriori had superior execution time compared to FP-Growth, while in other metrics (number of association rules generated and performance when splitting the database), both had similar performance. This work not only highlights important nuances about the performance and applicability of the techniques but also offers practical insights for companies to improve their marketing and sales strategies, promoting a more personalized approach.

Keywords: Data Science. Algorithms. Association.

LISTA DE FIGURAS

Figura 1: Bibliotecas de Python utilizadas na análise exploratória 29
Figura 2: Importação dos arquivos e concatenação 30
Figura 3: Mesclagem das bases de dados 30
Figura 4: Informações da tabela 31
Figura 5: Estatísticas descritivas da coluna de preços 32
Figura 6: Criação do histograma dos preços 32
Figura 7: Visualização do histograma de preços 33
Figura 8: Criação de um boxplot com a coluna de preços 33
Figura 9: Boxplot com a coluna de preços 33
Figura 10: Código de criação do top 10 marcas 34
Figura 11: Gráfico do top 10 marcas mais vendidas 34
Figura 12: Código para criar gráfico de vendas ao longo dos anos 35
Figura 13: Gráfico de vendas por ano 35
Figura 14: Criação do top 10 produtos mais recomendados 36
Figura 15: Gráfico do top 10 produtos mais recomendados 36
Figura 16: Código da criação dos gráficos de produtos mais recomendados por tipo de pele 37
Figura 17: Gráfico do top 5 produtos mais vendidos para pele seca 37
Figura 18: Gráfico do top 5 produtos mais vendidos para pele mista 38
Figura 19: Gráfico do top 5 produtos mais vendidos para pele normal 38
Figura 20: Gráfico do top 5 produtos mais vendidos para pele oleosa 38
Figura 21: Criação do top 20 categorias mais vendidas 39
Figura 22: Criação do top 20 categorias mais vendidas 39
Figura 23: Remoção de colunas 40
Figura 24: Informações da base após remoção das colunas 40
Figura 25: Código para remoção de linhas com valores nulos 41
Figura 26: Resultado da remoção de linhas com valores nulos 41
Figura 27: Remoção de valores duplicados 42
Figura 28: Resultado da remoção de valores duplicados 42
Figura 29: Importação da biblioteca e algoritmo 42
Figura 30: Tabela de transações 43
Figura 31: Algoritmo A Priori 43
Figura 32: Regras de associação 43

Figura 33: Código para exibir os resultados 44
Figura 34: Conjunto de categorias frequentes de suporte de 1% com uso do Apriori 44
Figura 35: Regras de Associação geradas com o uso do Apriori 44
Figura 36: Execução do algoritmo FP-Growth 45
Figura 37: Conjunto de itens frequentes gerados pelo FP-Growth 45
Figura 38: Regras de associação geradas pelo FP Growth 46
Figura 39: Função para medir tempo e medição do tempo do algoritmo Apriori 46
Figura 40: Medição do tempo do algoritmo FP-Growth 47
Figura 41: Código para criação do gráfico de barras da comparação entre algoritmos 47
Figura 42: Gráfico de barras da comparação entre algoritmos 47
Figura 43: Código para dados 1 e 2 do Apriori 49
Figura 44: Código para dados 1 e 2 do FP-Growth 49
Figura 45: Regras do Apriori encontradas no conjunto de 1 50
Figura 46: Regras do Apriori encontradas no conjunto de 2 50
Figura 47: Regras do FP-Growth encontradas no conjunto de 1 51
Figura 48: Regras do FP-Growth encontradas no conjunto de 2 51

LISTA DE QUADROS

Quadro 1: Tipos de análises e suas tarefas 24

LISTA DE TABELAS

Tabela 1: Tempos de execução entre Apriori e FP-Growth 48

LISTA DE ABREVIATURAS, SIGLAS E SÍMBOLOS

CD - Ciência de Dados

DM - *Data Mining*

TI - Tecnologia da Informação

SUMÁRIO

1. INTRODUÇÃO 14
1.1 CONSIDERAÇÕES INICIAIS 14
1.2 JUSTIFICATIVA 15
1.3 ESCOPO DO TRABALHO 16
1.4 ELABORAÇÃO DOS OBJETIVOS 17
1.5 DEFINIÇÃO DA METODOLOGIA 18
1.6 ESTRUTURA DO TRABALHO 19
2. CIÊNCIA DE DADOS 20
2.1 CONCEITOS DE CIÊNCIA DE DADOS 20
2.2 ETAPAS DE DATA MINING E TIPOS DE ANÁLISES SUPERVISIONADA E NÃO SUPERVISIONADA 22
2.2.1 TIPOS DE ANÁLISE SUPERVISIONADA E NÃO SUPERVISIONADA 24
2.3 ALGORITMOS (FERRAMENTAL MATEMÁTICO, ESTATÍSTICO E COMPUTACIONAL) 24
2.3.1 APRIORI 25
2.3.2 FP-GROWTH 27
2.3.3 RECURSOS COMPUTACIONAIS 27
3. DESENVOLVIMENTO 29
3.1 PROCESSO DE EXTRAÇÃO DOS DADOS 29
3.2 ANÁLISE EXPLORATÓRIA 29
3.3 LIMPEZA DA BASE DE DADOS 40
3.4 APLICAÇÃO DOS ALGORITMOS 42
3.5 COMPARAÇÃO DOS ALGORITMOS 46
4. CONCLUSÃO 53
REFERÊNCIAS 54
ANEXO A – TERMO DE AUTENTICIDADE 57

1. INTRODUÇÃO

1.1 CONSIDERAÇÕES INICIAIS

De acordo com a Abihpec (2023), a indústria brasileira de Higiene Pessoal, Perfumaria e Cosméticos é o quarto maior mercado consumidor do mundo com participação de US$26,9 bilhões. Para além do cenário atual, o mercado cosmético continua a apresentar um forte potencial de crescimento nos próximos anos, impulsionado pelas mudanças nas preferências dos consumidores e pela adoção crescente de tecnologias avançadas para personalização de produtos e melhoria da experiência do cliente (Vale *et al.*, 2023). Em vista disso, empresas do país necessitam de um foco atento em sua estratégia competitiva futura, o que inclui o alcance de públicos cada vez mais diversos e o entendimento sobre o perfil de consumo deles.

Vale destacar que, ao observar as tendências globais e o posicionamento de grandes marcas, o setor busca uma segmentação constantemente mais diversa, a fim de alcançar novos consumidores. De acordo com o Abihpec e Sebrae (2023), mais do que abraçar e celebrar a diversidade, é preciso entender os reais desejos e necessidades desse novo consumidor e aprender a melhor forma de se comunicar com ele. Nesse sentido, o setor apresenta a tendência da inclusão de segmentos que possuem pouca representatividade na compra de cosméticos. Por exemplo, mesmo que seja um mercado geralmente ligado ao público feminino, este segmento apresentou um crescimento da participação do público masculino. Sob esse viés, produtos e serviços para o público masculino são alvo de um interesse cada vez maior, e os consumidores estão mais abertos a experimentar esses itens (Sebrae, 2023).

O uso das tecnologias da informação para descrever esses comportamentos de mercado no cenário apresentado se tornou uma estratégia essencial para as empresas. Por isso, é válido destacar como as empresas estão aplicando a Tecnologia da Informação (TI) em suas estratégias de negócios. Em seu estudo, Vale *et al*. (2023) faz uma análise de diversas pesquisas que relacionam a inteligência artificial com o mercado de cosméticos, como no caso da lista de empresas: MAC, Estée Lauder, Clinique, L'Oréal e Neutrogena que utilizam Inteligência Artificial (IA) para competir no mercado global e antecipar tendências emergentes. As gigantes de cosméticos, com o objetivo de ganho de competitividade, utilizam o aprendizado de máquina e outras ferramentas da Ciência de Dados como centro de suas decisões de negócio. No trabalho de Valley *et al.*, (2022), autor de um dos artigos

analisados, as principais discussões do texto foram sobre a crescente adoção de IA na indústria de cosméticos e cuidados pessoais, bem como as possíveis aplicações da IA para melhorar a experiência do consumidor (Vale *et al.*, 2023).

Em vista dessas tendências da aplicação de TI nas empresas e de um mercado segmentado e diverso, a presente pesquisa visou entender, para além de uma visão genérica do mercado de cosméticos, como aplicar as ferramentas de Data Mining com objetivo de definir detalhadamente públicos de consumo do setor e qual o perfil de compra desses.

Para contextualização do Data Mining (DM), foi descrito o universo da Ciência de Dados, que é essencial para o entendimento dessa ferramenta. Nesse sentido, para Rautenbach; Kock; Grobler (2022) o termo Data Science ainda recai sobre o fenômeno da "buzzword", já que existe muita confusão sobre o conceito. Por isso, a presente pesquisa também propôs delimitar o perímetro desse escopo, para, assim, apresentar mais detalhadamente como o *Data Mining* se encaixa nesse contexto. A escolha da ferramenta se dá porque DM é provavelmente o conceito mais próximo de Ciência de Dados (CD) (Provost; Fawcett, 2013). Sob essa ótica, existem outras técnicas também valiosas que podem ser utilizadas na era do Big Data, todavia, DM é uma das alternativas mais efetivas para extrair conhecimento de um grande volume de dados, descobrindo relações ocultas, padrões e gerando regras para prever e correlacionar dados, que pode auxiliar as instituições a tomada de decisão mais veloz , ou até alcançar um maior grau de confiança (Galvão; Marin, 2009).

Ademais, existem diversas técnicas que podem ser utilizadas no universo de CD. Assim, a ciência de dados é a junção de estatística, informação, matemática, tecnologia, informática, comunicação e gestão para transformar dados em insights e decisões (Cao, 2018). Nesse sentido, o trabalho apresenta em sua revisão quais algoritmos e métodos podem ser utilizados em cada uma delas.

1.2 JUSTIFICATIVA

Com a vasta quantidade de dados disponíveis no século XXI, empresas em quase toda indústria ao redor do mundo estão focadas em explorar dados para vantagem competitiva (Provost; Fawcett, 2013). A relevância do trabalho para a Engenharia de Produção está na junção da expertise em gestão com o conhecimento técnico em métodos estatísticos, computacionais e matemáticos, a fim de retirar conhecimento útil para os negócios e torná-los mais competitivos, através de longas bases de dados.

Então, realizou-se um estudo sobre a relevância do *Data Mining* e como se dá a sua aplicação especificamente na indústria cosmética, a fim de apresentar os *insights* que a ferramenta pode gerar ao ser aplicada nesse contexto. A ênfase é dada na indústria cosmética pois essa continua a apresentar um forte potencial de crescimento nos próximos anos, impulsionada pelas mudanças nas preferências dos consumidores e pela adoção crescente de tecnologias avançadas para personalização de produtos e melhoria da experiência do cliente (Vale *et al.*, 2023). Sob essa ótica, esse trabalho torna-se relevante para acrescentar à literatura existente, mostrando como aplicar essas ferramentas no setor escolhido.

A autora tem interesse na área de Ciência de Dados por utilizar o pensamento analítico e pela crescente relevância da disciplina no mercado de trabalho, o que gera um diferencial em sua formação. Para Provost; Fawcett (2013), companhias entre diversos setores perceberam que elas precisam contratar mais cientistas de dados, e que esses devem ser capazes de ver problemas de negócios pela perspectiva dos dados.

1.3 ESCOPO DO TRABALHO

Diante do que foi previamente exposto, o presente trabalho utiliza conceitos da Ciência de Dados como embasamento teórico. A partir disso, adota ferramentas de *Data Mining*, inclusa no universo de CD, a fim de apresentar a usabilidade delas na prática e exemplificar como podem gerar vantagem competitiva para as empresas.

Dessa forma, a pesquisa focaliza em uma empresa que comercializa cosméticos e busca analisar uma base de dados que fornece informações sobre o seu histórico de vendas no período analisado. Essa base de dados foi extraída em agosto de 2023 de um site de dados abertos, onde é possível escolher bases filtrando por nível de usabilidade. Dessa forma, foi selecionada a base de produtos cosméticos com maior usabilidade e também maior número de comentários no *site* sobre a aplicação de algoritmos, porque os comentários de outras pessoas que já usaram a base auxiliam na descoberta de possíveis análises que podem ser realizadas. Em relação ao recorte temporal, a base contém informações de 2008 até março de 2023 e no trabalho foi usado esse histórico, com objetivo de estudar toda a base extraída. Esse histórico de vendas contém informações sobre o que foi comprado, avaliação do produto e perfil do cliente. Entretanto, a base possui limitações, pois não possui todos os campos 100% preenchidos, já que nem todos os clientes detalham suas informações pessoais na compra.

Assim, foram removidas as transações que possuem informações relevantes em branco. Nesse contexto, pretende-se descobrir quais algoritmos de mineração de associação são mais eficientes no caso estudado, para responder questões como: quais são as principais regras de associação encontradas no conjunto de dados? Que informações geradas pelas regras podem ser usadas para a tomada de decisão estratégica nas empresas? Qual a diferença de desempenho entre os algoritmos para o caso escolhido? Em relação ao recorte geográfico, os dados referentes ao estudo se limitam ao histórico dos Estados Unidos.

Para realizar tal trabalho, foi utilizada a linguagem Python através do Jupyter Notebook, escolhido por conta de sua facilidade no uso e por ser um sistema de código aberto. A base de dados foi inserida no sistema e tratada com dois algoritmos de mineração de dados: Apriori e *FP-Growth*. Após a execução dos algoritmos, a pesquisa também propõe uma análise de desempenho de ambos, com objetivo de definir as oportunidades de desenvolvimento, como melhorias no código e aplicações futuras em outras bases. Além disso, também busca mensurar tempo de execução e acurácia quando divide-se a base entre dois conjuntos. A partir dessas análises, ao final foi possível realizar uma comparação entre o desempenho de ambos algoritmos utilizados, para que se definisse qual deles é o mais indicado para o objetivo principal estabelecido no trabalho.

1.4 ELABORAÇÃO DOS OBJETIVOS

O trabalho teve como objetivo geral avaliar a aplicação dos algoritmos de associação Apriori e FP-Growth em uma base de dados de cosméticos, por meio da ferramenta *Data Mining*, de modo a descobrir questões evidenciadas no escopo.

Além disso, por se tratar de uma pesquisa que aplica um algoritmo, existem tarefas menores dentro das macro etapas. Por conta dessa segmentação, pode-se citar os seguintes objetivos específicos:

-Aplicar e avaliar os algoritmos de *Data Mining* escolhidos;

-Desenvolver um programa em Python composto dos algoritmos citados e que possa ser usado em novas bases de dados;

-Traduzir o resultado das análises em *insights* sobre o padrão de compra dos produtos;

-Comparar os algoritmos quanto à eficiência, aplicação e às oportunidades futuras, para definir qual deles tem melhor aplicabilidade no caso estudado.

1.5 DEFINIÇÃO DA METODOLOGIA

A fim de atingir os objetivos propostos e satisfazer as questões levantadas anteriormente, o trabalho foi realizado através de pesquisas que se valem de ferramental estatístico, computacional e matemático.

