PROJETO FINAL DE GRADUAÇÃO

Proposta de modelo de categorização de Tweets para identificar o discurso de ódio contra a mulher

Geovana de Melo Silva

Júlia Jamile Oliveira Gonçalves

Brasília, Fevereiro de 2023

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
Engenharia de Redes de Comunicação

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

PROJETO FINAL DE GRADUAÇÃO

Proposta de modelo de categorização de Tweets para identificar o discurso de ódio contra a mulher

Geovana de Melo Silva

Júlia Jamile Oliveira Gonçalves

Relatório submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Engenheira de Redes de Comunicação

Banca Examinadora

Dr. Georges Daniel Amvame Nze, EnE/UnB
Orientador

Msc. Valério Aymoré Martins, EnE/UnB
Examinador interno

Esp. Diego Martins de Oliveira, IFB/Brasília
Examinador externo

"Não fui eu que ordenei a você? Seja forte e corajoso! Não se apavore nem desanime, pois o Senhor, o seu Deus, estará com você por onde você andar."

Josué 1:9

"Eu não sonhei com sucesso. Eu trabalhei para ele"

Estée Lauder

"Tenho visto muitas meninas buscando empoderar as outras e ocupando mais espaço. Mas, ao mesmo tempo, ainda há uma longa caminhada"

Serena Fernandes, Méliuz

Agradecimentos

Ao longo da minha jornada acadêmica não faltaram motivos para agradecer a todos que contribuíram para que eu pudesse chegar até aqui. As minhas singelas palavras não irão expressar o quão sou grata por todo incentivo e apoio que recebi durante minha graduação.

Primeiramente, agradeço a Deus por me sustentar em todos os Seus planos e me manter firme, mesmo nos momentos difíceis. Agradeço à Sua misericórdia e bondade. Certamente, eu jamais chegaria até esse momento sem Ele que, com Sua graça, me trouxe vida e esperança.

Aos meus pais agradeço com toda a sinceridade pelo amor e apoio que me dão desde que nasci. Louvo a Deus pelas suas vidas e pela graça que tenho de tê-los como pais e exemplos. Obrigada por todo o esforço que fizeram para eu me tornar quem sou hoje.

Aos meus irmãos e amigos. por terem me aguentado até aqui, por terem me consolado nos meus choros e por terem me feito sorrir minutos depois. Sou grata a Deus por ter vocês em minha vida!

Em especial, agradeço ao meu orientador, Georges Daniel Amvame Nze, cujo apoio, paciência e conselhos foram inestimáveis. Também sou grata aos demais docentes pelo essencial papel desempenhado na transferência de conhecimento.

À minha colega de equipe Júlia Jamile, que trabalhou incansavelmente ao meu lado, tornando este projeto uma realidade.

Por fim, mas, não menos importante, aos meus colegas de curso, prestativos ajudadores nos momentos de dificuldade em projetos e trabalhos.

Geovana de Melo Silva

Em primeiro lugar, quero agradecer a Deus pela força e apoio que recebi até aqui, sem os quais nada teria conseguido. Minha mãe Núbia Regina, meus avós José Joaquim e Neli e meus padrinhos Nília Raquel e José Geraldo, me ensinaram tudo que eu sabia, fizeram o possível e o impossível para que eu aceitasse a melhor educação, desde chegar cedo no ponto de ônibus até o melhor conselho para não abandonar as aulas. Agradeço aos meus primos, Caio e Gustavo, por estarem ao meu lado em todos os momentos da minha vida, bons e ruins. Obrigada a toda minha família que cuidou de mim com todo carinho do mundo e me ensinou o que é o amor e como cuidar do próximo.

Gostaria de agradecer ao meu orientador, Prof. Dr. Georges Daniel Amvame Nze, por aceitar participar deste projeto comigo. Agradeço também à minha parceira de trabalho e amiga, Geovana Silva, por abordar este tema tão importante e desafiador para nós mulheres.

A todos os amigos e amigas que fiz durante a minha vida e, a todos os amigos que trilharam esse caminho comigo. Quero agradecer à empresa Globo por confiar em mim e me dar a oportunidade

mais incrível da minha vida, me proporcionando lições que jamais poderei alcançar. Agradeço ao Departamento de Engenharia Elétrica da Universidade de Brasília por me ajudar e me permitir continuar meus estudos na Universidade. Obrigada a todas as professoras que conheci quando me formei, infelizmente não foram muitas, foram meus espelhos e referências e me fizeram acreditar nos meus sonhos.

Dedico este trabalho a todas as mulheres dentro e fora da engenharia que estão lutando por espaço e sonhando em ser referência e espelho para muitas outras mulheres, para que a sociedade entenda e perceba que somos e podemos ser quem quisermos.

Júlia Jamile Oliveira Gonçalves

Resumo

O desenvolvimento e a disseminação da Internet e dos telefones celulares mudaram a forma como as pessoas se socializam. Enquanto as redes sociais e outras áreas da comunicação digital proporcionam ambientes úteis e convenientes para troca de informações e debate, elas também se tornaram espaços para difundir a violência contra as mulheres.

Vários fóruns de comunicação online, incluindo mídias sociais, permitem que os usuários se expressem livremente, gerando, assim, o aumento de ocorrência de crimes em ambientes virtuais, por gerar no criminoso a sensação de impunidade, uma vez que pode se esconder atrás da tela, mantendo o anonimato.

É incontestável que a violência contra a mulher é um grave problema em quase todos os países do mundo, a qual atinge todas as classes sociais e tipos de pessoas. Diante desse fato, o presente trabalho tem como objetivo identificar, através de um modelo de aprendizado de máquina, mensagens com teor de violência contra a mulher no Twitter. Para auxiliar na obtenção e análise dos dados será utilizado a ferramenta Elastic Stack.

Palavras-chave: Aprendizado de máquina, Violência contra a mulher, Elastic Stack, Twitter

Abstract

The development and spread of the Internet and cell phones have changed the way people socialize. While social networks and other areas of digital communication provide useful and convenient environments for information exchange and debate, they have also become spaces for spreading violence against women.

Several online communication forums, including social media, allow users to express themselves freely, thus generating an increase in the occurrence of crimes in virtual environments, by generating in the criminal a feeling of impunity, since he can hide behind the screen, maintaining anonymity.

It is undeniable that violence against women is a serious problem in almost every country in the world and affects all social classes and types of people. Given this fact, this paper aims to identify, through a machine learning model, messages with violence against women on Twitter. To assist in obtaining and analyzing the data, the ELK Stack tool will be used.

Keywords: Machine Learning, Violence Against Women, ELK Stack, Twitter

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SÍMBOLOS

API	*Application Programming Interface*
CSV	*Comma-separated values*
FN	Falsos negativo
FP	Falsos positivos
HTTPS	*Hyper Text Transfer Protocol Secure*
IBGE	Instituto Brasileiro de Geografia e Estatística
IDF	Índice de importância inversa dos documento
LGPD	Lei Geral de Proteção de Dados
LP	Linguagem Natural
ML	*Machine Learning*
NLP	Aprendizagem Supervisionado para linguagem natural
RPC	Chamadas de procedimento remoto
RLNL	Aprendizagem por Reforço para Linguagem Natural
SSL-NLP	Aprendizagem Semi-Supervisionada para Linguagem Natural
TI	Tecnologia da informação
TLS	*Transport Layer Security*
TN	Verdadeiros negativo
TP	Verdadeiros positivo
TF-IDF	*Term Frequency – Inverse Document Frequency*
TF	Frequência de termo

1 Introdução

1.1 Motivação

Muito se fala sobre a importância da igualdade entre gêneros e sobre como existe essa desigualdade nos dias atuais. Mas, se pensássemos mais a fundo: por que existe mais preconceito contra as mulheres do que com os homens? Por que os direitos das mulheres são diferentes daqueles inerentes aos homens? Tais perguntas podem ser respondidas pela própria história. Ao longo dos anos, a mulher sempre foi submetida ao homem e vista à margem da sociedade pelo simples fato de ser mulher. Tal condição sempre a acompanhou, sendo inegável a influência desta circunstância nas relações de trabalho e na vida social [1].

O papel da mulher estava restrito a ficar dentro de casa, cuidando dos afazeres domésticos e submetida ao homem. Assim, durante muito tempo, ela não participou e não esteve presente na sociedade [2]. A fala da mulher era considerada abominável e a sua presença perante a sociedade era considerada um escândalo. Com o passar dos anos as mulheres adquiriram alguns diretos, por exemplo: o voto feminino, o direito de estudar, a licença maternidade, a Lei Maria da Penha, entre outros. Apesar das conquistas, ainda é necessário lutar pelos direitos iguais. As desigualdades entre homens e mulheres são históricas e cabe à sociedade estudar para mudar o comportamento e o pensamento sobre a igualdade de gênero.

Existe muito preconceito e intolerância contra a mulher nos mais diversos contextos e um exemplo disso são as redes sociais. Muitas vezes, os casos de violência são legitimados por visões estereotipadas da mulher. O preconceito da vida real está passando para as redes sociais [3], onde as pessoas compartilham os sentimentos e suas atividades diárias. Os usuários das redes sociais se sentem confortáveis para fazer e/ou falar qualquer assunto sem serem punidos.

A internet é considerada a "terra sem lei", onde os usuários das redes sociais pensam que podem falar ou gerar conteúdos ofensivos e que não serão punidos. A tarefa de detecção de discurso de ódio nas redes sociais é bem recente, assim como a regularização desses crimes online. A lei do "Marco Civil da internet"(Lei nº 12.965, de 23 de abril de 2014) cria princípios, garantias, direitos e deveres para o uso da internet no Brasil [4].

