UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

RICARDO FRANCO DE ALMEIDA FILHO

TÉCNICAS DE APRENDIZADO DE MÁQUINA E MINERAÇÃO DE TEXTOS APLICADAS PARA A PREVISÃO DA AVALIAÇÃO DE CONSUMIDORES DE E-COMMERCE

LONDRINA

2023

RICARDO FRANCO DE ALMEIDA FILHO

TÉCNICAS DE APRENDIZADO DE MÁQUINA E MINERAÇÃO DE TEXTOS APLICADAS PARA A PREVISÃO DA AVALIAÇÃO DE CONSUMIDORES DE E-COMMERCE

Application of machine learning and text mining techniques to predict e-commerce consumer evaluation

Trabalho de Conclusão de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia de Produção, Universidade Tecnológica Federal do Paraná (UTFPR).
Orientador: Pedro Rochavetz de Lara Andrade.

LONDRINA

2023

RICARDO FRANCO DE ALMEIDA FILHO

TÉCNICAS DE APRENDIZADO DE MÁQUINA E MINERAÇÃO DE TEXTOS APLICADAS PARA A PREVISÃO DA AVALIAÇÃO DE CONSUMIDORES DE E-COMMERCE

Trabalho de Conclusão de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia de Produção, Universidade Tecnológica Federal do Paraná (UTFPR).

Data de aprovação: 22/novembro/2023

_____

Pedro Rochavetz de Lara Andrade
Doutor
Universidade Tecnológica Federal do Paraná

_____

Bruno Samways dos Santos
Doutor
Universidade Tecnológica Federal do Paraná

_____

Rafael Henrique Palma Lima
Doutor
Universidade Tecnológica Federal do Paraná

LONDRINA

2023

AGRADECIMENTOS

Agradeço, em primeiro lugar, a Deus, por ser minha fonte de força ao longo dessa jornada acadêmica, em meio aos desafios e momentos de incerteza.

À minha família, que sempre esteve ao meu lado, apoiando-me incondicionalmente. A vocês, meus pais, avós, irmã e demais familiares, devo minha educação, valores e alicerces que me permitiram chegar até aqui.

À minha namorada, por todo o seu apoio, carinho e compreensão. Sua presença torna cada conquista mais significativa e cada desafio mais superável.

Aos amigos que compartilharam comigo os altos e baixos deste percurso, expresso minha sincera gratidão. Seus encorajamentos, discussões e risadas trouxeram leveza aos momentos mais desafiadores.

Aos professores que me guiaram e me ensinaram ao longo desta trajetória, em especial ao meu orientador Pedro Rochavetz. Suas orientações, conhecimento e incentivo foram fundamentais para o desenvolvimento deste trabalho e para o meu crescimento como estudante e como pessoa.

RESUMO

O *e-commerce* tem crescido nos últimos anos no Brasil, estando cada dia mais presente no cotidiano de milhões de pessoas. Em um mercado cada vez mais competitivo as empresas buscam se diferenciar proporcionado um melhor atendimento ao cliente, para isso a análise dos dados gerados por seus consumidores é vital, independentemente de seus formatos. A análise de sentimentos une conceitos de processamento de linguagem natural e aprendizagem de máquinas de forma a extrair as opiniões contidas nos textos através de uma classificação, tradicionalmente associada a um sentimento “positivo” ou “negativo”. Este trabalho visa analisar sentimentos de clientes de um determinado produto da *Amazon* utilizando técnicas de processamento de linguagem natural e aprendizado de máquina para a análise de avaliações/opiniões de consumidores, de forma a classificar seu sentimento em relação ao produto. Foram testados cenários para apurar o impacto de diferentes técnicas de pré-processamento nos resultados obtidos. Além disso, foram utilizadas cinco técnicas de aprendizagem de máquina para classificar os textos: Decision Tree, Naïve Bayes Bernoulli e Naïve Bayes Multinomial, K-nearest Neighbors e *Support Vector Machines*. Os resultados da aplicação dos diferentes cenários de pré-processamento mostraram que as técnicas afetam diretamente e de diferentes formas cada algoritmo. Os algoritmos obtiveram desempenhos considerados adequados nos quatro cenários testados, sendo que o melhor resultado geral foi obtido pelo Naïve Bayes Multinomial, que obteve 90%, 92%, 91% e 89% de acurácia em cada um dos cenários.

Palavras-chave: Mineração de Textos; Processamento de Linguagem Natural; Análise de Sentimento; Aprendizado de Máquina.

ABSTRACT

E-commerce has grown in recent years in Brazil, becoming increasingly present in the daily lives of millions of people. In an ever-competitive market, companies seek to differentiate themselves by providing better customer service, making the analysis of data generated by their consumers vital, regardless of its formats. Sentiment analysis combines concepts of natural language processing and machine learning to extract opinions contained in texts through classification, traditionally associated with a "positive" or "negative" sentiment. This thesis aims to analyze the sentiments of customers for a specific product on Amazon using techniques of natural language processing and machine learning to analyze consumer reviews/opinions, classifying their sentiment regarding the product. Different pre-processing techniques were tested to assess the impact on the obtained results. Additionally, five machine learning techniques were used to classify the texts: Decision Tree, Naïve Bayes Bernoulli and Naïve Bayes Multinomial, K-nearest Neighbors, and Support Vector Machines. The results of applying different pre-processing scenarios showed that the techniques directly and differently affect each algorithm. The algorithms achieved performances considered adequate in the four scenarios tested, with the overall best result obtained by Naïve Bayes Multinomial, achieving 90%, 92%, 91%, and 89% accuracy in each of the scenarios.

Keywords: Text Mining; Natural Language Processing; Sentiment Analysis; Machine Learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

PNL	Processamento de Linguagem Natural
RI	Recuperação de Informação
BoW	*Bag of Words*
TF-IDF	*Term Frequency - Inverse Document Frequency*
KNN	*K - Nearest Neighbors*
SVM	*Support Vector Machines*
SVC	*Support Vector Classifier*

SUMÁRIO

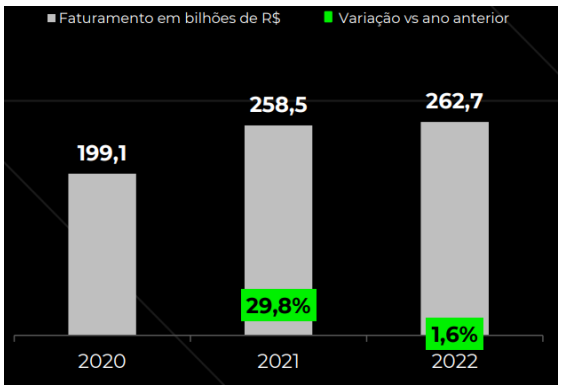
1. INTRODUÇÃO

Em um mundo globalizado em que o acesso à internet se torna a cada dia mais presente e necessário no cotidiano, as empresas têm a oportunidade de um novo segmento de mercado, o *e-commerce* (comércio eletrônico), em que o consumidor dos conteúdos online da empresa, ao se deparar com produtos de seu interesse, podem efetivar a compra pelos meios digitais, sem a necessidade de um vendedor lhe apresentando as ofertas e produtos.

Segundo dados da NielsenIQ Ebit (2023), houve em 2022 um aumento de 24% no número de consumidores em *e-commerce* no país, em comparação com o ano anterior. Tal forma de venda gera em conjunto uma grande quantidade de informações textuais fornecidas pelos usuários, como por exemplo: avaliações de produtos nos *websites* das companhias, respostas online em pesquisas de satisfação, postagens em *blogs* ou redes sociais sobre suas compras, além de interações com o Serviço de Atendimento ao Cliente (SAC).

Nesse contexto de pandemia global vivenciada nos últimos anos (COVID-19), observou-se um aumento nas vendas online, alavancas pelo isolamento necessário para evitar maior contágio. Como mostrado na Figura 1, dados divulgados pela NielsenIQ Ebit (2023) indicam um crescimento de 29,8% em faturamento em 2021 no setor de e-commerce, para 2022, ano de retorno a uma certa normalidade, houve um crescimento modesto de 1,6%, porém mantendo-se o cenário forte no segmento gerado pela pandemia, representando um novo recorde para o setor.

Figura 1 - Faturamento do e-commerce nos últimos anos



Fonte: NielsenIQ Ebit - Webshoppers 47ª Edição

Tal cenário traz relevância a este estudo, devido à mudança imposta pela pandemia aos consumidores em sua forma de compra, e por consequência, no meio como as empresas recebem o *feedback* de seu consumidor final.

Tan (1999) já afirmava que 80% das informações de uma empresa estão em formato textual, sendo necessário o surgimento de métodos de tratamento para essas informações. Visando realizar esse tratamento, é feita a utilização de técnicas de aprendizado de máquina para a análise de avaliações/opiniões de consumidores em formatos textuais não estruturados, de forma a classificar seu sentimento em relação ao produto, a fim de melhorar o marketing da empresa e aperfeiçoar a experiência do usuário com o produto, corrigindo seus defeitos, e evitando o desperdício de recursos em ações não eficazes.

Para Gupta e Lehal (2009) a análise dos dados textuais gerados pelos clientes, após um tratamento correto, pode fornecer informações de vital importância para as empresas sobre seus produtos ou serviços, além de uma melhor visão sobre os produtos e serviços da concorrência.

O comércio eletrônico no Brasil vem se consolidando nos últimos anos, tendo grande potencial de futuramente, se tornar o principal meio de realização de compras, não sendo mais necessária a ida a lojas físicas pelo comprador, seja qual for o produto. Com essa visão, este estudo buscou aplicar técnicas de aprendizado de máquina que as empresas podem utilizar para estarem mais preparadas para esse mercado exigente, competitivo e com uma alta demanda.

1.1 Objetivo Geral

Este trabalho tem como objetivo a aplicação de técnicas de mineração de textos em uma base de avaliações textuais de consumidores, avaliando os desempenhos das técnicas em classificar o sentimento expresso na avaliação.

1.1.1 Objetivos Específicos

- Coletar os dados textuais ainda não estruturados que serão utilizados na pesquisa através de sites de *e-commerce*;

- Realizar a estruturação dos dados por meio da aplicação de pré-processamento;
- Implementar técnicas de aprendizado de máquina de forma a preverem os sentimentos das avaliações textuais;
- Comparar através de métricas pré-estabelecidas as técnicas aplicadas com o intuito de definir a mais indicada para a previsão de sentimentos em dados textuais.

1.2 Estrutura do Trabalho

Após a introdução apresentada nesse capítulo, o Capítulo 2 apresenta a base conceitual utilizada para o desenvolvimento da metodologia e sua aplicação. Em seguida, o Capítulo 3, explica a metodologia de pesquisa utilizada. O Capítulo 4 é destinado à análise e discussão dos resultados da pesquisa. No Capítulo 5 são colocadas as considerações finais sobre o estudo.

2. REFERENCIAL TEÓRICO

Este capítulo aborda o embasamento teórico do trabalho, contendo conceitos necessários para seu entendimento, sendo eles: Comércio Eletrônico, Mineração de Textos, Análise de Sentimento e Aprendizado de Máquina, a seção finaliza com Trabalhos Correlatos.

2.1 Comércio Eletrônico

Segundo Turban e King (2004) o comércio eletrônico pode ser definido como o processo de compra, venda e troca de produtos, serviços e informações por redes de computadores. Para Chaffey (2006), essa definição mostra que o comércio eletrônico não se limita às transações financeiras entre empresas e consumidores, o autor considera todas as transações realizadas digitalmente entre uma empresa e um comprador como integrante do comércio eletrônico, sendo as solicitações de informações e *feedbacks* dos consumidores também consideradas como parte do processo.

Turban e King (2004) destacam três tipos de transações, podendo elas acontecerem no meio virtual ou não:

- *business-to-business* (B2B): Comércio entre empresas;
- *business-to-consumer* (B2C): Comércio entre empresa e consumidor;
- *consumer-to-consumer* (C2C): Comércio entre consumidores.

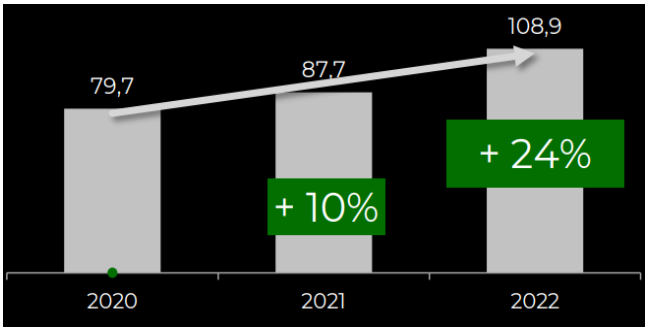
Na transação B2C, Turban e King ainda afirmam que a empresa não deve dar ênfase apenas no que está acontecendo no fechamento da compra do produto ou serviço pelo consumidor, mas no processo como um todo. Para se entender o consumidor, é preciso analisar o antes, o durante e o pós compra, para identificar características que podem favorecer a ocorrência de uma segunda compra e a fidelização do cliente (SILVA, OLIVEIRA, 2019).

