Preditor de E-mails SPAM: Um *Framework* Preditivo de Alta Precisão com *Machine Learning*[*]

João Vítor Batistella[1], **Andrws Aires Vieira**[1]

[1]Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - *Campus* Ibirubá
Rua Nelsi Ribas Fritsch, 1111 – CEP: 98200-000 – Ibirubá – RS – Brasil

Abstract. *This paper proposes the development of a machine learning model to classify emails as SPAM or HAM (not SPAM). Considering the growing relevance of unwanted emails in global data traffic, the research aims to develop a solution that optimizes computational resources and supports digital marketing companies. Using Python, the study was based on the application of Logistic Regression and Naïve Bayes to analyze email content, aiming to promote more sustainable digital marketing practices, reducing the impact of unwanted communications, and preserving the sender reputation. The results demonstrated the superiority of the Logistic Regression algorithm, which achieved consistent accuracy between 98% and 99% in the different datasets analyzed, evidencing methodological robustness and technical feasibility for the implementation of automated SPAM filters in large-scale corporate environments.*

Resumo. *O presente trabalho propõe desenvolver um modelo de aprendizado de máquina para classificar e-mails como SPAM ou HAM (não SPAM). Considerando a relevância crescente do problema dos e-mails indesejados no tráfego global de dados, a pesquisa busca criar uma solução que otimize recursos computacionais e auxilie empresas de marketing digital. Utilizando Python, o estudo foi fundamentado na aplicação de Regressão Logística e Naïve Bayes para analisar o conteúdo dos e-mails, buscando promover práticas mais sustentáveis de marketing digital, reduzindo o impacto das comunicações indesejadas e preservando a reputação dos remetentes. Os resultados demonstraram a superioridade do algoritmo de Regressão Logística, que alcançou precisão consistente entre 97% e 99% nos diferentes datasets analisados, evidenciando robustez metodológica e viabilidade técnica para implementação de filtros automatizados de SPAM em ambientes corporativos de larga escala.*

1. Introdução

O *e-mail* é uma ferramenta essencial no marketing digital, mas a alta taxa de mensagens classificadas como SPAM tem gerado desafios significativos. Segundo Statista (2024a), em abril de 2024, usuários nos Estados Unidos enviaram 9,7 bilhões de *e-mails* em um único dia. Já em dezembro, a média diária de *e-mails* classificados como SPAM chegou a 7,8 bilhões (STATISTA, 2024b). O envio em massa, de baixo custo, favorece a disseminação de SPAM, que inclui anúncios comerciais, golpes financeiros e mensagens fraudulentas, além de resultar em desperdício de recursos computacionais (SHERWIN, 2023).

[*]Trabalho de Conclusão de Curso (TCC) do curso Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul – Campus Ibirubá, 2025

Os *e-mails* indesejados correspondem a 45,6% do tráfego total de dados na internet (STATISTA, 2024c). Esse volume de dados desnecessários traz prejuízos significativos para os servidores quanto à utilização da CPU, espaço em memória e a largura de banda de rede (DADA et al., 2019). O cenário é particularmente desafiador para empresas que dependem do *marketing* digital, pois a eficácia de suas campanhas está diretamente ligada à reputação dos domínios utilizados para o envio.

Com o aprimoramento contínuo dos algoritmos de detecção de SPAM pelos principais provedores, manter uma boa reputação de domínio tornou-se crucial para garantir a entregabilidade das mensagens (BOSTOGANASHVILI, 2024). A classificação incorreta de mensagens legítimas como SPAM pode resultar em prejuízos econômicos significativos, comprometer a imagem corporativa e prejudicar relacionamentos comerciais importantes. Além disso, o atual cenário de proteção de dados e privacidade, com regulamentações como a Lei Geral de Proteção de Dados Pessoais, torna ainda mais crítica a necessidade de práticas responsáveis de *e-mail marketing* (NASCIMENTO, 2021).

Considerando os dados estatísticos expostos até aqui, foi desenvolvido um modelo de aprendizado de máquina capaz de classificar um conteúdo de *e-mail* como SPAM ou HAM, termo em inglês utilizado para nomear *e-mails* que geralmente são desejados, ou seja, o oposto de SPAM. O modelo pode ser empregado, com caráter preventivo, por empresas especializadas no envio de *e-mails* em massa, contribuindo para uma melhor utilização dos recursos de infraestrutura e prevenindo o envio de conteúdos SPAM aos provedores, o que poderia afetar negativamente a reputação do domínio remetente. Dessa forma, Dessa forma, evita-se danos aos clientes, que de forma involuntária, acabam escrevendo *e-mails* tipicamente classificados como SPAM.

O objetivo deste trabalho é desenvolver um *framework* preditivo capaz de classificar um corpo de *e-mail* como SPAM ou HAM (não SPAM). Isso envolve a definição de uma metodologia para predição de SPAM, o treinamento e avaliação de modelos de aprendizado de máquina, testando e comparando diferentes técnicas e algoritmos de classificação para determinar a abordagem mais eficaz.

Esta pesquisa possui foco especificamente no desenvolvimento de um modelo de *machine learning* para a classificação de *e-mails* SPAM, com ênfase na análise do corpo da mensagem como principal elemento de predição. O estudo se concentrou na aplicação de técnicas de processamento de linguagem natural e aprendizado de máquina, utilizando Regressão Logística como algoritmo principal e *Naïve Bayes* como algoritmo de comparação.

O escopo da pesquisa se concentrou na aplicação de técnicas de pré-processamento e aprendizado de máquina para classificação de *e-mails* SPAM, utilizando *datasets* públicos e disponíveis para treinamento dos modelos. As metodologias incluíram técnicas de pré-processamento de texto, como *tokenização*, remoção de *stopwords* e vetorização, para preparar os dados para treinamento do modelo. Embora a construção de um *dataset* próprio não fosse o objetivo principal, houve uma análise dos conjuntos de dados disponíveis para garantir a qualidade e representatividade das amostras utilizadas na pesquisa.

A pesquisa delimita sua contribuição tecnológica ao desenvolvimento de um *framework* preditivo que possa ser utilizado por empresas de marketing digital para avaliar

previamente se seus *e-mails* poderão ser classificados como SPAM. Não foram desenvolvidas soluções para bloqueio ou eliminação automática de *e-mails*. O método não contemplou a análise de anexos, imagens ou elementos multimídia dos *e-mails*, concentrando-se exclusivamente no conteúdo textual do corpo da mensagem como fonte primária de análise.

O restante do artigo está organizado da seguinte forma: a Seção 2 aborda o referencial teórico, incluindo conceitos sobre SPAM, reputação de domínio e técnicas de filtragem com aprendizado de máquina. A Seção 3 discute os trabalhos correlatos que fundamentam a pesquisa. A Seção 4 detalha a metodologia, que engloba a coleta de dados, o pré-processamento, o desenvolvimento e a avaliação do modelo, além de apresentar o fluxo metodológico seguido para o desenvolvimento da pesquisa. Na sequência, a Seção 5 apresenta e discute os resultados alcançados. Por fim, a Seção 6 conclui o trabalho e sugere trabalhos futuros.

2. Referencial Teórico

Esta seção busca estabelecer as bases conceituais necessárias para a compreensão e análise do tema em estudo. Por meio de uma revisão da literatura, são apresentados os principais conceitos, teorias e discussões que fundamentam esta pesquisa.

2.1. Filtragem de SPAM

Os filtros de SPAM podem ser implantados no lado do cliente, do servidor e de computadores intermediários (DADA et al., 2019). Um dos itens considerados por filtro de SPAM é a reputação do domínio do remetente. Segundo Dossetto (2022), a reputação de domínio é a opinião que os destinatários, incluindo provedores e serviços *anti-spam*, têm sobre um domínio. Os domínios utilizados nos remetentes de *e-mail* sofrem constantes análises quanto à sua qualidade. A reputação é uma maneira de determinar se os *e-mails* são confiáveis. Quem faz isso são os ISPs (Provedor de Serviço de Internet), analisando o histórico de envios em massa e envio de *phishing* e *spoofing* (DADA et al., 2019).

O *Phishing* é um cibercrime, cujo objetivo é roubar informações pessoais, como senhas ou número de cartões (KOSINSKI, 2024). O *spoofing* está comumente atrelado ao *phishing*, porém ele faz referência ao uso de técnicas de falsificação, onde os criminosos se passam por familiares ou empresas que o destinatário provavelmente conhece a fim de induzir a vítima a entregar as informações pretendidas (PRONNUS, 2024). Um exemplo amplamente conhecido é a solicitação de pagamentos de taxa para o Correios. Sendo que, nesses casos, os autores escrevem o modelo de *e-mail* de forma que o destinatário não perceba que o remetente não é o Correios.

Além dos aspectos relacionados à reputação e à autenticidade do remetente, é igualmente importante considerar a estrutura e o conteúdo da mensagem para a detecção de SPAM. O conteúdo de um *e-mail* é composto por diferentes partes que precisam ser consideradas na mineração de dados. O cabeçalho e o corpo do *e-mail* são elementos fundamentais nesse processo. O corpo pode conter marcação HTML, imagens, arquivos anexos e outros tipos de dados, exigindo um tratamento adequado durante o pré-processamento (DADA et al., 2019). Já o cabeçalho inclui metadados importantes, como o assunto do e-mail que podem ser utilizados para auxiliar na classificação e análise das mensagens.