O acesso à internet está cada vez mais comum na sociedade brasileira e, conforme a pesquisa do Instituto Brasileiro de Geografia e Estatística (IBGE) de 2021, 90,0% dos domicílios do país possuem Internet [5].

Com o aumento de pessoas com acesso à rede mundial, a Internet, foi necessária a criação de leis de combate a crimes cibernéticos. Exemplos desse tipo de crime no mundo virtual são a invasão de privacidade, a clonagem de contas bancárias e de contas nas redes sociais, a divulgação de notícias falsas, a pedofilia, crimes contra a honra, os discursos de ódio, o preconceito, o racismo e muitos outros. Segundo o site do Poder Judiciário do Estado do Rio de Janeiro: "O mundo das leis não acompanhou no mesmo ritmo esse crescimento vertiginoso da internet e dos crimes

virtuais. Apesar de existir comercialmente no Brasil desde meados dos anos 90, somente em 2008 uma lei alterou o Estatuto da Criança e do Adolescente, para penalizar a pornografia infantil virtual. Leis específicas de combate a crimes virtuais, alterando o Código Penal, só entraram em vigor em 2012."[6].

Muitas vezes as pessoas usam do anonimato, ou seja, um perfil anônimo, para realização do preconceito, seja contra a mulher ou outras minorias, não demostrando, assim, sua verdadeira identidade. Com isso, torna-se ainda mais difícil aos órgãos responsáveis punir tais atos ilícitos na internet, gerando, assim, o pensamento de que tudo que é ilegal nas redes sociais não tem condenação [7].

1.2 Objetivos do trabalho

1.2.1 Objetivos Gerais

Utilizar *Machine Learning* para identificar discurso de ódio contra a mulher em amostras de dados da rede social Twitter, a fim de melhorar o entendimento da extensão e da natureza desses discursos e contribuir para o desenvolvimento de estratégias dirigidas ao combate do ódio online.

1.2.2 Objetivos Específicos

- Coletar amostras de dados através da API do Twitter que contenham possíveis discursos de ódio contra a mulher.
- Processar e analisar esses dados para identificar discursos de ódio contra a mulher.
- Avaliar a eficácia dos algoritmos de *Machine Learning* usados para selecionar os discursos de ódio.
- Criar relatórios e apresentar os resultados das análises.
- Analisar e visualizar alguns dados usando Kibana e Elasticsearch.

1.3 Questão de Pesquisa

O intuito deste trabalho é usar a API para coletar textos no aplicativo "Twitter", detectando nesses textos, através de *Machine Learning*, quais dessas passagens têm o teor de preconceito contra as mulheres. Em outras palavras, será implementado um modelo de categorização de *tweets* com a utilização de aprendizado de máquina, de forma a auxiliar a identificação de publicações que estejam promovendo discurso de ódio contra as mulheres. Com esses dados, é possível ter uma análise matemática de quantas mensagens promovem o preconceito.

2 Revisão bibliográfica e Trabalhos Relacionados

2.1 Discurso de ódio

A definição de discurso de ódio não é universalmente aceita, tampouco os aspectos de sua definição. A linha entre o discurso de ódio e a liberdade de expressão adequada é uma linha tênue, o que torna algumas pessoas cautelosas em dar uma definição precisa de discurso de ódio[8].

Segundo Nockleby JT, discurso de ódio é o discurso que ataca uma pessoa ou grupo com base em atributos como raça, religião, origem étnica, nacionalidade, sexo, deficiência, orientação sexual ou identidade de gênero [9]. Já Álvaro Paul Diaz destaca que o discurso do ódio deve ser mais que uma manifestação de antipatia, ou seja, indicar a hostilidade contra determinado grupo [10].

Pode-se dizer que a produção de ódio passa também por fases preparatórias, como o estímulo ao preconceito, na perspectiva de ativar no grupo dominante "percepções mentais negativas em face de indivíduos e grupos socialmente inferiorizados".[11]

O discurso de ódio está presente em todas as esferas da sociedade, inclusive no meio virtual. De acordo com uma visão amplamente difundida na mídia e no discurso público em geral, o ódio online é um problema social em escala global. Segundo [12] o ódio online é a atividade de postar online uma avaliação explicitamente negativa de uma pessoa ou objeto, principalmente com o propósito de expressar uma atitude negativa em relação a aquela pessoa ou objeto, independentemente de causar dano real a uma pessoa concreta, provocar a resposta de outros ou diminuir o valor de um determinado grupo social.

2.2 Preconceito contra as mulheres

A violência contra as mulheres não é recente na história. Ela faz parte de um sistema sócio-histórico que condicionou as mulheres a uma posição hierarquicamente inferior na escala de perfeição metafísica, produzindo relações assimétricas entre homens e mulheres em nossa sociedade [13].

A discussão acerca das desigualdades entre homens e mulheres, como sabemos, não é recente, muito pelo contrário. Por exemplo, na idade antiga, a democracia ateniense não considerava a mulher cidadã, equiparando-a a escravos e estrangeiros e, na idade média, a igreja católica considerava a mulher amaldiçoada, sendo terrivelmente perseguida como bruxa no tempo da inquisição [14].

O fenômeno da violência contra a mulher se configura a partir da discussão sobre gênero, em que há "a construção social e histórica produzida sobre as características biológicas" [15]. Inegavelmente, homens e mulheres são diferentes em termos biológicos, no entanto, o problema

está em como, por intermédio da socialização, se concebem essas diferenças, de modo que há questões que acontecem particularmente com mulheres, por serem mulheres[16].

Atualmente, embora ela tenha conquistado diversos direitos de ordem internacional e nacional, além de estar equiparada ao homem em direitos e obrigações, de acordo com o artigo quinto da Magna Carta Nacional, a mulher ainda sofre opressão social, tendo em vista que o pensamento de que ela seja inferior ao homem ainda persiste, manifestando-se através dos altos números de violência física, sexual, moral e psíquica sofrida pela mulher brasileira [14].

2.3 Formas de violência contra a mulher

A violência contra a mulher é um comportamento violento ou abusivo que é perpetrado contra uma mulher, seja por um parceiro íntimo, familiar, conhecido ou estranho. Isso pode incluir agressão física, sexual, psicológica, econômica ou negligência. É uma questão grave de direitos humanos que possui um impacto profundo e duradouro na vida das mulheres.

2.3.1 Violência Física

A violência física contra a mulher é um comportamento violento ou agressivo que é perpetrado contra o sexo feminino, seja por um parceiro íntimo, familiar, conhecido ou estranho. Isso pode incluir agressão, sequestro, lesões corporais, entre outros. A violência física tem um impacto significativo e duradouro na saúde e segurança das mulheres, podendo levar a sequelas físicas e psicológicas graves [17].

2.3.2 Violência Sexual

A violência sexual contra a mulher é qualquer forma de abuso ou assédio sexual perpetrado contra a mulher, seja por parceiro íntimo, familiar, conhecido ou estranho. Isso pode incluir estupro, agressão sexual, assédio sexual, etc. A violência sexual possui efeitos profundos e duradouros na saúde e no bem-estar das mulheres, podendo acarretar sérias consequências físicas e psicológicas. É importante tomar medidas para prevenir a violência sexual contra as mulheres e garantir que elas tenham acesso a serviços de apoio e proteção quando vivenciam a violência sexual [18].

2.3.3 Violência moral e psicológica

A violência moral e psicológica contra a mulher se refere ao abuso psicológico ou emocional que é perpetrado contra uma mulher, geralmente por um parceiro íntimo ou familiar, podendo incluir humilhação, ameaças, controle, isolamento social, difamação, culpa, entre outras formas de abuso psicológico. Referidas formas de violência podem ter um impacto profundo e duradouro na saúde mental e bem-estar da vítima, podendo ser igualmente tão prejudiciais quanto outras modalidades de violência física [19].

É importante que as mulheres tenham acesso a recursos e apoio para lidar com a violência moral e psicológica. Também tem que ser tomadas medidas para prevenir este tipo de abuso [20].

2.4 Redes sociais

Com o advento da tecnologia da informação (TI) e dos meios de comunicação, as redes sociais tornaram-se ferramentas cada vez mais populares na Internet [21]. Para [22], este momento é caracterizado pela cibercultura, que se consolidou entre as décadas de 1980 e 1990 com a informática de massa e a popularização da internet, as quais ganharam força após a criação da World Wide Web (WWW), em 1991.

A comunicação está cada dia mais complexa e tornando a informação cada vez mais acessível. Com isso, as redes sociais não se limitam mais ao relacionamento, mas também como fonte de pesquisa e notícias, tendo como atributos a interatividade e participação, possibilitando não só o acesso à informação, mas a capacidade de produzi-la [23].

As diversas formas de comunicar-se aproximaram os contatos entre as pessoas, tanto as próximas quanto as distantes, possibilitando descobertas e interações dificilmente pensadas antes de seu advento [24]. Exemplo dessa mudança pode ser verificada nas empresas que passaram a utilizar as redes não só como um canal de publicidade, mas também como um meio de comunicação e aproximação com seu público, agregando valor ao seu produto e/ou serviço, gerando assim um diferencial competitivo e alavancando o negócio [21].

2.4.1 API do Twitter

O Twitter é uma rede social visitada por milhões de pessoas e é uma das redes sociais mais utilizadas no mundo. Como a maioria das redes sociais, o objetivo do Twitter é criar e compartilhar conteúdo. Na maioria das vezes, isso é feito com frases, mas fotos ou vídeos também podem ser compartilhados.