Para Silva e Oliveira (2019) conhecer e traçar o perfil de seus consumidores é vital para que as empresas se qualifiquem de acordo com o público que deseja atender, facilitando o acesso dos clientes em encontrar os produtos e serviços que necessitam. Os autores concluem que tendo em vista o crescimento exponencial do

comércio eletrônico, é importante reconhecer que essa tendência é mundial e acontece em conjunto com a globalização.

Panda e Swar (2013) afirmam que com o crescimento na quantidade de pessoas conectadas ao meio virtual, as compras online se tornam uma forma alternativa de aquisição de bens ou serviços. O crescimento mostrado na Figura 2 pela NielsenIQ Ebit (2023) no número de consumidores online, auxiliado pelo cenário ainda recente de pandemia global, causado por um novo vírus (SARS-CoV-2) (Organização Mundial da Saúde, 2020), em que quase todas as atividades comerciais consideradas não essenciais, como lojas de roupas e restaurantes, foram fechadas, tornaram em tempos de incerteza, o meio de compra online a opção mais conveniente. Sendo a adesão dos vendedores ao comércio eletrônico um fator essencial para a sobrevivência de seu negócio, independentemente de seu setor de atuação.

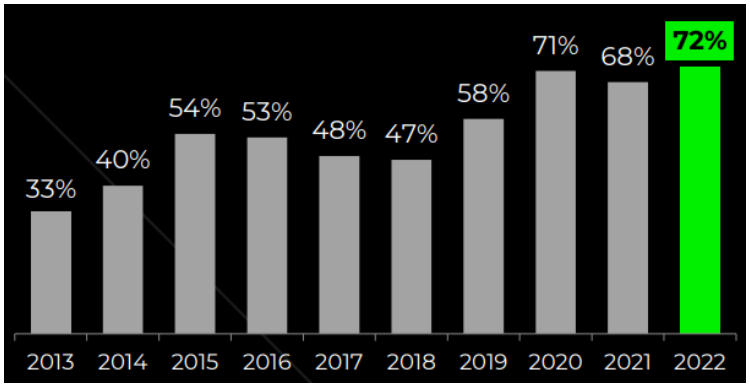
Figura 2 - Crescimento de consumidores do e-commerce (em bilhões)



Fonte: NielsenIQ Ebit - Webshoppers 47ª Edição

Dados recentes do Ebit (2023) também mostram o grande crescimento de vendas em sites internacionais nos últimos anos no mercado consumidor brasileiro, sendo acentuado nos anos de pandemia. Conforme a Figura 3, houve um aumento significativo nos números de compradores de *e-commerce* nesses sites ao longo dos anos.

Figura 3 - Percentual de consumidores brasileiros que fazem compras em sites internacionais



Fonte: NielsenIQ Ebit - Webshoppers 47ª Edição

2.2 Mineração de Textos

Com o crescimento do e-commerce pós-pandemia, as empresas têm enfrentado o desafio significativo de lidar com uma enorme quantidade de dados, incluindo informações textuais que expressam as opiniões dos clientes sobre produtos, serviços e a própria empresa. Sendo que a cada minuto essa quantidade cresce à medida que novos dados são gerados de diversas fontes. Um desafio para as organizações é o processamento de toda essa vasta informação, destacando-se as textuais, formato em que muitas vezes se encontram as opiniões dos clientes sobre a empresa, produto ou serviço.

Para Zhang (2009), essa informação se torna um recurso valioso quando tratada, pois, auxilia no processo de estabelecer uma melhor relação com os clientes alvo da empresa, além de gerar análises sobre o público em geral para a pesquisa de novos mercados.

Buscando o processamento desses dados textuais surge a mineração de textos, sendo uma variante da mineração de dados e conhecida como *Knowledge Discovery in Texts* (KDT). Utilizando-se de técnicas de análise e extração de dados a partir de textos, frases ou palavras, busca extrair padrões interessantes e não triviais ou a aquisição de conhecimento a partir de documentos em textos não estruturados (SEBASTIANI, 2002).

Segundo Sebastiani (2002), esta descoberta de conhecimento envolve diversas aplicações tais como análise de sentimentos, extração de informações, sumarização, classificação, agrupamentos, linguística computacional, dentre outras.

Embora a mineração de texto e a recuperação de informações (RI) compartilhem a área de Processamento de Linguagem Natural (PLN), a RI não é equiparada à mineração de texto. A RI é um ramo da ciência desenvolvido nas décadas de 1950 e 1960 que visa a busca de informações em documentos (OSINSKI E WEISS, 2005).

Aplicativos comuns de RI são encontrados em sistemas de biblioteca e mecanismos de pesquisa. Segundo Aggarwal (2018), no primeiro, o objetivo é fornecer aos usuários acesso a livros, periódicos e outros documentos, ao passo que, no segundo, os mecanismos de busca da Web, como Google e Yahoo, retornam páginas da Web de acordo com as palavras-chave inseridas pelos usuários.

Para Haravu e Neelameghan (2003), o uso de rótulos pré-determinados para recuperar informações está em nítido contraste com o caráter exploratório da mineração de texto. Em vez disso, o mecanismo de descoberta examina o corpus cada vez que uma consulta é feita e, portanto, é potencialmente capaz de descobrir relacionamentos e nós de rede previamente desconhecidos.

Mais importante ainda, retornar uma lista de resultados de pesquisa em RI, que podem ou não ser relevantes para o assunto em estudo, não traz uma contribuição substantiva para a descoberta de novos conhecimentos, um dos objetivos principais da mineração de texto. Estima-se que o Google, sem dúvida o mecanismo de pesquisa mais poderoso, poderia retornar apenas 30% dos documentos relevantes de todos os resultados. Em outras palavras, cada vez que uma pesquisa é realizada, mais de 70% dos documentos na saída são irrelevantes (RZHETSKY, SERINGHAUS, GERSTEIN, 2008).

A Análise de Sentimentos é uma das aplicações da mineração de textos e pode ser usada para determinar o sentimento expresso em um texto. Sendo útil para compreender opiniões, sentimentos e atitudes dos consumidores em relação a um produto ou serviço.

O fluxograma de um processo de mineração de texto para análise de sentimento bem-sucedido é montado por Patel e Soni (2012), uma adaptação desse fluxograma é apresentada na Figura 4.

Figura 4 - Fluxograma de um processo de mineração de textos

Fonte: Adaptado de Patel e Soni (2012)

2.3 Análise de Sentimento

Existe uma diferença sutil entre sentimento e opinião. O primeiro significa uma atitude ou julgamento baseado em um sentimento, enquanto o último é uma visão ou avaliação sobre um assunto específico (LIU E CHEN, 2015). Por exemplo, a frase "Não gosto de *tablets*" expressa um sentimento, enquanto a frase "Acho que tablets não são fáceis de utilizar" transmite uma opinião. No entanto, geralmente a análise de sentimento e a mineração de opinião são usadas de forma intercambiável.

Para Bahrainian e Dengel (2015) um processo típico de análise de sentimentos é composto de duas tarefas principais: primeiro, identificar se uma determinada frase ou sentença é subjetiva ou objetiva. Em segundo lugar, se a entidade textual for subjetiva, o algoritmo tentará localizar sua polaridade (positiva, negativa ou neutra). Os autores ainda citam três abordagens principais para detectar a polaridade de sentimento, conforme mostrado a seguir:

1. Métodos de aprendizado de máquina supervisionados: Nestes métodos, exemplos já classificados são fornecidos ao algoritmo para fins de treinamento;
2. Métodos de aprendizado de máquina não supervisionados: Nessas abordagens, o algoritmo tenta encontrar o padrão sem exemplos fornecidos. Um dos métodos populares do aprendizado não supervisionado é o aprendizado profundo, que usa redes neurais artificiais de várias camadas;
3. Métodos de nível de conceito: Esses métodos baseiam-se em conhecimento comum, como ontologias da *Web* ou redes de conceitos.

Em alguns casos, classificar o sentimento como positivo, negativo ou neutro pode ser desafiador. Alguns exemplos de obstáculos são citados por alguns pesquisadores:

- Sarcasmo: quando o escritor fez expressões sarcásticas, isso poderia enganar um pacote de software de mineração de texto regular (FELDMAN, 2013). Como exemplo de sarcasmo em que o algoritmo pode se enganar sobre o real sentimento do autor: “O ar-condicionado que comprei é tão bom que parece que estou sempre na Antártida”;
- Negação: A polaridade positiva é revertida por uma palavra negativa (ROJAS-BARAHONA, 2016). Por exemplo, “Esse sorvete não é bom”, neste caso, embora “bom” seja uma palavra positiva, a palavra “não” altera sua conotação;
- Mudança de significado: em alguns casos, uma palavra pode ter significados completamente distintos em frases diferentes (NAKAGAWA, INUI, KUROHASHI, 2010). Como exemplo temos: “Essa manga está doce” e “A manga dessa camiseta é muito curta”;

Segundo Liu e Chen (2015), o sentimento expresso por uma pessoa é representado por três variáveis:

1. Tipo de sentimento: pode ser, por exemplo, baseado na psicologia, baseado na linguagem ou baseado na pesquisa do consumidor;
2. Polaridade do sentimento: refere-se à orientação do sentimento e pode ser positiva (P) ou negativa (N) e (em alguns casos) neutra (O).
3. Intensidade da polaridade: representa os níveis de força com os quais a polaridade é expressa.

O objetivo principal da análise de sentimento é classificar um texto de acordo com seu conteúdo, tendo como a tarefa mais comum a classificação de polaridade, ou seja, dado um texto de entrada (T) determinar, por seu conteúdo, se contém uma opinião positiva (P) ou negativa (N). O problema descrito é considerado uma tarefa de classificação binária, podendo ser uma classificação multiclasse, caso uma categorização de polaridade neutra (O), texto sem sentimento negativo nem positivo, também for considerada.

Outro exemplo de tarefa de detecção de polaridade multiclasse é dado ao lidar com a intensidade da polaridade. Segundo Tsytsarau e Palpanas (2012), neste caso, a variável alvo não é nominal, mas uma variável ordinal / contínua tomando valores em uma faixa geralmente simétrica [−X, X], onde os valores positivos correspondem à polaridade positiva, os valores negativos correspondem à polaridade negativa, e o valor "0" (zero) corresponde a uma polaridade neutra, a Tabela 1 exemplifica uma classificação de polaridade e intensidade.

Tabela 1 - Exemplo de classificação de polaridade

Frase	Polaridade	Intensidade
Gostei muito do filme!	Positiva	2
Achei o filme legal.	Positiva	1
O filme foi indiferente para mim.	Neutra	0
Não gostei do filme.	Negativa	-1
Odiei o filme!	Negativa	-2

Fonte: Elaborado pelo Autor (2023).

Para Nandwani e Verma (2021) além da detecção de polaridade, outra tarefa central, relativamente nova para o campo da análise de sentimento, é a tarefa de reconhecimento de emoções, presente principalmente no contexto de pesquisa psicológica / emocional da análise de sentimento. Uma emoção é um sentimento especial que caracteriza um estado de espírito, como alegria, raiva, amor, medo e assim por diante. Nesse contexto, o tipo de emoção pode ser representado como um par (tipo de emoção, intensidade da emoção), em que o tipo de emoção expressa o nome da emoção (como felicidade e raiva), e a intensidade da emoção representaria a força expressada por ela (pouca, mediana ou muita por exemplo).

2.4 Pré-processamento de dados textuais

O pré-processamento de dados textuais desempenha um papel fundamental no campo de PLN, sendo a etapa que precede a análise e modelagem de textos. Essa fase envolve uma série de técnicas que visam aprimorar a qualidade e relevância dos dados textuais. Entre as práticas comuns de pré-processamento estão a remoção de pontuações, a conversão para minúsculas, a tokenização e a remoção de stopwords. Além disso, técnicas avançadas podem incluir a lematização ou o stemming para normalizar as palavras.