2.2. KDD (*Knowledge Discovery in Databases*)

A Descoberta de Conhecimento em Bancos de Dados — KDD (do inglês, *Knowledge Discovery in Databases*) — é o processo não trivial de identificar padrões válidos, novos, potencialmente úteis e, em última análise, compreensíveis nos dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Este processo é composto por cinco etapas essenciais, apresentadas na Figura 1 e descritas na sequência.

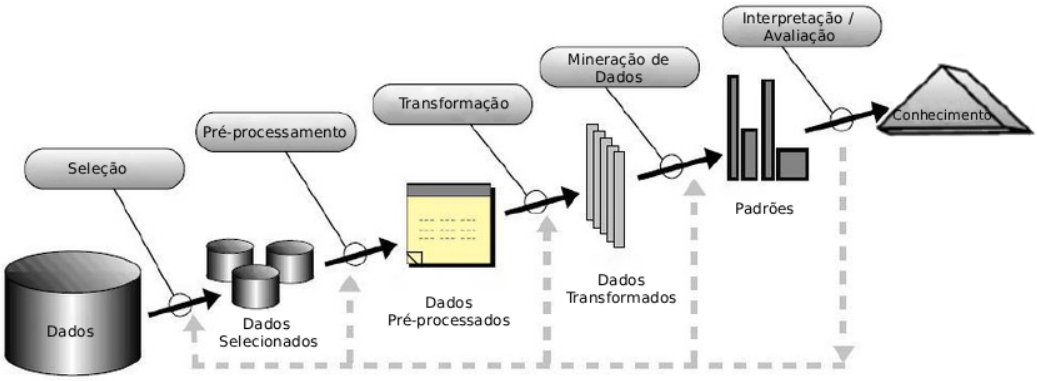

Figura 1. KDD (Knowledge Discovery in Databases). Fonte: (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996)

Responsável pela definição do subconjunto de dados que será analisado no processo de KDD, a seleção de dados é uma etapa essencial que pode ser realizada de duas maneiras: por meio da escolha dos atributos relevantes ou da filtragem dos registros que serão submetidos à análise (GOLDSCHMIDT, 2015).

As bases de dados são compostas por diversas atributos. Após definir os atributos de interesse, a etapa de pré-processamento tem o propósito de preparar os dados para uma análise mais eficiente e precisa. Nessa fase, é essencial detectar e corrigir inconsistências ou ruídos, realizar a limpeza dos dados, preencher valores ausentes com métodos apropriados e eliminar elementos indesejáveis que possam impactar negativamente a extração de conhecimento (MARIANO; MARQUES; SILVA, 2021).

A etapa de transformação tem como objetivo ajustar os dados para formatos compatíveis com os métodos que serão utilizados na próxima etapa, a mineração de dados. Esse processo pode incluir, por exemplo, a conversão de dados contínuos em discretos quando o modelo exige essa estrutura e a normalização dos dados para garantir uma análise mais eficaz (GOLDSCHMIDT, 2015).

A etapa de Mineração de Dados compreende a busca efetiva por conhecimentos úteis no contexto da aplicação de KDD. Esta etapa envolve a aplicação de algoritmos sobre os dados em busca de conhecimento implícito e útil (GOLDSCHMIDT, 2015). Algoritmos de classificação são comumente utilizados para a tarefa de identificar SPAM. (TAN; STEINBACH; KUMAR, 2009) definem a classificação como a tarefa de aprender uma função alfa f que mapeie cada conjunto de atributos x para um dos rótulos de classes y pré-determinados. Exemplos incluem classificadores de árvore de decisão, classificadores baseados em regras, redes neurais, máquinas de vetores de suporte (SVM) e classificadores *Bayesianos*.

2.3. Regressão Logística

Regressão Logística é um método estatístico de classificação que, apesar do nome, não é utilizado para regressão, ele é utilizado para problemas de classificação binária. Utiliza a função sigmoide (Equação 2) para transformar uma combinação linear das variáveis de entrada em uma probabilidade entre 0 e 1, onde x representa a soma do intercepto (b_0) com o produto de cada variável ($x_1, x_2, \ldots$) por seus respectivos coeficientes ($b_1, b_2, \ldots$), conforme Equação 1. Esses coeficientes e o intercepto são aprendidos pelo algoritmo durante o treinamento do modelo através de um processo de otimização (geralmente gradiente descendente), que busca minimizar o erro entre as previsões e os valores reais dos dados de treinamento (JURAFSKY; MARTIN, 2009).

$$x = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n \tag{1}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

2.4. *Naïve bayes*

O *Naïve Bayes* é um algoritmo de classificação probabilística amplamente usado em processamento de linguagem natural e classificação de texto. Ele funciona com base no teorema de *Bayes*, calculando a probabilidade de um documento pertencer a uma determinada classe através da Equação 3. Aplicando o teorema de Bayes, essa probabilidade é decomposta como mostrado na Equação 4, que pode ser simplificada para a Equação 5, removendo o denominador constante. O algoritmo é chamado de "*naïve*"(ingênuo) porque faz uma suposição simplificadora: assume que as características são condicionalmente independentes entre si, dada a classe.

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) \tag{3}$$

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)} \tag{4}$$

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(d|c)P(c) \tag{5}$$

Para classificar um novo documento, o *Naïve Bayes* calcula a probabilidade posterior $P(c|d)$ para cada classe *c*, escolhendo aquela com maior probabilidade. Usando o teorema de *Bayes*, essa probabilidade é decomposta em dois componentes principais: a probabilidade prior $P(c)$ - que representa a frequência de cada classe no conjunto de treinamento - e a probabilidade condicional $P(d|c)$ - que mede quão provável é observar as palavras do documento em uma determinada classe. O treinamento do algoritmo é relativamente simples, baseando-se em estimativas de máxima probabilidade condicional calculadas a partir das frequências das palavras nos documentos de treinamento de cada classe.

3. Trabalhos Correlatos

Esta seção apresenta uma análise de estudos relacionados à classificação de *e-mails* como SPAM ou HAM baseados em técnicas de *machine learning*. Esses trabalhos foram publicados entre 2019 e 2023 e fornecem uma base teórica e prática para o desenvolvimento do presente estudo, descrevendo as diferentes abordagens, algoritmos e desafios da detecção de SPAM.

O trabalho de Dada et al. (2019) revisa as abordagens de *machine learning* para a filtragem de SPAM, destacando que, apesar dos avanços significativos, um dos maiores desafios enfrentados é a capacidade dos modelos de acompanhar a rápida evolução das técnicas utilizadas por *spammers* (pessoas que utilizam da técnica de enviar *e-mails* indesejados em massa). Para eles, o uso de abordagens híbridas, que combinam diferentes algoritmos de *machine learning*, pode ser uma solução promissora para aumentar a precisão na classificação de SPAM. No entanto, os autores também apontam uma lacuna importante: a adaptação desses modelos a diferentes idiomas e contextos linguísticos ainda é limitada, o que afeta a eficácia dessas soluções em cenários específicos.

O estudo de Kuchipudi, Nannapaneni e Liao (2020), em complemento, analisa como a manipulação de palavras-chave em modelos de *e-mails* pode influenciar a classificação de mensagens como SPAM ou HAM. Usando o algoritmo *Naïve Bayes*, os autores investigaram três técnicas de evasão: substituição de sinônimos HAM, injeção de palavras HAM e espaçamento de palavras SPAM. Eles mostram que, em 60% dos casos, é possível contornar os filtros ao aplicar uma dessas estratégias, expondo uma vulnerabilidade significativa em modelos tradicionais de classificação.

Adicionalmente, o estudo de Yaseen et al. (2021) investiga modelos avançados na detecção de *e-mails* de SPAM, destacando a eficácia do modelo *Transformer BERT Base Cased*. Esse modelo se mostrou superior em comparação com abordagens anteriores, como o *BiLSTM*, devido à sua capacidade de considerar o contexto das palavras por meio de camadas de atenção. Os resultados indicam que esses modelos não apenas se ajustam bem a novos dados, mas também apresentam robustez, evitando problemas comuns como o sobreajuste.

O artigo de Jayapandian et al. (2023), apresenta um modelo de detecção de SPAM usando Regressão Logística, comparando seu desempenho com o algoritmo *Naïve Bayes*. Os resultados mostraram que a Regressão Logística alcançou uma precisão significativamente maior, variando entre 97,41% e 99,35% em diferentes conjuntos de dados, enquanto o *Naïve Bayes* obteve precisão entre 82,63% e 88,26%. O modelo proposto no trabalho de Jayapandian et al. (2023) se destacou pela sua interpretabilidade, permitindo entender quais atributos dos *e-mails* têm maior influência na classificação, além de demonstrar boa adaptabilidade a novas estratégias de burlar filtros de SPAM.

A Tabela 1 apresenta um comparativo entre os estudos citados, evidenciando as diferenças de objetivos, algoritmos utilizados em cada estudo e os desafios encontrados pelos autores. A forte presença do algoritmo *Naïve Bayes* e o desafio de aperfeiçoar o modelo de *machine learning*, a fim de combater as vulnerabilidades do filtro de SPAM, se destacam entre os estudos.

Tabela 1. Trabalhos Correlatos. Fonte: Autor.