O Twitter é um ótimo lugar para as pessoas abordarem e publicarem sobre qualquer coisa e como elas se sentem em relação a algum assunto, fornecendo, assim, comportamentos, emoções e avaliações. Diante disso, percebe-se que essa plataforma possui uma grande quantidade de dados, fazendo com que tal rede seja uma ótima fonte para analisar os dados de todos os tipos [25], pois:

- A Interface de Programação de Aplicação (API) é organizada e vem com ferramentas de desenvolvimento avançadas;
- Os dados que ele fornece são repletos de informações e possuem uma estrutura de database bem estabelecida para a análise;
- Os dados do Twitter estão disponíveis de forma fácil e gratuita para qualquer pessoa com direitos apropriados para usá-los.

A API do Twitter permite realizar consultas complexas, como obter tweets sobre um determinado tópico ou segmentar usuários que moram em um determinado local. Portanto, essa API será de grande valia para fornecer dados com o intuito de detectar atos discriminatórios contra a mulher [26].

2.4.2 O preconceito contra a mulher nas redes sociais

Atualmente é notório o preconceito que a mulher sofre num mundo ainda machista, mesmo com todo espaço conquistado ao longo do tempo. Com o desenvolvimento da tecnologia, surgem cada vez mais formas de preconceito, ou seja, a tecnologia de hoje se apresenta como facilitadora e até mediadora do preconceito, seja contra a mulher ou contra outras minorias, emergindo mais facilmente na internet e, mais precisamente nas redes sociais [27].

Como mencionado anteriormente, as redes sociais tornaram-se um grande centro de disseminação de informações e ideias, devido a fatores como a sensação de anonimato, a velocidade com que as informações se espalham e a recente facilidade de acesso aos planos de internet. Com esses fatores não é de se espantar que este meio também seria utilizado para pessoas propagarem discursos de ódio [28].

O foco principal deste escrito é identificar exemplos de discursos de ódio proferidos contra mulheres. Para tanto, foram retirados alguns exemplos de falas de pessoas públicas na rede social Twitter. As Figuras 2.1, 2.2, 2.3 e 2.4 mostram esses discursos de ódio:

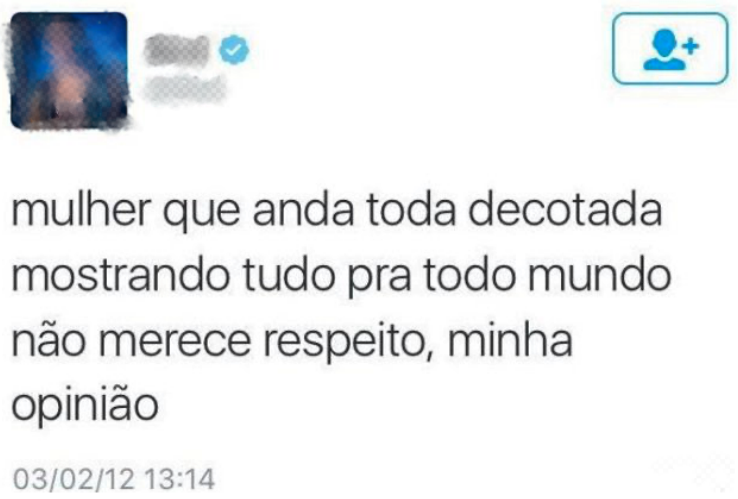

Figura 2.1 – Tweet com conteúdo preconceituoso feito por um cantor famoso. [29]

Figura 2.2 – Tweet com conteúdo preconceituoso feito por um Youtuber famoso. [29]

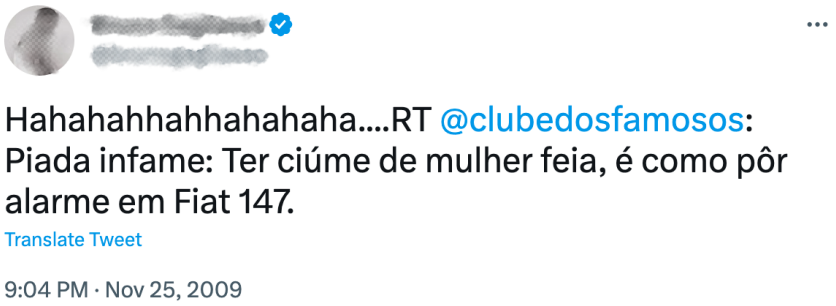

Figura 2.3 – Tweet com conteúdo preconceituoso feito por um ator famoso. [29]

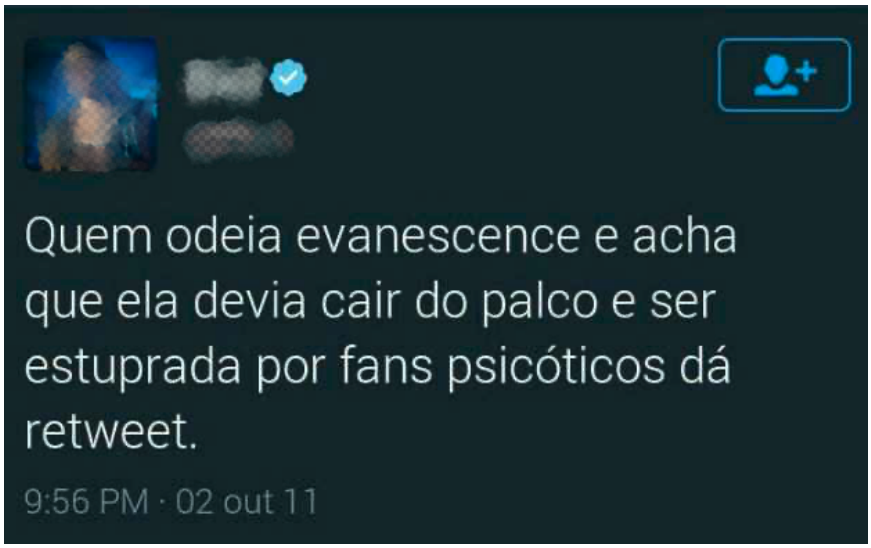

Figura 2.4 – Tweet com conteúdo preconceituoso feito por um cantor famoso. [29]

Essas Figuras foram importantes porque ajudam a formar um dicionário de palavras que são utilizadas pelas pessoas para propagarem seu discurso de ódio. Este dicionário será importante no momento de se construir os modelos de aprendizado de máquina [30]. Para a preservação de dados foram alterados as citações e os links, objetivando seguir a Lei Geral de Proteção de Dados (LGPD).

2.5 Machine Learning

O aprendizado de máquina (ML) é um tipo de inteligência artificial que permite que aplicativos de software prevejam resultados com mais precisão sem serem explicitamente programados [31].

De acordo com [32], ML é o estudo de métodos computacionais que mecanicamente contribuem para melhorar o desempenho, adquirindo conhecimento por meio da experiência. Seu objetivo é substituir atividades demoradas e executadas por humanos pela automatização da tecnologia, objetivando melhorar a precisão e a eficiência do processo, descobrindo e explorando padrões nos dados usados no treinamento.

A ML utiliza ferramentas de análise de dados para modelar padrões complexos e prever resultados futuros. Utilizando processos de aprendizado estatístico, os algoritmos de ML identificam

padrões nas informações de entrada e aprendem a prever resultados a partir destes. [32]

O aprendizado de máquina é um processo contínuo no qual os modelos analíticos são continuamente aprimorados e reimplantados ao longo do tempo. As previsões podem ser implantadas em um aplicativo ou microsserviço de várias maneiras. Uma possibilidade é incorporar modelos analíticos diretamente em aplicativos de processamento de fluxo, como aqueles que usam o Kafka Streams [33].

Técnicas baseadas em aprendizado de máquina foram aplicadas com sucesso em muitos campos, desde reconhecimento de padrões, visão computacional, radiologia, exames médicos, engenharia espacial, redes sociais, finanças, entretenimento e biologia computacional até aplicações biomédicas e médicas. É fácil encontrar o ML no dia a dia das pessoas. Por exemplo, o Google usa ML para melhorar a precisão de suas pesquisas. Os algoritmos de ML são usados para classificar o conteúdo e sites como o YouTube usam o ML para classificar com precisão as pesquisas do usuário [34].

2.5.1 Tipos de aprendizado

A ML é usada para resolver problemas como reconhecimento de voz e imagem, tradução de idiomas, previsão de preços, análise de sentimento e detecção de fraudes. Os algoritmos de ML podem ser aplicados para a tomada de decisões, permitindo que os sistemas sejam mais reativos e adaptáveis.

O aprendizado de máquina clássico geralmente é categorizado pela forma como um algoritmo aprende para se tornar mais preciso em suas previsões. Assim, temos vários tipos de aprendizado.

- Aprendizado supervisionado

  O aprendizado supervisionado usa um conjunto de treinamento para ensinar o modelo a produzir a saída desejada. Esse conjunto de dados de treinamento inclui entradas e saídas corretas, permitindo que o modelo aprenda com o tempo. O algoritmo mede sua precisão por meio de uma função de perda e se autoajusta até que o erro seja suficientemente minimizado. [35]

  Abaixo estão descritos, com mais detalhes, alguns tipos de características dos conjuntos de dados.

  - Underfitting

    Underfitting ocorre quando o modelo de aprendizado supervisionado não consegue capturar completamente as características do conjunto de dados. Isso ocorre geralmente quando o modelo é muito simples para o problema em questão. Por exemplo, se o modelo não tiver suficientes parâmetros para ajustar os dados, isso pode levar a um underfitting. Em geral, existem diversas técnicas que podem ser usadas para identificar e corrigir o underfitting, como a regularização, a adição de mais parâmetros, a seleção de variáveis e a redução de dimensionalidade. Outra maneira de lidar com o underfitting é aumentar a complexidade do modelo.