2.4.1 Tokenização

É a primeira etapa do pré-processamento e tem como objetivo dividir o texto em unidades, mais conhecidas como *tokens*. Essas unidades podem ser frases, símbolos, números, espaços, palavras ou termos compostos por mais de uma palavra (LEE, 2002). Quando a unidade é formada por uma palavra, é chamada de unigrama, quando formada por duas palavras, bigrama, e assim por diante. Este processo requer uma programação inteligente que seja capaz de compreender o contexto.

Lee (2002) afirma que os dados textuais são apenas um bloco de caracteres no início, sendo que os processos de recuperação de informações ou mineração de textos requerem as palavras do conjunto de dados. Sendo o requisito para um analisador a realização da tokenização dos documentos, mesmo o texto já estando armazenado em formatos legíveis pela máquina. O uso principal de tokenização é identificar as palavras-chave significativas.

Os desafios da tokenização dependem da linguagem utilizada, sendo separadas em sua maior parte em linguagens delimitadas por espaço, como as originadas do Latim, por terem o espaço como separador de palavras e linguagens não segmentadas, sendo exemplos as linguagens asiáticas, que utilizam símbolos sequenciais, requerendo para a aplicação do pré-processamento um conhecimento maior de informações léxicas e morfológicas sobre a linguagem (PALMER, 2010).

Comrie et al. (1996) sugere a existência de três categorias principais nas quais as estruturas de palavras podem ser colocadas, e cada categoria se apresenta tanto em sistemas de linguagem não segmentada, quanto em sistemas delimitados por espaço. Para os autores, a morfologia das palavras em um idioma pode ser:

- Isoladora: onde as palavras não se dividem em unidades menores;
- Aglutinante: nas quais as palavras se dividem em unidades menores (morfemas), com limites entre os morfemas;
- Flexional: onde os limites entre os morfemas não são claros e seus componentes podem expressar mais de um significado gramatical.

O autor conclui que a maioria das línguas possui vestígio dos três tipos, porém cita exemplos de línguas com predominância por um tipo: a linguagem chinesa mandarim é predominantemente isoladora, o japonês é fortemente aglutinante e o latim é amplamente flexional.

Em relação à tokenização em linguagens delimitadas por espaço, a maioria das ambiguidades existem entre o uso de sinais de pontuação, tais como ponto final, vírgulas, aspas, apóstrofos e hífens, uma vez que o mesmo sinal de pontuação pode servir para diferentes funções em uma mesma sentença.

A remoção de sinais de pontuação, acentos e caracteres especiais (colchetes, hífens, etc) são exemplos de processos necessários a serem feitos após a tokenização para a correta padronização. Além das remoções, diferentes formatos de número e hora, assim como abreviações e acrônimos devem ser padronizados.

Na frase presente na Figura 5 é possível demonstrar tais ambiguidades, ao colocar espaços, vírgulas e pontos finais como separadores dos tokens:

Figura 5 - Exemplo de ambiguidades de separadores de tokens

O celular foi comprado por R$ 2.000,00 na loja localizada na Av. Santos Dumont no dia 5.

Fonte: Elaborado pelo Autor (2023).

Sendo exemplos a vírgula em “2.000,00” representando o divisor decimal, cuja remoção alterará o valor numérico e a remoção do ponto em “Av.” deixando de caracterizar uma sigla.

2.4.2 *Stopwords*

A remoção de *stopwords* é uma importante técnica de pré-processamento com o intuito de retirar as palavras de pouco ou nenhum valor semântico, listas de *stopwords* comuns foram criadas para diversos idiomas, incluindo o português. A Figura 6 abaixo nos dá exemplos de *stopwords* da língua portuguesa.

Figura 6 - Exemplos de *stopwords*

a , à , ao , aos , aquela , aquelas , aquele , aqueles , aquilo , as, às , até , com , como , da , das , de , dela , delas , dele , deles , depois , do , dos , e , é , ela , elas , ele , eles , em , entre , era , eram , éramos , essa , essas , esse , esses , esta

Fonte: Biblioteca NLTK de *stopwords*

Porém, é necessário cuidado ao utilizar listas prontas, em algumas ocasiões raras, informações importantes podem ser perdidas devido à remoção incorreta da palavra de parada, como por exemplo a consideração da palavra “não” como *stopword* pode mudar completamente o sentido original da frase.

Raulji e Saini (2016) sugerem os seguintes passos para o procedimento da remoção de *stopwords*:

- Passo 1: O documento base é tokenizado e as palavras são armazenadas individualmente (*tokens*).
- Passo 2: Uma única *stopword* é lida da lista de *stopwords*
- Passo 3: A *stopword* é comparada com os *tokens*.
- Passo 4: Se encontrada em um ou mais *tokens*, eles são removidos.
- Passo 5: Após a remoção completa da *stopword*, retorna-se ao passo 2, seguindo para a próxima *stopword* até que se compare toda a lista.
- Passo 6: Finalizado todo o comparativo, os *tokens* restantes são mostrados, assim como as *stopwords* removidas, números de contagem de tokens restantes e *tokens* de *stopwords* também podem ser disponibilizados.

2.4.3 Lematização

É o processo de usar um conjunto de regras para reduzir as formas flexionais de uma palavra à sua forma raiz (*lemma*). Por exemplo, “amigos” e “amigão” podem ser alterados para “amigo” (IGNATOW, MIHALCEA, 2018). O *lemma* corresponde à forma singular no caso de um substantivo, à forma infinitiva no caso de um verbo e à forma positiva no caso de um adjetivo ou advérbio.

A lematização é um processo de normalização no qual diferentes variantes morfológicas de uma palavra são mapeadas no mesmo *lemma* para que possam ser analisadas como um único termo (KHYANI, 2021). Ao reduzir o número total de termos distintos, a complexidade do texto analisado é reduzida, trazendo benefícios para o processamento de texto.

2.4.4 *Stemming*

Stemming é uma técnica que utiliza um conjunto de regras para remover inflexões, como sufixos e prefixos. Por exemplo, “computador”, “computacional” e “computação” podem ser reduzidos ao mesmo radical, “comput” (IGNATOW, MIHALCEA, 2018).

Além de ser utilizada no PLN, também é comumente utilizada em muitos outros campos que envolvem linguística geral. O *stemming* fornece uma maneira eficiente de gerar palavras-chave genéricas para mecanismos de pesquisa ou *tags* para mapas conceituais.

Jivani (2011) explica que existem três grupos de algoritmos que realizam *stemming:*

- Algoritmos de Truncamento: Sendo considerado a forma mais simples de se realizar o *stemming*, sendo o corte de prefixos e sufixos.
- Algoritmos Estatísticos: Apesar de cortar os prefixos e sufixos na maioria das vezes, apenas realiza tal corte após aplicação de modelos estatísticos para analisar o *token*.
- Algoritmos Mistos: Sendo a combinação dos outros dois grupos, possibilita o ganho de velocidade na análise estatística após o corte inicial de prefixos e sufixos.

Paice (1994) pontua que existem dois erros a serem observados: *over-stemming* e *under-stemming*. *Over-stemming* ocorre quando duas palavras com radicais diferentes são derivadas da mesma raiz, sendo considerado um falso positivo. Já o *under-stemming* ocorre quando duas palavras que deveriam ser derivadas da mesma raiz não são, sendo um falso negativo.

Paice (1994) conclui que utilizar algoritmos mais leves no *stemming* reduz os erros de *over-stemming*, mas aumenta os erros de *under-stemming*. Sendo o inverso verdadeiro, algoritmos de *stemming* mais pesados reduzem os erros de *under-stemming* e aumentam os erros de *over-stemming*.

2.5 Técnicas de representação das variáveis

2.5.1 *Bag of Words*

A abordagem de *Bag of Words* (*BoW*) é uma técnica amplamente utilizada no campo do PLN e análise de texto. Tendo como premissa a desconsideração da estrutura gramatical e ordem das palavras, essa abordagem trata um texto como um conjunto desordenado de palavras (ZHANG, JIN E ZHOU, 2010). Ao aplicar a técnica, um vetor é construído a partir de todos os termos únicos presentes nos documentos analisados, sendo contabilizada a frequência de cada termo em cada documento.

Os valores incluídos na *BoW* podem ser modificados usando técnicas como contagem de termos, frequência de termos e frequência de documento inversa de frequência de termos (JULURU, SHIH e MURTHY, 2021). Os autores concluem que a experiência prévia e a experimentação orientarão nas decisões sobre quais técnicas específicas deverão ser escolhidas.

Um exemplo de *BoW* é montado utilizando de base as frases:

- Frase 1 - O homem comprou um celular com uma ótima câmera
- Frase 2 - A foto tirada pela câmera do celular ficou ótima

Aplicando-se a tokenização e a remoção das *stopwords*, tem-se o seguinte vetor de palavras que serão nossas colunas na *BoW*, formando a Tabela 2.

Tabela 2 - Exemplo de *BoW*

	homem	**foto**	**Comprou**	**tirada**	**celular**	**Câmera**	**ficou**	**ótima**
Frase 1	1	0	1	0	1	1	0	1
Frase 2	0	1	0	1	1	1	1	1

Fonte: Elaborado pelo Autor (2023).

2.5.2 *Term Frequency - Inverse Document Frequency*

O *Term Frequency - Inverse Document Frequency* (*TF-IDF*) é uma medida que considera a importância de uma palavra em relação a um documento específico e ao *corpus* como um todo. O *TF* calcula a frequência de uma palavra em um documento, enquanto o *IDF* mede a raridade dessa palavra em todo o corpus.

A multiplicação desses dois fatores resulta no valor *TF-IDF*, que representa a relevância da palavra em um determinado documento. Ao utilizar o *TF-IDF* na análise de sentimento, podemos identificar palavras-chave que possuem um alto valor *TF-IDF* em documentos positivos ou negativos, indicando a contribuição dessas palavras na polaridade emocional.

A Equação 1 representa o cálculo realizado:

$$w_{i,j} = tf_{i,j} \times log(\frac{N}{df_i}) \quad (1)$$

em que:

- $w_{i,j}$ é a importância do termo i no documento j;
- $tf_{i,j}$ é o número de ocorrências do termo i no documento j;
- df_i é o número de documentos que contém o termo i;
- N é o número total de documentos.

A Figura 7 demonstra um exemplo simplificado de utilização do *TF-IDF* na análise de sentimento.

Figura 7 - Exemplo de aplicação do *TF-IDF*

Corpus

Frase 1	O gato pulou sobre o cachorro
Frase 2	O gato pulou sobre o outro gato.
Frase 3	O cachorro pulou sobre o gato.

TF (Gato , Frase 1) = 1 / 6 = 0,17	**TF-IDF** = 0,17 x 1 = **0,17**
TF (Gato , Frase 2) = 2 / 6 = 0,33	**TF-IDF** = 0,33 x 1 = **0,33**
TF (Gato , Frase 3) = 1 / 6 = 0,17	**TF-IDF** = 0,17 x 1 = **0,17**

IDF (Corpus , Gato) = *log* (3 / 3) = 1

Fonte: Elaborado pelo próprio autor (2023).

2.6 Aprendizado de Máquina

Nos tempos da tecnologia moderna, em que temos uma grande quantidade de informações estruturadas e não estruturadas à disposição, o aprendizado de máquina surge como uma subárea da inteligência artificial envolvendo o desenvolvimento de algoritmos autodidatas que ganham conhecimento a partir de dados previamente fornecidos para o cálculo de previsões (RASHKA, 2015).

Segundo Naqa e Murphy (2015), um algoritmo de aprendizado de máquina é um processo computacional que utiliza dados de entrada para alcançar uma tarefa desejada sem ser programado em sua totalidade para produzir um determinado resultado. Sendo algoritmos adaptáveis, eles automaticamente se alteram em sua arquitetura por meio de experiências anteriores, justificando o termo aprendizado, para que se tornem cada vez melhores na realização da tarefa desejada.

O cenário ideal do aprendizado de máquina é replicar a maneira como os seres humanos aprendem a processar sinais sensoriais (de entrada) para realizar uma meta. Esse objetivo pode ser uma tarefa de reconhecimento de padrões, na qual o algoritmo deseja distinguir maçãs de laranjas. Cada maçã e laranja é única, mas ainda estamos capazes (geralmente) de distinguir um do outro (NAGA E MURPHY, 2015).

Em vez de codificar uma máquina com representações exatas de maçãs e laranjas, ele pode ser programado para que aprenda a distingui-las através da experiência repetida com maçãs reais e laranjas. Sendo este um bom exemplo de aprendizado supervisionado, em que cada treinamento de dados de entrada (cor, forma, odor etc.) é conectado com seu rótulo de classificação conhecido (maçã ou laranja).