Por se tratar de uma pesquisa aplicada, é necessário descrever quais as etapas que serão seguidas e o detalhamento delas. Abaixo segue de maneira sequencial quais as etapas propostas:

A. Escolha e extração da base de dados: a escolha se deu através do *website* https://www.kaggle.com/, onde existem inúmeras bases de dados gratuitas disponibilizadas para uma comunidade que pratica e compartilha conhecimento sobre *Machine Learning* e Inteligência Artificial. Primeiramente, foi feita uma pesquisa na busca do site de bases existentes que trouxessem informações sobre o ramo de cosméticos. Após estudar as bases disponíveis, foi escolhida aquela com nota máxima em usabilidade (o site classifica de 0 a 10 a usabilidade), e também que fosse robusta o suficiente para retirar informações valiosas. O histórico de vendas da empresa contém informação sobre mais de 8.000 produtos vendidos no site online da empresa, além de 1.000.000 de avaliações de clientes sobre esses produtos.
B. Pré-processamento dos dados, que envolve a preparação, limpeza e tratamento: A base foi baixada e manipulada adotando a biblioteca Pandas para que sejam removidas informações faltantes ou errôneas, pois alguns registros possuem células vazias em colunas específicas. Além disso, algumas colunas foram renomeadas, filtros foram realizados e algumas colunas que não foram utilizadas nas análises foram removidas, para auxiliar na eficiência do programa.
C. Análise exploratória: Nessa etapa, por meio da biblioteca Pandas, Matplotlib e Numpy, foram gerados gráficos e tabelas a partir da base extraída com o objetivo de entender informações que não são triviais a primeiro olhar. Nesse sentido, foram gerados gráficos sobre a quantidade vendida dos produtos mensalmente, tabelas que mostram a média de avaliação dos produtos, gráficos dos produtos mais comprados por pessoas de um tipo ou tom de pele específico, entre outras análises que a autora pode perceber relevante ao explorar os dados.

D. Análise preditiva: Foram aplicados os algoritmos de *Data Mining*, o Apriori e o *FP-Growth*, utilizando da linguagem Python de programação. Para que fosse possível a aplicação desses algoritmos, o trabalho usou as bibliotecas Pandas e Mlxtend.

1.6 ESTRUTURA DO TRABALHO

O presente trabalho foi dividido em 5 capítulos, onde este, o primeiro, foi desenvolvido os aspectos gerais da pesquisa.

No segundo capítulo, é apresentada a revisão bibliográfica do tema, onde são abordados conceitos sobre Ciência de Dados e *Data Mining*, além de descrever detalhadamente o ferramental matemático, estatístico e computacional para aplicação dos conceitos.

O terceiro capítulo corresponde ao desenvolvimento do trabalho, ou seja, quais são as etapas realizadas e como se deu a aplicação do ferramental descrito no capítulo 2 em uma base de dados extraída previamente.

O quarto capítulo apresenta os resultados da pesquisa, que entende-se pela conclusão da análise preditiva após a aplicação dos algoritmos, além de comparar o desempenho dos algoritmos usados.

O quinto e último capítulo é uma conclusão realizada pela autora dos resultados descritos no capítulo 4, evidenciando principais aprendizados na condução da pesquisa e direcionamentos futuros sobre o uso dos algoritmos.

2. CIÊNCIA DE DADOS

2.1 CONCEITOS DE CIÊNCIA DE DADOS

O século 21 é marcado pela quantidade expressiva de dados que a humanidade produz. De acordo com o *International Data Corporation* (IDC, 2018), entre o ano de 2018 até 2025 estima-se que crescerá em mais de 500% na quantidade de dados armazenados, saindo de 33 zettabytes para 175 zettabytes (um zettabyte corresponde a um sextilhão de bytes). Nesse sentido, é notória a era do *Big Data, Analytics* e Ciência de Dados (CD) (Cao, 2018).

Sob essa ótica, o dado pode ser considerado um novo ativo estratégico. De acordo com Cao (2018), ele é estratégico porque pode determinar o futuro da economia, ciência, tecnologia e, possivelmente, de tudo no mundo. Isso ocorre porque o dado suportou diversas transformações em indústrias, empresas e organizações com a sua capacidade de gerar vantagem competitiva e aumentar a produtividade. Nesse contexto, a Ciência de Dados surge como uma das soluções para gerenciar esse ativo e extrair o máximo de informação e conhecimento útil dele. Entretanto, o termo ciência de dados ainda é frequentemente discutido, caindo no fenômeno da "*buzzword*", já que existe muita confusão da extensão do seu escopo (Rautenbach; Kock; Grobler, 2022). Ele pode ser confundido com a estatística, ciência da informação e diversos outros conceitos como: *Big Data*, análise de dados, inteligência artificial, *Data Decision Making*, etc.

Em primeiro momento, é preciso diferenciar Ciência de Dados de Estatística, pois há diversas considerações sobre as similaridades em seus escopos. A estatística é a ciência que se preocupa com a organização, descrição, análise e interpretação de dados experimentais (Neto, 2002). Já a Ciência de Dados trata de estudar o dado em todo o seu ciclo de vida, da produção ao descarte, além de incluir várias outras ciências, modelos, tecnologias, processos e procedimentos relacionados ao dado (Amaral, 2016). Assim, é possível perceber que a ciência de dados é mais abrangente e inclui mais técnicas e tecnologias que não são necessariamente utilizadas pela estatística em sua totalidade. Da mesma forma, Cao (2018) também afirma a abrangência desse escopo, porém complementa que os diversos desafios da economia até podem ser resolvidos por estatística, matemática ou inteligência artificial, porém eles não podem ser acomodados efetivamente sem criar uma nova disciplina. Assim, a ciência de dados é a junção de estatística, informação, matemática, tecnologia, informática, comunicação e gestão para transformar dados em insights e decisões (Cao, 2018).

Ademais, existem diversos outros conceitos que estão ligados à Ciência de Dados e que também precisam ser definidos para entender essa disciplina. Para efeitos deste trabalho foram esclarecidos, de maneira não exaustiva, os termos: *Big Data, Data Mining, Data Analytics e Data Decision Making* (Tomada de Decisão Orientada a Dados).

- *Big Data*: Diz respeito a bases de dados que são grandes e/ou complexas demais para serem tratadas por métodos tradicionais;
- *Data Analytics*: Corresponde a todos os processos, ferramentas e tecnologias que auxiliam a descoberta de informação útil em dados;
- *Data Decision Making*: Tomar decisões baseadas em dados, ao invés de usar a intuição humana (Cao, 2018);
- *Data Mining*: Possivelmente o conceito mais próximo da ciência de dados (Provost; Fawcett, 2013). Refere-se a técnicas ou processos para buscar padrões em bases de dados e, assim, extrair conhecimento útil. Ela é menos abrangente que a ciência de dados, pois não envolve áreas como gestão e comunicação.

Definidos os conceitos que permeiam o universo da Ciência de Dados, é essencial entender a sua importância no contexto econômico. Mesmo que sua aplicação esteja presente em diversas áreas, as indústrias e empresas, particularmente, transformaram a sociedade com o uso dessa disciplina no cerne de seus negócios. Ela é o novo motor de inovação para produtividade (Cao, 2018) e, com a grande quantidade de dados disponíveis, empresas em quase todo setor estão focadas em explorar dados para vantagem competitiva (Provost; Fawcett, 2013). Dessa forma, as empresas estão em uma "corrida" para alcançar uma estrutura de gestão e análise de dados eficiente, que forneça insumos para uma tomada de decisão precisa e acurada.

Dentre os setores, a CD pode ser aplicada em *marketing*, anúncios online, pontuação de crédito, negociação financeira, recomendação de produto, detecção de fraude, etc (Provost; Fawcett, 2013). Em marketing, a CD produz campanhas com maior probabilidade de chamar atenção do consumidor, os anúncios poderão ter mais cliques, as negociações financeiras podem ter seus riscos reduzidos através de análises de predição, produtos podem ter suas vendas elevadas ao serem recomendados para clientes que os desejam por meio da análise de histórico de compras e fraudes podem ser detectadas com a análise de dados bancários.

Nesse sentido, ciência de dados é uma ferramenta poderosa e sua aplicação gera resultados palpáveis para as organizações em vários setores, desde resultados financeiros até ganhos de eficiência e produtividade. Todavia, ainda existem limitações para essa disciplina,

visto que ela não pode ser utilizada em todas as situações. Um fundamento trivial é que, se você olhar para uma base de dados o suficiente, você irá encontrar algo - mas pode não ser generalizado para além dos dados que está olhando (Provost; Fawcett, 2013). Existem uma série de etapas e considerações para que seja possível tomar uma decisão a partir de dados e não haja equívocos ao formular predições. Por isso, é necessário entender contexto e realidade dos dados estudados e seguir as etapas necessárias para gerar máximo proveito da ferramenta.

2.2 ETAPAS DE *DATA MINING* E TIPOS DE ANÁLISES SUPERVISIONADA E NÃO SUPERVISIONADA

A descoberta de conhecimento e tomada de decisão através de longas bases de dados (*Data Decision Making*) passa, geralmente, por etapas principais: pré-processamento, *Data Mining* e resultados. O pré-processamento tem como objetivo organizar e tratar os dados, visto que dificilmente eles estão na configuração ideal para realizar a mineração. Sob essa ótica, as bases podem possuir informações desnecessárias para análises, algumas variáveis com a configuração incorreta ou informações inconsistentes, o que faz ser necessário realizar um tratamento para ajustar as incoerências. Para Mariano *et al*. (2021), existem três tipos de problemas encontrados em bases de dados sem tratamento: incompletude, ruído e inconsistência. A incompletude diz respeito a objetos faltantes, enquanto o ruído são variações não explicadas na amostra. Já a inconsistência são casos onde há violação do domínio ou informações que remetem os mesmos valores, porém com configuração diferente.

Para realizar essa etapa de limpeza, o pré processamento geralmente passa pelas seguintes fases (Goldschmidt; Passos, 2005):

a) Seleção de dados: Tem como objetivo escolher os dados que entrarão no processo de mineração posterior. Por exemplo, em uma base de dados do histórico de compra de produtos, a base pode fornecer as informações de CPF do cliente, produto comprado, quantidade comprada, valor e data da compra. Para realizar uma análise, a informação do CPF do cliente pode ser descartada, pois não será possível retirar *insights* com esse dado. Por isso, seria mais proveitoso selecionar apenas os dados de produto, quantidade, valor e data da compra.
b) Limpeza dos dados: Essa etapa assegura a qualidade dos dados, através da remoção de informações errôneas, ausentes ou inconsistentes nas bases de dados. No exemplo do histórico de compra de produtos, caso algum dos clientes na lista esteja com a

informação de produto comprado faltante, é importante remover esse caso da base, pois ele não agrega valor nas análises futuras.

c) Codificação dos dados:Os dados devem ser codificados para ficarem numa forma que possam ser usados como entrada de algoritmos de Mineração de Dados.

d) Enriquecimento dos dados: Conseguir mais informação que possa ser agregada aos registros existentes, enriquecendo dados, seja buscando bases de dados externas ou complementação de dados.

Além das etapas citadas pelo autor acima, Mariano *et al*. (2021) complementam com as fases de integração, discretização e transformação. A etapa de integração se refere ao tratamento de dados inconsistentes, ou seja, integrar dados que representam a mesma informação. Um exemplo é no caso de uma empresa de cosméticos que possui duas bases de dados de franqueados diferentes e as duas representam a informação da cidade da venda de modos distintos: a primeira escreve “Rio de Janeiro” e a segunda “RJ”. A discretização pode ser resumida como a conversão de dados numéricos em dados categóricos, como no uso de histogramas para representação de informação. Já na etapa de transformação é necessário análise das colunas das bases de dados e a formatação do tipo de dado que está contido nelas, como no caso da mesma empresa cosmética que ao cadastrar o telefone dos clientes o faz de duas maneiras: colocando o DDD sem parênteses e com parênteses.

É válido destacar nessa etapa, além de técnicas básicas de pré-processamento, métodos que envolvem o *Text Mining* são relevantes para as análises, principalmente na aplicação de avaliação de clientes, pois envolvem campos textuais longos. Para Vijayarani *et al.* (2016), são três métodos de pré-processamento para a mineração de textos:

a) Extração: Também conhecida como tokenização, esse método visa simplificar os textos de um campo em uma única palavra, chamada de *token*.

b) *Stop Words*: Remover palavras do vocabulário que são as divisões da linguagem, como pronomes, preposições, artigos. As “palavras de parada” mais comuns são “de”, “e”, “a”, entre outras.

c) *Stemming*: Diz respeito a encontrar as raízes ou forma base das palavras, removendo os sufixos ou prefixos. Nesse método as palavras devem ser semanticamente relacionadas, caso contrário o método não pode ser aplicado.

Ao realizar a organização dos dados por meio desse processamento, a próxima fase é a Mineração de Dados, a qual há mais complexidade em tarefas e técnicas a serem aplicadas. É importante distinguir o que é uma tarefa e uma técnica de Mineração de Dados (Amo, 2004).

A tarefa diz respeito à identificação de que tipo de padrões queremos encontrar na base de dados, enquanto as técnicas são os métodos que mostram como encontrar esses padrões. As tarefas podem ser classificadas com base em duas categorias: análise supervisionada e análise não supervisionada.

2.2.1 TIPOS DE ANÁLISE SUPERVISIONADA E NÃO SUPERVISIONADA

A análise supervisionada é aquela em que o objetivo da análise é utilizar uma base de dados para prever a ocorrência de certo atributo em casos onde eles ainda não foram vistos (Bramer, 2007). Tal análise é utilizada quando os dados são rotulados, ou seja, o atributo a ser estudado já foi designado previamente para cada caso na base de dados. Em contraponto, os dados que não têm nenhum atributo especialmente designado são chamados de não rotulados, e a mineração de dados nessas situações é conhecida como análise não supervisionada (Bramer, 2007).

As principais tarefas de análise supervisionada e não supervisionada são apresentadas na tabela 1 abaixo:

Quadro 1: Tipos de análises e suas tarefas

Tipo de análise	**Tarefas de *Data Mining***
Análise supervisionada	Regressão
	Classificação e Predição
Análise não supervisionada	Agrupamento
	Regras de associação

Fonte: Adaptado de Bramer (2007)

O presente trabalho tem como foco as análises que envolvem regras de associação, que são um tipo de análise não supervisionada. Os algoritmos desta análise têm como objetivo encontrar conjuntos de itens que aconteçam simultaneamente e de forma frequente em um banco de dados. De acordo com Goldschmidt; Passos; Bezerra (2015), uma regra de associação é formada por X→Y, onde o X é o antecedente da regra e Y é o consequente, tal que a interseção de X e Y é igual a um conjunto vazio. Considere, por exemplo, o caso de uma loja de cosméticos onde existe um conjunto (X={Hidratante}) e (Y={Sabonete}). Uma

regra de associação entre os dois conjuntos implica que a compra de hidratante pode levar a compra de sabonete: Hidratante → Sabonete.