Estudo	Objetivo	Algoritmos Utilizados	Desafios
Dada et al. (2019)	Revisar diferentes abordagens de *Machine Learning* para filtros de SPAM	*Naïve Bayes*; Redes Neurais; *Firefly*; Árvore de Decisão; *NBTree*; Regressão Logística	Acompanhar o aperfeiçoamento das técnicas de SPAM
Kuchipudi, Nannapaneni e Liao (2020)	Analisar a manipulação de palavras-chave, abordando técnicas para burlar os filtros de SPAM	*Naïve Bayes*	Diminuir a incidência de tentativas concretas de invasão a filtros de SPAM
Yaseen et al. (2021)	Investigar modelos avançados de detecção de SPAM	*BERT Base Cased*; *BiLSTM*	Lidar com a insuficiência dos recursos computacionais
Jayapandian et al. (2023)	Analisar e comparar performance de algoritmo de Regressão Logística com o *Naïve Bayes*	*Regressão Logística*; *Naïve Bayes*	Potenciais vulnerabilidades a padrões específicos de SPAM

Os trabalhos correlatos de Dada et al. (2019), Kuchipudi, Nannapaneni e Liao (2020), Yaseen et al. (2021) e Jayapandian et al. (2023), divergem deste estudo em escopo e aplicação. Enquanto Dada et al. (2019) revisa abordagens gerais e Kuchipudi, Nannapaneni e Liao (2020) analisa fragilidades do *Naïve Bayes*, este trabalho emprega Regressão Logística para classificar *e-mails* com foco em pré-processamento textual e baixo custo computacional. Yaseen et al. (2021) utiliza modelos complexos como BERT, contrastando com a escolha do presente trabalho por algoritmos simples, e Jayapandian et al. (2023) compara desempenho em *datasets* amplos, enquanto este delimita-se ao corpo do *e-mail*, visando aplicabilidade prática e preservação de reputação de domínios.

4. Metodologia

O presente trabalho adota uma abordagem quantitativa experimental para desenvolver um classificador de *e-mails* utilizando o algoritmo de Regressão Logística. Seguindo o processo do KDD, o desenvolvimento foi realizado com a linguagem de programação *Python*, amplamente utilizada para tarefas de aprendizado de máquina. Para isso, foram empregadas bibliotecas como *pandas*, *numpy* e *sklearn*, utilizadas respectivamente para manipulação de dados, operações matriciais, divisão do conjunto de dados e implementação dos algoritmos de *machine learning*. Para efeito de comparação, também foi treinado um modelo utilizando o algoritmo *Naïve Bayes*.

4.1. Fluxo Metodológico

A Figura 2 ilustra o fluxo base utilizado para a realização do trabalho, que seguiu a metodologia do KDD. Desta forma, conforme a ilustração, foram utilizados conjuntos de dados de domínio público. A partir destes dados, iniciou-se o pré-processamento dos dados, que desempenha um papel fundamental, envolvendo a limpeza, normalização e eventual transformação dos dados brutos, de modo a prepará-los adequadamente para as etapas subsequentes de mineração e avaliação. A mineração de dados, etapa seguinte, utiliza os algoritmos de Regressão Logística e *Naïve Bayes*.

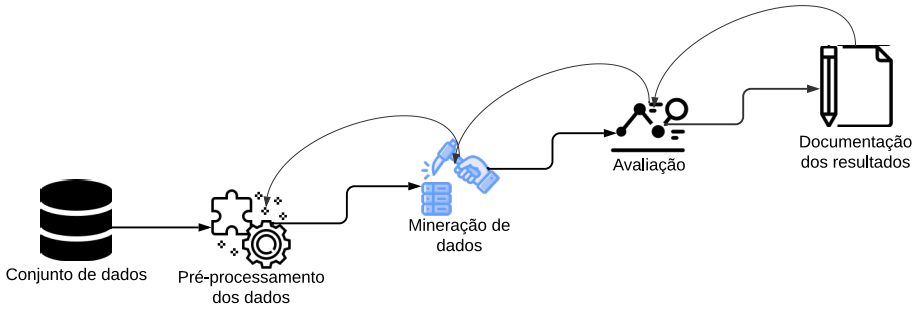

Figura 2. Fluxo Metodológico. Fonte: Autor.

Finalmente, a documentação dos resultados foi parte essencial do projeto, consolidando as descobertas e percepções geradas a cada iteração durante o processo de análise. Esse material serve como insumo para a etapa final de verificação dos resultados obtidos. É importante ressaltar que o processo de mineração de dados não é linear, mas iterativo, permitindo retornar a etapas anteriores sempre que necessário, garantindo assim um processo flexível e robusto de análise de dados.

4.2. Descrição dos *datasets*

Foram escolhidos três conjuntos de dados, dois deles estão disponíveis na plataforma *Kaggle* publicados por Almeida e Hidalgo (2012) e Garnepudi (2019), que serão referidos respectivamente como *Dataset 1* e *Dataset 2*. O terceiro está disponível na plataforma *Hugging Face* publicado por Stone (2023), que será referido como *Dataset 3*. A Tabela 2 traz uma visão detalhada do tamanho e distribuição de classes dos *datasets*.

Tabela 2. Detalhes dos *datasets*

	Número de Amostras	Tamanho	SPAM	HAM
Dataset 1	5572	505 kB	747 (13,4%)	4825 (86,6%)
Dataset 2	5171	5,5 MB	1499 (28,9%)	3672 (71,1%)
Dataset 3	10749	39,2 MB	3795 (35,3%)	6954 (64,7%)

4.3. Seleção e Preparação dos Dados

o *Dataset 1* e o *Dataset 2* já haviam passado por uma normalização mínima, como o *parse* do HTML e resolução de codificação de caracteres, ou seja, padrões de caracteres que claramente deveriam ser UTF-8, mas foram decodificados erroneamente. O *Dataset 3* possuía somente algumas poucas amostras normalizadas, então ele precisou passar pelo parse do HTML, a remoção de rodapés e termos que identificavam a origem ou o

patrocinador da amostra. A escolha de seguir com alguns *datasets* minimamente normalizados, segue o objetivo específico de testar e comparar diferentes técnicas e algoritmos. Portanto, a necessidade de um *dataset* com mais amostras e mais variedade de *features*, motivou a busca pelo terceiro *dataset*.

A partir da obtenção dos dados e um pré-processamento inicial, foi realizada uma análise para remoção de *stopwords*, que são palavras que aparecem com frequência em textos, mas carregam pouca informação. A remoção destas palavras pode aumentar a relação sinal-ruído em textos não estruturados e, assim, aumentar a significância estatística de termos que podem ser importantes para uma tarefa específica (SARICA; LUO, 2021) .

Na etapa de transformação dos dados, foi aplicado o processo de tokenização, que consiste na segmentação do texto em unidades significativas, conhecidas como tokens. Essa substituição de informações por símbolos identificadores, os *tokens*, permite a extração de atributos e termos do *e-mail*, sem considerar seu significado real (DADA et al., 2019). Na sequência, foi aplicada a vetorização, onde os *tokens* são convertidos em representações numéricas. Essas representações, os vetores, serviram como entrada para o algoritmo de mineração de dados. Para a tokenização e vetorização, foi utilizada a classe *TfidfVectorizer* da biblioteca *sklearn*, que implementa a técnica TF-IDF (*Term Frequency - Inverse Document Frequency*). O TF-IDF extrai informações com base na frequência do termo no documento e na coleção de dados, ajudando a destacar palavras que são importantes em um documento (MARTINS et al., 2020).

4.4. Desenvolvimento do Modelo

O desenvolvimento do modelo de classificação foi conduzido de forma iterativa e exploratória, visando identificar a configuração mais eficaz para a detecção de *e-mails* SPAM. Esta fase da pesquisa foi dividida em duas etapas principais: escolha dos algoritmos e otimização dos classificadores.

4.4.1. Escolha dos algoritmos de classificação

A Regressão Logística foi escolhida como algoritmo principal, devido à sua flexibilidade matemática e capacidade de produzir resultados interpretáveis e eficazes em problemas de classificação binária, conforme descrito por (HASTIE; TIBSHIRANI; FRIEDMAN, 2001). Além disso, o modelo fornece probabilidades diretamente interpretáveis por meio da transformação $logit$[1], permitindo *feedback* detalhado para o usuário final, o que enriquece a análise preditiva. A implementação foi conduzida de forma progressiva, iniciando com uma configuração básica e adaptando-se com base nos resultados observados, buscando melhorar a precisão do modelo.

Para ter uma base comparativa, o algoritmo *Naïve Bayes* também foi escolhido. Devido à sua popularidade na classificação de texto, especialmente na detecção de e-mails SPAM, considerou-se relevante treinar um modelo com esse algoritmo, que é mais simples e computacionalmente eficiente. Diferentemente da Regressão Logística, o Naïve Bayes baseia-se na suposição de independência condicional entre as variáveis preditoras,

[1] A função logit é o log natural das probabilidades de que Y seja igual a uma das categorias (GRACE-MARTIN, 2019)

o que reduz a complexidade do treinamento e permite lidar bem com grandes volumes de dados textuais. No entanto, essa simplificação pode limitar seu desempenho quando as variáveis apresentam forte correlação, cenário em que a Regressão Logística tende a oferecer maior flexibilidade e precisão. Dessa forma, a comparação entre os dois algoritmos permite avaliar não apenas a acurácia, mas também os *trade-offs* entre *precision* e *recall*, métricas explicadas na Seção 4.5.