Permitindo ao algoritmo lidar com semelhanças e diferenças quando os objetos a serem classificados possuem muitas propriedades variáveis dentro de suas próprias classes, mas têm qualidades fundamentais que as identificam. Mais importante ainda, um algoritmo supervisionado bem-sucedido deve ser capaz de reconhecer uma maçã ou uma laranja que nunca viu antes.

O processo de adaptação do algoritmo de um aprendizado supervisionado é chamado de treinamento, no qual são fornecidas amostras de dados de entrada juntamente com os resultados desejados (MAHESH, 2020). O algoritmo então se configura de forma a não apenas produzir o resultado desejado quando apresentado com as entradas de treinamento, mas de modo a generalizar os dados de entrada, possibilitando a produção do resultado desejado a partir de novos dados. Tal como acontece com os humanos, um bom algoritmo pode praticar o aprendizado “ao longo de sua vida” à medida que processa novos dados e aprende de seus erros.

O principal objetivo do aprendizado supervisionado é aprender com uma base de dados previamente estabelecida de forma que permita a previsão de dados não vistos e/ou futuros, supervisionado se refere às amostras já conhecidas. Considerando o exemplo do filtro de spam de um determinado e-mail, nós podemos treinar o modelo de forma que ele classifique novos e-mails como spam ou não (classificação binária, o algoritmo define regras para distinguir entre duas classes possíveis), sendo esse um exemplo de tarefa de classificação, outra subcategoria é a regressão, onde o valor retornado é sempre contínuo (RENUKA ET AL, 2011).

A classificação tem como objetivo predizer os “rótulos” de classe previamente definidos das novas instâncias baseada nas observações passadas. Já na regressão recebemos uma variável explicativa e uma variável contínua de saída (resposta), tentamos então achar a correlação entre elas de forma que possamos predizer as futuras saídas. Por exemplo, imaginemos que queremos prever as notas finais de uma turma de estudantes, se houver uma relação entre o tempo que passaram estudando para a prova e a nota final, nós poderíamos usar como dados de treinamento para predizer as notas finais dos futuros alunos que irão fazer a prova (GHADAVI E PATEL, 2017).

Um segundo tipo de aprendizado de máquina é o chamado de algoritmo não supervisionado. Este pode ter o objetivo de tentar atirar um dardo no centro de um alvo (COULAND, HAMON E GEORGE, 2020). O algoritmo tem uma variedade de graus de escolha no mecanismo que controla o caminho do dardo. Ao invés de ter calculado exatamente o grau correto de acerto, o algoritmo pratica o lançamento do dardo.

Para cada tentativa, os graus são ajustados para que o dardo se aproxime cada vez mais do centro do alvo. Isso não é supervisionado no sentido de que o treinamento não associa uma determinada configuração de entrada a um determinado resultado. O algoritmo encontra seu próprio caminho a partir de seu treinamento. Um algoritmo não supervisionado bem-sucedido deve ser capaz de ajustar o grau para um cenário, por exemplo, de mudança na posição do alvo.

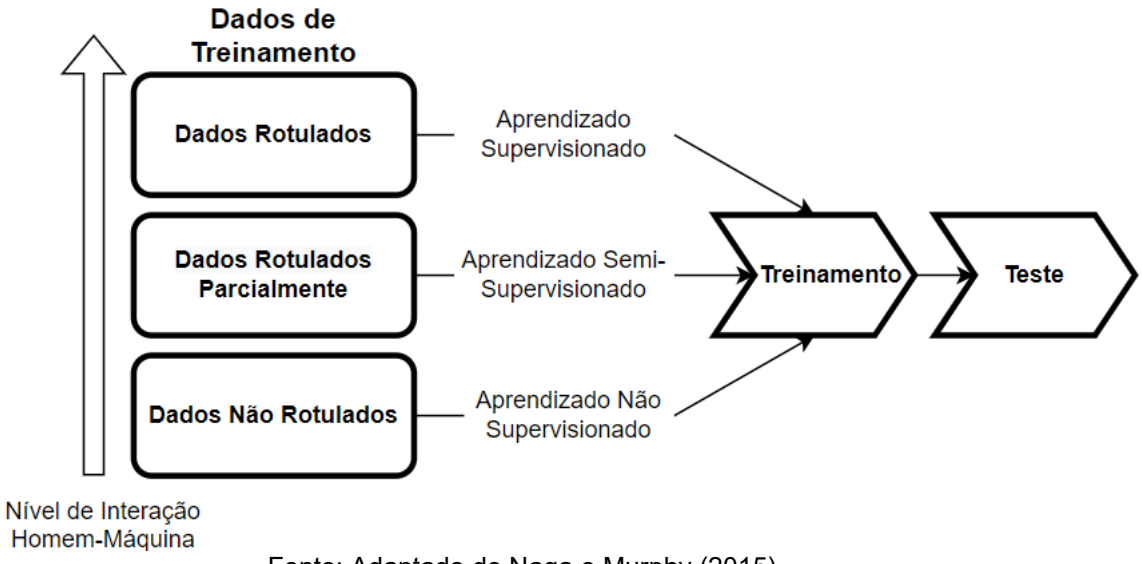
Um terceiro tipo de aprendizado de máquina é o aprendizado semi-supervisionado, onde parte da os dados são rotulados e outras partes não são

rotuladas. Nesse aprendizado, a peça rotulada pode ser usada para auxiliar o aprendizado da parte não rotulada, sendo o algoritmo que simula melhor o método como os humanos desenvolvem suas habilidades.

No aprendizado não supervisionado, como não existem dados previamente rotulados, as técnicas usadas buscam extrair informações importantes da nossa base de dados sem o conhecimento prévio de uma variável de saída. Um método utilizado é a clusterização em que organizasse as informações em subgrupos que compartilham determinadas similaridades, mas são mais diferentes dos outros subgrupos, sendo essa técnica também chamada de classificação não supervisionada. Sendo ótima para estruturar as informações e estabelecer relações entre os dados (RASHKA, 2015). Como exemplo, a formação de grupos de consumidores que têm interesses comuns, de forma a criar planos de marketing específicos para cada grupo. A Figura 8 é uma representação feita por Naga e Murphy (2015) dos tipos de aprendizado em relação aos seus dados de treinamento.

Para Kubat (2017), além de uma boa noção das forças e fraquezas dos métodos de aprendizado e suas técnicas, as peculiaridades dos diferentes casos devem ser analisadas. De forma que seja previsto previamente que circunstâncias algumas técnicas irão ser bem-sucedidas ou não. Apenas dessa forma eles poderão fazer as escolhas corretas quando lidarem com aplicações reais.

Figura 8 - Tipos de Aprendizado de Máquina de acordo com a natureza dos dados de treinamento



Fonte: Adaptado de Naga e Murphy (2015).

2.7 Técnicas de Classificação

Esse tópico explorará as técnicas de classificação, que são fundamentais para o entendimento do funcionamento e aplicação desses algoritmos em uma variedade de contextos. Os algoritmos de classificação têm a capacidade de automatizar a tarefa de agrupar dados em categorias distintas, com base em critérios predefinidos e diferentes abordagens conforme a técnica base do algoritmo. As técnicas apresentadas serão: *Decision Tree*, *K-Nearest Neighbour*, *Support Vector Machines* e *Naïve Bayes*.

2.7.1 *Decision Tree*

Decision Tree (ou Árvore de decisão) é uma técnica que classifica as instâncias classificando-as com base em valores de condições. Cada nó em uma árvore de decisão representa uma condição em uma instância a ser classificada, e cada ramo representa um valor que o nó pode assumir. As instâncias são classificadas a partir do nó raiz e classificadas com base em seus valores de características (MURTHY, 1998).

Murthy (1998) afirma que existem vários métodos para encontrar o recurso que melhor divide os dados de treinamento, mas a maioria dos estudos sobre o tema concluiu que não existe um método melhor. Sendo a comparação de métodos individuais ainda importante para se decidir qual métrica deve ser usada em um conjunto de dados específico. O mesmo procedimento é então repetido em cada divisão dos dados, criando ramos até que os dados de treinamento sejam divididos em subconjuntos da mesma classe.

Para Song e Ying (2015) uma dificuldade encontrada em uma árvore de decisão ocorre quando a mesma super ajusta (*overfitting*) os dados de treinamento se outra hipótese h2 existe que tem um erro maior que a hipótese h1 quando testada nos dados de treinamento, mas um erro menor que h1 quando testada em todo o conjunto de dados. Se as duas árvores empregam os mesmos tipos de testes e têm a mesma precisão de previsão, aquele com menos folhas é geralmente escolhido. A maneira mais simples de lidar com o *overfitting* é pré-podar a árvore de decisão, não permitindo que ela cresça até seu tamanho máximo.

2.7.2 *K-Nearest Neighbor*

O algoritmo *K-Nearest Neighbor* (*KNN*) é baseado no princípio de que as instâncias dentro de um conjunto de dados geralmente existirão próximo a outras instâncias que possuem propriedades semelhantes, sendo considerado um dos algoritmos de aprendizado baseado em instâncias mais simples.

Se as instâncias são marcadas com um rótulo de classificação, então o valor do rótulo de uma instância não classificada pode ser determinado observando a classe de seus vizinhos mais próximos (WETTSCHERECK ET AL., 1997). O *KNN* localiza as K instâncias mais próximas da instância de consulta e determina sua classe com o rótulo da classe mais frequente nos vizinhos

O poder do *KNN* foi demonstrado em vários domínios reais, mas existem algumas ressalvas sobre a utilidade do *KNN*, tais como: eles têm grandes requisitos de armazenamento, são sensíveis à escolha da função de similaridade que é usada para comparar instâncias, não têm uma maneira de escolher K, exceto por validação cruzada ou semelhante, técnica computacionalmente cara (GUO ET AL., 2003).

O autor lista dois motivos pelos quais o *KNN* pode classificar incorretamente uma instância de consulta:

- Quando o ruído está presente na localidade da instância da consulta, a(s) instância(s) ruidosa(s) ganha(m) os votos da maioria, resultando na previsão da classe incorreta. Um K maior poderia resolver isso problema.
- Quando a região que define a classe é muito pequena, ocasionado que as instâncias pertencentes à classe que circundam a região ganham a maioria dos votos. Um K menor poderia resolver este problema.

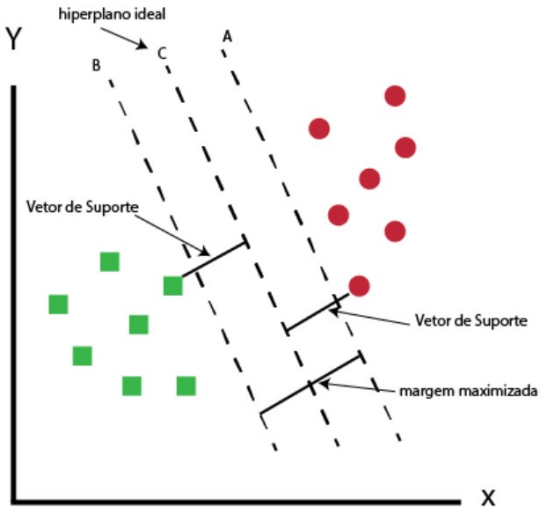
Wettschereck et al. (1997) investigaram o comportamento do *KNN* na presença de instâncias ruídos. Os experimentos mostraram que o desempenho do *KNN* não foi sensível a escolha exata de K quando K era grande. Eles descobriram que para pequenos valores de K, o algoritmo *KNN* foi mais robusto do que o algoritmo do vizinho mais próximo único (*1NN*) para a maioria dos grandes conjuntos de dados testados. No entanto, o desempenho do *KNN* foi inferior ao alcançado pelo *1NN* em conjuntos de dados pequenos (menores que 100 instâncias).

Okamoto e Yugami (2003) representaram a precisão de classificação esperada de KNN como uma função de características de domínio, incluindo o número de instâncias de treinamento, o número de atributos relevantes e irrelevantes, a probabilidade de cada atributo, a taxa de ruído para cada tipo de ruído, e K. Eles também exploraram as implicações comportamentais das análises, apresentando os efeitos das características do domínio na precisão esperada de *KNN* e no valor de K para domínios artificiais.

2.7.3 *Support Vector Machines*

A técnica *Support Vector Machines* (*SVM*) busca encontrar um hiperplano de separação ótimo que maximize a margem entre diferentes classes de dados. Ao criar a maior distância possível entre o hiperplano de separação e as instâncias em qualquer lado dele, Cristianini e Shawe-taylor (2000) provam que ocorre uma redução no limite superior do erro de generalização esperado.