Uma associação é considerada frequente se o número de vezes em que a união do conjunto de itens (X ∪ Y) ocorrer em relação ao número total de transações do banco de dados for superior a uma frequência mínima (denominada suporte mínimo). No caso da loja de cosméticos, uma transação pode representar a compra de um hidratante, por exemplo. Uma associação é válida se o número de vezes em que X ∪ Y ocorrer em relação ao número de vezes em que X ocorrer for superior a um valor determinado confiança mínima (Goldschmidt; Passos; Bezerra, 2015). A confiança determina a qualidade de uma regra, pois compreende o quanto a ocorrência do antecedente pode ocasionar a ocorrência do consequente. No caso do setor de cosméticos, a aplicação de regras de associação pode ser essencial para entendimento do comportamento do consumidor. Isso porque é possível entender através dessa regra quais produtos são comprados juntos, ao analisar as transações da base de dados. Muitos clientes podem escolher comprar um sabonete para o rosto e na mesma transação comprar um hidratante para o rosto, como será observado pelos resultados deste estudo. Tais tipos de resultados tornam possível gerar estratégias de marketing para induzir mais compradores a fazer o mesmo pedido.

2.3 ALGORITMOS (FERRAMENTAL MATEMÁTICO, ESTATÍSTICO E COMPUTACIONAL)

Para realizar as tarefas de mineração de dados, são utilizados diversos algoritmos em cada tipo de análise, supervisionada ou não. Como existe uma gama extensa de algoritmos e cada um é aplicado para uma necessidade específica, serão apresentados apenas os algoritmos a serem adotados neste trabalho, que são os algoritmos de associação.

2.3.1 APRIORI

O algoritmo Apriori foi o primeiro a ser utilizado e é um dos mais conhecidos para mineração por regras de associação (Agrawal; Imielinski; Swami, 2014). Ele é utilizado para encontrar, em grandes bases de dados, itens que aconteçam simultaneamente. De acordo com Goldschmidt; Passos (2005), o algoritmo passa por duas etapas:

1. Encontrar todos os conjuntos de itens frequentes que levam em consideração o suporte mínimo. Um conjunto frequente pode ser chamado de *itemset.* O

itemset é um conjunto de itens, como no caso da loja de cosméticos onde X={Hidratante}. Esses conjuntos são sempre representados da maneira *k-itemset*, onde k é o número de elementos do conjunto. O conjunto X é chamado de *1-itemset,* por conter um elemento

2. Gerar as regras de associação com base nos itens frequentes.

Primeiramente, o usuário deve estabelecer os valores de suporte e confiança. Suponha que uma regra de associação é formada pelo conjunto Y={Sabonete facial} e W={Sérum}. O suporte será o número de vezes em que o conjunto {Y} →{W} aparece na base de dados com todas as transações de vendas de produtos de cuidado para a pele. Dessa forma, o suporte é dado pela frequência desse conjunto sobre o total de transações (T):

$$\text{Suporte } (Y \rightarrow W) = (\text{frequência de Y e W}) / (\text{total de T})$$

Já a confiança é uma medida estatística que calcula a probabilidade condicional P(Y|W) de uma transação que contenha Y, dado que contém W (Mariano *et al*., 2021). No caso acima, será calculado a probabilidade de um cliente comprar sabonete facial {Y}, dado que ele comprou um sérum {W}. É possível observar a probabilidade condicional na fórmula:

$$\text{Confiança } (Y \rightarrow W) = \text{suporte } (Y \cup W) / \text{suporte } (Y)$$

Depois de estabelecer os suportes, a primeira etapa é feita de forma iterativa, ou seja, cada *itemset* é resultado da combinação do *itemset* anterior. Com os conjuntos frequentes definidos, a próxima fase é encontrar regras de associação válidas. Para isso, é necessário que as regras tenham uma confiança superior à confiança mínima definida pelo usuário (Mariano *et al*., 2021). Todas as regras válidas geram como saída as associações, podendo se analisar quais são os padrões nas transações do banco de dados.

Como é visto na pesquisa de Şimşek (2018), o trabalho com esse algoritmo se inicia criando um *dataset* com os dados de transações. Após ter essas informações, o autor aplica o algoritmo, com níveis de suporte mínimos. Para escolha do nível de suporte, que será uma métrica usada no presente trabalho, Şimşek (2018) afirma que essa métrica é escolhida com base na observação das vendas 3 a 4 vezes ao dia e avaliando se o suporte escolhido resulta em regras que condizem com a realidade observada. Tal calibração deve ser feita até encontrar um nível de suporte que imprima regras que façam sentido. Após essas definições, o autor encontrou 10 regras de associação altamente informativas, que podem auxiliar em decisões de *marketing* como: escolha de promoções, controle de inventário e campanhas de venda cruzada. Ademais, no trabalho de Bharadhwaj *et al.* (2023), foram 16 regras encontradas, e a conclusão do trabalho também foca no uso desses resultados para iniciativas como venda

cruzada, além de sugerir a aplicação no varejo e até *e-commerce,* que será o foco da presente pesquisa.

2.3.2 FP-GROWTH

O algoritmo *FP-Growth* é uma alternativa, na maioria das vezes, mais veloz que o Apriori. O algoritmo *FP-Growth* gera dados de conjuntos de itens frequentes a partir da *FP-Tree* usando o método de Dividir e Conquistar (Munfarijah; Lucia, 2020). Esse processo busca identificar conjuntos de itens frequentes sem criar candidatos, analisando e filtrando itens com um nível de suporte abaixo do mínimo. Dessa forma, os itens com suporte suficientes são ordenados por frequência, e o *FP-Growth* é capaz de rastrear as transações ao longo de um caminho da árvore, vinculando itens a vértices em uma hierarquia. Tal conjuntura, através da formação da *FP-Tree*, demonstra que, com o *FP-Growth*, o conjunto de dados ou conjunto de itens mais frequentemente ocorrentes pode ser determinado mais rapidamente, especialmente ao lidar com grandes conjuntos de dados (Wahyuningsih *et al*., 2023).

Sua eficiência é observada no estudo de Cabreira (2023), onde o Apriori teve um tempo de execução quase 5 cinco vezes maior que o *FP-Growth*. Na pesquisa de Dedy *et. al (*2023), observa-se que, na aplicação dos algoritmos de maneira tradicional, o tempo de execução do *FP-growth* foi, na verdade, menor que do Apriori. Entretanto, o estudo propõe um reajuste das métricas convencionais usadas nos algoritmos, como o "*Bi-Confidence"*, uma alternativa à métrica de confiança e que tem como objetivo reduzir o número de regras inválidas. Ao realizar uma nova mensuração através dessas novas métricas, o algoritmo *FP-Growth* obteve um tempo de execução menor.

2.3.3 RECURSOS COMPUTACIONAIS

A linguagem de programação utilizada para aplicação dos algoritmos será a Python. Guido Van Rossum a inventou no início dos anos 1990 e ela é uma linguagem de programação de alto nível e de uso geral para resolver problemas em sistemas de computadores modernos (Lambert, 2022). Ser de alto nível significa ser mais próxima da linguagem natural, enquanto uma linguagem de baixo nível é mais próxima dos conceitos do hardware.

O Python tem obtido vasta aceitação entre os profissionais de Ciência de Dados desde o começo dos anos 2000 (Netto; Maciel, 2021). Para Grus (2021), ela é a melhor opção dentre as linguagens disponíveis, pois é gratuita, relativamente simples de programar e dispõe de muitas bibliotecas úteis relacionadas ao *Data Science.* Além dessas características, Maciel (2020) também complementa que a linguagem se tornou popular pelas seguintes características:

- Eficiência: os códigos em Python podem realizar diversas tarefas mesmo com poucas linhas, o que promove alta automação;
- Comunidade ativa: A comunidade é presente e está crescendo, o que faz crescer o número de pacotes com todas as ferramentas para o desenvolvedor;
- Forte presença na academia: Vários cursos de computação já possuem disciplinas voltadas para o Python;
- Tendência: Muitas ferramentas conhecidas foram criadas em Python, o que aumenta sua popularidade e a tendência de crescer o uso da linguagem.

Sob esse viés, Python também ganha destaque por sua aplicação em *Data Mining* e *Machine Learning*. Por meio de sua comunidade ativa, bibliotecas são criadas com o fim de agregar às funcionalidades já existentes dentro da instalação padrão do Python. De acordo com Behrman (2023), as bibliotecas de terceiros citadas em seu livro tornam a linguagem dominante em Ciência de Dados, além de criar um ecossistema vibrante que mantém o código influente no mundo da programação. Por isso, serão descritas as principais bibliotecas a serem usadas na pesquisa:

- Numpy: A comunidade Numpy (2022) descreve a biblioteca que consegue prover objetos de matrizes e vetores multidimensionais, vários objetos derivados e uma variedade de rotinas para rápidas operações em matrizes.
- Pandas: O Pandas é, provavelmente, a estrutura de dados mais utilizada, de acordo com Behrman (2023). Essa biblioteca também é fundamental para a realização de ciência de dados.
- Matplotlib**:** É um pacote de gráficos 2D usado para desenvolvimento e publicação de imagens de alta qualidade em interfaces e sistemas operacionais (Hunter, 2007).
- Mlxtend: Essa biblioteca implementa uma variedade de algoritmos para aplicação de *Data Mining* e *Machine Learning* (Raschka, 2018).

3. DESENVOLVIMENTO

3.1 PROCESSO DE EXTRAÇÃO DOS DADOS

A base de dados utilizada no trabalho pertence à Sephora, a maior loja de perfumes e cosméticos do mundo, com informações de compras no e-commerce de 2008 até 2023. A base foi retirada do Kaggle, site que fornece mais de 300 mil bases de dados públicos para utilização em projetos de tecnologia e programação.

Esse conjunto de dados é encontrado em formato csv (https://www.kaggle.com/datasets/nadyinky/sephora-products-and-skincare-reviews). No endereço web foram baixadas as 4 bases de histórico de vendas e uma base de dados com informações e características dos produtos, que foram concatenadas em apenas uma base de dados. As informações foram úteis para a análise exploratória e aplicação dos algoritmos nos capítulos que se seguem.

3.2 ANÁLISE EXPLORATÓRIA

A análise exploratória tem como objetivo entender os possíveis padrões nos dados extraídos, relações entre variáveis e adquirir maior conhecimento sobre as informações contidas, para que o trabalho seja mais direcionado durante a etapa de aplicação dos algoritmos.

Após a extração da base, ela foi inserida no *Jupyter Notebook*, uma ferramenta de código aberto que combina elementos de código, visualização e documentação em um único ambiente interativo. A ferramenta possui aplicações para várias linguagens de programação, porém o trabalho irá utilizar o Python para testar os algoritmos.

Primeiramente, foram exportadas as bibliotecas que foram utilizadas durante a análise, apresentadas na figura 1:

Figura 1: Bibliotecas de Python utilizadas na análise exploratória

```
#importação das bibliotecas

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Fonte: Autora

Depois, como mostra a figura 2, foi utilizada a função *read_csv()* da biblioteca Pandas para importar as 5 bases de dados que foram salvas na pasta do projeto. As 4 bases que contém as informações de vendas de produtos possuem as mesmas colunas e dizem respeito ao mesmo conjunto de dados, só foram separadas em 4 arquivos diferentes. Por conta dessa característica, foi feita uma concatenação das 4 bases, através da função *concat* para que se tornassem apenas uma, facilitando, assim, a pesquisa.

Figura 2: Importação dos arquivos e concatenação

```
# Importar as bases
base1 = pd.read_csv('reviews_0-250.csv')
base2 = pd.read_csv('reviews_500-750.csv')
base3 = pd.read_csv('reviews_750-1250.csv')
base4 = pd.read_csv('reviews_1250-end.csv')
base5 = pd.read_csv('product_info.csv')

# Juntar as bases de dados
base_final = pd.concat([base1, base2, base3, base4], ignore_index=True)
```

Fonte: Autora

A quinta e última base chamada “product_info.csv” não possui as mesmas colunas que as outras 4, pois ela se refere a uma lista de todos os produtos vendidos no *e-commerce* e suas principais características. Primeiramente, foi necessário selecionar apenas as colunas diferentes entre a “base_final” e “product_info.csv”, por meio da função *difference()*, para que não fossem exportadas colunas duplicadas. Depois disso, foi preciso mesclar esses dados com os demais, para que se tivesse uma única base de dados com todas as colunas a serem analisadas. Tal mesclagem foi realizada por meio da função *merge()* com base na coluna “product_id” e método “*outer*”, que indica que todas as linhas dos *DataFrames* originais foram incluídas no resultado final. Dessa forma, foi possível usar novamente a função *read_csv()* para leitura dos dados concatenados e mesclados.

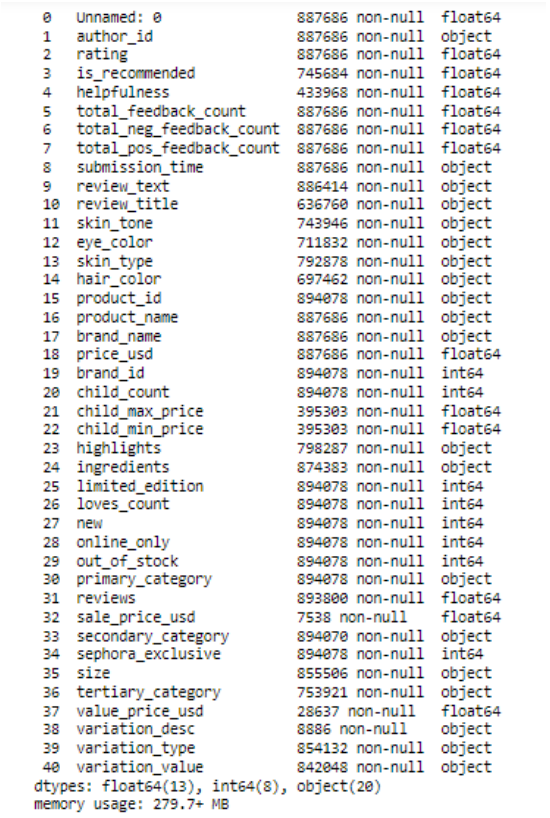
Figura 3: Mesclagem das bases de dados

```
# Encontrar colunas diferentes entre as bases
colunas_diferentes = base5.columns.difference(base_final.columns)
colunas_diferentes = list(colunas_diferentes)
colunas_diferentes.append('product_id')
print(colunas_diferentes)

# Mesclar as bases de dados final e base5 com base no nome do produto
base_final = pd.merge(base_final, base5[colunas_diferentes], on = 'product_id', how='outer')

# Salvar a base final
base_final.to_csv('base_final.csv', index=False)

base = pd.read_csv('base_final.csv')
```

Fonte: Autora

A fim de explorar a base, foram analisadas os atributos gerais da tabela com a função *info()*.

Figura 4: Informações da tabela