4.4.2. Otimização do Modelo

Para a otimização do modelo, foi utilizada a classe *GridSearchCV* da biblioteca *sklearn*. Esta classe recebe uma grade de hiperparâmetros e suas respectivas variações, de modo que, com base em uma métrica pré-definida, ela se encarregue de fazer o treinamento e a validação cruzada com cada uma das variações de hiperparâmetros. Ao final, devolve a melhor configuração de hiperparâmetros, baseada na configuração que obteve a maior pontuação para a métrica escolhida.

Os hiperparâmetros do *TfidfVectorizer* utilizados na busca por grade estão descritos na Tabela 3.

Tabela 3. Grade de hiperparâmetros utilizados na classe *TfidfVectorizer*

Hiperparâmetro	Definição
max_features	Quantidade máxima de recursos que serão vetorizados
ngram_range	Tamanho mínimo e máximo das sequências de palavras consecutivas que serão extraídas como características do texto
min_df	Frequência mínima que um termo deve aparecer nos documentos para ser incluído no vocabulário
max_df	Frequência máxima que um termo pode aparecer nos documentos antes de ser ignorado
stop_words	Remove *stopwords* com base no idioma ou em um dicionário de palavras
sublinear_tf	Aplica escala logarítmica à frequência dos termos (tf = 1 + $\log(tf)$) para reduzir o impacto de palavras que aparecem muitas vezes no mesmo documento
norm	Tipo de normalização aplicada aos vetores TF-IDF, para que todos os documentos tenham a mesma magnitude

Os hiperparâmetros da classe *LogisticRegression* da biblioteca *sklearn*, utilizada para implementar o algoritmo de Regressão Logística, que sofreram variações da sua configuração padrão estão listados na Tabela 4.

Tabela 4. Grade de hiperparâmetros utilizados na classe *LogisticRegression*

Hiperparâmetro	Definição
C	Inverso da força de regularização, onde valores menores aplicam mais regularização (mais penalização aos coeficientes) e valores maiores aplicam menos regularização
solver	Algoritmo de otimização usado para encontrar os coeficientes do modelo, cada um adequado para diferentes tipos de dados e regularização
penalty	Tipo de regularização aplicada aos coeficientes, para controlar a complexidade do modelo e prevenir *overfitting*
class_weight	Pesos para as classes para lidar com *datasets* desbalanceados
max_iter	Número máximo de iterações que o algoritmo de otimização pode executar antes de parar, mesmo que não tenha convergido para a solução ótima

Os hiperparâmetros da classe *MultinomialNB* da biblioteca *sklearn*, utilizada para implementar o algoritmo *Naïve Bayes*, que sofreram variações da sua configuração padrão estão listados na Tabela 5.

Tabela 5. Grade de hiperparâmetros utilizados na classe *MultinomialNB*

Hiperparâmetro	Definição
alpha	Parâmetro de suavização que adiciona uma contagem mínima a todas as características para evitar probabilidades zero quando uma palavra não aparece em uma classe específica
force_alpha	Força a aplicação da suavização mesmo quando o parâmetro *alpha* é automaticamente ajustado
fit_prior	Define se as probabilidades a priori das classes são calculadas com base na frequência das classes nos dados de treino ou se todas as classes são consideradas igualmente prováveis

4.5. Avaliação do Modelo

O modelo foi avaliado seguindo um processo metódico, investigando o impacto de diferentes parâmetros no desempenho do classificador. A validação do modelo foi realizada através de um processo iterativo de experimentação, a validação cruzada, que conforme apresentado por James et al. (2013), a técnica envolve dividir repetidamente o conjunto de dados em partes distintas. A principal vantagem dessa abordagem é fornecer uma estimativa mais realista do erro de generalização, evitando o otimismo excessivo que pode ocorrer ao calcular apenas o erro de treinamento. A função *cross_val_score* e *learning_curve* da biblioteca *sklearn* foram usadas para a validação cruzada, elas implementam a técnica *K-Fold*, que divide o *dataset* em k partes iguais, usando $k - 1$ partes

para treino e 1 para teste em cada iteração, repetindo o processo k vezes para obter uma avaliação mais robusta do modelo.

As métricas utilizadas para avaliar o modelo estão descritas e definidas nas subseções a seguir. Para compreensão das equações descritas abaixo, considere: verdadeiro positivo (TP), verdadeiro negativo (TN), falso positivo (FP) e falso negativo (FN).

4.5.1. Acurácia

A acurácia, representada na Equação 6, é formada pela proporção de predições corretas em relação ao total de predições. Ela não pode ser utilizada sozinha, caso as classes estejam desbalanceadas, ela pode indicar alta acurácia mesmo com o modelo tendo errado todas previsões da classe com menos amostras.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

4.5.2. *Precision*

Precisão, representada na Equação 7, calcula a proporção de predições corretas entre todas as predições positivas feitas pelo modelo. Filtros SPAM buscam minimizar falsos positivos (FP), ou seja, diminuir a quantidade de *e-mails* HAM classificados como SPAM. Desta maneira, a precisão é a métrica ideal para avaliar isso.

$$PREC = \frac{TP}{TP + FP} \tag{7}$$

4.5.3. *Recall*

O *recall*, representada na Equação 8, calcula a proporção de casos positivos corretamente identificados pelo modelo. O *recall* camufla o erro na classe negativa, por este motivo, ela não foi a principal métrica de avaliação deste modelo.

$$REC = \frac{TP}{TP + FN} \tag{8}$$

4.5.4. *F1-Score*

Quando é necessário considerar tanto a precisão quanto o *recall* na avaliação, uma maneira eficaz de comparar dois modelos é por meio da métrica *F1-Score*, representada na Equação 9. Essa métrica é calculada como a média harmônica entre precisão e *recall*, equilibrando o desempenho de ambos os aspectos em uma única medida (SICSU; SAMARTINI; BARTH, 2023).

$$F1 = \frac{2 * PREC * REC}{PREC + REC} \tag{9}$$

4.5.5. AUC

A AUC (*Area Under the Curve*) representa a área sob esta curva, variando de 0 a 1, onde valores próximos a 1 indicam excelente capacidade discriminativa do modelo. Quanto maior a AUC, melhor o modelo no sentido que separa melhor as probabilidades estimadas das duas categorias da variável alvo (SICSU; SAMARTINI; BARTH, 2023).

5. Resultados e Discussão

Visando à construção de um modelo de *machine learning* para a classificação de *e-mails* como SPAM ou HAM, foram realizados experimentos comparativos com os algoritmos Regressão Logística e *Naïve Bayes*. A seguir, apresentam-se os resultados obtidos durante o desenvolvimento do *framework* preditivo.

5.1. Seleção dos Hiperparâmetros dos Algoritmos

A escolha dos hiperparâmetros dos algoritmos foi conduzida de forma sistemática, utilizando o recurso de busca em grade com validação cruzada da classe *GridSearchCV* da biblioteca *sklearn*.

A busca pelos hiperparâmetros ótimos por meio do *GridSearchCV* apresenta complexidade computacional significativa, uma vez que o tempo de execução está diretamente relacionado ao produto cartesiano de todos os valores dos hiperparâmetros testados. O número total de execuções é determinado pela multiplicação entre a quantidade de combinações possíveis de hiperparâmetros e o número de dobras da validação cruzada ($k = 5$ em todos os experimentos realizados).

Para a busca em grade foram definidas as grades de hiperparâmetros que passariam pelo treinamento e validação, a fim de encontrar a melhor configuração para cada algoritmo. As Tabelas 6, 7 e 8 ilustram as grades de hiperparâmetros escolhidos para os testes.

Tabela 6. *Grid* de hiperparâmetros utilizado para o tokenizador TF-IDF.

Parâmetro	Valores
max_features	1000, 3500, 5000, 10000, 20000
ngram_range	(1,1), (1,2)
min_df	0.0005, 0.001, 10% do *dataset*[2]
max_df	0.7, 0.8, 0.9
stop_words	'english'
sublinear_tf	True
norm	'l2', 'l1'

[2]Foi utilizado um valor inteiro percentual das amostras do conjunto de treino, para que somente palavras presentes em pelo menos 10% das amostras fossem utilizadas

Tabela 7. *Grid* de hiperparâmetros para Regressão Logística.

Parâmetro	Valores
C	0.1, 1.0, 5.0, 10.0, 15.0
solver	'liblinear'
penalty	'l1', 'l2'
class_weight	{0: 1, 1: 5}, 'balanced'
max_iter	1000, 1500

Tabela 8. *Grid* de hiperparâmetros para o *Naïve Bayes*

Parâmetro	Valores
alpha	0.1, 3.0, 5.0, 10.0
force_alpha	True, False
fit_prior	True, False

Para otimizar o processo de busca e reduzir o custo computacional, foi adotada uma estratégia iterativa de refinamento. Quando se fez necessário expandir o espaço de busca com novos valores para determinado hiperparâmetro, os valores que não fizeram parte da melhor configuração identificada na iteração anterior foram removidos do *grid* antes da inclusão dos novos valores candidatos. Esta abordagem permitiu explorar de forma mais eficiente o espaço de hiperparâmetros.