Figura 9 - Exemplo de hiperplano ideal ao utilizar o algoritmo SVM



Fonte: Cavalcanti (2019)

No caso de dados linearmente separáveis, uma vez encontrado o hiperplano de separação ótimo, pontos de dados que se encontram em sua margem são conhecidos como pontos de vetor de suporte e a solução é representada como uma combinação linear apenas desses pontos. Outros pontos de dados são ignorados para que a complexidade do modelo de um *SVM* não seja afetada pelo número de recursos encontrados nos dados de treinamento (o número de vetores de suporte selecionados pelo algoritmo de aprendizado *SVM* é geralmente pequeno). Por esta

razão, o *SVM* é adequado para lidar com tarefas de aprendizagem onde o número de recursos é grande em relação ao número de instâncias de treinamento (DERIS, ZAIN E SALLEHUDDIN, 2011).

Mesmo que a margem máxima permita que o *SVM* selecione entre vários candidatos hiperplanos, para muitos conjuntos de dados, o *SVM* pode não encontrar nenhum hiperplano de separação porque os dados contêm instâncias classificadas incorretamente. O problema pode ser resolvido usando uma margem que aceita alguns erros de classificação das instâncias de treinamento (VEROPOULOS ET AL. 1999).

2.7.4 *Naïve Bayes*

Naïve Bayes é um algoritmo de aprendizado simples que utiliza o Teorema de Bayes juntamente com a suposição de que os atributos são condicionalmente independentes dada a classe. Devido a essa suposição estar frequentemente errônea, o nome *Naïve* (ingênuo) é dado ao algoritmo. No entanto, outras características do algoritmo fazem com que seja amplamente aplicado em prática.

Naïve Bayes fornece um mecanismo para usar as informações de dados de treinamento para estimar a probabilidade posterior P(y | x) de cada classe y dado um objeto x. Assim que criadas tais estimativas, podem ser usadas para classificação ou outras aplicações de suporte à decisão. Como mostrado na Equação 2:

$$\mathrm{P}(y \mid \mathbf{x}) = \mathrm{P}(y)\mathrm{P}(\mathbf{x} \mid y)/\mathrm{P}(\mathbf{x}) \qquad (2)$$

em que:

- $\mathrm{P}(y \mid \mathbf{x})$ é probabilidade de y acontecer dado que $\mathbf{x}$ ocorreu;
- $\mathrm{P}(y)$ é a probabilidade de y ocorrer;
- $\mathrm{P}(\mathbf{x} \mid y)$ é probabilidade de $\mathbf{x}$ acontecer dado que y ocorreu;
- $\mathrm{P}(\mathbf{x})$ é probabilidade de $\mathbf{x}$ ocorrer, $\mathrm{P}(\mathbf{x}) \neq 0$.

Segundo Webb et al (2010), são características do algoritmo Naïve Bayes:

- Eficiência computacional: O tempo de treinamento é linear tanto em relação ao número de exemplos de treinamento e o número de atributos e o tempo de

classificação é linear em relação ao número de atributos e não é afetado pelo número de exemplos de treinamento.

- Baixa variância ao custo de um viés alto.
- Aprendizagem incremental: *Naïve Bayes* opera a partir de estimativas de probabilidades de baixa ordem derivados dos dados de treinamento. Podendo ser atualizados com novos dados de treinamento conforme são adquiridos.
- Predição direta de probabilidades posteriores.
- Robustez face ao ruído: *Naïve Bayes* usa sempre todos os atributos para todas as previsões e, portanto, é relativamente insensível ao ruído nos exemplos a serem classificados. A mesma insensibilidade vale para os dados de treinamento.
- Robustez diante de valores ausentes: por utilizar sempre todos os atributos para todas as previsões, se um valor de atributo estiver faltando, a informação de outro atributo ainda é utilizada, resultando em uma pequena degradação no desempenho.

2.8 Métricas para avaliação dos resultados

Nesta seção, serão apresentas as métricas de avaliação mais utilizadas em uma pesquisa que utiliza aprendizado de máquina, sendo a Matriz de Confusão a ferramenta que permite visualizar o desempenho de um modelo de classificação, e a Precisão, a Sensibilidade, o *F1-Score* e a Acurácia as métricas de avaliação.

2.8.1 Matriz de Confusão

A matriz de confusão é uma forma de avaliar modelos de classificação, para que se possa analisar métricas específicas de um algoritmo, comparando esse algoritmo com outros algoritmos. Busca comparar os verdadeiros positivos e os verdadeiros negativos em uma amostra de teste com os dados positivos previstos e negativos previstos, uma variedade de estatísticas resumidas pode ser calculada. Abaixo uma breve explicação dos termos:

- Verdadeiros Positivos (VP): casos em que a previsão foi correta em relação a ocorrência do evento.
- Verdadeiros Negativos (VN): casos em que a previsão foi correta em relação a não ocorrência do evento.

- Falsos Positivos (FP): casos em que a previsão não foi correta em relação a ocorrência do evento, conhecidos como "erros do tipo 1".
- Falsos Negativos (FN): casos em que a previsão não foi correta em relação a não ocorrência do evento, conhecidos como "erros do tipo 2".

O autor conclui que muitos estudos incluem uma forma limitada dessas informações. Por exemplo, fornecendo apenas a matriz de confusão, de forma a possibilitar o cálculo das métricas não apresentadas, para comparativo de resultados entre os modelos.

Tabela 3 - Exemplo de Matriz de Confusão

Matriz de Confusão		**Previsto**		**Total**
		Negativo	**Positivo**	
Real	**Negativo**	15 – VN	10 - FP	25
	Positivo	5 – FN	20 - VP	25
Total		20	30	50

Fonte: Elaborado pelo Autor (2023).

Na Tabela 3, podemos observar um cenário no qual ocorrem 35 previsões corretas, sendo 15 de verdadeiros negativos e 20 verdadeiros positivos. Em uma amostra de 50 previsões, temos então 15 previsões erradas, com 5 representando os falsos negativos e 10 representando os falsos positivos.

2.8.2 Métricas de avaliação

São algumas das métricas mais comumente calculadas a partir da matriz de confusão:

- Precisão**:** É a quantidade de chamadas corretas (verdadeiro-positivo e verdadeiro-negativo) que foram feitas em proporção ao conjunto de dados total. O valor preditivo positivo ou precisão é a probabilidade de que a predição positiva seja realmente positiva; da mesma forma, o valor preditivo negativo é a probabilidade de que a predição negativa é realmente negativa. Seu cálculo é apresentado na Equação 3.

- Sensibilidade**:** A taxa de verdadeiros positivos também é conhecida como *recall* ou sensibilidade e é a probabilidade de o modelo detectar casos verdadeiramente positivos. Seu cálculo é apresentado na Equação 4.

- *F1-Score*: A pontuação F1 é função da taxa de verdadeiro-positivo e do valor preditivo positivo para fornecer uma indicação geral do desempenho do classificador. Seu cálculo é apresentado na Equação 5.

- Acurácia**:** É a medida que mostra o quão preciso um modelo é, indicando a proporção de previsões corretas em relação ao número total de previsões feitas. Seu cálculo é apresentado na Equação 6.

$$precisão = \frac{VP}{VP + FP} \quad (3)$$

$$sensibilidade = \frac{VP}{VP + FN} \quad (4)$$

$$f1 = 2 * \frac{precisão * sensibilidade}{precisão + sensibilidade} \quad (5)$$

$$acurácia = \frac{VP + VN}{VP + FN + VN + FP} \quad (6)$$

2.9 Trabalhos Correlatos

Vários estudos analisaram a influência de vários métodos de pré-processamento e aprendizado de máquina na classificação de textos. Lematização, *stemming* e remoção de *stopwords* foram utilizados por Toman et al (2006) usando apenas o classificador *MultinomialNB* em dois conjuntos de dados: 8000 documentos em inglês selecionados da *Reuters Corpus* (Volume 1) divididos em seis categorias e 8.000 documentos tchecos fornecidos por Agência de Notícias Tcheca divididos em cinco categorias. Eles concluíram que a melhor abordagem de pré-processamento foi apenas utilizando a remoção de *stopwords*, em que ocorria a melhoria a precisão da classificação na maioria dos casos. Além disso, a lematização e o *stemming* foram mais negativos do que positivos para ambas as línguas.

Já Méndez et al (2005) utilizou de remoção de *stopwords*, lematização e esquemas de tokenização diferentes na filtragem de e-mails de spam para dois corpora de e-mails. Eles usaram três métodos de aprendizado de máquina: *Naïve Bayes*, *Decision Tree* e *SVM*. A principal conclusão foi que o desempenho do *SVM* é surpreendentemente eficaz quando a remoção de *stopwords* e o *stemming* não

são feitos. Sendo um dos motivos pontuados pelos autores a raridade de *stopwords* em mensagens de spam e não devem ser removidas para melhorar o desempenho da filtragem.

Srividhya e Anitha (2013) avaliaram quatro métodos de pré-processamento: remoção de *stopwords*, *stemming*, ponderação *TF-IDF* e frequência de documentos. Suas principais conclusões foram que a remoção de *stopwords* melhora o desempenho da classificação, e o *TF-IDF* é necessário para criar o arquivo de índice a partir dos termos resultantes.

Haddi et al. (2010) investigaram o papel do pré-processamento de texto na análise de sentimentos de dois conjuntos de dados on-line de resenhas de filmes (Dat-1400 e Dat-2000). Eles usaram uma combinação de diferentes métodos de pré-processamento (remoção de *tags* HTML, remoção de sinais não alfabéticos, remoção de espaços em branco, expansão de abreviação, lematização e remoção de *stopwords*) para reduzir o ruído no texto. Eles concluem que a análise de sentimentos pode ser significativamente melhorada com a utilização do *SVM*.

Ayedh et al. (2016) investigaram o efeito de três métodos de pré-processamento (remoção de *stopwords*, lematização e normalização de certas letras árabes que têm formas diferentes na mesma palavra) para um corpus interno contendo 32.620 documentos de notícias dividido em dez categorias baixadas de diferentes sites de notícias árabes. No estudo do autor, três métodos de aprendizado de máquina foram aplicados: *Naïve Bayes*, *KNN* e *SVM*. A análise experimental revelou que o pré-processamento tem um impacto significativo na precisão da classificação, especialmente com a complicada estrutura morfológica da língua árabe. A escolha de combinações apropriadas de tarefas de pré-processamento proporciona uma melhoria significativa na precisão dependendo do tamanho dos dados e os métodos de aprendizado de máquina escolhidos.

Uysal e Gunal (2014) estudaram o impacto do pré-processamento na classificação de textos usando quatro métodos: tokenização, remoção de *stopwords*, lematização e conversão para letras minúsculas. Todas as combinações possíveis dos métodos de pré-processamento foram aplicadas em quatro conjuntos de dados: e-mails em turco, e-mails em inglês, notícias em turco e notícias em inglês. A

principal conclusão dos autores foi que combinações apropriadas de tarefas de pré-processamento, dependendo do conteúdo da base e da linguagem, podem proporcionar uma melhoria significativa na precisão da classificação, enquanto combinações inadequadas podem degradar a precisão. Diferentemente de outros autores, seus resultados concluem que as *stopwords* não devem ser removidas, tendo relevância na correta classificação dos textos.

3. METODOLOGIA

Neste capítulo, a metodologia do trabalho é descrita em quatro partes. Na seção 3.1 a coleta de dados é explicada, seguida pela seção 3.2, que detalha o pré-processamento dos dados, que envolve a limpeza e organização dos dados. Na seção 3.3 a representação numérica dos textos é apresentada através da aplicação dos modelos *BoW* e *TF-IDF*. O capítulo finaliza com a seção 3.4 referente a aplicação das técnicas de aprendizado de máquina.

3.1 Coleta de Dados

Neste estudo, foram coletados dados relacionados a uma *webcam* vendida na *Amazon*, sendo 435 avaliações textuais. Para a extração dos dados foi utilizado um *webscraper* com o auxílio da biblioteca *Selenium* disponível na linguagem de programação *Python*. Importante ressaltar que toda a extração, pré-processamento e modelos de análise de sentimentos empregados nos dados foram desenvolvidos em *Python3*.

O *webscraper* é um programa (ou *script*) que extrai dados de páginas da web de forma automatizada. Projetado de forma a navegar por diferentes páginas, fazer o download do conteúdo HTML e extrair informações específicas desejadas, como texto, imagens, links ou qualquer outro elemento relevante.