```
 0   Unnamed: 0               887686 non-null  float64
 1   author_id                887686 non-null  object
 2   rating                   887686 non-null  float64
 3   is_recommended           745684 non-null  float64
 4   helpfulness              433968 non-null  float64
 5   total_feedback_count     887686 non-null  float64
 6   total_neg_feedback_count 887686 non-null  float64
 7   total_pos_feedback_count 887686 non-null  float64
 8   submission_time          887686 non-null  object
 9   review_text              886414 non-null  object
 10  review_title             636760 non-null  object
 11  skin_tone                743946 non-null  object
 12  eye_color                711832 non-null  object
 13  skin_type                792878 non-null  object
 14  hair_color               697462 non-null  object
 15  product_id               894078 non-null  object
 16  product_name             887686 non-null  object
 17  brand_name               887686 non-null  object
 18  price_usd                887686 non-null  float64
 19  brand_id                 894078 non-null  int64
 20  child_count              894078 non-null  int64
 21  child_max_price          395303 non-null  float64
 22  child_min_price          395303 non-null  float64
 23  highlights               798287 non-null  object
 24  ingredients              874383 non-null  object
 25  limited_edition          894078 non-null  int64
 26  loves_count              894078 non-null  int64
 27  new                      894078 non-null  int64
 28  online_only              894078 non-null  int64
 29  out_of_stock             894078 non-null  int64
 30  primary_category         894078 non-null  object
 31  reviews                  893800 non-null  float64
 32  sale_price_usd           7538 non-null    float64
 33  secondary_category       894070 non-null  object
 34  sephora_exclusive        894078 non-null  int64
 35  size                     855506 non-null  object
 36  tertiary_category        753921 non-null  object
 37  value_price_usd          28637 non-null   float64
 38  variation_desc           8886 non-null    object
 39  variation_type           854132 non-null  object
 40  variation_value          842048 non-null  object
dtypes: float64(13), int64(8), object(20)
memory usage: 279.7+ MB
```

Fonte: Autora

A partir dos atributos, algumas análises foram feitas. Ao todo são 887.686 linhas na tabela e quase todas as colunas possuem todas as linhas preenchidas por valores não nulos. Entretanto, as colunas que possuem o número de valores não nulos menor que o número de linhas são colunas que possuem valores vazios e precisam ser tratados na limpeza da base de dados.

Posteriormente, foram realizadas análises de algumas colunas. Primeiramente, é essencial avaliar a coluna de preço dos produtos. Para tal, foi utilizada a função *describe(),* que gera estatísticas descritivas dos dados, como a média, desvio padrão, valores mínimos e máximos e os quartis. A partir dessas estatísticas mostradas na figura 5, um histograma foi criado por meio do código apresentado na figura 6.

Figura 5: Estatísticas descritivas da coluna de preços

```
print(base['price_usd'].describe())
```

```
count    887686.000000
mean         48.545258
std          41.043800
min           3.000000
25%          24.000000
50%          39.000000
75%          60.000000
max        1900.000000
Name: price_usd, dtype: float64
```

Fonte: Autora

Ao início do código, foram definidas as estatísticas descritivas a serem inseridas no gráfico, como a média, o desvio padrão, a assimetria e curtose, através das funções *mean(), std(), skew() e kurtosis()*, respectivamente. Esses valores foram concatenados em uma única variável chamada "texto", utilizada para incluir uma caixa de texto no histograma. Assim, como é percebido na figura 6, foi feito o histograma através da função hist() e, logo depois dos ajustes de tamanho e *layout*, usou-se a função *show*() para apresentar os resultados.

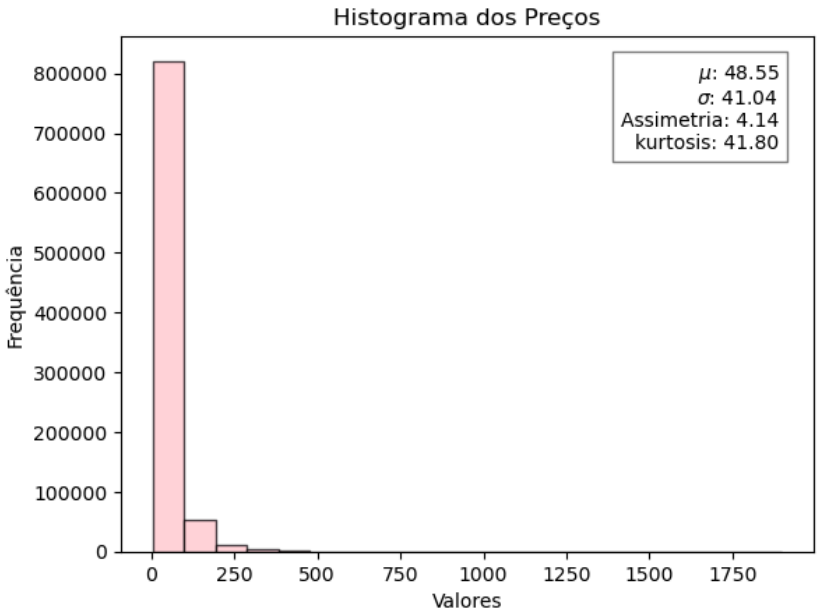
Figura 6: Criação do histograma dos preços

```
#Criando histograma para a coluna de preços
media = base['price_usd'].mean()
std = base['price_usd'].std()
skew = base['price_usd'].skew()
kurt = base['price_usd'].kurtosis()

texto = f'$\mu$: {media:.2f}\n$\sigma$: {std:.2f}\nAssimetria: {skew:.2f}\nkurtosis: {kurt:.2f}'

plt.hist(base['price_usd'], bins=20, alpha=0.7, color='pink', edgecolor='black')
plt.text(0.95, 0.95, texto, verticalalignment='top', horizontalalignment='right', transform=plt.gca().transAxes, bbox=dict(facecc
plt.title('Histograma dos Preços')
plt.xlabel('Valores')
plt.ylabel('Frequência')
plt.show()
```

Fonte: Autora

A partir do histograma da figura 7 é possível perceber que o desvio padrão é alto, além de que 75% dos valores se encontram a menos de 1 desvio padrão da média. Além disso, a assimetria de 4.14 indica uma forte assimetria positiva, ou seja, há uma concentração maior de valores abaixo da média e uma dispersão maior de valores acima da média, contribuindo para o valor alto de desvio padrão. Tal formato de histograma é comum para empresas de varejo, visto que geralmente se tem muitos produtos vendidos com preço baixo e poucos produtos vendidos com preço mais alto. Por isso, a fim de reconhecer a quantidade de outliers dos valores acima média, foi feito um boxplot através da função *boxplot*(), como mostra a figura 8.

Figura 7: Visualização do histograma de preços



Fonte: Autora

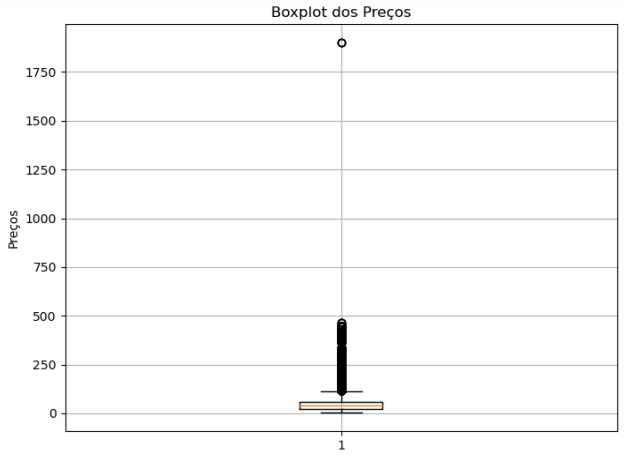
Figura 8: Criação de um *boxplot* com a coluna de preços

```
plt.figure(figsize=(8, 6))
plt.boxplot(base['price_usd'])
plt.title('Boxplot dos Preços')
plt.ylabel('Preços')
plt.grid(True)
plt.show()
```

Fonte: Autora

Foi possível observar, na figura 9 abaixo, uma concentração de *outliers* até 500 dólares e um valor muito acima da média de 1900 dólares.

Figura 9: *Boxplot* com a coluna de preços



Fonte: Autora

Dessa forma, pode-se concluir que a maior parte das compras realizadas são valores mais baixos (75% são valores até 60 dólares), enquanto existem alguns valores muito altos que destoam.

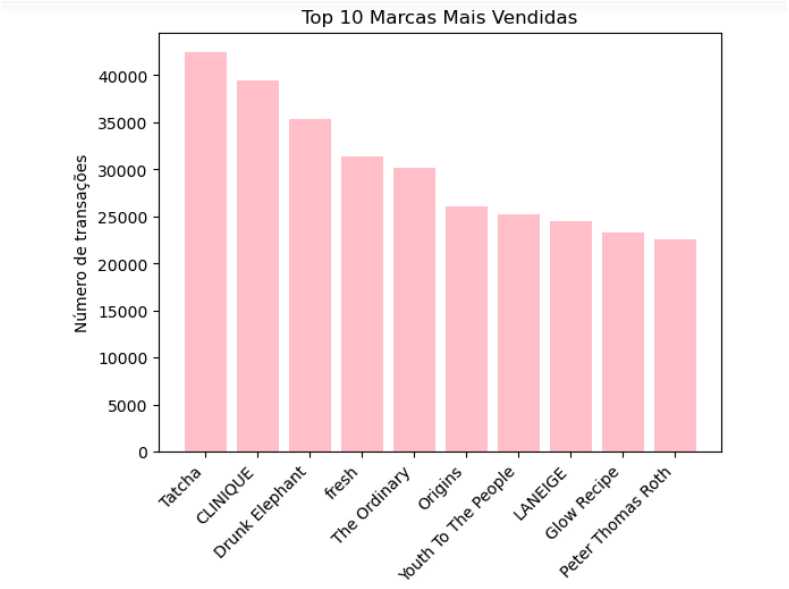
Ao finalizar a análise da coluna de preços, foi feita uma avaliação da coluna de marcas dos produtos, para que se entenda se existe algum padrão na escolha das marcas. Assim, através da função *value_counts(),* contabilizou-se quantas vezes cada marca foi adquirida, para que fossem selecionadas as 10 marcas mais compradas, com a função *head()*. Por fim, foi feito o gráfico de barras, indicado para casos onde há comparação de categoria, por meio da função *bar()*.

Figura 10: Código de criação do top 10 marcas

```
#(10 primeiras marcas mais comprada)

vendas_por_marca = base['brand_name'].value_counts()

# Selecionar as 10 marcas mais vendidas
top_10_marcas = vendas_por_marca.head(10)

# Plotar o gráfico de barras
plt.bar(top_10_marcas.index, top_10_marcas.values, color='pink')

# Adicionar título e rótulos dos eixos
plt.title('Top 10 Marcas Mais Vendidas')
plt.xlabel('Marcas')
plt.ylabel('Número de transações')

# Rotacionar os rótulos do eixo x para facilitar a leitura
plt.xticks(rotation=45, ha='right')

# Exibir o gráfico
plt.show()
```

Fonte: Autora

Figura 11: Gráfico do top 10 marcas mais vendidas



Fonte: Autora

A partir da visualização, é possível afirmar que não há uma marca que se destaque muito comparada às demais. A marca mais vendida, Tatcha, vendeu pouco mais de 40 mil produtos, enquanto a concorrente mais próxima vendeu quase 40 mil.

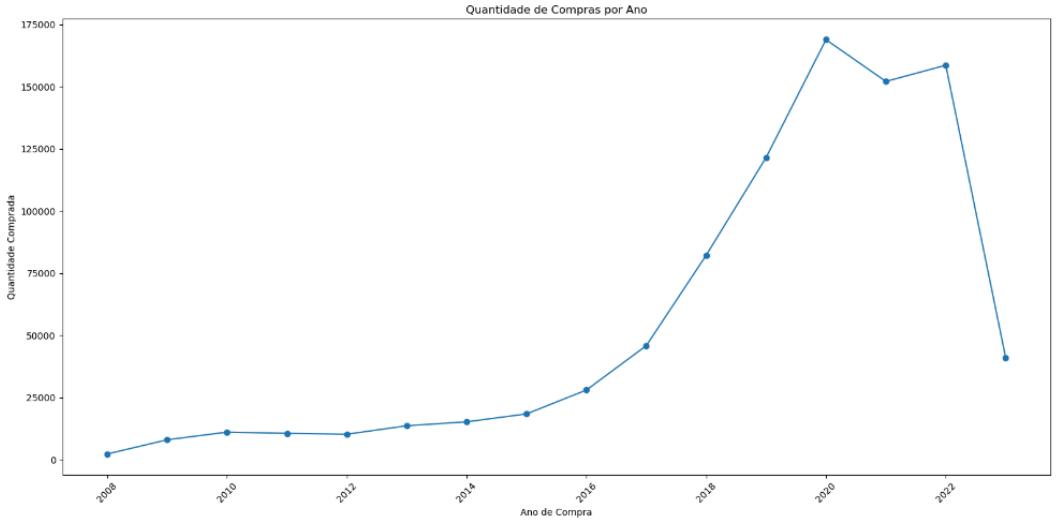
Para analisar a relação entre a quantidade de vendas e a data, foi feito um gráfico em linha. Para criá-lo, primeiro foi utilizada a função *groupby()* para agrupar os dados por ano e depois calcular a quantidade comprada em cada data. Logo após esse passo, utilizou-se a função *plot()* para a construção do gráfico de linhas.

Figura 12: Código para criar gráfico de vendas ao longo dos anos

```
base['submission_time'] = pd.to_datetime(base['submission_time'])

# Agrupar os dados pela data e calcular a quantidade comprada em cada data
compras_por_data = base.groupby(base['submission_time'].dt.date).size()

# Criar o gráfico de linha
plt.figure(figsize=(16, 8))
plt.plot(compras_por_data.index, compras_por_data.values, linestyle='-')
plt.title('Quantidade de Compras por Data')
plt.xlabel('Data de Compra')
plt.ylabel('Quantidade Comprada')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Fonte: Autora

Figura 13: Gráfico de vendas por ano



Fonte: Autora

Ao observar o resultado na figura 13, é possível perceber um aumento considerável na quantidade de compras a partir do ano de 2018. Após esse ano, as vendas declinam, com destaque para 2020, quando aconteceu a covid-19. É válido destacar que as vendas declinam muito em 2023 pois a base de dados só vai até março desse ano.