Isso pode ser feito aumentando o número de camadas ou adicionando mais conexões entre elas [36], bem como aplicando técnicas de regularização, como a regularização L1 ou L2. Outras técnicas comuns para lidar com o underfitting incluem adicionar mais variáveis, reduzir a dimensionalidade dos dados e acrescentar mais parâmetros ao modelo. Finalmente, aumentar o número de amostras de treinamento também pode ajudar a melhorar a capacidade do modelo de aprendizado supervisionado dirigido a capturar as características do conjunto de dados [37].

- Overfitting

  O overfitting é uma situação no aprendizado supervisionado em que um modelo é treinado com um conjunto de dados demasiadamente específicos, resultando em um alto desempenho no conjunto de dados de treinamento, mas um desempenho inferior nos dados de teste. Isso ocorre porque o modelo não generalizou corretamente e aprendeu detalhes específicos do conjunto de treinamento. É por esse motivo que é importante testar o modelo com dados novos para determinar se ele está sobreajustado [38] .

  A prevenção do overfitting envolve a regulação (regularização) dos parâmetros do modelo, a redução da dimensionalidade dos dados e a seleção de características relevantes. A regularização penaliza parâmetros com valores maiores para limitar o ajuste excessivo dos dados de treinamento. A redução da dimensionalidade remove características irrelevantes e redundantes, o que minimiza o ajuste excessivo. A seleção de características relevantes seleciona apenas aquelas mais importantes que contribuem para a generalização do modelo. Outras técnicas incluem o uso de métodos de validação cruzada e o uso de conjuntos de dados maiores [39] .

  Por fim, aumentar o número de dados de treinamento para que o modelo possa generalizá-los melhor também é uma boa prática para prevenir o overfitting. Por exemplo, se a quantidade de dados de treinamento for pequena, o modelo pode acabar se ajustando aos dados de treinamento e não generalizando corretamente para os dados de teste. Ao adicionar mais variáveis de treinamento, o modelo pode se ajustar aos dados e, ainda assim, generalizar corretamente para os dados de teste [39].

- Aprendizagem não supervisionada:

  Esse tipo de aprendizado de máquina envolve algoritmos treinados em dados não rotulados. O algoritmo verifica o conjunto de dados procurando conexões significativas. Os dados nos quais os algoritmos são treinados e as previsões ou recomendações que eles produzem são predeterminados. A partir desses dados, o ML descobriu padrões que poderiam ajudar a resolver problemas de agrupamento ou associação. Isso é especialmente útil quando os experts no assunto não têm certeza dos atributos comuns em um conjunto de dados [40].

- Aprendizagem semi-supervisionada:

  O aprendizado semi-supervisionado ocorre quando apenas parte dos dados de entrada fornecidos é rotulada. O aprendizado não supervisionado e o semi-supervisionado podem ser opções mais atraentes, pois confiar na experiência de domínio para rotular adequadamente os dados para aprendizado supervisionado pode ser demorado e caro [41].

- Aprendizagem por reforço:

  Os cientistas de dados normalmente usam aprendizado por reforço para ensinar máquinas por meio de um processo de várias etapas com regras bem definidas. Os cientistas de dados escrevem algoritmos para realizar tarefas e fornecem dicas positivas ou negativas, à medida que descobrem como fazê-las. Mas, muitas vezes, o algoritmo decide por si mesmo quais etapas seguir ao longo do caminho [42] [43].

2.6 Machine Learning para linguagem natural

O aprendizado de máquina supervisionado para a linguagem natural é uma área de pesquisa que investiga como os computadores podem compreender e processar linguagem natural humana. Isso envolve usar algoritmos de aprendizado de máquina para treinar modelos que possam reconhecer e classificar o significado de palavras, frases e contextos. Os algoritmos podem ser treinados com dados de treinamento que contenham amostras de linguagem natural humana, como notícias, documentos legais, textos acadêmicos e outros. Ao usar esses dados, os algoritmos podem aprender a identificar e classificar palavras, frases e contextos, fornecendo uma forma de entender e processar melhor a linguagem natural humana [35].

Aprendizado de máquina não supervisionado para linguagem natural é um ramo da inteligência artificial que estuda métodos computacionais para processar e analisar grandes volumes de dados textuais. O termo "não supervisionado"significa que o algoritmo não foi treinado com dados previamente classificados e, portanto, deve aprender por meio de seu próprio processo de descoberta. Os algoritmos de aprendizado de máquina não supervisionados podem ser usados para extrair informações úteis e padrões ocultos dos dados textuais, como análise de sentimentos, detecção de tópicos e extração de palavras-chave [40].

A Aprendizagem Semi-supervisionada para Linguagem Natural (SSL-NLP) é um ramo da Inteligência Artificial que visa aplicar aprendizado de máquina semi-supervisionado em tarefas de processamento de linguagem natural. O aprendizado semi-supervisionado consiste na combinação de técnicas de aprendizado supervisionado e não supervisionado. Por um lado, as técnicas de aprendizado supervisionado exigem que os dados de treinamento sejam rotulados, o que pode ser um processo demorado e caro. Por outro lado, as técnicas de aprendizado não supervisionado não precisam de dados rotulados, mas, por vezes, tendem a produzir resultados menos precisos. Assim, o SSL-NLP visa aproveitar o melhor dos dois mundos, usando tanto dados rotulados quanto não rotulados para obter resultados melhores [41].

Algumas das principais técnicas de SSL-NLP incluem a aprendizagem por reforço, a transferência de aprendizado, a aprendizagem ativa, o agrupamento e a anotação Semi-supervisionada. A aprendizagem por reforço usa recompensas para incentivar o sistema a aprender a partir de dados não rotulados. A transferência de aprendizado faz uso de modelos já treinados para ajudar a treinar outros modelos que não possuem dados suficientes para treinar. A aprendizagem ativa envolve a seleção ativa de dados para treinamento. O agrupamento usa algoritmos de clustering para agrupar dados não rotulados para facilitar o treinamento. Por fim, a anotação semi-supervisionada usa algoritmos para inferir rótulos para dados não rotulados [41].

A Aprendizagem por Reforço para Linguagem Natural (RLNL) é uma área de pesquisa emergente que se concentra na aplicação de técnicas de aprendizado de máquina de aprendizado por reforço a problemas de processamento de linguagem natural. A RLNL oferece novas possibilidades para a criação de sistemas inteligentes que possam aprender a interagir com o usuário de maneiras mais eficientes e precisas. Esta área de pesquisa se concentra na aplicação de técnicas de aprendizado de máquina de aprendizado por reforço para problemas de processamento de linguagem natural, como classificação de texto, geração de texto, tradução, detecção de sentimentos e compreensão de linguagem natural. As técnicas de RLNL permitem aos sistemas aprender a partir de exemplos e ajustar-se de acordo com a experiência, permitindo assim que os sistemas se ajustem e se adaptem às necessidades específicas dos usuários [42] [43].

2.6.1 Modelos de Machine Learning

A seguir, estão explanados os modelos de *Machine Learning* usados neste trabalho para realizar a classificação dos textos.

- *MultinomialNB*

  Multinomial Naive Bayes ou também conhecido como *MultinomialNB* é um classificador de Naive Bayes que tem sido amplamente utilizado como modelo de aprendizado probabilístico para classificação de textos. Ele é um modelo de aprendizado supervisionado baseado na teoria de probabilidade de Bayes, o qual assume que as palavras nos documentos são independentes umas das outras [44]. O algoritmo calcula a probabilidade da palavra pertencer a uma dada categoria, utilizando-a para classificar o texto. É bastante útil para classificar textos em categorias como notícias, artigos, comentários, textos preconceituosos, entre outros.

- *SGDClassifier*

  Stochastic Gradient Descent - Classifier (SGDClassifier) é um classificador de descida de gradiente estocástico que é usado para realizar a classificação de texto. Ele é baseado no conceito de descida de gradiente, utilizado para achatar o custo da função de custo usando gradientes [45]. O algoritmo *SGDClassifier* usa um conjunto de dados de treinamento para aprender o gradiente de descida que será usado para classificar novos dados. Ele funciona bem para problemas de grande escala e tem uma boa precisão de classificação.

- *TfidfVectorizer*

  TfidfVectorizer é usado para converter texto em vetores de recursos usando o TF-IDF. O TF-IDF é um método de ponderar as palavras em um documento, o qual é usado para classificar o texto. O algoritmo usa o TF-IDF para calcular a importância relativa de cada palavra no documento e usa essa informação para converter o texto em vetores [46].

- *CountVectorizer*

  CountVectorizer é usado para converter texto em vetores de recursos, utilizando a contagem de palavras. O algoritmo conta o número de vezes que cada palavra aparece em um

documento e usa essa informação para converter o texto em vetores. Os vetores são construídos com base na dimensionalidade do tamanho do vocabulário do texto. Com base nesse vocabulário criado, ele cria uma matriz de termos do documento [47].

2.7 Elastic Stack

O *Elastic Stack* é um grupo de projetos *open source* da *Elastic* projetado para ajudar os usuários a obter dados de qualquer tipo de fonte e em qualquer formato e também pesquisar, analisar e visualizar esses dados. Anteriormente, esse grupo de projetos era conhecido como *ELK Stack*, pois era o acrônimo para os principais produtos do grupo, *Elasticsearch, Logstash e Kibana*, mas foi renomeado como *Elastic Stack*, por causa de um quarto projeto, *Beats*, que foi posteriormente adicionado à pilha. A seguir será descrito com mais detalhes o propósito de cada uma delas:

- *Elasticsearch:*

  O *Elasticsearch* é um mecanismo de pesquisa de código aberto altamente escalável. Ele fornece resultados ao longo de uma Interface RESTful, portanto pode ser utilizado em vários ambientes de programação. Dentre as suas funcionalidades, destacam-se as consultas complexas, como sintaxe SQL, recursos geográficos, núcleo para armazenamento de dados, pesquisa de texto completo, além de lidar com altas cargas de consulta [48].