Por fim, a Tabela 9 apresenta a melhor configuração de hiperparâmetros para o algoritmo de Regressão Logística e *Naïve Bayes* respectivamente.

Tabela 9. Melhores hiperparâmetros obtidos via GridSearchCV para cada algoritmo

Componente	Parâmetro	Logistic Regression	Multinomial NB
TfIdfVectorizer	*max_features*	20.000	20.000
	min_df	0,001	0,005
	max_df	0,7	0,7
	ngram_range	(1, 1)	(1, 1)
	sublinear_tf	True	True
	norm	'l2'	'l2'
	stop_words	'english'	'english'
Classificador	*C*	20,0	-
	solver	'liblinear'	-
	penalty	'l2'	-
	max_iter	1.000	-
	alpha	-	0,1
	fit_prior	-	True
	force_alpha	-	True

5.2. Comparação entre modelos

Esta seção apresenta uma análise abrangente da eficácia dos algoritmos de classificação treinados e testados para detecção de *e-mails* SPAM. A análise comparativa é apresentada inicialmente através de visualizações que facilitam a identificação de padrões e diferenças de performance entre os métodos, seguida por uma discussão detalhada dos resultados numéricos obtidos. Os resultados são organizados de forma a permitir uma compreensão progressiva, partindo de uma visão geral da performance dos algoritmos até análises específicas que revelam características particulares de cada método e *dataset* utilizado.

O *threshold* ou limiar de decisão é um ponto de corte usado para mapear uma saída sigmoide de uma classificação binária para uma categoria binária. Para apresentar o resultado comparativo entre os algoritmos, foi selecionado o *threshold* com *F1-score* mais alto para cada algoritmo em cada *dataset*, foram testados os valores limiares 0.3, 0.5, 0.6 e 0.7.

A Figura 3 apresenta um *heatmap* comparativo da eficácia dos algoritmos *Naïve Bayes* e Regressão Logística testados nos três *datasets*. O eixo Y da Figura 3, representa o modelo, onde NB é a sigla para *Naïve Bayes*, LR para Regressão Logística. D1, D2 e D3 são os *datasets*, enquanto T representa o *threshold escolhido*. A visualização utiliza uma escala de cores onde tons mais escuros de azul representam melhor eficácia, permitindo identificação rápida de padrões nos resultados. Observa-se que ambos os algoritmos demonstram performance consistentemente elevada, com valores de acurácia superiores a 96% em todos os cenários testados.

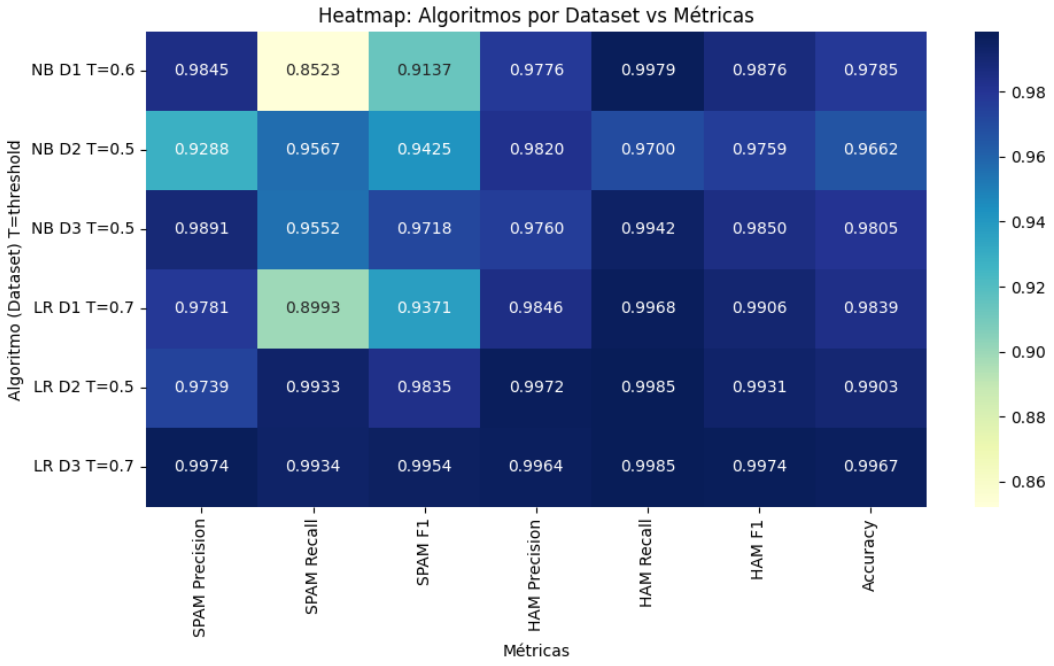

Figura 3. *Heatmap*: comparação de métricas dos melhores resultados para cada *dataset*

A análise comparativa dos algoritmos de classificação revela distintos padrões de comportamento entre os algoritmos *Naïve Bayes* e Regressão Logística. O algoritmo de Regressão Logística demonstra superior estabilidade de performance, particularmente quando utilizado para treinar com os *Datasets* 2 e 3, mantendo valores consistentemente elevados (superiores a 0,97) em todas as métricas avaliadas. Em contraste, o classificador *Naïve Bayes* apresenta maior variabilidade no desempenho, onde o modelo treinado com o *Dataset 1* exibe pontos de menor performance, sugerindo maior sensibilidade às características intrínsecas dos conjuntos de dados utilizados.

A análise das métricas de avaliação revela padrões comportamentais significativos no desempenho classificatório. As métricas HAM *Recall* e HAM F1 consistentemente apresentam valores superiores em comparação às métricas SPAM correspondentes, indi-

cando que os modelos demonstram maior eficácia na identificação correta de instâncias da classe negativa (HAM). Embora a métrica acurácia mantenha valores consistentemente elevados em todos modelos experimentais, esta pode estar mascarando desequilíbrios inerentes às classes, tornando as métricas F1 mais informativas para uma avaliação robusta da performance real dos classificadores.

A observação dos *trade-offs* entre métricas revela a existência de uma correlação inversa sutil entre SPAM *Precision* e SPAM *Recall* em determinadas configurações, evidenciando o *trade-off* clássico entre precisão e *recall*, característico de problemas de classificação binária.

5.3. Análise do Melhor Modelo

O melhor modelo que é analisado a seguir, foi escolhido com base no maior *F1-score* e taxa de precisão. Dessa forma, é analisado o desempenho do método de Regressão Logística treinado com o *Dataset 3*, que obteve *F1-score* de 0,9954 e *precision* de 0,9974.

5.3.1. Matriz de Confusão

As matrizes de confusão obtidas na classificação de *e-mails* SPAM utilizando Regressão Logística no *Dataset 3*, conforme Figuras 4 e 5, evidenciam um desempenho eficaz do modelo, com uma diminuição dos falsos positivos com o aumento do limiar de decisão.

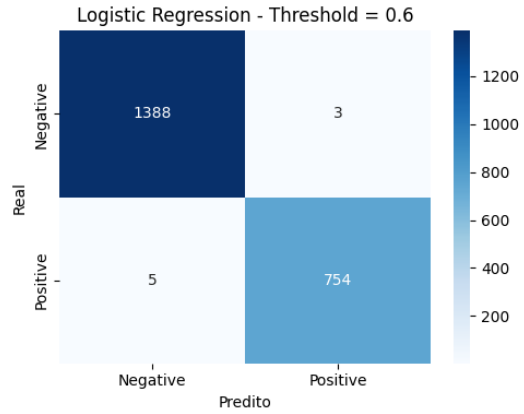

Figura 4. Matriz de Confusão com limiar de decisão 0.6

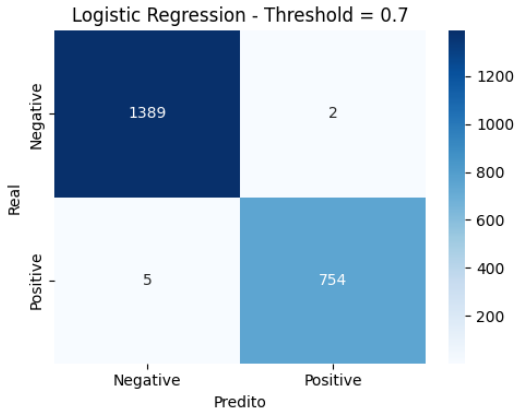

Figura 5. Matriz de Confusão com limiar de decisão 0.7

Com *threshold* de 0.6, o modelo apresentou uma precisão de 99,60%, com 1388 verdadeiros negativos (*e-mails* legítimos corretamente identificados), 754 verdadeiros positivos (SPAM corretamente classificados), apenas 3 falsos positivos e 5 falsos negativos. Já com *threshold* de 0.7, observou-se um aumento na precisão (99,74%), com redução de falsos positivos para apenas 2, sem significar aumento nos falsos negativos.

Esses resultados mostram que o ajuste do *threshold* impacta diretamente o equilíbrio entre precisão e *recall*: um valor mais alto tende a reduzir os falsos positivos (FP), mas pode aumentar ligeiramente a chance de o modelo deixar de identificar um SPAM

(FN), como será apresentado na seção 5.5. Ainda assim, ambos os limiares resultaram em métricas de desempenho elevadas, evidenciando a eficácia do modelo na tarefa proposta.