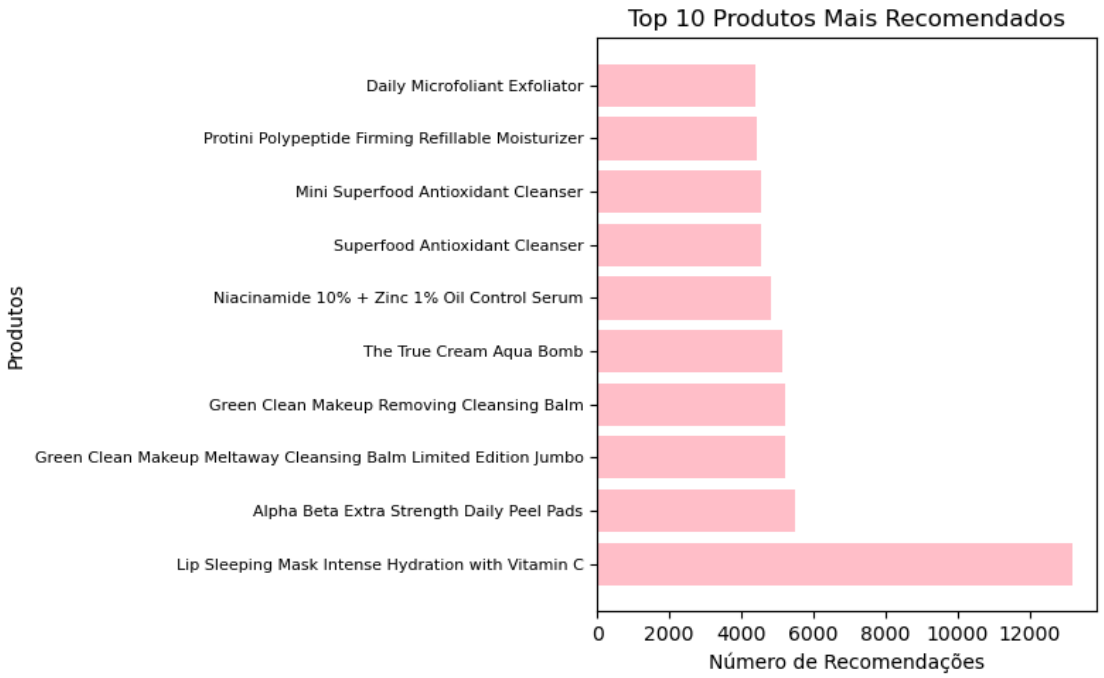
Em seguida, foi feita uma análise dos produtos mais recomendados pelos clientes. Para isso, foi utilizada a função *groupby()* que agrupou os produtos somando todas as recomendações, que são representadas por valores booleanos - 1 representa uma recomendação e 0 representa que o produto não foi recomendado.

Figura 14: Criação do top 10 produtos mais recomendados

```
# Contagem de recomendações por produto
recommendation_count = base.groupby('product_name_x')['is_recommended'].sum(numeric_only=True).reset_index()

# Ordenar os produtos pela contagem de recomendações
recommendation_count = recommendation_count.sort_values(by='is_recommended', ascending=False).head(10)

# Criar o gráfico de barras
plt.figure(figsize=(8, 5))
plt.barh(recommendation_count['product_name_x'], recommendation_count['is_recommended'], color='pink')
plt.xlabel('Número de Recomendações')
plt.ylabel('Produtos')
plt.title('Top 10 Produtos Mais Recomendados')
plt.gca().tick_params(axis='y', labelsize=8)

plt.tight_layout()
plt.show()
```

Fonte: Autora

Figura 15: Gráfico do top 10 produtos mais recomendados



Fonte: Autora

É possível observar que o produto com maior número de recomendações é recomendado mais que o dobro de vezes que os outros produtos presentes na lista. Além disso, os produtos que ficaram da segunda até a nona posição têm números de recomendação similares, de 4 a 6 mil recomendações.

Para complementar as análises relacionadas às recomendações de clientes, foram construídos gráficos de produtos mais recomendados por tipo de pele, com objetivo de observar padrões de escolha do produto. É válido destacar que o banco de dados possui 4 tipos de pele com a escrita em inglês: *dry* (seca), *oily* (oleosa), *combination* (mista) e *normal* (normal). Assim, foi utilizada a função *groupby()* para agrupar os dados por nome de produto e tipo de pele e somar a quantidade de recomendações nesses agrupamentos. Posteriormente, a função *unique()* foi inserida para separar os valores únicos dos 4 tipos de pele que estão contidos na base de dados. Por último, através da estrutura *for*, criou-se um gráfico de barras horizontais para cada tipo de pele.

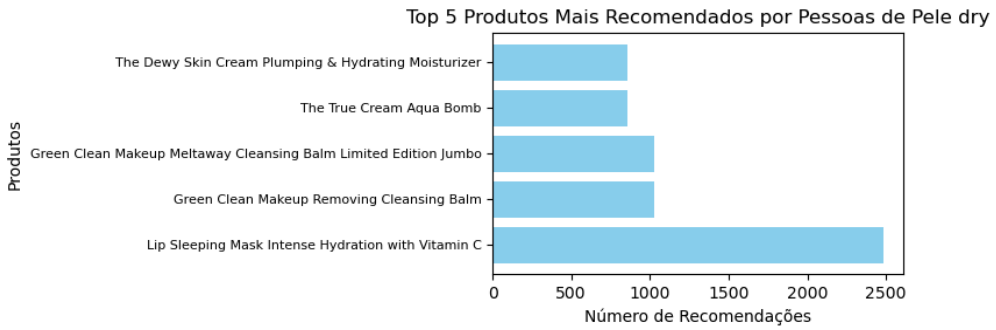
Figura 16: Código da criação dos gráficos de produtos mais recomendados por tipo de pele

```
# Agrupar por produto e contar recomendações por tipo de pele
recommendation_count = base.groupby(['product_name_x', 'skin_type'])['is_recommended'].sum().reset_index()

# Criar um gráfico separado para cada tipo de pele
skin_types = base['skin_type'].unique()

for skin_type in skin_types:
    data_skin_type = recommendation_count[recommendation_count['skin_type'] == skin_type]
    data_skin_type = data_skin_type.sort_values(by='is_recommended', ascending=False).head(5)

    plt.figure(figsize=(8, 3))
    plt.barh(data_skin_type['product_name_x'], data_skin_type['is_recommended'], color='skyblue')
    plt.xlabel('Número de Recomendações')
    plt.ylabel('Produtos')
    plt.title(f'Top 5 Produtos Mais Recomendados por Pessoas de Pele {skin_type}')
    plt.gca().tick_params(axis='y', labelsize=8)
    plt.tight_layout()
    plt.show()
```

Fonte: Autora

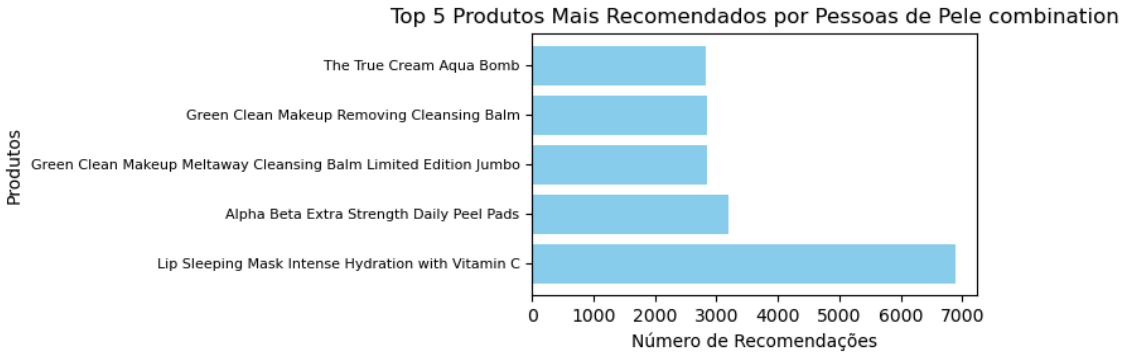
Ao gerar os 4 gráficos das figuras 17 até 20, pode-se observar que o produto "*Lip Sleeping Mask Intense Hydration with Vitamin C*" foi o produto mais recomendado para os 4 tipos de pele. Outros produtos como o "*The True Cream Aqua Bomb", "Green Clean Makeup Cleansing Balm", "Green Clean Makeup Cleansing Balm Limited Edition Jumbo", "Alpha Beta Extra Strenght Daily Peel Pads"*, também aparecem nas 4 categorias, porém em posições diferentes para cada tipo de pele.

Figura 17: Gráfico do top 5 produtos mais vendidos para pele seca



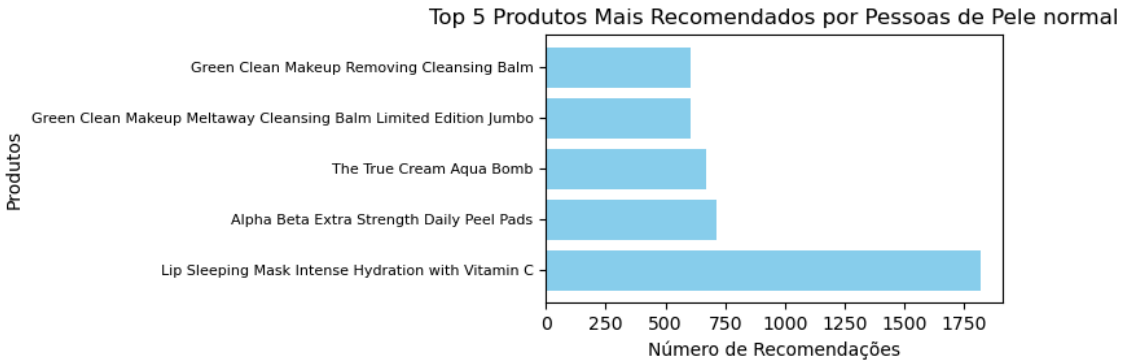
Fonte: Autora

Figura 18: Gráfico do top 5 produtos mais vendidos para pele mista



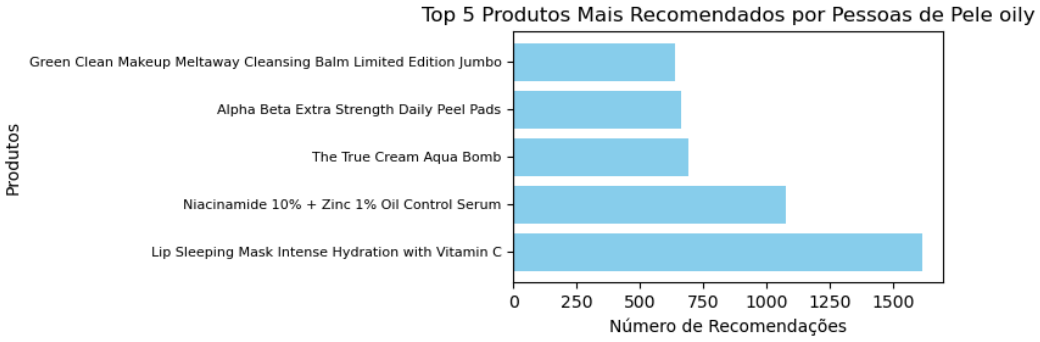
Fonte: Autora

Figura 19: Gráfico do top 5 produtos mais vendidos para pele normal



Fonte: Autora

Figura 20: Gráfico do top 5 produtos mais vendidos para pele oleosa



Fonte: Autora

Por fim, ao analisar os dois tipos de pele que mais se diferenciam, *oily* e *dry*, torna-se evidente que dois produtos se destacam, por aparecerem apenas em uma das categorias. O produto "*Niacinamide 10% + Zinc 1% Oil Control Serum*" aparece apenas como produto recomendado para pessoas de tipo de pele oleosa, enquanto o produto "*The Dewy Skin Cream*

Plumping & Hydrating Moisturizer" só aparece para produtos de pele seca. Dessa maneira, percebe-se uma relação entre os produtos destinados apenas a pele oleosa ou pele seca e o número de recomendações desses.

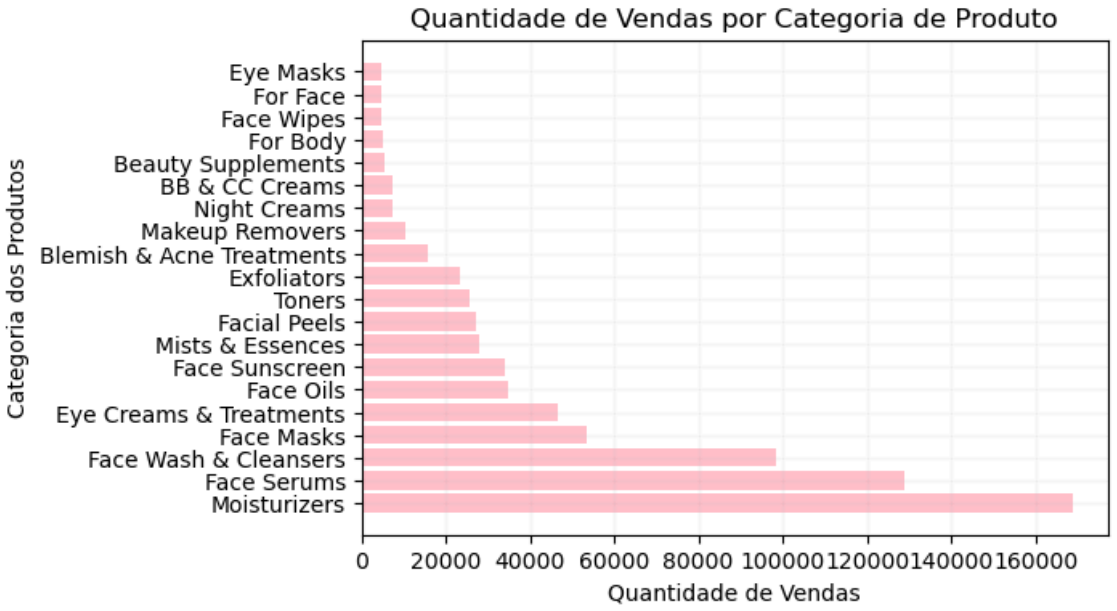
A última análise feita foi uma análise de vendas por categoria de produto. Para isso, foi realizada uma contagem da coluna *"tertiary_category"* por meio da função *value_counts()*. A coluna utilizada representa a categoria terciária, ou seja, existem categorias primárias e secundárias. Entretanto, foi utilizada a terciária pois todos os produtos pertencem às mesmas categorias primárias e secundárias, visto que a base possui registros de vendas de produtos de cuidados com a pele. Com os valores somados por categoria de produto, foi selecionado o top 20 de categorias mais vendidas, por meio da função *head()*. Por fim, através da função *barh()*, criou-se um gráfico de barras horizontais para apresentar as informações

Figura 21: Criação do top 20 categorias mais vendidas

```
plt.figure(figsize=(6, 4))
vendas_por_categoria = base['tertiary_category'].value_counts()
top_20_categorias = vendas_por_categoria.head(20)
plt.barh(top_20_categorias.index, top_20_categorias.values, color='pink')
plt.xlabel('Quantidade de Vendas')
plt.ylabel('Categoria dos Produtos')
plt.title('Quantidade de Vendas por Categoria de Produto')
plt.grid(True, linestyle='-', alpha=0.1)
plt.show()
```

Fonte: Autora

Figura 22: Criação do top 20 categorias mais vendidas



Fonte: Autora

A partir da visualização, fica evidente que há uma predominância de vendas de hidratantes para pele. Além disso, há um destaque para as categorias de séruns de rosto e produtos para limpar a pele.