  No presente trabalho o *Elasticsearch* será responsável por armazenar dados de forma centralizada, proporcionando buscas rápidas que podem ser facilmente analisadas e a possibilidade de customizar buscas, mesmo quando se trata de um grande volume de logs, de modo que é possível encontrar padrões e assim aprimorar as análises.

- *Logstash:*

  Logstash é um pipeline de dados gratuito que fornece uma estrutura integrada para coleta, centralização, análise e pesquisa de logs. Ele pode ser conectado a várias fontes e permite a criação de sistemas analíticos centrais altamente escaláveis. O *Logstash* recebe logs de diferentes fontes, realiza transformação, normalização e agrupamento [49].

  O *Logstash* é responsável pelo processamento dos dados que não estão necessariamente estruturados, independente da fonte e formato dos mesmos, fazendo, então, o detalhamento das informações do log, permitindo a escolha de como serão apresentados.

- *Kibana:*

  Kibana é um produto analítico de código aberto para pesquisar, visualizar e analisar dados. Ele atua como a camada superior do *Elasticsearch*, oferecendo uma variedade de opções para visualizar dados na forma de tabelas, gráficos, mapas, histogramas, entre outros. Fácil de configurar e usar, o *Kibana* ajuda você a analisar os dados armazenados no *Elasticsearch* em minutos, sem nenhuma codificação [50].

  Dessa forma, todos os dados que foram armazenados e processados pelos dois mecanismos anteriores serão apresentados no *Kibana*. Analisar os *logs* seria mais trabalhoso, pois seria

necessário a visualização das informações em várias linhas de um mesmo *log*, já no *Kibana* o resultado dos processamentos podem ser apresentado em gráficos e histogramas.

- *Beats:*

  Beat é um produto *open source* leve e desenvolvido para adquirir dados que alimentam, em seguida, o *Elasticsearch*. Ele envia os dados operacionais para o *Elasticsearch*, diretamente ou via *Logstash*, para que possam ser visualizados com o*Kibana*. Uma das características do *Beats* é realizar pouco processamento e manipulação de dados, fazendo com que isso o torne extremamente leve nos recursos do sistema. Significa que eles podem ser instalados onde forem necessários para que, junto com as métricas, possam ser enviados ao *Elasticsearch* em vez de ficarem presos nos *logs* da máquina [51].

2.8 Trabalhos relacionados

O propósito deste trabalho é identificar por meio do Machine Learning textos na mídia social Twitter que denigrem a imagem da mulher, portanto, foi necessário pesquisar trabalhos que tinham como objetivo identificar fragmentos de textos, utilizando esse tipo de método. Por meio deles observou-se o tipo de modelagem e os mecanismos que eles utilizaram para passar dos modelos até a implementação. Nosso foco nesta seção é explorar a metodologia intencional e identificar aspectos úteis das experiências aqui relatadas para o trabalho.

2.8.1 Predição de comentários em mídias sociais sobre discursos racistas

O trabalho [30] investiga como as interações racistas acontecem nas redes sociais, objetivando, assim, desenvolver um modelo para categorizar *tweets*, visando identificar rapidamente publicações que contenham discursos racistas. Para isso, utilizou-se a API do Twitter para a obtenção dos dados e, então, construiu-se um modelo de aprendizado de máquina para identificar comentários racistas.

A obtenção dos dados se deu pela biblioteca *open source* Tweepy que possibilita uma maneira simples e rápida de acesso à API do Twitter. Os *tweets* foram coletados a partir da observação de alguns perfis de pessoas públicas negras e políticos, observando tanto os *post*s dessas pessoas, como a interação dos seus seguidores com esses *posts*. Ao total, foram coletados 1271 *tweets* dentro do contexto de racismo e a rotulagem desses dados foi feita manualmente.

Em relação ao tratamento dos dados, a linguagem de programação Python juntamente com a biblioteca NLTK foram utilizadas para realizar algumas técnicas de tratamento dos dados, como a tokenização e remoção de *stopwords*.

Para o treinamento dos modelos, foi utilizada a biblioteca do *Scikit-Learn*. Essa biblioteca é bem completa e ela foi utilizada para a construção de diversos modelos de aprendizado de máquina, tanto utilizando uma estratégia *Unigram*, quanto *Unigram + Bigrams*. Além da construção dos modelos, a biblioteca também forneceu uma forma de validar o modelo e verificar sua acurácia, usando a validação cruzada *k-fold* com 10 subconjuntos de treinamento.

Foram utilizados três algoritmos de aprendizagem de máquina supervisionada: *Naive Bayes, SVM* e Regressão Logística. Esses três algoritmos, usando a estratégia *Unigram*, apresentaram desempenho satisfatório, todos com uma taxa de acurácia acima de 76%.

2.8.2 Machine Learning: metodologia de mineração automatizada com dados das redes sociais e processamento de linguagem natural

Atualmente, é raro encontrar alguém que não esteja registrado em uma rede social. Muitas pessoas as usam em seus computadores e dispositivos móveis, tornando-as parte de suas vidas. Isso transforma as redes sociais em uma ferramenta de colaboração para troca de informações. A grande quantidade de informações geradas facilmente torna essas mensagens em uma fonte valiosa para descobrir conhecimentos ocultos e secretos que seriam impossíveis de descobrir por métodos tradicionais.

Em [52] foi feito um estudo na rede social Twitter objetivando identificar registros nas redes sociais sobre crimes ocorridos na cidade do Rio de Janeiro. Muitas pessoas usam estas informações para tomar decisões cotidianas, como checar se houve algum incidente na rota de transporte para o trabalho e buscar uma alternativa se necessário.

Para lidar com essa quantidade de informações, uma metodologia de mineração automatizada com PLN (Processamento de Linguagem Natural) foi criada, que se concentra em identificar eventos de segurança. A metodologia consiste em três etapas: Parametrização e Iniciação, Preparação e Mineração e Marcação e Apresentação. Ela foi implementada através de um Bot feito em Python e usando o banco de dados PostgreSQL.

O autor explica e usa cinco algoritmos de aprendizado de máquina, sendo eles:

- Naive Bayes: É um algoritmo de classificação baseado em probabilidade que utiliza o teorema de Bayes para calcular a probabilidade de uma amostra pertencer a uma determinada classe. Ele supõe que as *features* são independentes entre si, o que é uma suposição "ingênua"(daí o nome), mesmo assim é uma abordagem eficaz em muitas situações.

- Máquina de Vetor de Suporte (SVM): É um algoritmo de classificação que procura encontrar a melhor fronteira de separação entre as classes. A fronteira é definida de forma a maximizar a margem entre as classes. Ele é eficaz em problemas de classificação de alta dimensionalidade e é capaz de lidar com dados linearmente inseparáveis usando técnicas como o Kernel Trick.

- Árvore de Decisão: É um algoritmo de aprendizado supervisionado que cria uma árvore de decisão para modelar a relação entre as *features* de entrada e a variável alvo. A árvore é construída de maneira recursiva, onde as perguntas são feitas sobre as *features* até que as amostras sejam classificadas de maneira satisfatória. É uma abordagem intuitiva e fácil de entender, mas pode ter problemas com *overfitting* se a árvore for muito profunda.

- Cadeia de Markov Escondida (HMM): É um modelo de sequência utilizado para prever a próxima etapa de uma sequência baseado nas etapas anteriores. Ele é "escondido"porque o

processo que gera a sequência não é diretamente observável, e a saída é modelada usando probabilidades condicionais. HMMs são amplamente utilizados em tarefas como reconhecimento de fala e análise de séries temporais.

- Agrupamento (Clusterização): É uma técnica de aprendizado não supervisionado que procura encontrar grupos de amostras similares (ou seja, *clusters)* a partir das *features* de entrada. O objetivo é separar as amostras em grupos de forma que as amostras dentro de um grupo sejam mais similares entre si do que com as amostras de outros grupos. Exemplos de algoritmos de clusterização incluem *K-Means, DBSCAN e Hierarchical Clustering.*

2.8.3 Análise de sentimento em redes sociais para a língua portuguesa utilizando algoritmos de classificação

Este trabalho [53] apresenta uma nova abordagem para verificação do humor dos usuários do Twitter em relação a assuntos como política, desastres mundiais, saúde e eventos esportivos, na língua portuguesa. Essa abordagem é baseada em um Comitê composto por vários algoritmos de aprendizado de máquina e foi comparada com abordagens já existentes para a língua portuguesa.

O Comitê foi construído a partir de algoritmos como *Naive Bayes Multinomial*, Máquina de vetor de suporte (SVM), *Random Forest* e Regressão Logística.

Os resultados dos testes iniciais mostraram que a abordagem proposta com o uso do Comitê teve um desempenho superior em termos de taxa de acerto, precisão e menor erro quando comparado com cada algoritmo utilizado isoladamente. Além disso, a abordagem foi comparada com soluções comerciais disponíveis e seus resultados foram superiores a todas as soluções disponíveis no mercado. Destaca-se que o Comitê apresentou uma acurácia acima de 80% no *dataset* utilizado.

3 Metodologia

3.1 Contexto

Um bom conjunto de dados ou *dataset*, em inglês, é essencial para o sucesso de um processo de aprendizado de máquina. A qualidade e a precisão dos resultados dependem da qualidade dos dados. Bons dados devem representar o problema com precisão, ter pontos de informações suficientes para capturar a complexidade do problema, estar livres de erros e discrepâncias.

Sem um bom *dataset*, o modelo de aprendizado de máquina pode ser impreciso ou incapaz de generalizar para dados não vistos. Além disso, um bom conjunto de dados permitirá que o modelo aprenda mais rápido e com melhor precisão. Um bom conjunto de dados também pode permitir que o modelo seja mais interpretável e ajudar a identificar padrões nos dados.