A biblioteca *Selenium* é uma ferramenta popular para automação de testes, sendo também amplamente utilizada como parte de um *webscraper* para interagir com páginas da web de forma automatizada. Permitindo controlar um navegador web real, como o Chrome, Firefox ou Safari, para simular ações humanas, como clicar em botões, preencher formulários, rolar a página e extrair dados.

Foram coletadas as avaliações comentadas dos compradores do produto, compondo assim uma base em formato *comma separated values* (ou CSV) com as seguintes colunas: data da postagem, título da avaliação, avaliação escrita e número de estrelas postado (de 1 a 5).

3.2 Pré-Processamento dos Dados

Inicialmente foi feita a divisão das 435 avaliações em dois grupos, sendo as avaliações com número de estrelas menores que 4 consideradas negativas e maiores ou iguais a 4 consideradas positivas. Resultando em 277 avaliações positivas e 158 negativas.

Tendo como base de sentimento essa nova classificação, iniciou-se a primeira etapa de pré-processamento do texto de avaliação, todas as etapas tiveram como objetivo a redução dos termos, de forma que ao fim do pré-processamento restem apenas os termos mais relevantes a análise de sentimento. A partir da função *isalnum* foi possível a remoção das pontuações, a função verifica se os caracteres da avaliação são alfanuméricos, ou seja, se são letras (maiúsculas ou minúsculas) ou números. De forma a evitar a perda dos espaços entre os termos já que o espaçamento não é considerado um caractere alfanumérico (será utilizado depois para a montagem dos tokens), cada termo não alfa numérico foi substituído por um espaço em branco.

A segunda etapa do pré-processamento buscou a remoção dos acentos, a partir do uso dos códigos *unicodes*, que é o padrão de codificação de caracteres em que se atribui um número único para cada caractere usado na escrita, incluindo letras, dígitos, símbolos, pontuação e caracteres especiais de diferentes alfabetos e sistemas de escrita. No processo, o caractere acentuado tem seu *unicode* dividido em dois *unicodes:* um sendo do caractere sem acento, e o outro sendo o *unicode* do acento. Divididos todos os casos, os *unicodes* de acentos são removidos. Em seguida, a conversão de todos os caracteres para minúsculas foi realizada, padronizando o texto e evitando problemas de duplicidade causados por letras maiúsculas e minúsculas.

Como a primeira parte da terceira etapa de pré-processamento foi feita a tokenização dos dados textuais, com o objetivo de dividir o texto em unidades (*tokens*), sendo o divisor utilizado os espaçamentos em branco entre as palavras. Como segunda parte, a remoção de *stopwords* e o *stemming* dos *tokens* restantes foram feitos buscando reduzir a quantidade de *tokens* e normalizar os restantes, melhorando assim a qualidade dos dados para a análise de sentimento.

Como base para o levantamento de *stopwords* foi utilizada a biblioteca *NLTK* (*Natural Language Toolkit*) do Python, que fornece recursos para o PLN. A biblioteca possui um conjunto de *stopwords* para diferentes idiomas, incluindo o português. Para o *stemming* foi utilizado o algoritmo *snowball stemmer*. Desenvolvido por Martin Porter e sendo uma melhoria do primeiro algoritmo de *stemming* de Porter (1980), é amplamente utilizado para a redução de palavras em inglês. O *snowball stemmer* foi projetado para ser facilmente adaptável a diferentes idiomas, permitindo a criação de implementações específicas para cada idioma.

Outras etapas de pré-processamento que não foram utilizadas podem incluir a detecção e correção de erros ortográficos e a manipulação de emojis e emoticons, que podem conter informações relevantes sobre o sentimento do texto.

As Tabelas 4 e 5 exibem uma amostra antes e após passar por todas as etapas do pré-processamento. Uma nuvem de palavras de toda a base coletada é apresentada na Figura 10, com a Figura 11 sendo a nuvem da mesma base após a aplicação de pré-processamento.

A nuvem da Figura 10 apresenta muitas *stopwords* como termos com maior frequência, sendo exemplos os 4 mais frequentes: “de”, “e”, “a”, “é”, isso demonstra a importância do pré-processamento para a correta análise, tendo em vista que as *stopwords* tem pouco a nenhum valor no estudo de análise de sentimento. Na Figura 11, alguns exemplos de destaque são os termos “nao”, “imag”, “qualidad”, “boa”, que possuem grande valor na análise de sentimento em questão. Importante ressaltar a não consideração do termo “nao” como *stopword*, devido ao seu grande valor em negativar os comentários em que está presente.

Tabela 4 - Amostras sem aplicação de pré-processamento

Frase 1	Gostei muito do produto, cumpre o que promete, qualidade top 👏👏👏👏
Frase 2	Muito ruim, não tem o que promete, e ainda não estão querendo trocar.

Fonte: Elaborado pelo Autor (2023).

Tabela 5 - Amostras com aplicação de pré-processamento

Frase 1	gost produt cumpr promet qualidad top
Frase 2	ruim nao promet aind nao esta quer troc

Fonte: Elaborado pelo Autor (2023).

Figura 10 - Nuvem de palavras sem aplicação de pré-processamento

Fonte: Elaborado pelo Autor (2023).

Figura 11 - Nuvem de palavras após aplicação de pré-processamento

Fonte: Elaborado pelo Autor (2023).

Outra visualização que pode ser feita é mostrada nas Tabelas 6 e 7. Em ambas são apresentados os 10 termos mais frequentes nas bases, tendo a tabela 6 como base os dados sem aplicação de pré-processamento, e a tabela 7 os dados já pré-processados.

Tabela 6 - Palavras mais frequentes sem aplicação de pré-processamento

Palavra	Frequência
de	309
e	241
a	223
é	208
não	173
que	170
o	157
imagem	122
qualidade	112
do	108

Fonte: Elaborado pelo Autor (2023).

Tabela 7 - Palavras mais frequentes com aplicação de pré-processamento

Palavra	Frequência
nao	247
imag	174
qualidad	154
cam	142
produt	139
boa	114
fic	64
bem	59
facil	59
compr	57

Fonte: Elaborado pelo Autor (2023).

Vale ressaltar também a análise da quantidade de frequências após o pré-processamento. Em que há uma diminuição perceptível na quantidade somada dos 10 termos mais frequentes com a remoção das *stopwords*, indo de um total de 1823 para 1209.

3.3 Aplicação da BoW e TF-IDF

Após o pré-processamento, os dados são retirados do conjunto de dados e armazenados em uma variável X usando a função *np.array*. Enquanto os sentimentos (Positivo ou Negativo) são armazenados em uma variável Y.

A função *countvectorizer* é utilizada em conjunto com a *fit_transform* para converter os dados armazenados na variável X em uma matriz esparsa, contendo

435 linhas e 1092 colunas, cada coluna representa um termo único ao final do pré-processamento enquanto cada linha representa cada documento presente na base.

A matriz resultante é uma representação numérica que reflete a ocorrência de cada termo em cada documento, sendo convertida em uma matriz densa por meio da função *toarray* e, em seguida, armazenada de volta em X. Um exemplo das etapas descritas é mostrado na Figura 12 utilizando o *Python*.

Figura 12 - Exemplo de criação de uma *BoW* no *Python*

```
from sklearn.feature_extraction.text import CountVectorizer

corpus = [
    'Essa é a primeira linha',
    'Essa é a segunda segunda linha',
    'E a terceira.',
    'Seria essa a primeira linha?']

vectorizer = CountVectorizer()
X = vectorizer.fit_transform(corpus)

print(X.toarray())
```

```
[[1 1 1 0 0 0]
 [1 1 0 2 0 0]
 [0 0 0 0 0 1]
 [1 1 1 0 1 0]]
```

```
Abaixo os Cabeçalhos das Colunas da BoW
```

```
vectorizer.get_feature_names()
```

```
['essa', 'linha', 'primeira', 'segunda', 'seria', 'terceira']
```

Fonte: Elaborado pelo Autor (2023).

Utilizando a ferramenta *TfidfVectorizer* e novamente aplicando a função fit_transform, foi criada uma matriz semelhante à gerada pelo modelo *BoW*. No entanto, desta vez, os valores na matriz representam os pesos atribuídos pelo algoritmo *TF-IDF*, esse algoritmo considera tanto a frequência de um termo em um documento quanto o número de documentos que o contêm.

Essa abordagem resulta em uma representação ponderada dos termos, destacando a relevância relativa de cada termo em relação não apenas aos documentos que o contém, mas sim a todo o *corpus* de documentos.

3.4 Aplicação de Técnicas de Aprendizado de Máquina

Para a aplicação das técnicas e criação dos resultados iniciais da pesquisa o *scikit-learn* foi utilizado, sendo uma biblioteca em *Python* amplamente utilizada para

aprendizado de máquina. Ela fornece uma ampla gama de algoritmos e ferramentas para tarefas de classificação, regressão, clusterização e pré-processamento de dados.

Foram utilizados neste trabalho cinco algoritmos, sendo eles: *MultinomialNB*, *BernoulliNB*, *KNeighborsClassifier*, *DecisionTreeClassifier* e *SVC*. Para o treino dos algoritmos, é dada a matriz TF-IDF como conjunto de dados.

A escolha desses cinco algoritmos busca abranger diversas técnicas de classificação. O *MultinomialNB* e o *BernoulliNB* são classificadores *Naive Bayes*, frequentemente utilizados em tarefas de PLN, enquanto o *KNeighborsClassifier* emprega o conceito de vizinhos mais próximos para classificação. O *DecisionTreeClassifier* constrói árvores de decisão para dividir os dados de forma iterativa, e o *SVC* baseia-se em máquinas de vetor de suporte para encontrar hiperplanos de separação ótimos.

4. RESULTADOS E DISCUSSÕES

Neste tópico os resultados dos modelos de aprendizagem de máquina serão apresentados e discutidos. Na seção 4.1, apresentam-se os resultados utilizando o pré-processamento convencional, em sequência, na seção 4.2 são apresentados os resultados utilizando diferentes abordagens de pré-processamento, finalizando com as limitações do trabalho na seção 4.3.

4.1 Resultados com Pré-Processamento Convencional

Com os resultados de cada técnica após a validação cruzada, o cálculo das métricas de avaliação pode ser feito. As Tabelas 8, 9, 10, 11 e 12 representam as matrizes de confusão resultantes de cada técnica.

Tabela 8 - Matriz de Confusão: *MultinomialNB*

Matriz de Confusão		**Previsto**		**Total**
		Negativo	**Positivo**	
Real	**Negativo**	25 - VN	5 - FP	30
	Positivo	4 - FN	53 - VP	57
Total		29	58	87

Fonte: Elaborado pelo Autor (2023).

Tabela 9 - Matriz de Confusão: *BernoulliNB*

Matriz de Confusão		**Previsto**		**Total**
		Negativo	**Positivo**	
Real	**Negativo**	12 - VN	18 - FP	30
	Positivo	3 - FN	54 - VP	57
Total		15	72	87

Fonte: Elaborado pelo Autor (2023).

Tabela 10 - Matriz de Confusão: *KNNClassifier*

Matriz de Confusão		**Previsto**		**Total**
		Negativo	**Positivo**	
Real	**Negativo**	20 - VN	10 - FP	30
	Positivo	6 - FN	51 - VP	57
Total		26	61	87

Fonte: Elaborado pelo Autor (2023).

Tabela 11 - Matriz de Confusão: *DecisionTreeClassifier*

Matriz de Confusão		Previsto		Total
		Negativo	**Positivo**	
Real	**Negativo**	18 - VN	12 - FP	30
	Positivo	8 - FN	49 - VP	57
Total		26	61	87

Fonte: Elaborado pelo Autor (2023).

Tabela 12 - Matriz de Confusão: *SVC*

Matriz de Confusão		Previsto		Total
		Negativo	**Positivo**	
Real	**Negativo**	21 - VN	9 - FP	30
	Positivo	7 - FN	50 - VP	57
Total		28	59	87

Fonte: Elaborado pelo Autor (2023).

A partir da análise das tabelas, o *MultinomialNB* foi o que obteve os menores valores de falso negativo e falso positivo, sendo este o algoritmo que mais classificou corretamente. Já o algoritmo *BernoulliNB*, apesar de ter um bom desempenho na previsão de comentários positivos, teve grandes dificuldades em classificar os comentários negativos, gerando muitas previsões erroneamente positivas. Entre os dois algoritmos *Naïve Bayes*, já era esperado o destaque do *MultinomialNB*, sendo o algoritmo indicado em estudos correlatos na área de mineração de textos (TOMAN, 2006).