5.3.2. Análise de *features* mais importantes

A análise dos coeficientes obtidos pelo modelo de Regressão Logística revela padrões linguísticos distintos que caracterizam *e-mails* SPAM e HAM. Os resultados, apresentados nas Figuras 6 e 7 demonstram a eficácia da técnica TF-IDF na identificação de *features* discriminativas para a classificação automática de SPAM.

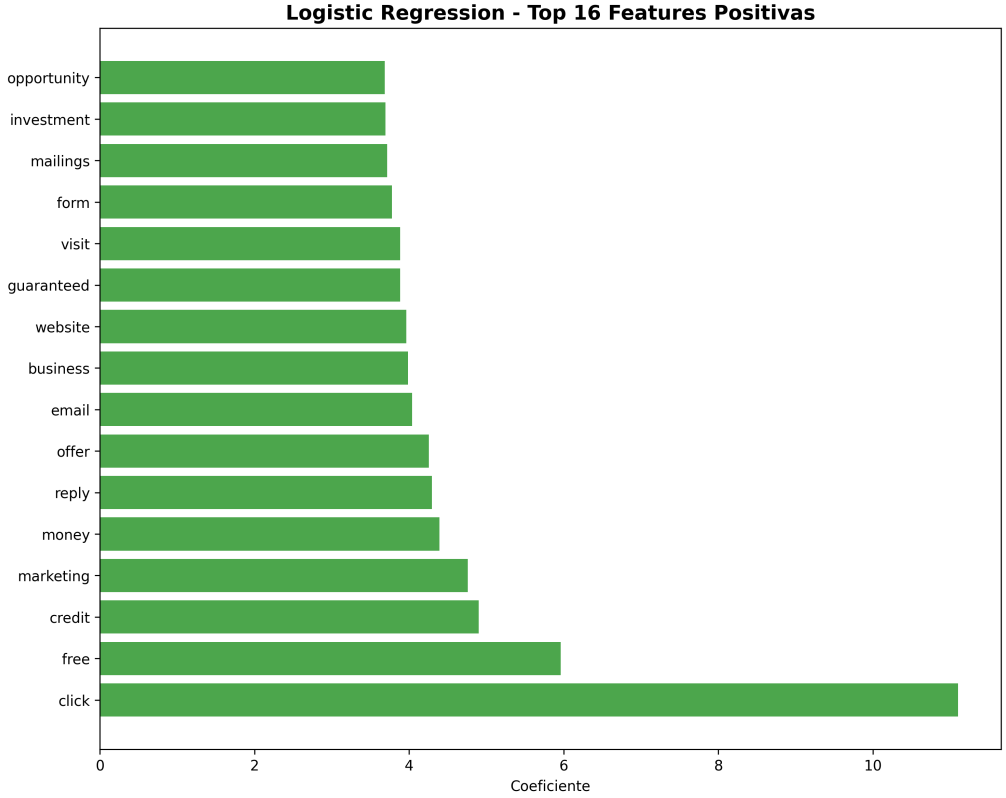

Figura 6. *Top* 16 *Features* Positivas do Modelo de Regressão Logística com o *Dataset 3*. Fonte: Autor.

O conjunto de *features* com coeficiente positivo identificado pelo modelo são apresentadas na Figura 6, onde é notório que a *feature click* apresentou o maior coeficiente (≈ 11), destacando-se como o indicador mais forte de SPAM no *dataset*. Termos relacionados a urgência e manipulação psicológica também apresentaram coeficientes elevados, incluindo *reply* ($\approx 4,3$), *offer* ($\approx 4,1$) e *guaranteed* ($\approx 3,8$). Estes resultados corroboram com o estudo de Khadka et al. (2023) sobre táticas persuasivas em SPAM, onde a criação de senso de urgência é uma estratégia comum.

As *features* relacionadas a ofertas comerciais também aparecem entre as *Top* 16 palavras de maior coeficiente no modelo. *free* (≈ 6), *credit* (≈ 5), *marketing* ($\approx 4,8$) e *money* ($\approx 4,5$). Esta distribuição reflete a natureza comercial agressiva típica de campanhas de SPAM, onde ofertas aparentemente vantajosas são utilizadas como isca para fraudes ou marketing não solicitado.

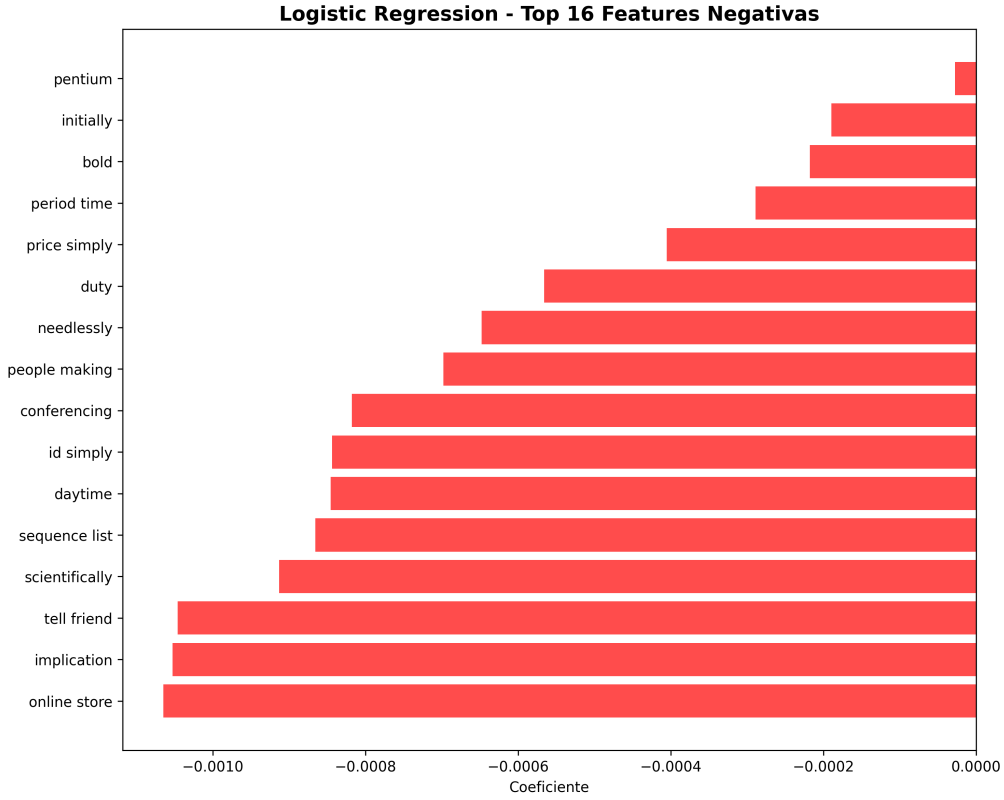

Figura 7. *Top* 16 *Features* Negativas do Modelo de Regressão Logística com o *Dataset 3*. Fonte: Autor.

Já as *features* negativas (Figura 7) que se destacam, ou seja, que indicam relação a *e-mails* legítimos, estão presentes em ambientes técnicos, corporativos e acadêmicos. Considerando que valores mais distantes de zero no eixo negativo sinalizam maior propensão à classe negativa, destacam-se os termos *online store*, *implication*, *tell friend* e *scientifically* com coeficientes entre -0,0008 e -0,0012.

Features relacionadas a discussões técnicas e acadêmicas apresentaram coeficientes negativos consistentes, incluindo *sequence list*, *daytime*, *conferencing* e *people making* com coeficientes entre -0,0007 e -0,0008. Esta distribuição sugere que comunicações técnicas especializadas são menos suscetíveis a manipulação por *spammers*, devido à barreira do contexto em que os destinatários estão inseridos e do conhecimento técnico necessário. Termos relacionados a contextos profissionais e acadêmicos, como *period time*, *price simply* e *duty* com coeficientes entre -0,0003 e -0,0006, demonstram que discussões estruturadas e formais são características de comunicação legítima.

5.4. Análise do Modelo com Maior Contraste de Performance

Com base no *heatmap* apresentando na Seção 5.3.1, onde é possível observar o maior contraste entre as métricas *precision* e *recall* no modelo treinado com o algoritmo *Naïve Bayes* utilizando o *Dataset 1* (NB DT 1 T=0.6), fez-se uma comparação com o modelo treinado com Regressão Logística no mesmo *dataset*.