Para o desenvolvimento de qualquer trabalho referente ao aprendizado de máquina, o primeiro passo é adquirir dados. Neste trabalho o acesso aos dados foi feito através da rede social Twitter. Entretanto, como não foi encontrado nenhum *dataset* que contemplasse o tema proposto no trabalho, foi necessário classificar manualmente os dados, o que pode ter acarretado uma categorização errônea dos textos que contenham discursos de ódio contra as mulheres.

Após a obtenção dos dados, foi possível processá-los, analisar possíveis estratégias, aplicá-las e, então, selecionar o modelo com os melhores resultados. A partir daí, é possível haver implementações desse modelo para classificar novos dados.

Todo o trabalho foi realizado como software, sob a licença GPL *(General Public License)* da GNU, todo o código será disponibilizado no GitHub. Assim, será possível promover melhorias no futuro. O software está disponível no GitHub em *juliajamile/detection-of-hate-speech* [54].

3.2 Fluxo da elaboração do Database

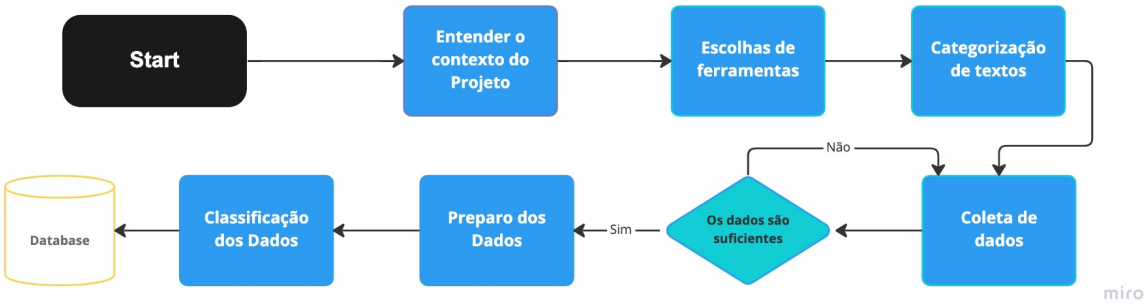

Figura 3.1 – Fluxograma de elaboração do Database.

- Entendimento do contexto: entendimento sobre o projeto a ser criado;

- Escolha de ferramenta: serão selecionadas ferramentas para satisfazer o processo de criação e execução do modelo;
- Categorização de texto: definição de que tipos de dados serão pesquisados na API do Twitter;
- Coleta de dados: esta atividade será para a execução da coleta dos dados;
- Preparo dos dados: a partir do momento que você tem um conjunto de dados de classificação, é necessário prepará-lo com uma estratégia de limpeza de dados para fins analíticos;
- Classificação dos Dados: definição de discurso de ódio nos textos.

3.3 Fluxo da elaboração do modelo

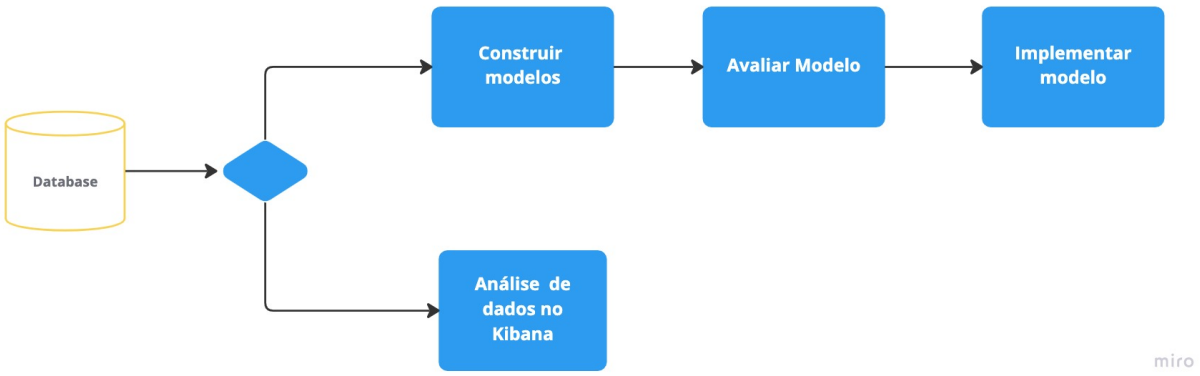

Figura 3.2 – Fluxograma de elaboração do modelo.

- Construir modelos: nesta atividade, vários modelos são construídos de acordo com a estratégia de análise definida;
- Avaliar modelo: nesta atividade, o modelo é analisado para verificar se atende aos objetivos. Se ele não for satisfatório, possíveis razões precisam ser identificadas;
- Implementar Modelo: com o modelo bem-sucedido, pode-se implementá-lo em outros twitters.
- Análise de dados no kibana: com o database bem-criado, a análise de dados pode ser feita usando o Kibana.

3.4 Categorização de texto

Segundo [55], a categorização, também conhecida como classificação de texto, é uma tarefa voltada para a classificação de padrões para mineração de texto, sendo esta uma tarefa necessária para o controle das informações textuais.

Envolve a atribuição de categorias ou rótulos predefinidos a um determinado documento de texto ou a um conjunto de documentos de texto [56]. O objetivo da classificação de texto é classificá-lo automaticamente em uma ou mais categorias predefinidas, com base em seu conteúdo.

Essa categorização é útil porque pode ajudar a dar sentido a grandes quantidades de dados de texto não estruturados, que são cada vez mais prevalentes na era digital atual. Também pode ser usado para melhorar mecanismos de busca, sistemas de recomendação e muitas outras áreas de processamento de linguagem natural e inteligência artificial. A categoria utilizada como parâmetro foi o discurso de ódio/violência contra as mulheres na rede social do Twitter.

A violência contra a mulher é uma questão grave e preocupante em todo o mundo. É importante que a sociedade tome medidas para prevenir e combater esse tipo de violência. Ela pode ser dividida em três categorias:

- Violência psicológica e moral: é um conjunto de comportamentos que buscam diminuir a autoestima, controlar e intimidar a vítima, como ameaças, xingamentos, humilhações, isolamento social, entre outros.
- Violência física: é qualquer tipo de agressão física contra uma pessoa como, por exemplo, espancamento, estrangulamento, queimaduras, entre outros.
- Violência sexual: é qualquer tipo de comportamento sexual não desejado ou imposto por outra pessoa, incluindo assédio, estupro, agressão sexual, entre outros.

Sabendo que existe uma vasta diversidade de palavras com teor de discursos de ódio/violência contra a mulher proferidos no Twitter, foi necessário escolher as mais relevantes para o problema proposto no presente trabalho. Também precisou se atentar a quantidade de frases e palavras, pois elas deveriam ser suficientes para possibilitar o treinamento dos modelos.

Sendo assim, para montar a lista de ofensas que foram passadas como parâmetros, foi utilizado como critério o resultado da pesquisa realizada sobre as principais palavras e frases proferidas por pessoas para disseminar o discurso de ódio/violência contra a mulher [57] [58] [59]. A partir dessa bibliografia foi possível criar uma boa lista para integrar o *dataset*.

A lista criada foi classificada entre as três categorias de violência, ou seja, entre violência psicológica e moral, física e sexual. A classificação dos grupos e das palavras de discurso de ódio e violência foi realizada pelas autoras desse trabalho. A separação dos dados está contida nas tabelas 3.1, 3.2, 3.3. Vale ressaltar que alguns termos correspondem a mais de um grupo de violência.

Tabela 3.1 – Formas de violência contra a mulher: Violência Física

Formas de violência contra a mulher: violência Física	
Eu levaria você para casa;	Doméstica;
Mulher que se respeita não é estuprada;	Mulher devia ficar em casa;
Feminicídio.	

Tabela 3.2 – Formas de violência psicológica: violência sexual

Formas de violência psicológica: violência sexual	
Vai chamar muita atenção;	Mulher que se respeita não é estuprada;
Eu levaria você para casa;	Mal comida;
Malcomida;	Vadia;
V4dia;	Puta;
Pu**;	Put*;
Put4;	Cachorra;
Gostosa;	Buceta;
Delicia;	Bucet4;
Prostituta;	Vagaba;
Piranha;	Vagabunda;
Prostiranha;	Peituda;
Cadela.	

Tabela 3.3 – Formas de violência contra a mulher: violência moral e psicológica

Formas de violência contra a mulher: violência moral e psicológica	
Eu levaria você para casa;	Mas é coisa de homem;
Mulher que se respeita não é estuprada;	Isso é muito agressivo para mulheres;
Você come muito para uma menina;	Só queria te elogiar;
Vai chamar muita atenção;	Você deve estar naqueles dias;
Se você se esforçasse ficaria linda;	Malcomida;
Mal comida;	Nega;
Magrela;	Gorda;
Feia;	Malamada;
Mal amada;	Vadia;
V4dia;	Puta;
Pu**;	Put*;
Put4;	Cachorra;
Sarnenta;	Estupida;
Ignorante;	Burra;
Incapaz;	Gordona;
Mulher devia ficar em casa;	Mulher só serve para cuidar de casa;
Chorona;	Prostituta;
Gostosa;	Piranha;
Delicia;	Vaca;
Dona encrenca;	Vagaba;
Encrenca;	Vagabunda;
Encrenqueira;	Prostiranha;
Cabelo ruim;	Peituda;
Cabelo de bombril;	Cadela;
Negona;	Jumenta;
Macaca;	Baranga;
Baleia.	