O *KNNClassifier* e o *SVC* tiveram resultados parecidos, tendo um desempenho um pouco pior que o *MultinomialNB*. O *DecisionTreeClassifier* se manteve na faixa de desempenho do *BernoulliNB*, tendo como diferença o equilíbrio no erro de suas previsões (entre Falsos Positivos e Falsos Negativos).

A Tabela 13 apresenta as métricas de avaliação de desempenho dos algoritmos de forma consolidada.

Tabela 13 - Métricas de avaliação com pré-processamento convencional

Algoritmo	Sentimento	Precisão	Sensibilidade	F1-Score	Acuracidade
MultinomialNB	Negativo	86%	83%	85%	**90%**
	Positivo	91%	93%	92%	
BernoulliNB	Negativo	80%	40%	53%	**76%**
	Positivo	75%	95%	84%	
KNNClassifier	Negativo	77%	67%	71%	**82%**
	Positivo	84%	89%	86%	
DecisionTreeClassifier	Negativo	69%	60%	64%	**77%**
	Positivo	80%	86%	83%	
SVC	Negativo	75%	70%	72%	**82%**
	Positivo	85%	88%	86%	

Fonte: Elaborado pelo Autor (2023).

A tabela mostra a superioridade do *MultinomialNB* quando utilizado para tarefas de classificação de texto, como análise de sentimentos. Tendo uma alta precisão e sensibilidade para ambas as classes, o que significa que ele é capaz de identificar tanto sentimentos negativos quanto positivos corretamente. Além disso, o *F1-Score* equilibrado indica que o modelo mantém um bom equilíbrio entre precisão e sensibilidade, finalizando com uma alta acurácia de 90%, resultante das ótimas métricas anteriores.

Em comparação com o *MultinomialNB*, o *SVC* apresenta um desempenho um pouco inferior no que diz respeito à sensibilidade para a classe Negativo, sendo mais baixa (70%). Isso significa que o SVC tende a errar mais na identificação dos sentimentos negativos em relação ao *MultinomialNB*. No entanto, possui uma precisão e *F1-Score* razoavelmente altos para a classe Positivo, indicando que é bom em identificar sentimentos positivos. O *KNNClassifier* tem resultados parecidos com o *SVC*, mostrando a mesma dificuldade em identificar corretamente os sentimentos negativos, porém apresenta bom desempenho na análise dos sentimentos positivos.

Já o *BernoulliNB* tem alta precisão, mas baixa sensibilidade para a classe Negativa, indicando que facilmente classifica erroneamente muitos exemplos negativos. Por outro lado, possui maior sensibilidade à classe Positiva, o que significa que pode identificar corretamente sentimentos positivos, mas com precisão menor quando comparado com o *MultinomialNB* ou *SVC*.

O desempenho geral do *DecisionTreeClassifier* foi parecido com o *BernoulliNB*, tendo um desempenho um pouco melhor em relação a sentimentos negativos. Porém, sua precisão para ambas as classes é relativamente baixa quando comparado aos outros algoritmos, o que significa que classifica incorretamente muitos exemplos.

4.2 Resultados com Diferentes Abordagens de Pré-Processamento

Este tópico buscou através de diferentes abordagens de pré-processamento mostrar o impacto nos resultados das métricas das técnicas de aprendizado de máquina ao se alterar o formato de seus dados de treinamento e teste. Foram testados 3 cenários diferentes além do 1º Cenário (aplicação convencional de pré-processamento):

- 2º Cenário: Pré-processamento apenas com a remoção de *stopwords*
- 3º Cenário: Pré-processamento sem a aplicação de *stemming*
- 4º Cenário: Pré-processamento considerando notas acima de três como avaliações positivas

As Tabelas 14, 15 e 16 apresentam as métricas obtidas em cada cenário para cada algoritmo, possibilitando uma comparação do desempenho e auxiliando na seleção da estratégia de pré-processamento mais apropriada para estudos futuros relacionados ao tema.

Tabela 14 - Métricas de avaliação com pré-processamento apenas com a remoção de *stopwords*

Algoritmo	Sentimento	Precisão	Sensibilidade	F1-Score	Acuracidade
MultinomialNB	Negativo	93%	83%	88%	**92%**
	Positivo	92%	96%	94%	
BernoulliNB	Negativo	80%	40%	53%	**76%**
	Positivo	75%	95%	84%	
KNNClassifier	Negativo	86%	40%	53%	**77%**
	Positivo	75%	96%	85%	
DecisionTreeClassifier	Negativo	58%	73%	65%	**72%**
	Positivo	84%	72%	77%	
SVC	Negativo	77%	67%	71%	**82%**
	Positivo	84%	89%	86%	

Fonte: Elaborado pelo Autor (2023).

No 2º Cenário mostrado na tabela 13 o desempenho geral do *MultinomialNB* melhorou tendo um aumento em todas as métricas de avaliação, tornando-o ainda

mais forte em termos de identificação de sentimentos tanto negativos quanto positivos. Para o *SVC* e *BernoulliNB*, quase não houve mudanças nas métricas, mantendo-se suas forças e fraquezas do 1º Cenário.

A precisão para a classe Negativo melhorou para o *KNNClassifier*, porém sua sensibilidade diminuiu grandemente, tornando-o menos eficaz na identificação de sentimentos negativos, tendo melhor resultado para a classe Positivo nesse cenário. O *DecisionTreeClassifier* continuou a ter um desempenho inferior, tendo uma redução em grande parte das métricas nesse cenário.

Tabela 15 - Métricas de avaliação com pré-processamento sem a aplicação de *stemming*

Algoritmo	Sentimento	Precisão	Sensibilidade	F1-Score	Acuracidade
MultinomialNB	Negativo	89%	83%	86%	**91%**
	Positivo	92%	95%	93%	
BernoulliNB	Negativo	83%	33%	48%	**75%**
	Positivo	73%	96%	83%	
KNNClassifier	Negativo	84%	70%	76%	**85%**
	Positivo	85%	93%	89%	
DecisionTreeClassifier	Negativo	73%	73%	73%	**82%**
	Positivo	86%	86%	86%	
SVC	Negativo	71%	67%	69%	**79%**
	Positivo	83%	86%	84%	

Fonte: Elaborado pelo Autor (2023).

O 3º Cenário mostrado na tabela 14 mantém os bons resultados do *MultinomialNB* obtidos no 2º Cenário, tendo uma leve piora no desempenho. O *BernoulliNB* também mantém resultados parecidos com o 2º Cenário, tendo como destaque uma piora em sua sensibilidade referente a Classe Negativo.

Nesse cenário, a precisão na classe Negativo foi destaque para o *SVC* e para o *DecisionTreeClassifier*, sendo o motivo do pior desempenho nesse cenário para o primeiro citado e da grande melhora no desempenho do segundo. Para o *KNNClassifier* destacou-se o aumento considerável da sensibilidade para a classe Negativa.

Tabela 16 - Métricas de avaliação com pré-processamento considerando notas acima de três como avaliações positivas

Algoritmo	Sentimento	Precisão	Sensibilidade	F1-Score	Acuracidade
MultinomialNB	Negativo	81%	74%	77%	**89%**
	Positivo	91%	94%	92%	
BernoulliNB	Negativo	73%	48%	58%	**82%**
	Positivo	83%	94%	88%	
KNNClassifier	Negativo	73%	70%	71%	**85%**
	Positivo	89%	91%	90%	
DecisionTreeClassifier	Negativo	72%	78%	75%	**86%**
	Positivo	92%	89%	90%	
SVC	Negativo	93%	57%	70%	**87%**
	Positivo	86%	98%	92%	

Fonte: Elaborado pelo Autor (2023).

O 4º Cenário analisado demonstra como os algoritmos respondem a uma mudança nas características da base de dados, passando a classificação intermediária como um sentimento positivo ao invés de negativo (uma outra abordagem interessante seria a criação da classe Neutro). O impacto dessa transição pode ser percebido nos cinco algoritmos, uma vez que foram treinados para identificar sentimentos negativos e terão dificuldades ao se depararem com uma grande quantidade de sentimentos positivos.

Todos os algoritmos apresentam altos valores de desempenho nesse cenário para a classe Positivo, enquanto para a classe Negativo, apenas o *SVC* apresentou ótimo desempenho em sua precisão. Essa mudança de cenário destaca a importância de escolher algoritmos que sejam flexíveis e adaptáveis a diferentes distribuições de classe e tipos de dados, para garantir um desempenho consistente em situações diversas de análise de sentimentos. A Tabela 17 compila as acuracidades obtidas nos 4 cenários.

Tabela 17 - Acurácia de cada modelo segmentada por cenário aplicado

Algoritmo	1º Cenário	2º Cenário	3º Cenário	4º Cenário
MultinomialNB	**90%**	**92%**	**91%**	**89%**
BernoulliNB	**76%**	**76%**	**75%**	**82%**
KNNClassifier	**82%**	**77%**	**85%**	**85%**
DecisionTreeClassifier	**77%**	**72%**	**82%**	**86%**
SVC	**82%**	**82%**	**79%**	**87%**

Fonte: Elaborado pelo Autor (2023).

Ao analisar a Tabela 17 é perceptível que os classificadores já possuem uma acurácia razoável com o pré-processamento convencional (1º Cenário), sendo que os valores foram próximos ou acima de 80%, sendo o melhor o *MultinomialNB*, com 90%. Nos outros 3 cenários, ao menos um algoritmo teve aumento em sua acurácia, o que demonstra a importância da grande influência do formato escolhido de pré-processamento para cada modelo.

O 2º Cenário, surtiu um efeito muito positivo no melhor classificador do 1º Cenário: aumentando a acurácia do *MultinomialNB* para 92%, enquanto o *BernoulliNB* e *SVC* não tiveram alteração em sua acurácia do 1º Cenário, *KNNClassifier* e *DecisionTreeClassifier* sofreram negativamente com o pré-processamento reduzido, cada um perdendo 5% de acurácia. Isso demonstra que, dependendo da técnica de pré-processamento, o resultado pode diferir para diferentes algoritmos.

O 3º Cenário mostrou-se benéfico aos classificadores que haviam tido queda em sua acurácia no segundo cenário. Nesse cenário, o *KNNClassifier* e o *DecisionTreeClassifier* aumentaram significativamente suas performances enquanto os outros modelos tiveram uma leve queda.

Por fim, o 4º Cenário mostrou uma melhor performance em quase todos os classificadores ao ser comparado ao 1º Cenário. Embora não seja um pré-processamento, este cenário foi criado, pois a modificação da premissa utilizada como divisora de sentimento da base influencia diretamente no resultado. É

interessante notar que o algoritmo *MultinomialNB*, criado com o objetivo de solucionar problemas de classificação de documentos a partir da contagem de palavras, foi o único que teve uma leve queda de performance com a alteração da premissa. O resultado do algoritmo *MultinomialNB* no 1º Cenário continuou sendo o melhor no comparativo geral apesar da grande melhora no 4º Cenário dos outros algoritmos.

4.3 Limitações do Trabalho

A utilização de uma base de dados pequena representa uma limitação significativa no trabalho. Bases reduzidas podem não capturar adequadamente a diversidade de expressões e opiniões presentes em contextos mais amplos. A falta de representatividade pode comprometer a capacidade do modelo em aprender padrões robustos, levando a previsões menos confiáveis e generalizações inadequadas para avaliações que saem do escopo da base de dados limitada.

Além disso, a escolha de utilizar apenas tokens únicos para compensar a limitação de tamanho da base para representar o texto acaba por introduzir outra limitação. O uso de *tokens* unigramas pode gerar a perda de informações valiosas presentes em relações mais complexas entre palavras, tendo o modelo dificuldade em capturar variações sintáticas e semânticas, limitando a profundidade da análise em relação ao sentimento real das avaliações.

Ao adotar uma única abordagem de divisão entre treinamento e teste, o modelo pode ficar suscetível a um viés específico dos dados utilizados. Essa abordagem pode resultar em avaliações otimistas do desempenho do modelo, uma vez que ele é testado em dados semelhantes aos utilizados durante o treinamento, mas pode falhar ao ser exposto a variações mais amplas presentes em diferentes conjuntos de dados. Essa limitação pode comprometer a generalização do modelo para situações fora do contexto analisado.

5. CONCLUSÃO

Esse trabalho de conclusão de curso teve como objetivo a aplicação de técnicas de aprendizado de máquina em uma base de avaliações textuais de consumidores, e posteriormente, feita a avaliação dos desempenhos das técnicas em classificar o sentimento expresso por cada consumidor.