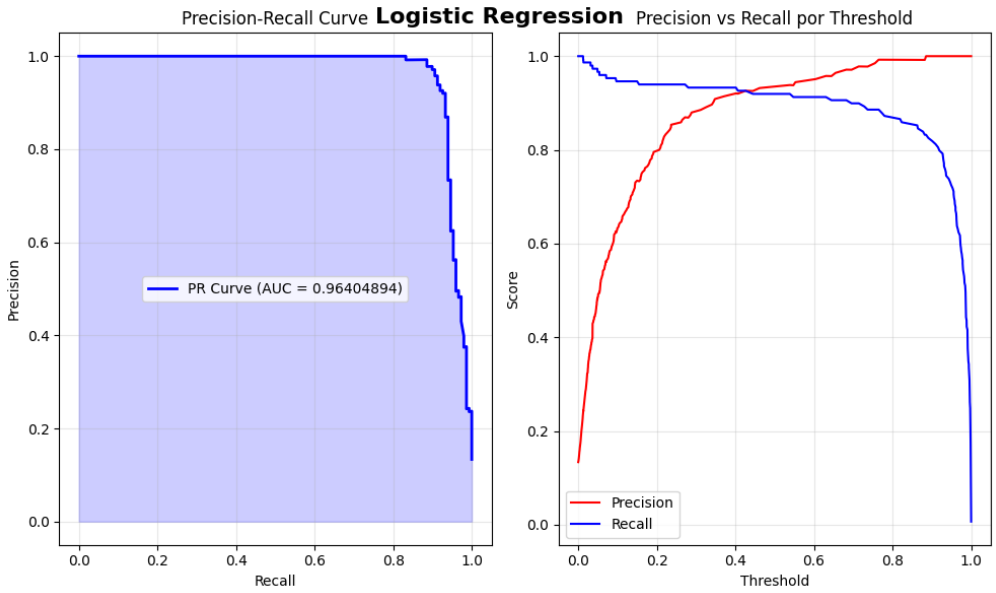

Figura 8. Curva *Precision* x *Recall* e gráfico de *trade-off* para o modelo treinado com Regressão Logística no *Dataset 1*

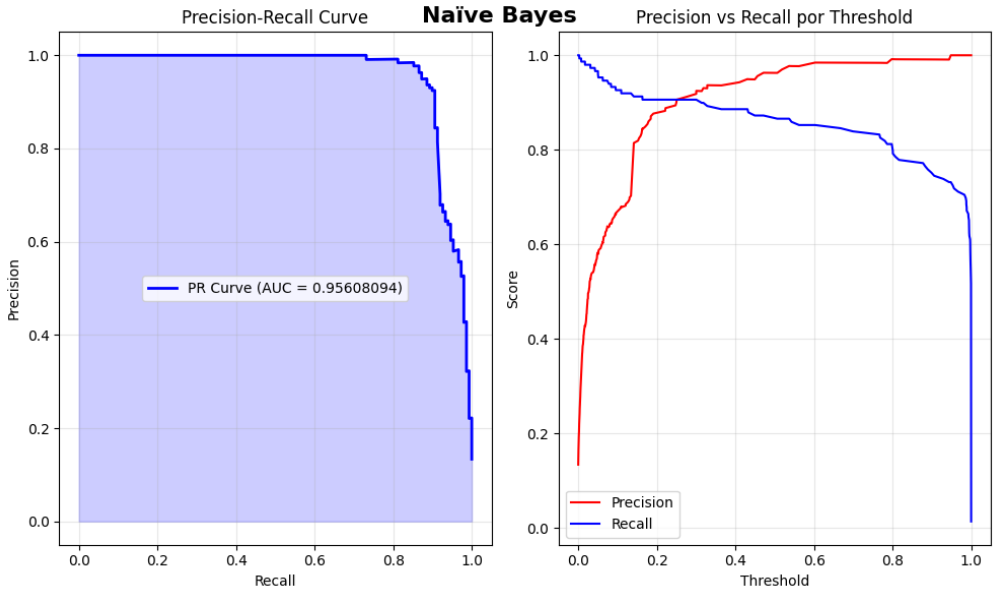

Figura 9. Curva *Precision* x *Recall* e gráfico de *trade-off* para o modelo treinado com *Naïve Bayes* no *Dataset 1*

A avaliação comparativa, ilustrada nos gráficos disponíveis nas Figuras 8 e 9, revela diferenças na capacidade discriminativa, embora ambos os modelos demonstrem eficácia elevada. A análise quantitativa das áreas sob a curva (AUC) *Precision-Recall* indica uma vantagem do algoritmo de Regressão Logística ($AUC = 0,9640$) em relação ao *Naïve Bayes* ($AUC = 0,9561$), representando uma melhoria de aproximadamente 0,8 pontos percentuais na capacidade de discriminação entre classes.

O comportamento das curvas *Precision-Recall* evidencia características operaci-

onais distintas entre os algoritmos quando submetidos às mesmas condições de treinamento. O classificador *Naïve Bayes* apresenta uma degradação mais acentuada da precisão em níveis elevados de *recall*, com declínio pronunciado iniciando aproximadamente no ponto de *recall* $0,75$. Esta característica sugere limitações na capacidade do modelo de manter alta precisão quando configurado para detectar uma maior proporção de instâncias positivas da classe SPAM. Em contraste, o algoritmo de Regressão Logística exibe maior robustez, mantendo precisão elevada ao longo de uma faixa mais ampla de *recall*, com degradação mais suave e controlada.

A análise das curvas de precisão e *recall* em função do *threshold* reforça que o algoritmo de Regressão Logística apresenta maior estabilidade operacional comparado ao *Naïve Bayes*. Ambos os modelos demonstram comportamento similar nos gráficos de *trade-off* em função do *threshold*, com o *Naïve Bayes* apresentando oscilações mais pronunciadas no *recall*, especialmente em *thresholds* elevados. O algoritmo de Regressão Logística, por sua vez, mantém maior estabilidade em ambas as métricas, com transições mais suaves e platôs bem definidos na faixa de *threshold* intermediária $(0,3 - 0,7)$, característica desejável para implementações práticas onde a calibração de *threshold* é crítica.

Os resultados obtidos demonstram que, embora ambos os algoritmos apresentem eficácia competitiva no *Dataset 1*, o algoritmo de Regressão Logística oferece vantagens operacionais significativas. A diferença na AUC *Precision-Recall*, embora numericamente modesta, representa uma melhoria consistente na capacidade de discriminação, particularmente relevante em aplicações de detecção de SPAM onde a minimização de falsos positivos é prioritária. A maior robustez da Regressão Logística em diferentes configurações de *threshold* sugere melhor adequação para cenários de produção onde a estabilidade de performance é fundamental.

Os resultados obtidos neste estudo demonstram a eficácia dos algoritmos de *machine learning* na classificação de *e-mails* SPAM, com destaque para o desempenho superior da Regressão Logística em relação ao *Naïve Bayes*. A análise comparativa revelou padrões consistentes que merecem discussão aprofundada, considerando tanto as implicações práticas quanto as limitações metodológicas identificadas.

A superioridade da Regressão Logística observada nos experimentos alinha-se com os achados de Jayapandian et al. (2023), que também reportaram precisão superior deste algoritmo (97,41% a 99,35%) em comparação ao Naïve Bayes (82,63% a 88,26%). No presente estudo, a Regressão Logística demonstrou maior estabilidade operacional, mantendo valores consistentemente elevados (>97%) em todas as métricas avaliadas, com exceção do *recall* e *F1-score* para o modelo treinado com o *Dataset 1* que ficou com 89% e 93% respectivamente. Por outro lado o Naïve Bayes apresentou maior variabilidade, especialmente no *Dataset 1*.

A análise dos três *datasets* utilizados revelou que as características intrínsecas dos dados impactaram significativamente o desempenho dos modelos. O *Dataset 3*, que apresentou os melhores resultados, possui maior volume de dados (10.749 amostras) e distribuição mais equilibrada entre as classes (64,7% HAM, 35,3% SPAM), corroborando com a literatura que enfatiza a importância do volume e qualidade dos dados para o treinamento de modelos de *machine learning*. O *Dataset 1*, por sua vez, apresentou os maio-

res contrastes de performance, particularmente para o *Naïve Bayes*. Este fenômeno está relacionado ao forte desequilíbrio de classes (86,6% HAM, 13,4% SPAM) e a menor quantidade de amostras, 5.572.

A análise das *features* mais importantes extraídas pelo modelo de Regressão Logística revelou padrões linguísticos consistentes com a literatura sobre técnicas de manipulação em *e-mails* SPAM. A identificação de termos como *click*, *free*, *credit* e *guaranteed* como indicadores fortes de SPAM confirma as estratégias de urgência e apelo comercial descritas por Kuchipudi, Nannapaneni e Liao (2020).

A identificação de *features* negativas, revelou uma associação a contextos técnicos e acadêmicos, como *daytime*, *conferencing*, *implication* e *scientifically*. Este achado sugere que comunicações especializadas são naturalmente menos suscetíveis a serem classificadas como SPAM, devido aos termos técnicos e ao contexto específico dessas mensagens não carregarem apelo comercial.

A análise do impacto do *threshold* na performance dos modelos revelou *trade-offs* importantes entre precisão e *recall*. O ajuste do *threshold* de 0,6 para 0,7 no melhor modelo resultou em redução significativa de falsos positivos. Este resultado é particularmente relevante para aplicações comerciais, onde a classificação incorreta de *e-mails* legítimos (HAM) pode ter consequências econômicas significativas, já que o *e-mail* precisa ser entregue para que vendas aconteçam.

Apesar dos resultados promissores, algumas limitações metodológicas devem ser consideradas. A delimitação do estudo ao corpo textual dos *e-mails*, excluindo anexos, imagens e metadados do cabeçalho, pode ter limitado a capacidade de detecção de técnicas mais sofisticadas de SPAM. Conforme discutido por Dada et al. (2019), a evolução constante das técnicas de *spamming* exige abordagens mais abrangentes que considerem múltiplas dimensões do *e-mail*. Adicionalmente, a utilização exclusiva de *datasets* em língua inglesa pode limitar a generalização dos resultados para contextos linguísticos distintos, particularmente relevante considerando a aplicabilidade em empresas brasileiras de marketing digital.