3.3 LIMPEZA DA BASE DE DADOS

A limpeza é etapa fundamental para o funcionamento ideal dos algoritmos de *machine learning*. Para iniciar, foram removidas todas as colunas que não são interessantes para o trabalho, através do método *drop():*

Figura 23: Remoção de colunas

```
# Lista das colunas que serão removidas
colunas_para_remover = ['is_recommended', 'helpfulness', 'total_feedback_count', 'total_neg_feedback_count', 'total_pos_feedback

# Remover as colunas ignorando erros
base = base.drop(columns=colunas_para_remover, errors='ignore')
print(base.info())
```

Fonte: Autora

Ao final do código, foram utilizadas as funções *print()* e *info()* com objetivo de visualizar as colunas restantes no *data frame:*

Figura 24: Informações da base após remoção das colunas

```
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Unnamed: 0        887686 non-null  float64
 1   author_id         887686 non-null  object
 2   rating            887686 non-null  float64
 3   submission_time   887686 non-null  object
 4   review_text       886414 non-null  object
 5   review_title      636760 non-null  object
 6   skin_tone         743946 non-null  object
 7   eye_color         711832 non-null  object
 8   skin_type         792878 non-null  object
 9   hair_color        697462 non-null  object
 10  product_id        894078 non-null  object
 11  product_name      887686 non-null  object
 12  brand_name        887686 non-null  object
 13  price_usd         887686 non-null  float64
 14  tertiary_category 753921 non-null  object
dtypes: float64(3), object(12)
memory usage: 102.3+ MB
None
```

Fonte: Autora

As colunas mantidas foram apenas aquelas que fornecem informações essenciais das características dos produtos e informações da compra e venda. Tais colunas formam a base do estudo apresentado no próximo capítulo, com a aplicação dos algoritmos.

Após essa etapa, foram removidas todas as linhas nulas da base em duas colunas específicas, por meio da função *dropna()*. A coluna de ‘author_id’ teve seus valores nulos

removidos, visto que a aplicação dos algoritmos de associação necessita de todos os valores de identificação dos compradores. Além disso, também removeu-se valores nulos da coluna “tertiary_category”, pois a coluna de categorias também é fundamental para entendermos a relação entre as categorias de produtos que os usuários compram.

Para visualizar a quantidade de valores nulos removidos, também foi aplicada a função *isnull()* para verificar o somatório. Por fim, foi discriminado o resultado.

Figura 25: Código para remoção de linhas com valores nulos

```
# Remoção das linhas com valores nulos
base = base.dropna(subset=['tertiary_category', 'author_id'])

# Número de valores nulos após a remoção
print("Valores nulos após a remoção:")
print(base.isnull().sum())
```

Fonte: Autora

Figura 26: Resultado da remoção de linhas com valores nulos

```
Valores nulos após a remoção:
Unnamed: 0                0
author_id                 0
rating                    0
submission_time           0
review_text            1046
review_title         211483
skin_tone            122108
eye_color            146388
skin_type             79952
hair_color           158730
product_id                0
product_name              0
brand_name                0
price_usd                 0
tertiary_category         0
dtype: int64
```

Fonte: Autora

É possível observar que a maior parte dos valores nulos foram removidos e a base tornou-se mais otimizada para ser trabalhada. Para finalizar a limpeza, foram removidas todas as linhas duplicadas neste *data frame.* Dessa forma, utilizou-se a função *duplicated()* para armazenar todas as duplicatas em uma única variável ‘duplicatas’. Após isso, foi impressa a soma total de valores duplicados, para se verificar a quantidade deles na base de dados. O número total de linhas duplicadas foi 308, número considerável para remoção. Assim, foi utilizada a função *drop_duplicates()* com *subset* em todas as colunas da base, para remover linhas exatamente iguais. Com isso, é possível visualizar o total de linhas após a remoção, com a função *len(),* para que fique evidente o número de dados que foram trabalhados na próxima etapa. É possível perceber que a redução de duplicatas teve um

número pouco significativo comparado ao número total de linhas, porém essa remoção é essencial para as análises feitas nos próximos capítulos.

Figura 27: Remoção de valores duplicados

```
# Verificar duplicatas
duplicatas = base.duplicated(subset=['author_id', 'submission_time', 'product_id', 'review_text', 'rating', 'review_title', 'skin

print(f"Número de linhas duplicadas: {duplicatas.sum()}")

# Remover duplicatas
base = base.drop_duplicates(subset=['author_id', 'submission_time', 'product_id', 'review_text', 'rating', 'review_title', 'skin_

print(f"\nNúmero de linhas após a remoção de duplicatas: {len(base)}")
```

Fonte: Autora

Figura 28: Resultado da remoção de valores duplicados

```
Número de linhas duplicadas: 308

Número de linhas após a remoção de duplicatas: 747885
```

Fonte: Autora

3.4 APLICAÇÃO DOS ALGORITMOS

Para aplicar os algoritmos, foi necessário importar a biblioteca *mlxtend*, além das bibliotecas já utilizadas na análise exploratória. A biblioteca *mlxtend* fornece uma implementação eficiente dos algoritmos Apriori e *FP-Growth* e a funcionalidade para gerar regras de associação.

Figura 29: Importação da biblioteca e algoritmo

```
from mlxtend.frequent_patterns import apriori, association_rules
```

Fonte: Autora

Após a importação da biblioteca, foi necessário transformar os dados no formato adequado para a aplicação do algoritmo em questão. O *data frame* utilizado possui os dados de venda de produtos comprados por cada cliente, enquanto o formato adequado depende de uma base de transações, onde cada transação representa uma compra única. Para realizar a transformação, o processo envolveu a criação de uma tabela de transações onde cada linha representa um autor (transação) e cada coluna representa uma categoria de produto, com valores binários indicando a presença ou ausência da categoria na transação. Essa transformação foi realizada utilizando a função *pd.crosstab()* do Pandas, seguida pela

binarização dos dados, onde valores superiores a zero foram substituídos por 1 e os demais por 0.

Figura 30: Tabela de transações

```
# Criar uma tabela de crosstab para converter as transações
transactions = pd.crosstab(base['author_id'], base['tertiary_category'])

# Transformar em dados binários
transactions = transactions.applymap(lambda x: 1 if x > 0 else 0)
```

Fonte: Autora

Com os dados preparados, aplicou-se o algoritmo Apriori utilizando a função apriori da biblioteca *mlxtend*. Para definir o suporte mínimo necessário para que um conjunto de itens fosse considerado frequente, foi utilizado o valor de 0,01 (ou 1%). A escolha desse valor de suporte foi baseada em um equilíbrio entre a identificação de padrões úteis e a manutenção de uma quantidade manejável de conjuntos frequentes. Produtos com um suporte de 1% estão presentes em 1% das transações.

Figura 31: Algoritmo A Priori

```
# Executar o algoritmo A Priori
frequent_itemsets = apriori(transactions, min_support=0.01, use_colnames=True)
```

Fonte: Autora

Após identificar os conjuntos frequentes de itens, foram geradas as regras de associação utilizando a função *association_rules()*, especificando métricas como "*lift*" e um limite mínimo para o valor do *lift*. Essa métrica é relevante pois ela dita o quanto o item consequente se torna mais frequente a partir do momento em que o antecedente ocorre. As regras de associação são derivadas dos conjuntos frequentes e ajudam a identificar relações significativas entre diferentes categorias de produtos.

Figura 32: Regras de associação

```
# Gerar as regras de associação
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
```

Fonte: Autora

Por fim, para visualizar o resultado e as regras de associação, foi utilizado o *print()* para exibir o conjunto de itens frequentes e as regras de associação geradas pelo algoritmo.

Figura 33: Código para exibir os resultados

```
# Exibir os resultados
print("Conjuntos frequentes de itens:")
print(frequent_itemsets)

print("\nRegras de associação:")
print(rules)
```

Fonte: Autora

Figura 34: Conjunto de categorias frequentes de suporte de 1% com uso do Apriori

Conjuntos frequentes de itens:

	support	itemsets
0	0.014839	(BB & CC Creams)
1	0.031515	(Blemish & Acne Treatments)
2	0.046500	(Exfoliators)
3	0.086765	(Eye Creams & Treatments)
4	0.096667	(Face Masks)
5	0.067191	(Face Oils)
6	0.213248	(Face Serums)
7	0.063848	(Face Sunscreen)
8	0.178907	(Face Wash & Cleansers)
9	0.053788	(Facial Peels)
10	0.020598	(Makeup Removers)
11	0.043659	(Mists & Essences)
12	0.287375	(Moisturizers)
13	0.014835	(Night Creams)
14	0.048645	(Toners)
15	0.011509	(Moisturizers, Exfoliators)
16	0.018988	(Eye Creams & Treatments, Face Serums)
17	0.010523	(Face Wash & Cleansers, Eye Creams & Treatments)
18	0.023904	(Moisturizers, Eye Creams & Treatments)
19	0.013944	(Face Masks, Face Serums)
20	0.011889	(Face Masks, Face Wash & Cleansers)
21	0.016966	(Face Masks, Moisturizers)
22	0.010807	(Moisturizers, Face Oils)
23	0.012124	(Face Serums, Face Sunscreen)
24	0.023020	(Face Wash & Cleansers, Face Serums)
25	0.011807	(Facial Peels, Face Serums)
26	0.041216	(Moisturizers, Face Serums)
27	0.013629	(Moisturizers, Face Sunscreen)
28	0.010336	(Makeup Removers, Face Wash & Cleansers)
29	0.029293	(Moisturizers, Face Wash & Cleansers)
30	0.010361	(Facial Peels, Moisturizers)
31	0.011106	(Moisturizers, Eye Creams & Treatments, Face S...
32	0.011289	(Moisturizers, Face Wash & Cleansers, Face Ser...

Fonte: Autora

Figura 35: Regras de Associação geradas com o uso do Apriori

Regras de associação:

	antecedents	consequents
0	(Face Serums)	(Eye Creams & Treatments)
1	(Eye Creams & Treatments)	(Face Serums)
2	(Facial Peels)	(Face Serums)
3	(Face Serums)	(Facial Peels)
4	(Makeup Removers)	(Face Wash & Cleansers)
5	(Face Wash & Cleansers)	(Makeup Removers)
6	(Moisturizers, Face Serums)	(Eye Creams & Treatments)
7	(Moisturizers, Eye Creams & Treatments)	(Face Serums)
8	(Face Serums, Eye Creams & Treatments)	(Moisturizers)
9	(Moisturizers)	(Face Serums, Eye Creams & Treatments)
10	(Face Serums)	(Moisturizers, Eye Creams & Treatments)
11	(Eye Creams & Treatments)	(Moisturizers, Face Serums)
12	(Face Wash & Cleansers, Moisturizers)	(Face Serums)
13	(Face Wash & Cleansers, Face Serums)	(Moisturizers)
14	(Moisturizers, Face Serums)	(Face Wash & Cleansers)
15	(Face Wash & Cleansers)	(Moisturizers, Face Serums)
16	(Moisturizers)	(Face Wash & Cleansers, Face Serums)
17	(Face Serums)	(Face Wash & Cleansers, Moisturizers)

Fonte: Autora

Para aplicação do algoritmo *FP-Growth*, os passos realizados acima foram replicados, com a alteração apenas do algoritmo utilizado. Ao fim, foram impressos os resultados do algoritmo: conjunto de itens frequentes e as regras de associação.

Figura 36: Execução do algoritmo FP-Growth

```
# Executar o algoritmo FP-Growth
frequent_itemsets2 = fpgrowth(transactions, min_support=0.01, use_colnames=True)

# Gerar as regras de associação
rules2 = association_rules(frequent_itemsets, metric="lift", min_threshold=1)

# Exibir os resultados
print("Conjuntos frequentes de itens:")
print(frequent_itemsets2)

print("\nRegras de associação:")
print(rules2)
```

Fonte: Autora

Figura 37: Conjunto de itens frequentes gerados pelo FP-Growth

	support	itemsets
0	0.067191	(Face Oils)
1	0.031515	(Blemish & Acne Treatments)
2	0.213248	(Face Serums)
3	0.178907	(Face Wash & Cleansers)
4	0.287375	(Moisturizers)
5	0.046500	(Exfoliators)
6	0.020598	(Makeup Removers)
7	0.086765	(Eye Creams & Treatments)
8	0.014835	(Night Creams)
9	0.063848	(Face Sunscreen)
10	0.096667	(Face Masks)
11	0.043659	(Mists & Essences)
12	0.053788	(Facial Peels)
13	0.048645	(Toners)
14	0.014839	(BB & CC Creams)
15	0.010807	(Moisturizers, Face Oils)
16	0.041216	(Moisturizers, Face Serums)
17	0.023020	(Face Wash & Cleansers, Face Serums)
18	0.029293	(Moisturizers, Face Wash & Cleansers)
19	0.011289	(Moisturizers, Face Wash & Cleansers, Face Ser...
20	0.011509	(Moisturizers, Exfoliators)
21	0.010336	(Makeup Removers, Face Wash & Cleansers)
22	0.023904	(Moisturizers, Eye Creams & Treatments)
23	0.010523	(Face Wash & Cleansers, Eye Creams & Treatments)
24	0.018988	(Eye Creams & Treatments, Face Serums)
25	0.011106	(Moisturizers, Eye Creams & Treatments, Face S...
26	0.013629	(Moisturizers, Face Sunscreen)
27	0.012124	(Face Serums, Face Sunscreen)
28	0.011889	(Face Masks, Face Wash & Cleansers)
29	0.016966	(Face Masks, Moisturizers)
30	0.013944	(Face Masks, Face Serums)
31	0.011807	(Facial Peels, Face Serums)
32	0.010361	(Facial Peels, Moisturizers)

Fonte: Autora

Na figura 38 é possível visualizar as regras de associação geradas pelo *FP-Growth*, em que temos os produtos antecedentes e consequentes, formando 17 regras de associação válidas.

Figura 38: Regras de associação geradas pelo FP Growth

Regras de associação:

	antecedents	consequents
0	(Eye Creams & Treatments)	(Face Serums)
1	(Face Serums)	(Eye Creams & Treatments)
2	(Facial Peels)	(Face Serums)
3	(Face Serums)	(Facial Peels)
4	(Makeup Removers)	(Face Wash & Cleansers)
5	(Face Wash & Cleansers)	(Makeup Removers)
6	(Moisturizers, Eye Creams & Treatments)	(Face Serums)
7	(Moisturizers, Face Serums)	(Eye Creams & Treatments)
8	(Eye Creams & Treatments, Face Serums)	(Moisturizers)
9	(Moisturizers)	(Eye Creams & Treatments, Face Serums)
10	(Eye Creams & Treatments)	(Moisturizers, Face Serums)
11	(Face Serums)	(Moisturizers, Eye Creams & Treatments)
12	(Moisturizers, Face Wash & Cleansers)	(Face Serums)
13	(Moisturizers, Face Serums)	(Face Wash & Cleansers)
14	(Face Wash & Cleansers, Face Serums)	(Moisturizers)
15	(Moisturizers)	(Face Wash & Cleansers, Face Serums)
16	(Face Wash & Cleansers)	(Moisturizers, Face Serums)
17	(Face Serums)	(Moisturizers, Face Wash & Cleansers)

Fonte: Autora

3.5 COMPARAÇÃO DOS ALGORITMOS

Os dois algoritmos foram comparados, para que seja possível analisar qual deles desempenha melhor no caso particular que trata este trabalho. É relevante destacar que ambos resultaram na mesma quantidade de regras de associação, ou seja, não houve um desempenho diferente nesse sentido. Para avaliar o desempenho para além das regras, foi feita uma função que mede o tempo de execução de cada algoritmo e, posteriormente, uma comparação dividindo a base entre conjunto 1 (80% da base) e conjunto 2 (20% da base).