Um produto específico com avaliações mistas foi escolhido do site da *Amazon* para compor a base de dados a ser classificada por sua vantagem de não tendenciar o treinamento do aprendizado de máquina. Tendo como ferramenta utilizada a linguagem de programação *Python* e seu conjunto de bibliotecas específicas para problemas de análise de sentimentos e aprendizagem de máquina.

Buscando uma maior exploração na pesquisa, uma comparação entre os algoritmos de aprendizado de máquina foi realizada. De forma a determinar quais algoritmos demonstravam melhor desempenho na categorização de sentimentos em textos.

Além disso, foram exploradas diversas abordagens de pré-processamento, com o objetivo de otimizar a qualidade e a utilidade dos dados textuais. Essas iniciativas visavam aperfeiçoar o desempenho dos algoritmos em sua capacidade de interpretação e classificação das informações presentes nos textos. Após a realização dessas etapas, os algoritmos foram empregados para classificar instâncias ainda não vistas, e os resultados gerados foram comparados, permitindo a avaliação do desempenho e da eficácia das abordagens adotadas no trabalho.

Os testes com diferentes técnicas de processamento de texto revelaram que essas técnicas influenciam os algoritmos de diversas maneiras. Por exemplo, a não aplicação do *stemming* demonstrou influência positiva nas acurácias dos algoritmos, reduzindo levemente apenas a do algoritmo *SVC*. Já o algoritmo *MultinomialNB*, teve melhora em sua acurácia ao se aplicar apenas a remoção de *stopwords*, cenário que resultou em uma diminuição significativa da acurácia nos algoritmos *KNNClassifier* e *DecisionTreeClassifier*. Isso realça a influência específica do tipo de tratamento de texto na performance de diferentes algoritmos.

Adicionalmente, a análise feita com o pré-processamento considerando notas acima de três como avaliações positivas destaca a importância de escolher cuidadosamente as premissas ao lidar com dados. Essa abordagem específica para a categorização de avaliações positivas tem um impacto direto na interpretação e classificação dos dados. Com a alteração de premissas, pode-se influenciar significativamente a forma como o algoritmo interpreta e rotula as avaliações, o que, por sua vez, afeta a qualidade das análises e resultados obtidos.

Por fim, os resultados dos algoritmos foram analisados e comparados. As comparações dos resultados revelaram que a acurácia do algoritmo *MultinomialNB* se destacou em todos os cenários, tendo apenas demonstrando apenas uma singela dificuldade na classificação correta de comentários negativos que tendem a neutralidade, sendo essa a grande dificuldade demonstrada pelo algoritmo *BernoulliNB*, que afetou de forma significativa em sua acurácia. O *SVC* não só compartilhou das mesmas dificuldades identificadas, mas também cometeu erros ao classificar erroneamente comentários como negativos quando, na realidade, não o eram. Essa imprecisão na classificação dos comentários negativos é vista também no *DecisionTreeClassifier*.

Uma das principais contribuições deste estudo reside na viabilização da aplicação de técnicas de aprendizado de máquina e mineração de textos na avaliação de consumidores de e-commerce. Nesse contexto, sugere-se que pesquisas futuras explorem diferentes técnicas e premissas de pré-processamento no contexto da previsão das avaliações de consumidores em plataformas de e-commerce. Uma área promissora para investigação é o uso de bibliotecas mais avançadas, como *BERT* (*Bidirectional Encoder Representations from Transformers*) e *RoBERTa* (*Robustly optimized BERT approach*).

A aplicação dessas tecnologias avançadas pode resultar em melhorias significativas na confiabilidade das previsões de avaliações de consumidores, permitindo uma análise mais profunda das opiniões expressas. Visando aprimorar a capacidade de antecipar e interpretar as avaliações dos consumidores, a fim de proporcionar uma melhor experiência ao cliente e eficiência nas estratégias de negócios no *e-commerce*.

REFERÊNCIAS

ABCOMM (2021). **Associação Brasileira de Comércio Eletrônico**. Recuperado de https://abcomm.org/.

AYEDH A.; TAN G.; ALWESABI K.; RAJEH H. **The effect of preprocessing on arabic document categorization**. Algorithms, 2016.

AGGARWAL, C. et al. **Neural networks and deep learning**. Springer, v. 10, p. 978, 2018.

BAHRAINIAN, S. A.; DENGEL, A. **Sentiment analysis of texts by capturing underlying sentiment patterns**. Web Intelligence. IOS Press, p. 53-68, 2015.

CHAMBERS, N.; JURAFSKY, D. **Template-based information extraction without the templates**. Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies. p. 976-986, 2011.

COMRIE, B. **The unity of noun-modifying clauses in Asian languages**. Pan-Asiatic Linguistics: Proceedings of the Fourth International Symposium on Languages and Linguistics. Salaya, Thailand: Institute of Language and Culture for Rural Development, Mahidol University at Salaya, p. 1077-1088, 1996.

CRISTIANINI, N.; SHAWE-TAYLOR, J. **An introduction to support vector machines and other kernel-based learning methods**. Cambridge university press, 2000.

DERIS, A. M.; ZAIN, A. M.; SALLEHUDDIN, R. **Overview of support vector machine in modeling machining performances**. Procedia Engineering, v. 24, p. 308-312, 2011.

DIGIACOMO, R. A.; KREMER, J. M.; SHAH, D. M. **Fish-oil dietary supplementation in patients with Raynaud's phenomenon: a double-blind, controlled, prospective study**. The American journal of medicine, v. 86, n. 2, p. 158-164, 1989.

FELDMAN, R. **Techniques and applications for sentiment analysis**. Communications of the ACM, v. 56, n. 4, p. 82-89, 2013.

GUO, G.; WANG, H.; BELL, D.; BI, Y.; GREER, K. **KNN Model-Based Approach in Classification.** OTM 2003. Lecture Notes in Computer Science, vol 2888, 2003.

GUPTA, V.; LEHAL, G. S. **A survey of text mining techniques and applications**. Journal of emerging technologies in web intelligence, v. 1, n. 1, p. 60-76, 2009.

HADDI, E.; LIU, X.; SHI, Y. **The role of text pre-processing in sentiment analysis**. Procedia Computer Science, 17, 26-32, 2013.

HARAVU, L. J.; NEELAMEGHAN, A. **Text mining and data mining in knowledge organization and discovery: the making of knowledge-based products**. Cataloging & classification quarterly, v. 37, n. 1-2, p. 97-113, 2003.

IGNATOW, G.; MIHALCEA, R. **The Philosophy and Logic of Text Mining**, 2018.

JIVANI, A. G. et al. **A comparative study of stemming algorithms**. Int. J. Comp. Tech. Appl, v. 2, n. 6, p. 1930-1938, 2011.

JULURU K. et al. **Bag-of-words technique in natural language processing: A primer for radiologists**. RadioGraphics, 41(5), 1420-1426, 2021.

KHYANI, D. et al. **An interpretation of lemmatization and stemming in natural language processing**. Journal of University of Shanghai for Science and Technology, v. 22, n. 10, p. 350-357, 2021.

KUBAT, M. **An introduction to machine learning**. Cham, Switzerland: Springer International Publishing, 2017.

LEE, Y. K.; NG, H. T. **An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation**. Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002), p. 41-48, 2002.

LIU, S. M.; CHEN, J. H. **A multi-label classification-based approach for sentiment classification.** Expert Systems with Applications, v. 42, n. 3, p. 1083-1093, 2015.

MÉNDEZ, J. R.; IGLESIAS, E. L.; FDEZ-RIVEROLA, F.; DÍAZ, F.; CORCHADO, J. M. **Tokenising, stemming and stopword removal on anti-spam filtering domain**. Conference of the Spanish Association for Artificial Intelligence, p. 449-458, 2005.

MURTHY, S. K. **Automatic Construction of Decision Trees from Data: A Multidisciplinary Survey**. Data Mining and Knowledge Discovery, 2, 345-389, 1998.

NAKAGAWA, T.; INUI, K.; KUROHASHI, S. **Dependency tree-based sentiment classification using CRFs with hidden variables**. Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, p. 786-794, 2010.

NANDWANI, P; VERMA, R. **A review on sentiment analysis and emotion detection from text**. Social Network Analysis and Mining, v. 11, n. 1, p. 81, 2021.

NIELSENIQ EBIT. **47ª Edição – Webshoppers**. Dados disponíveis em: <www.ebit.com.br> Acesso em: março de 2023.

OKAMOTO, S.; YUGAMI, N. **Effects of domain characteristics on instance-based learning algorithms**. Theoretical Computer Science, v. 298, n. 1, p. 207-233, 2003.

OSINSKI, S.; WEISS, D. **Carrot 2: Design of a flexible and efficient web information retrieval framework**. International atlantic web intelligence conference. Springer, Berlin, Heidelberg, p. 439-444, 2005.

PAICE, C. D. **An evaluation method for stemming algorithms**. SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University. Springer London, p. 42-50, 1994.

PANDA, R.; SWAR, B. **Online Shopping: An Exploratory Study to Identify the Determinants of Shopper Buying Behaviour**. International journal of business insights & transformation, v. 7, n. 1, 2013.

PALMER, D. **Text Preprocessing**. Handbook of natural language processing, v. 2, p. 9-29, 2010.

PORTER M. F. **An Algorithm for Suffix Stripping.** Program, v. 14, n. 3, p. 130-137, 1980.

RASCHKA, S. **Python machine learning**. Packt publishing ltd, 2015.

RAULJI, J. K.; SAINI, J. R. **Stop-word removal algorithm and its implementation for Sanskrit language**. International Journal of Computer Applications, v. 150, n. 2, p. 15-17, 2016.

ROJAS-BARAHONA, L. M. **Deep learning for sentiment analysis**. Lang Linguist Compass, n. 10, p. 701-719, 2016.

RZHETSKY, A.; SERINGHAUS, M.; GERSTEIN, M. **Seeking a new biology through text mining**. Cell, v. 134, n. 1, p. 9-13, 2008.

SEBASTIANI, F. **Machine learning in automated text categorization**. ACM computing surveys (CSUR), v. 34, n. 1, p. 1-47, 2002.

SILVA, L. F; OLIVEIRA, P. S. G. **Fatores que influenciam o comportamento do consumidor em lojas virtuais**. Revista de Administração Unimep, n. 17, 2019.

SONG, Y.; YING, L. **Decision tree methods: applications for classification and prediction**. Shanghai archives of psychiatry, v. 27, n. 2, p. 130, 2015.

SRIVIDHYA, V.; ANITHA, R. **Evaluating preprocessing techniques in text categorization.** International journal of computer science and application, 47(11), 49-51, 2010.

SWANSON, D. R. **Fish oil, Raynaud's syndrome, and undiscovered public knowledge**. Perspectives in biology and medicine, v. 30, n. 1, p. 7-18, 1986.

TAN, A. et al. Text mining: The state of the art and the challenges. **Proceedings of the pakdd 1999 workshop on knowledge discovery from advanced databases**, p. 65-70, 1999.

THOMAS, J.; THOMAS, E.; TOMB, E. **Serum and erythrocyte magnesium concentrations and migraine**. Magnesium Research, v. 5, n. 2, p. 127-130, 1992.

TOMAN; M.; TESAR, R.; JESEK, K. **Influence of word normalization on text classification.** Proceedings of InSciT, n. 4, 354-358, 2006.

TURBAN, E.; KING, D.; LEE; J.; VIEHLAND, D. **Electronic Commerce: a managerial perspective**. Pearson Education, 2004.

TSYTSARAU, M.; PALPANAS, T. **Survey on mining subjective data on the web**. Data Mining and Knowledge Discovery, 24, p. 478-514, 2012.

UYSAL, A. K.; GUNAL, S. **The impact of preprocessing on text classification**. Information Processing & Management, 50(1), 104-112, 2014.

VEROPOULOS, K. et al. **Controlling the sensitivity of support vector machines**. Proceedings of the international joint conference on AI, p. 60, 1999.

ZHANG, H. H. et al. **Principles and theory for data mining and machine learning**. New York: Springer, 2009.

ZHANG, Y.; JIN, R.; ZHOU, Z. **Understanding bag-of-words model: a statistical framework**. International journal of machine learning and cybernetics, v. 1, p. 43-52, 2010.

WEBB, G. I.; KEOGH, E.; MIIKKULAINEN, R. **Naïve Bayes**. Encyclopedia of machine learning, v. 15, n. 1, p. 713-714, 2010.

WETTSCHERECK, D.; AHA, D.; MOHRI, T. **A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms**. Artificial Intelligence Review, 11, 273-314, 1997.