Ao executar essa etapa de coleta dos dados como discurso de ódio, criou-se um *dataset* inicialmente de 29717 *tweets*.

3.5 Coleta de dados

A coleta de dados é uma parte crítica do processo de aprendizado de máquina, pois fornece os dados de entrada usados para treinar os modelos de aprendizado de máquina. Ela é importante porque permite que o algoritmo de aprendizado de máquina aprenda com os dados e faça previsões ou decisões. Além disso, a obtenção dos dados ajuda a criar um modelo mais robusto, que pode lidar melhor com entradas desconhecidas e prever com precisão. Sem dados, o algoritmo de aprendizado de máquina não teria base para tomar decisões, portanto, a primeira etapa para realizar o treinamento da máquina foi a coleta dos dados, para isso foi necessário ter acesso à API do Twitter.

Para acessar a API do Twitter, precisa-se de uma conta de desenvolvedor do Twitter, portanto, após criar uma conta na rede social é necessário criar um aplicativo no site da conta de desenvolvedor do Twitter [60]. Em seguida, utilizou-se o console do Twitter para gerar as chaves de autenticação necessárias para acessar a API.

Com as credenciais, realizou-se um *script* em Python para realizar a consulta na API. Os dados foram obtidos utilizando a biblioteca *Twython* [61], por ela ser quem fornece, de maneira fácil, o acesso aos dados do Twitter. A coleta dos dados foi realizada entre o período de 20 de novembro a 4 de dezembro de 2022. A seguir, temos uma parte do código realizado para obter os dados.

```
    #Criando o dicionario, onde vai ser hospedado a API
dict_ = {'id':[], 'user':[], 'date': [], 'text': [], 'favorite_count': [],
'retweet_count':[], 'location':[] }

def buscar_tweets(query_term, max_id, max_iters):
  for call in range(0, max_iters):
    #configurar a query
    query = {'q': query_term,
          'result_type': 'recent',
          'count': 100,
          'lang':'pt',
          'max_id': max_id,
          'tweet_mode': 'extended',
          'include_entities': False
    }
    #adicionando os dados no dicionario
    for status in python_tweets.search(**query)['statuses']:
      if 'RT @'not in (status['full_text']):
        dict_[ 'id'].append(status['id'])
        dict_['user'].append(status['user']['screen_name'])
        dict_['date'].append(status['created_at'])
        dict_['text'].append(status['full_text'])
        dict_['favorite_count'].append(status['favorite_count'])
        dict_['retweet_count'].append(status['retweet_count'])
        dict_['location'].append(status['user']['location'])
        max_id = status['id']
```

Trecho de código 3.1 – Script para busca de tweets

Este código acima é um *script* Python que busca tweets com base em uma determinada consulta. A busca é realizada por meio da API do Twitter usando a biblioteca *Twython*. O *script* cria um dicionário para armazenar os dados dos tweets e, então, executa um *loop* para buscar os tweets de acordo com os parâmetros especificados, como o termo de consulta, o número máximo de interações e a identificação máxima (max_id) Os dados dos tweets são adicionados ao dicionário.

Os parâmetros definidos para a consulta da API foram escolhidos de acordo com a documentação dessa aplicação, a qual retorna uma coleção de Tweets relevantes que correspondem à pesquisa específica dos referidos parâmetros [62]. Vale ressaltar que o número de Tweets a serem retornados por consulta é de no máximo 100, segundo a documentação da API, portanto, foi necessário utilizar um *for* para realizar várias requisições, a fim de obter o dataset completo.

Com o objetivo de extrair tweets de palavras ofensivas e organizá-los em um *dataframe* para formar um dataset, criou-se a função $palavras\_para\_dataset()$ que recebe uma lista de ofensas como parâmetro que, em seguida, executa um *loop* para buscar tweets correspondentes a essas ofensas. Depois de buscar e coletar os tweets, os dados são organizados usando a biblioteca pandas em um dataframe que, após, é salvo em um arquivo CSV. O código também inclui um tempo de espera aleatório para evitar que o Twitter detecte atividades suspeitas. A função descrita está a seguir:

```
    def palavras_para_dataset(ofensas):
  print('Serao extraidos tweets de ' + str(len(ofensas)) + ' palavras')

  #rodando array de palavras para colher o dataset
  for ofensas in ofensas:
    print('Iniciando processamento: ' + ofensas)
    buscar_tweets(ofensas, "", 5)
    #estruturar o dataframe com o pandas
    df = pd.DataFrame(dict_)
    pd.set_option('display.max_colwidth', None)
    df.to_csv('/content/drive/MyDrive/TCC Ju e Ge/Data base/Ofensas11_12.csv',
    index=False)

    print('Final de processamento: ' + ofensas)

    randomico = random.randrange(10, 40)
    print('Vamos aguardar '+ str(randomico) + ' segundos')
    time.sleep(randomico)
```

Trecho de código 3.2 – Função para extrair tweets ofensivos

3.6 Preparo dos dados

A preparação de dados é o processo de limpeza, transformação, manipulação e organização dos dados antes de realizar qualquer análise. O objetivo é torná-los mais facilmente compreensíveis e úteis para análises futuras. Alguns dos passos envolvidos na preparação de dados incluem a sua limpeza, a correção de erros, a normalização, a padronização e a transformação desses dados. O

processo também pode incluir a criação de novos conjuntos de dados para fins de análise, a partir de fontes de dados existentes [30] [52].

3.6.1 Stopwords, Lemmatization e tokenização

Stopwords são palavras comuns usadas na língua que não têm significado específico quando usadas sozinhas. São comumente ignoradas por mecanismos de busca, pois não adicionam significado a uma frase. Exemplos de *stopwords* incluem palavras como "o", "em", "de", "um", "você", entre outras. Essas palavras são frequentemente usadas como conectivos de ligação, mas não têm significado quando utilizadas sozinhas. Por isso, elas são removidas quando usadas em mecanismos de busca para melhorar os resultados de modelos de ML [53].

No ML, as *stopwords* são usadas para reduzir a dimensionalidade dos dados e remover palavras irrelevantes que podem afetar a precisão de um modelo de aprendizado de máquina [63]. Essas palavras são identificadas como sendo frequentemente usadas, mas não trazem significado relevante para a análise, então, elas são removidas do conjunto de dados. Isso ajuda a reduzir o ruído nos dados e aumentar a precisão do modelo.

Lemmatização é uma forma de processamento de linguagem natural que envolve a redução de palavras para seu formulário básico. A lemmatização é usada para simplificar os dados de entrada para melhorar a precisão da classificação, reduzir a dimensionalidade e melhorar a eficiência dos modelos de Machine Learning. A lemmatização permite que as palavras sejam mapeadas para um formulário básico, normalizando palavras diferentes para a mesma raiz. Isso ajuda a reduzir a dimensionalidade de um conjunto de dados e a manter a precisão de classificação. Além disso, a lemmatização reduz a necessidade de alimentar o modelo com uma grande quantidade de palavras diferentes [64].

Lemmatization é uma técnica de processamento de linguagem natural que visa agrupar as formas flexionadas de uma palavra para uma única forma básica. O objetivo é reduzir o número de palavras diferentes em um texto para facilitar a análise léxica. Por exemplo, "correr", "correndo", "correu"e "corre"são todas formas de "correr"que seriam lematizadas para "correr"[64].

Tokenização é o processo de dividir uma cadeia de texto em *tokens*, ou seja, em sequências de palavras ou símbolos. É um algoritmo computacional utilizado em Inteligência Artificial para reconhecer padrões e identificar os elementos que compõem uma sentença, por exemplo [64].

A *tokenização* pode ser utilizada para muitas aplicações, como análise de sentimentos, extração de informação, geração automática de conteúdo, mecanismos de busca, etc. É comum ver essa técnica sendo usada para dividir palavras de um texto em palavras-chave para facilitar a análise de texto e a busca por informações por meio de um motor de busca.

A tokenização é fundamental para aplicações de *Machine Learning* e linguagem natural. Ela permite que os computadores entendam e processem os textos, classificando-os e interpretando-os para identificar padrões. Sem tokenização, os computadores não seriam capazes de processar textos corretamente.

A tokenização ainda é usada para aplicações de *Machine Learning* e linguagem natural para

identificar as palavras-chave mais importantes e relevantes em um texto. Esse processo ajuda a criar modelos de *Machine Learning* mais precisos e eficientes, que são capazes de reconhecer o significado de certas palavras e frases.

```
def remove_Stop_Words(texto):
  #stopwords em portuguese
  lista_stop = nltk.corpus.stopwords.words('portuguese')

  pontuacao = string.punctuation
   #biblioteca em portugues
  pln_texto = spacy.load('pt_core_news_sm')
  tokens = pln_texto(texto)
  lista_palavras = []

  for token in tokens:
    lista_palavras.append(token.lemma_)

  #tirando pontuacao, lematizacao e stopwords
  lista_palavras = [token for token in lista_palavras if token not in pontuacao
    and token not in lista_stop]

  lista_palavras = " ".join(lista_palavras)

  return lista_palavras
```

Trecho de código 3.3 – Função stopwords

A biblioteca NLTK [65] é um algoritmo em Python para processamento de linguagem natural. Ela fornece ferramentas para acesso, manipulação e modelagem de dados de linguagem natural. A biblioteca NLTK inclui vários componentes, modelos de linguagem e ferramentas, como recursos de *tokenização, tagger*, reconhecedor, *stemmers*, etc. A biblioteca também fornece suporte para análise de sentimentos, análise de texto e outras tarefas relacionadas ao processamento de linguagem natural [66].