Figura 39: Função para medir tempo e medição do tempo do algoritmo Apriori

```
import time

# Função para medir o tempo de execução de um algoritmo
def measure_time(func, *args):
    start_time = time.time()
    result = func(*args)
    end_time = time.time()
    return result, end_time - start_time

# Executar o algoritmo A Priori e medir o tempo de execução
apriori_itemsets, apriori_time = measure_time(apriori, transactions, 0.01, True)
print(f"Tempo de execução do algoritmo A Priori: {apriori_time} segundos")
```

Fonte: Autora

A função é declarada na figura 39 como “measure_time”, em que é utilizado cada algoritmo como parâmetro para a execução do código. A função inicia com a medição de tempo ao início e armazena na variável “start_time”, depois calcula o resultado do algoritmo, por meio da função dele e seus argumentos. Assim, depois de executado, é possível armazenar o tempo novamente na variável “end_time”, em que é calculado novamente o tempo. Por fim,

a função “measure_time” retorna ao resultado do algoritmo e a duração de execução, que é o “end_time” menos o “start_time”.

Primeiramente foi calculado o tempo do algoritmo Apriori, e posteriormente os mesmos passos foram realizados para calcular o tempo de execução do algoritmo FP-Growth.

Figura 40: Medição do tempo do algoritmo FP-Growth

```
# Executar o algoritmo FP-Growth e medir o tempo de execução
fpgrowth_itemsets, fpgrowth_time = measure_time(fpgrowth, transactions, 0.01, True)
print(f"Tempo de execução do algoritmo FP-Growth: {fpgrowth_time} segundos")
```

Fonte: Autora

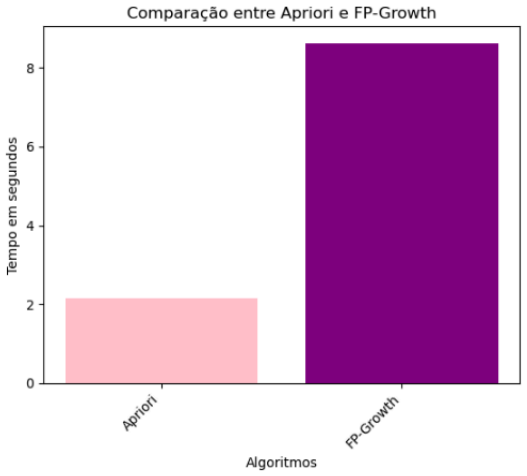
Para visualizar os resultados e gerar melhor comparação, foi criado um gráfico de barras que compara os tempos de execução dos dois algoritmos. Dessa maneira, dois vetores foram criados, um que armazena o nome em texto dos algoritmos e outro que armazena o tempo de cada um, em tipo *float*. O gráfico foi plotado por meio da função *bar(),* adicionando títulos e rótulos para eixo x e y. Por fim, o gráfico foi gerado usando a função *show()*.

Figura 41: Código para criação do gráfico de barras da comparação entre algoritmos

```
algoritmos = ['Apriori','FP-Growth']
tempo = [apriori_time, fpgrowth_time]

# Plotar o gráfico de barras
plt.bar(algoritmos, tempo, color=['pink', 'purple'])

# Adicionar título e rótulos dos eixos
plt.title('Comparação entre Apriori e FP-Growth')
plt.xlabel('Algoritmos')
plt.ylabel('Tempo em segundos')

# Rotacionar os rótulos do eixo x para facilitar a leitura
plt.xticks(rotation=45, ha='right')

# Exibir o gráfico
plt.show()
```

Fonte: Autora

Figura 42: Gráfico de barras da comparação entre algoritmos



Fonte: Autora

A partir da visualização é possível perceber que o algoritmo Apriori teve um desempenho superior ao FP-Growth, pois apresentou um tempo de execução menor. Entretanto, para confirmar a eficácia do Apriori, os algoritmos foram rodados 5 vezes, a fim de garantir que teria-se o mesmo padrão mesmo em múltiplas rodadas. Dessa forma, foi possível gerar a tabela abaixo com os tempos de execução em cada rodada para os algoritmos:

Tabela 1: Tempos de execução entre Apriori e FP-Growth

Rodada	Tempo Apriori	Tempo FP-Growth
1	2.06	8.9
2	2.11	8.8
3	2.03	8.82
4	1.96	8.74
5	2.14	8.61
Média	2.06	8.774

Fonte: Autora

Como é possível observar na tabela 1, o padrão de desempenho foi mantido para todas as rodadas aplicadas e, em média, o Apriori tem um desempenho superior em execução. O algoritmo FP-Growth é estruturado para que tenha uma execução mais veloz que o Apriori, por conta da sua estrutura em árvores, como é observado no resultado da pesquisa de Cabreira, onde o Apriori teve um tempo de execução quase 5 cinco vezes maior que o FP-Growth (Cabreira, 2023). Não obstante, o resultado do presente estudo pode ser explicado pela estrutura dos dados escolhidos, característica influente para o desempenho de algoritmos. Uma explicação que possui fortes evidências é a esparsidade dos dados da base, o que significa uma quantidade alta de transações, porém poucos conjuntos frequentes e baixa variedade. Na figura 33 é possível perceber que muitos dos conjuntos de itens frequentes são similares uns com os outros, trazendo pouca variedade na análise. Além disso, o desempenho dos algoritmos também depende do sistema computacional e outras características da máquina utilizada.

Por fim, foi feita uma análise para comparar a acurácia entre dados dividindo a base em 80% e 20% para os dois algoritmos, com objetivo de entender se haverá diferença nas regras encontradas. Para isso, a base de dados de transações foi embaralhada e depois dividida entre uma base conjunto 1 “*conjunto1_transactions*”, que representa 80% das transações, e uma base conjunto 2 “*conjunto2_transactions*”, que representa 20% das transações. Com isso, nas duas bases, foram aplicados os algoritmos Apriori, como mostra a figura 43, para que seja

possível imprimir na tela as regras descobertas no conjunto 1 e as regras validadas no conjunto 2.

Figura 43: Código para dados do conjunto 1 e 2 do Apriori

```
# Embaralhar os dados
shuffled_transactions = transactions.sample(frac=1, random_state=42).reset_index(drop=True)

# Determinar o ponto de divisão
conjunto1_size = int(0.8 * len(shuffled_transactions))

# Dividir os dados
conjunto1_transactions = shuffled_transactions[:conjunto1_size]
conjunto2_transactions = shuffled_transactions[conjunto1_size:]

# Descobrir conjuntos frequentes no conjunto 1
conjunto1_frequent_itemsets = apriori(conjunto1_transactions, min_support=0.01, use_colnames=True)

# Gerar regras de associação a partir do conjunto 1
conjunto1_rules = association_rules(conjunto1_frequent_itemsets, metric="lift", min_threshold=1)

# Descobrir conjuntos frequentes no conjunto 2
conjunto2_frequent_itemsets = apriori(conjunto2_transactions, min_support=0.01, use_colnames=True)

# Gerar regras de associação a partir do conjunto 2
conjunto2_rules = association_rules(conjunto2_frequent_itemsets, metric="lift", min_threshold=1)

# Resultados
print(f"Regras descobertas no conjunto 1: \n{conjunto1_rules}\n")
print(f"Regras validadas no conjunto 2: \n{conjunto2_rules}\n")
```

Fonte: Autora

O mesmo foi feito para o algoritmo *FP-Growth,* como mostra a figura 44, para que seja possível comparar a acurácia dos dois algoritmos no resultado entre conjunto 1 e 2.

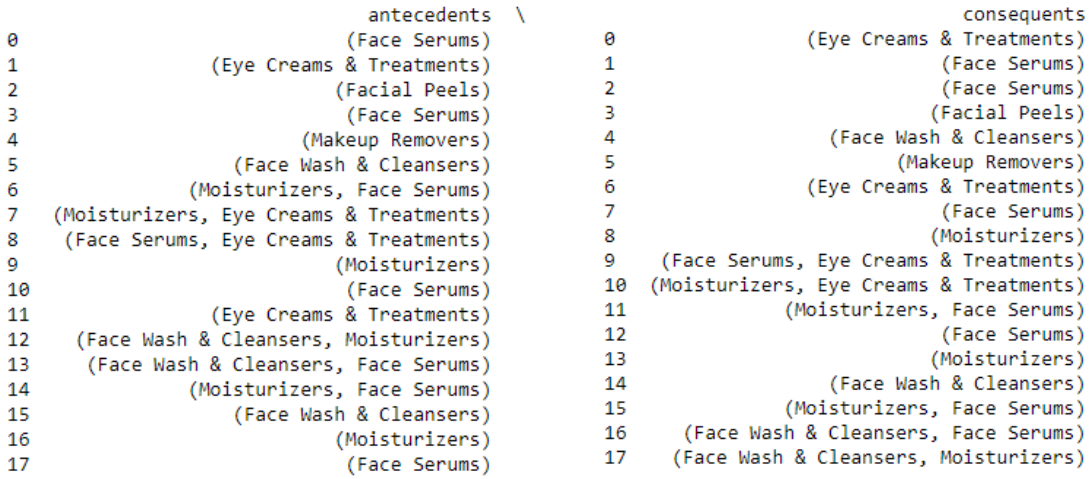
Figura 44: Código para dados do conjunto 1 e 2 do *FP-Growth*

```
# Embaralhar os dados
shuffled_transactions = transactions.sample(frac=1, random_state=42).reset_index(drop=True)

# Determinar o ponto de divisão
conjunto1_size = int(0.8 * len(shuffled_transactions))

# Dividir os dados
conjunto1_transactions = shuffled_transactions[:conjunto1_size]
conjunto2_transactions = shuffled_transactions[conjunto1_size:]

# Descobrir conjuntos frequentes no conjunto1
conjunto1_frequent_itemsets = fpgrowth(conjunto1_transactions, min_support=0.01, use_colnames=True)

# Gerar regras de associação a partir do conjunto1
conjunto1_rules = association_rules(conjunto1_frequent_itemsets, metric="lift", min_threshold=1)

# Descobrir conjuntos frequentes no conjunto2
conjunto2_frequent_itemsets = fpgrowth(conjunto2_transactions, min_support=0.01, use_colnames=True)

# Gerar regras de associação a partir do conjunto2
conjunto2_rules = association_rules(conjunto2_frequent_itemsets, metric="lift", min_threshold=1)

# Resultados
print(f"Regras descobertas no conjunto 1 FP-Growth: \n{conjunto1_rules}\n")
print(f"Regras validadas no conjunto 2 FP-Growth: \n{conjunto2_rules}\n")
```

Fonte: Autora

Como resultado do algoritmo Apriori, pode-se observar nas figuras 45 e 46 que as regras encontradas tanto no conjunto 1 quanto no conjunto 2 são as mesmas. Isso significa que o modelo teve um bom desempenho para o caso, pois as regras são consistentes em ambos os casos. Isso também significa que as regras encontradas são fortes o suficiente para serem vistas tanto em 80% da base quanto em 20% da base.

Figura 45: Regras do Apriori encontradas no conjunto 1

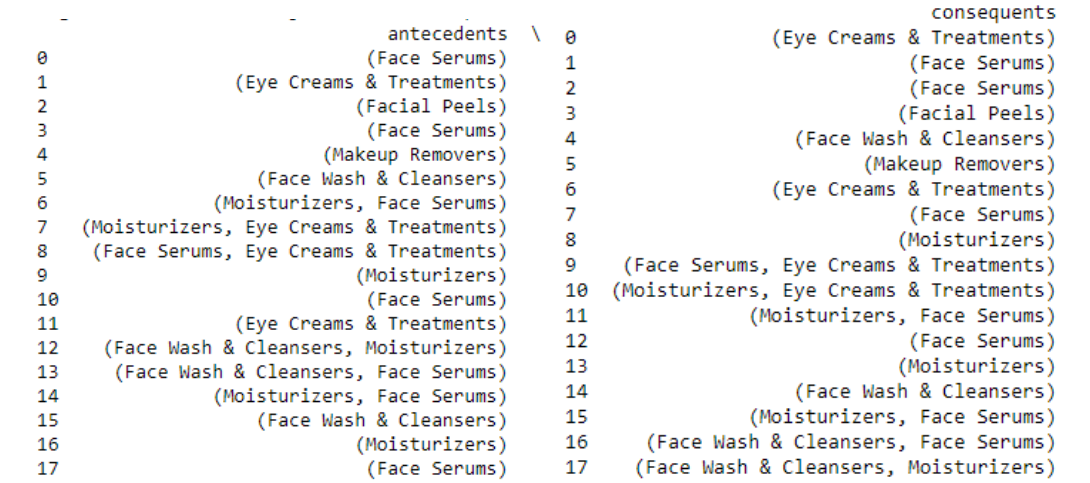


	antecedents	consequents
0	(Face Serums)	(Eye Creams & Treatments)
1	(Eye Creams & Treatments)	(Face Serums)
2	(Facial Peels)	(Face Serums)
3	(Face Serums)	(Facial Peels)
4	(Makeup Removers)	(Face Wash & Cleansers)
5	(Face Wash & Cleansers)	(Makeup Removers)
6	(Moisturizers, Face Serums)	(Eye Creams & Treatments)
7	(Moisturizers, Eye Creams & Treatments)	(Face Serums)
8	(Face Serums, Eye Creams & Treatments)	(Moisturizers)
9	(Moisturizers)	(Face Serums, Eye Creams & Treatments)
10	(Face Serums)	(Moisturizers, Eye Creams & Treatments)
11	(Eye Creams & Treatments)	(Moisturizers, Face Serums)
12	(Face Wash & Cleansers, Moisturizers)	(Face Serums)
13	(Face Wash & Cleansers, Face Serums)	(Moisturizers)
14	(Moisturizers, Face Serums)	(Face Wash & Cleansers)
15	(Face Wash & Cleansers)	(Moisturizers, Face Serums)
16	(Moisturizers)	(Face Wash & Cleansers, Face Serums)
17	(Face Serums)	(Face Wash & Cleansers, Moisturizers)

Fonte: Autora

Pode-se observar que, para ambas as regras, o conjunto também deu o mesmo resultado gerado pelo algoritmo quando aplicado em 100% da base, o que significa que temos uma boa aproximação de quais regras são melhores aplicadas no contexto desta base de dados de cosméticos estudada.

Figura 46: Regras do Apriori encontradas no conjunto 2



	antecedents	consequents
0	(Face Serums)	(Eye Creams & Treatments)
1	(Eye Creams & Treatments)	(Face Serums)
2	(Facial Peels)	(Face Serums)
3	(Face Serums)	(Facial Peels)
4	(Makeup Removers)	(Face Wash & Cleansers)
5	(Face Wash & Cleansers)	(Makeup Removers)
6	(Moisturizers, Face Serums)	(Eye Creams & Treatments)
7	(Moisturizers, Eye Creams & Treatments)	(Face Serums)
8	(Face Serums, Eye Creams & Treatments)	(Moisturizers)
9	(Moisturizers)	(Face Serums, Eye Creams & Treatments)
10	(Face Serums)	(Moisturizers, Eye Creams & Treatments)
11	(Eye Creams & Treatments)	(Moisturizers, Face Serums)
12	(Face Wash & Cleansers, Moisturizers)	(Face Serums)
13	(Face Wash & Cleansers, Face Serums)	(Moisturizers)
14	(Moisturizers, Face Serums)	(Face Wash & Cleansers)
15	(Face Wash & Cleansers)	(Moisturizers, Face Serums)
16	(Moisturizers)	(Face Wash & Cleansers, Face Serums)
17	(Face Serums)	(Face Wash & Cleansers, Moisturizers)

Fonte: Autora

No caso do algoritmo *FP-Growth,* as figuras 47 e 48 mostram que as regras encontradas para ambos conjuntos são as mesmas, porém elas aparecem em ordens diferentes. A mudança na ordem das regras não indica que houve um desempenho melhor ou pior entre as duas aplicações, visto que o algoritmo não imprime as regras em ordem específica. Dessa forma, é possível afirmar que o *FP-Growth* também apresentou uma boa acurácia ao ser submetido ao conjunto 1 e ao conjunto 2, já que as regras impressas são as mesmas.