Este estudo contribui para o campo da classificação de SPAM ao demonstrar a eficácia de técnicas de pré-processamento baseadas em TF-IDF combinadas com Regressão Logística. A metodologia de otimização iterativa de hiperparâmetros utilizando *GridSearchCV* mostrou-se eficiente na identificação de configurações ótimas, com destaque para parâmetros como *max_features*, *min_df* e *max_df*, que se mostraram consistentemente importantes nos diferentes *datasets*, pois permite que somente as n *features* mais importantes sejam utilizadas, eliminando aqueles termos que aparecem várias vezes mas em apenas alguns documentos da mesma classe.

6. Conclusão e Trabalhos Futuros

O presente estudo apresentou evidências empíricas consistentes sobre a eficácia da aplicação de algoritmos de *machine learning* na classificação de *e-mails* SPAM. A análise experimental revelou que a Regressão Logística supera o algoritmo *Naïve Bayes*, oferecendo uma solução computacionalmente eficiente, interpretável e tecnicamente viável para ambientes corporativos.

Com métricas de precisão superiores a 92% em todos os *datasets*, os resultados

reforçam o potencial de implementação em escala industrial, contribuindo para o aprimoramento de sistemas de segurança digital. A metodologia adotada, baseada em técnicas de pré-processamento com TF-IDF, otimização de hiperparâmetros via *GridSearchCV* e avaliação multimétrica, estabelece um protocolo sistemático para o desenvolvimento de classificadores robustos.

Além disso, a análise das *features* evidenciou padrões linguísticos alinhados à literatura sobre manipulação textual em mensagens não solicitadas, oferecendo subsídios teóricos para o aprimoramento contínuo desses sistemas. A inclusão de técnicas complementares, como a análise de metadados presentes nos cabeçalhos dos *e-mails* e a aplicação de análise de sentimentos, desponta como uma estratégia promissora para ampliar a capacidade discriminativa dos modelos, abordando limitações observadas neste trabalho. Tais aprimoramentos podem fortalecer a resiliência dos classificadores frente à evolução das técnicas de evasão utilizadas por emissores de SPAM.

Conforme sugerido na versão preliminar deste artigo, publicada na XX Escola Regional de Banco de Dados (ERBD) com autoria de Batistella e Vieira (2025), a presente pesquisa cumpriu com o objetivo futuro de abranger a otimização dos hiperparâmetros e uma avaliação mais abrangente dos dados.

Os achados obtidos abrem diversas possibilidades para pesquisas futuras. A investigação de técnicas de *ensemble*[3] que combinem Regressão Logística com outros algoritmos pode mitigar deficiências individuais e aprimorar o desempenho geral dos sistemas. A adaptação dos modelos para contextos multilíngues, especialmente no idioma português brasileiro, constitui uma extensão natural e relevante desta pesquisa. Por fim, a exploração de abordagens de aprendizado online, que permitam a atualização contínua dos modelos diante de novas estratégias de envio de SPAM, representa uma direção promissora para garantir a eficácia de longo prazo desses sistemas em ambientes reais e dinâmicos.

7. Disponibilidade do Código Fonte

O código fonte do *framework* definido neste estudo está disponível publicamente no repositório *GitHub*: <https://github.com/joaovitorbatistella/tcc_2_spam_filter_ml>.

Referências

ALMEIDA, T.; HIDALGO, J. *SMS Spam Collection Dataset*. 2012. Último acesso em: 10/02/2025. Disponível em: <https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset>.

BATISTELLA, J.; VIEIRA, A. Análise e classificação de e-mails spam com machine learning. In: *Anais da XX Escola Regional de Banco de Dados*. Porto Alegre, RS, Brasil: SBC, 2025. p. 30–39. ISSN 2595-413X. Disponível em: <https://sol.sbc.org.br/index.php/erbd/article/view/35402>.

[3]Ensemble é uma técnica que combina vários modelos individuais de aprendizado de máquina para criar um modelo preditivo mais poderoso e preciso

BOSTOGANASHVILI, K. *Email IP Reputation: Everything You Need to Know to Get Emails Delivered*. 2024. <https://mailtrap.io/blog/email-ip-reputation>. Último acesso em: 06/12/2024.

DADA, E. G. et al. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, Elsevier, v. 5, n. 6, 2019.

DOSSETTO, F. *Domain reputation, explained*. 2022. Último acesso em: 03/01/2025. Disponível em: <https://postmarkapp.com/glossary/domain-reputation>.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. Knowledge discovery and data mining: towards a unifying framework. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. [S.l.]: AAAI Press, 1996. (KDD'96), p. 82–88.

GARNEPUDI, V. *Spam Mails Dataset*. 2019. Último acesso em: 10/02/2025. Disponível em: <https://www.kaggle.com/datasets/venky73/spam-mails-dataset>.

GOLDSCHMIDT, R. *Data Mining*. 2ª. ed. Rio de Janeiro, RJ, BRA: GEN LTC, 2015. ISBN 9788595156395.

GRACE-MARTIN, K. *What is a Logit Function and Why Use Logistic Regression?* 2019. <https://www.theanalysisfactor.com/what-is-logit-function>. Último acesso em: 22/03/2025.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. New York, NY, USA: Springer New York Inc., 2001. (Springer Series in Statistics).

JAMES, G. et al. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. Disponível em: <https://faculty.marshall.usc.edu/gareth-james/ISL/>.

JAYAPANDIAN, N. et al. Machine learning based spam e-mail detection using logistic regression algorithm. In: IEEE. *2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)*. Indore, India: IEEE, 2023. p. 1–6.

JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing (2nd Edition)*. USA: Prentice-Hall, Inc., 2009. ISBN 0131873210.

KHADKA, K. et al. A survey on the principles of persuasion as a social engineering strategy in phishing. In: *2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. [S.l.: s.n.], 2023. p. 1631–1638.

KOSINSKI, M. *What is phishing?* 2024. Último acesso em: 27/10/2024. Disponível em: <https://www.ibm.com/topics/phishing>.

KUCHIPUDI, B.; NANNAPANENI, R. T.; LIAO, Q. Adversarial machine learning for spam filters. In: *Proceedings of the 15th International Conference on Availability, Reliability and Security*. New York, NY, USA: Association for Computing Machinery, 2020. p. 1–6.

MARIANO, D. C. B.; MARQUES, L. T.; SILVA, M. S. *Data Mining*. Porto Alegre, RS, BRA: SAGAH, 2021. ISBN 9786556900292.

MARTINS, J. S. et al. *Processamentos de linguagem natural.* Porto Alegre: SAGAH, 2020. ISBN 9786556900575.

NASCIMENTO, M. *Como a LGPD pode ajudar a combater spam? (Primeira fase de uma "experiência")*. 2021. <https://www.jusbrasil.com.br/artigos/como-a-lgpd-pode-ajudar-a-combater-spam-primeira-fase-de-uma-experiencia/1243715213>. Último acesso em: 06/12/2024.

PRONNUS. *Segurança digital: Você sabe a diferença entre Phishing e Spoofing?* 2024. Último acesso em: 27/10/2024. Disponível em: <https://g1.globo.com/sc/santa-catarina/especial-publicitario/pronnus/noticia/2024/05/24/seguranca-digital-voce-sabe-a-diferenca-entre-phishing-e-spoofing.ghtml>.

SARICA, S.; LUO, J. Stopwords in technical language processing. *PLOS ONE*, Public Library of Science, v. 16, n. 8, p. 1–13, 08 2021. Disponível em: <https://doi.org/10.1371/journal.pone.0254937>.

SHERWIN, R. *Report Spam, Misclassified, Viral Email Messages*. 2023. Último acesso em: 27/10/2024. Disponível em: <https://www.cisco.com/c/en/us/support/docs/security/email-security-appliance/214133-how-to-submit-email-messages-to-cisco.html>.

SICSU, A. L.; SAMARTINI, A.; BARTH, N. L. *Técnicas de machine learning*. São Paulo, SP, BR: Editora Blucher, 2023. ISBN 9786555063974. Disponível em: <https://app.minhabiblioteca.com.br/reader/books/9786555063974/>.

STATISTA. *Daily number of emails sent worldwide as of April 2024 by country*. 2024. Último acesso em: 26/10/2024. Disponível em: <https://www.statista.com/statistics/1270459/daily-emails-sent-by-country/>.

STATISTA. *Daily number of spam emails sent worldwide as of August 2024, by country*. 2024. Último acesso em: 26/10/2024. Disponível em: <https://www.statista.com/statistics/1270488/spam-emails-sent-daily-by-country/>.

STATISTA. *Global spam volume as percentage of total e-mail traffic from 2011 to 2023*. 2024. Último acesso em: 26/10/2024. Disponível em: <https://www.statista.com/statistics/420400/spam-email-traffic-share-annual>.

STONE, R. *apamassassin*. 2023. <https://huggingface.co/datasets/talby/spamassassin>. Último acesso em: 21/06/2025.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. *Introdução ao Data Mining - Mineração de Dados*. 1ª. ed. Rio de Janeiro, RJ: Editora Ciência Moderna Ltda., 2009.

YASEEN, Q. et al. Spam email detection using deep learning techniques. *Procedia Computer Science*, Elsevier, v. 184, p. 853–858, 2021.