Figura 47: Regras do *FP-Growth* encontradas no conjunto de 1

	antecedents	consequents
0	(Face Wash & Cleansers, Moisturizers)	(Face Serums)
1	(Face Wash & Cleansers, Face Serums)	(Moisturizers)
2	(Moisturizers, Face Serums)	(Face Wash & Cleansers)
3	(Face Wash & Cleansers)	(Moisturizers, Face Serums)
4	(Moisturizers)	(Face Wash & Cleansers, Face Serums)
5	(Face Serums)	(Face Wash & Cleansers, Moisturizers)
6	(Facial Peels)	(Face Serums)
7	(Face Serums)	(Facial Peels)
8	(Face Serums)	(Eye Creams & Treatments)
9	(Eye Creams & Treatments)	(Face Serums)
10	(Moisturizers, Face Serums)	(Eye Creams & Treatments)
11	(Moisturizers, Eye Creams & Treatments)	(Face Serums)
12	(Face Serums, Eye Creams & Treatments)	(Moisturizers)
13	(Moisturizers)	(Face Serums, Eye Creams & Treatments)
14	(Face Serums)	(Moisturizers, Eye Creams & Treatments)
15	(Eye Creams & Treatments)	(Moisturizers, Face Serums)
16	(Makeup Removers)	(Face Wash & Cleansers)
17	(Face Wash & Cleansers)	(Makeup Removers)

Fonte: Autora

Um exemplo da diferença na ordem é a regra 7 da figura 47, *Face Serums →Facial Peels,* aparecer na regra 3 da figura 48. São as mesmas regras, porém com posições diferentes na impressão feita no Jupyter Notebook. Como o *FP-Growth* gera as regras em estrutura de árvore, isso pode ter ocasionado no resultado aparecer em uma ordem diferente para as regras de associação.

Figura 48: Regras do *FP-Growth* encontradas no conjunto 2

	antecedents	consequents
0	(Face Serums)	(Eye Creams & Treatments)
1	(Eye Creams & Treatments)	(Face Serums)
2	(Facial Peels)	(Face Serums)
3	(Face Serums)	(Facial Peels)
4	(Makeup Removers)	(Face Wash & Cleansers)
5	(Face Wash & Cleansers)	(Makeup Removers)
6	(Moisturizers, Face Serums)	(Eye Creams & Treatments)
7	(Moisturizers, Eye Creams & Treatments)	(Face Serums)
8	(Face Serums, Eye Creams & Treatments)	(Moisturizers)
9	(Moisturizers)	(Face Serums, Eye Creams & Treatments)
10	(Face Serums)	(Moisturizers, Eye Creams & Treatments)
11	(Eye Creams & Treatments)	(Moisturizers, Face Serums)
12	(Face Wash & Cleansers, Moisturizers)	(Face Serums)
13	(Face Wash & Cleansers, Face Serums)	(Moisturizers)
14	(Moisturizers, Face Serums)	(Face Wash & Cleansers)
15	(Face Wash & Cleansers)	(Moisturizers, Face Serums)
16	(Moisturizers)	(Face Wash & Cleansers, Face Serums)
17	(Face Serums)	(Face Wash & Cleansers, Moisturizers)

Fonte: Autora

Vale ressaltar, para fins de pesquisa futura, que não é estritamente necessário realizar a distinção de análise de dados em conjuntos diferentes para algoritmos como o Apriori e *FP-Growth.* Essa análise foi feita neste trabalho com o objetivo de entender mais profundamente o funcionamento dos algoritmos e se o modelo se mantém fiel quando isolamos uma parte específica da base. Os dados do conjunto 2 então são usados para "simular" se nesse caso isolado o padrão visto na maior parte da base se manteria. Como visto

acima, podemos afirmar que o modelo tem um bom desempenho nesse sentido. Caso o desempenho resultasse em regras muito discrepantes, seria interessante uma análise do porquê ele não atende às expectativas.

4. CONCLUSÃO

Ao longo deste trabalho, foram exploradas diversas ferramentas que contribuem para a compreensão de grandes bases de dados. Inicialmente, realizou-se uma análise exploratória com o objetivo de identificar possíveis padrões no conjunto de dados selecionado. Nesta etapa, foi possível extrair informações relevantes, tais como a concentração de produtos vendidos por até R$ 250,00, a preferência dos consumidores por produtos de hidratação, como hidratantes e séruns faciais, que se destacaram em quantidade de vendas, e a relação entre o tipo de pele do consumidor e o produto adquirido por ele.

Após essa etapa, aplicaram-se dois algoritmos de associação, Apriori e FP-Growth, ao mesmo conjunto de dados. A partir dessas aplicações, os algoritmos geraram regras essenciais para entender o comportamento de compra dos consumidores e outros *insights* importantes para empresas do setor, como, por exemplo, a associação que compras de *Face Serums* são frequentemente acompanhadas da compra de *Eye Creams and Treatments*. Ademais, observou-se que ambos os algoritmos produziram resultados semelhantes, com 17 regras de associação cada e associações bastante similares.

Para estudar a diferença de desempenho entre os dois algoritmos, foram analisados os critérios de tempo de execução e acurácia com dados de diferentes conjuntos. Verificou-se que o Apriori apresentou um tempo de execução inferior ao FP-Growth, o que contribui para compreender melhor o funcionamento desses algoritmos em diferentes contextos. Este resultado evidencia um caso em que o FP-Growth não obteve superioridade. Em relação à acurácia, ambos os algoritmos produziram regras de associação similares tanto no conjunto 1 (80% da base), quanto no conjunto 2 (20% da base). A única distinção observada foi na ordem de impressão das regras pelo FP-Growth entre os dois conjuntos.

Este estudo revelou-se relevante para explorar técnicas de *Data Mining*. Os resultados obtidos não apenas confirmaram a robustez dessas técnicas, mas também evidenciaram nuances importantes sobre desempenho e aplicabilidade em diferentes cenários de análise de dados. Assim, este trabalho não só amplia o entendimento teórico sobre mineração de dados, mas também oferece *insights* práticos que podem ser diretamente aplicados por empresas para melhorar suas estratégias de *marketing* e vendas, promovendo uma abordagem mais personalizada e eficiente para atender às necessidades dos consumidores.

REFERÊNCIAS

ABIHPEC. **A Indústria de Higiene Pessoal, Perfumaria e Cosméticos.** Disponível em: https://abihpec.org.br/publicacao/panorama-do-setor/. Acesso em: 4 dez. 2023.

ABIHPEC; SEBRAE. **Tendências para o mercado de beleza**. Disponível em: https://digital.sebraers.com.br/blog/estrategia/tendencias-para-o-setor-de-beleza-em-2023/. Acesso em: 4 dez. 2023.

AMARAL, Fernando. **Introdução à ciência de dados**. Rio de Janeiro: Alta Books, 2016.

BEHRMAN, Kennedy R. **Fundamentos de Python para ciência de dados**. [Porto Alegre]: Grupo A, 2023. E-book. ISBN 9788582605974. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788582605974/. Acesso em: 26 nov. 2023.

BHARADHWAJ REDDY LEKIREDDY et al. **Market-Based Analysis: Apriori approach to analyze purchase patterns**. ICST Transactions on Scalable Information Systems, 4 jul. 2023.

BRAMER, M. **Principles of Data Mining.** London: Springer London, 2007.

CAO, L. **Data science: A comprehensive overview.** ACM computing surveys, v. 50, n. 3, p. 1–42, 2018.

DEDY DWIPUTRA et al. **Evaluating the Performance of Association Rules in Apriori and FP-Growth Algorithms: Market Basket Analysis to Discover Rules of Item Combinations.** Journal of World Science, v. 2, n. 8, p. 1229–1248, 30 ago. 2023.

GALVÃO, N. D.; MARIN, H. DE F. **Técnica de mineração de dados: uma revisão da literatura**. Acta Paulista de Enfermagem, v. 22, n. 5, p. 686–690, out. 2009.

GOLDSCHMIDT, R.; PASSOS, E. **Data mining: Um guia prático**. [s.l.] Gulf Professional Publishing, 2005.

GOLDSCHMIDT, R.; PASSOS, E.; BEZERRA, E. **Data Mining: Conceitos, Técnicas, Algoritmos, Orientações e Aplicações**. [s.l.] Elsevier Brasil, 2015.

GRUS, Joel. **Data Science do Zero.** [Rio de Janeiro]: Editora Alta Books, 2021. E-book. ISBN 9788550816463. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788550816463/. Acesso em: 25 nov. 2023.

HUNTER, J. D. **Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering,** v. 9, n. 3, p. 90–95, 2007.

LAMBERT, Kenneth A. **Fundamentos de Python: primeiros programas.** [São Paulo]: Cengage Learning Brasil, 2022. E-book. ISBN 9786555584301. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9786555584301/. Acesso em: 26 nov. 2023.

MACIEL, Francisco Marcelo de B. **Python e Django**. [Rio de Janeiro]: Editora Alta Books, 2020. E-book. ISBN 9786555200973. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9786555200973/. Acesso em: 25 nov. 2023.

MARIANO, Diego César B.; MARQUES, Leonardo T.; SILVA, Marcel S.; et al. **Data Mining**. [Porto Alegre]: Grupo A, 2021. E-book. ISBN 9786556900292. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9786556900292/. Acesso em: 12 nov. 2023.

MUNFARIJAH, V.; LUCIA, D. **Implementation of FP-Growth Algorithm in Determining Food Package Recommendation in Sunan Giri Ribs Meatball Restaurant**. International Journal of Computer Applications, v. 176, n. 24, p. 15–20, 15 maio 2020.

ŞIMŞEK, YILDIRIM MURAT. **Market basket analysis using apriori algorithm.** İstanbul: MEF Üniversitesi Fen Bilimleri Enstitüsü, 2018.

NETTO, Amilcar; MACIEL, Francisco. **Python para Data Science e Machine Learning Descomplicado**. [Rio de Janeiro]: Editora Alta Books, 2021. E-book. ISBN 9786555203172. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9786555203172/. Acesso em: 25 nov. 2023.

NUMPY. **What is NumPy? —** NumPy v1.19 Manual. 2022. Disponível em: https://numpy.org/doc/stable/user/whatisnumpy.html. Acesso em: 26 nov. 2023.

PROVOST, F.; FAWCETT, T. **Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking**. [s.l.] "O'Reilly Media, Inc.", 2013.

RASCHKA, S. MLxtend: **Providing machine learning and data science utilities and extensions to Python's scientific computing stack**. Journal of Open Source Software, v. 3, n. 24, p. 638, 22 abr. 2018.

RAUTENBACH, S.; DE KOCK, I.; GROBLER, J. **Data science for small and medium-sized enterprises: A structured literature review.** The South African Journal of Industrial Engineering, v. 32, n. 3, 2022.

REINSEL, D.; GANTZ, J.; RYDNING, J. Data Age 2025: **The Evolution of Data to Life-Critical Don't Focus on Big Data** (I. Headquarters, Ed.). Framingham, MA: IDC, 2017.

SciPy. **User Guide — SciPy v1.7.1 Manual**. 2023. Disponível em: https://docs.scipy.org/doc/scipy/tutorial/index.html#user-guide. Acesso em: 26 nov. 2023.

SEBRAE. **Tendências para o setor de beleza**. 2023. Disponível em: https://digital.sebraers.com.br/blog/estrategia/tendencias-para-o-setor-de-beleza-em-2023/. Acesso em: 4 dez. 2023.

VALE, J. et al. **A aplicação da inteligência artificial na indústria de cosméticos: cenário atual e oportunidades para o futuro.** Foco, v. 16, n. 6, p. e2225–e2225, 12 jun. 2023.

VALLEY, P.; BAKI, G.; MAJMUDAR, G. **AI Technology: Current and Future Applications in Cosmetics.** Cosmetics&Toiletries, v. 137, p. 18-23, 2022.

CABREIRA, P. **Análise comparativa e complementar do emprego de algoritmos de clusterização e associação em base de dados e-commerce**. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Produção) - Faculdade de Engenharia, Universidade Federal de Juiz de Fora, Juiz de Fora, p 82, 2023.

VIJAYARANI, S.; JANANI, R. **Text mining: open source tokenization tools: an analysis. Advanced Computational Intelligence: An International Journal**, p. 37-47, 2016.

WAHYUNINGSIH, R.; SUHARSONO, A.; IRIAWAN, N. **Comparison of Market Basket Analysis Method Using Apriori Algorithm, Frequent Pattern Growth (FP-Growth) and Equivalence Class Transformation (ECLAT) (Case Study: Supermarket "X" Transaction Data for 2021)**. Business and Finance Journal, v. 8, n. 2, p. 192–201, 30 nov. 2023.

ANEXO A – TERMO DE AUTENTICIDADE

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
FACULDADE DE ENGENHARIA

Termo de Declaração de Autenticidade de Autoria

Declaro, sob as penas da lei e para os devidos fins, junto à Universidade Federal de Juiz de Fora, que meu Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia de Produção é original, de minha única e exclusiva autoria. E não se trata de cópia integral ou parcial de textos e trabalhos de autoria de outrem, seja em formato de papel, eletrônico, digital, áudio-visual ou qualquer outro meio.
Declaro ainda ter total conhecimento e compreensão do que é considerado plágio, não apenas a cópia integral do trabalho, mas também de parte dele, inclusive de artigos e/ou parágrafos, sem citação do autor ou de sua fonte.
Declaro, por fim, ter total conhecimento e compreensão das punições decorrentes da prática de plágio, através das sanções civis previstas na lei do direito autoral[1] e criminais previstas no Código Penal[2], além das cominações administrativas e acadêmicas que poderão resultar em reprovação no Trabalho de Conclusão de Curso.

Juiz de Fora, 02 de Setembro de 2024.

Maria Luiza Damas Corrim	201949055
NOME LEGÍVEL DO ALUNO (A)	Matrícula
Maria Luiza D. Corrim	148.161.087-25
ASSINATURA	CPF

[1] LEI Nº 9.610, DE 19 DE FEVEREIRO DE 1998. Altera, atualiza e consolida a legislação sobre direitos autorais e dá outras providências.
[2] Art. 184. Violar direitos de autor e os que lhe são conexos: Pena - detenção, de 3 (três) meses a 1 (um) ano, ou multa.