A biblioteca de pontuação [67] do Python fornece acesso a um conjunto de caracteres de pontuação. Estes caracteres são usados para realçar a formatação de strings, os quais podem ser usados em qualquer código Python para realçar a leitura e a compreensão do código. Estes caracteres incluem vírgulas, ponto e vírgula, dois pontos, ponto de interrogação, ponto de exclamação, aspas, parênteses e vários sinais de pontuação [68].

A biblioteca *Spacy* carregada é a *pt_core_news_sm*, que contém modelos de processamento de língua portuguesa para *tokenização, part-of-speech tagging, lemmatização* e outras tarefas. O texto fornecido é passado para a biblioteca *Spacy* e cada token é extraído. O lema de cada token é adicionado à lista de palavras. Depois que todos os tokens são processados, a lista de palavras é retornada.

A biblioteca *lemma* é uma biblioteca Python para *lemmatização* de palavras. Ela possui um conjunto de regras para identificar e classificar as palavras corretamente no contexto apropriado. É usado para ajudar a reduzir o tamanho dos dados ao trabalhar com texto. Ele pode ser utilizado

para remover as formas flexionadas das palavras dentro de um texto, para que possam ser tratadas como palavras únicas, reduzindo a complexidade de trabalhar com um texto.

O código 3.3 tem como objetivo remover as *stopwords* de um texto, para isso é utilizada a biblioteca NLTK. O código começa lendo as *stopwords* da língua portuguesa, que consistem em palavras comuns que não adicionam nenhum significado à frase, como "a", "e", "o", "e", "de"e outras.

Após isso, é criada uma lista de pontuação para ser removida da frase. Em seguida, é carregado o pacote de língua portuguesa para o Spacy e o texto é convertido em tokens. Então, é adicionado cada token a uma lista de palavras.

O código 3.3 acima faz *lemmatização* e tokenização de um texto dado. A *lemmatização* é o processo de transformar as palavras para a sua forma base (ou seja, o seu lema). Por exemplo, as palavras "caminhando"e "caminhou"podem ser transformadas para "caminhar"que é a forma base das duas palavras. Tokenização é o processo de dividir um texto em pequenas partes (tokens). Para salvar o resultado no database foi tirada a tokenização para, posteriormente, aplicá-la melhor no momento de treinar o ML.

3.6.2 Remoção de caracteres indesejado

Remoção de caracteres em aprendizado de máquina é o processo de remover caracteres indesejáveis de um conjunto de texto. Por exemplo, você pode usar a remoção de caracteres para remover pontuação, espaços em branco e caracteres especiais de um conjunto de dados de texto, antes de treinar um modelo de aprendizado de máquina. Isso pode ajudar a reduzir ruído e melhorar as taxas de sucesso da previsão [69].

A remoção de caracteres indesejados pode ser feita usando funções de manipulação de strings, que podem ser usadas para substituir caracteres indesejados por espaços em branco ou outros caracteres, ou para deletar completamente um certo caractere. Por exemplo, se for necessário remover todos os caracteres de espaço em branco de uma *string*, é possível usar a função *trim()* para executar essa tarefa. Outras funções de *string*, como a função de substituição de *substring()* ou a função de separação de *string split()*, também podem ser usadas para remover caracteres indesejados.

No caso, foi usada a função re.sub em Python é uma ferramenta útil para remover caracteres indesejados do texto. Utilizando-a, é possível remover caracteres específicos, como letras, números, símbolos, etc., de um conjunto de texto. Isso pode ajudar a torná-lo mais limpo e fácil de trabalhar.

```
1 def unwanted_characters(texto):
2   #colocar todo o texto em minusculo
3   texto = texto.lower()
4 
5   #tirando a citacoes de perfils
6   texto = re.sub(r'@\S+', '', texto)
7 
8   #tirando o link inclusos o tweets
9   texto = re.sub(r'http\S+', '', texto)
```

```
10
11  #tirando espaco adicional
12  texto = re.sub(r" +", " ", texto)
13
14  return texto
```

Trecho de código 3.4 – Função de remoção de caracteres indesejado

O código 3.4 é uma função chamada *"unwanted_ characters"* que remove caracteres indesejados de um conjunto de texto. A função começa por converter todos os caracteres do conjunto de texto para minúsculas. Em seguida, ela usa expressões regulares para remover perfis mencionados (@) e links (http) do conjunto de texto. Por último, o código remove qualquer espaço em excesso do texto. A função retorna o texto sem as partes mencionadas acima.

3.6.3 Stemming

Stemming é um processo usado para reduzir as formas flexionadas de qualquer palavra para a forma raiz. O processo de *Stemming* é amplamente utilizado na análise de texto, como no processamento de linguagem natural, que usa técnicas automatizadas para trabalhar com a linguagem humana. Por exemplo, a palavra 'caminhava' pode ser reduzida à forma raiz 'caminhar', o que pode ser útil se o desejo for processar o texto para encontrar todas as ocorrências de palavras que significam a mesma coisa [70].

O *Stemming* [70] é amplamente usado no *Machine Learning* para simplificar textos complexos. Por exemplo, ao classificar documentos, um algoritmo de aprendizagem de máquina pode não ser capaz de identificar todas as formas de uma mesma palavra (por exemplo, 'caminhando', 'caminhou', 'caminhava' etc.). O uso de técnicas de *Stemming* pode ajudar a reduzir essas palavras à sua forma raiz (neste caso, 'caminhar'), aumentando a precisão das classificações de documentos. Além disso, o *Stemming* pode ser usado para reduzir o tamanho dos documentos classificados, tornando-os mais fáceis de ser processados.

A técnica de *Stemming* não foi utilizada, pois, na língua portuguesa, o sufixo das palavras pode ser alterado como forma de ofensa à mulher [30].

3.6.4 Remoção de linhas duplicadas na base de dados

Uma das etapas na preparação de dados é remover linhas duplicadas na base de dados. Isso é necessário para evitar que os resultados de análise sejam influenciados por informações redundantes. A remoção de linhas duplicadas pode ser feita usando uma variedade de ferramentas, dependendo do banco de dados ou linguagem de programação em uso. Estas ferramentas permitem que os usuários especifiquem quais colunas devem ser verificadas para detectar linhas duplicadas e quais linhas devem ser mantidas. Em alguns casos, é possível configurar a ferramenta para ignorar determinados campos durante a verificação, o que reduz a possibilidade de exclusão de dados importantes [30].

```
1    # verificando duplicatas de text nos tweets
2    df = pd.read_csv('/content/drive/MyDrive/TCC Ju e Ge/Data base/
    Translated_collected_twitter')
```

```
    df = df.drop_duplicates('text')
    df.count()
    df = df.reset_index()
```

Trecho de código 3.5 – Método de tirar texto repetidos do banco

O código 3.5 tem por objetivo remover linhas duplicadas de um conjunto de dados de uma planilha CSV. O código lê o arquivo CSV, remove linhas duplicadas com base em uma coluna específica ("texto"), redefine os índices e remove a coluna de índices antigos. Em seguida, o código salva o arquivo com as linhas duplicadas removidas. Importante ressaltar que esse código foi executado no final do tratamento do banco de dados, logo, o tamanho do *dataset* foi reduzido.

Para chamar todas as funções descritas acima, foi criado uma função chamada de *"data_ preparation"*, podendo ser visualizada no código 3.6.

```
def data_preparation(texto):
  texto = unwanted_characters(texto)
  texto = remove_Stop_Words(texto)

  return texto
```

Trecho de código 3.6 – Método de tirar texto repetidos do banco

3.7 Classificação dos dados

Após a realização da preparação dos dados, o dataset reduziu 1131 dados, portanto, o conjunto de dados coletado possou a possuir 28586 tweets. Com a quantidade de dados definida, pode-se começar o processo de classificação destes.

A classificação é uma tarefa de aprendizado de máquina que visa atribuir uma classe ou categoria a cada exemplo de entrada. É uma forma de aprendizado supervisionado, o que significa que o algoritmo de aprendizado de máquina é treinado com dados rotulados previamente, ou seja, exemplos de entrada já classificados em categorias conhecidas. O objetivo é que o modelo aprenda a identificar as características associadas a cada categoria, de modo que possa classificar corretamente novos exemplos de entrada.

A rotulagem dos dados foi realizada de forma manual pelas autoras desse trabalho e o critério utilizado para classificar os tweets foi baseado na bagagem obtida através das experiências vividas por elas e pela vivência na rede social Twitter.

3.8 Elastic Stack

O Elastic Stack é um grupo de produtos open source da Elastic, criado para ajudar os usuários a obter dados e pesquisar, analisar e visualizá-los em tempo real. O Elastic Stack é composto por 4 produtos, Elasticsearch, Kibana, Beats e Logstash, porém para a finalidade desse projeto serão utilizados apenas os produtos Elasticsearch e Kibana.

3.8.1 Instalação

Para a instalação foi utilizado o sistema operacional Windows 11 e o primeiro passo foi verificar se o Java 8 estava instalado na máquina, pois ele é um requisito básico para instalar as soluções da Elastic. Feito isso, iniciou-se a instalação do Elastic Stack.

A versão utilizada para instalar o Elasticsearch e o Kibana foi a mais recente até a data da realização deste projeto, portanto, escolheu-se a versão 8.6.1. Um dos motivos de ter escolhido essa versão foi porque a versão 8 da Stack inclui, por padrão, algumas funções importantes de segurança como, por exemplo, autenticação de usuário, comunicação criptografada com a API Elasticsearch em HTTPS, autorização de usuário com Role-Based Access Control (RBAC), entre outros.

Para realizar o download do binário gerado pelo Elastic Stack, foi acessado o site oficial da Elastic [71], selecionou-se a plataforma Windows e realizou-se o download do binário do Elasticsearch e do Kibana. Após o download foi necessário extrair os arquivos.

3.8.1.1 Instalação do Elasticsearch

- Após extrair os arquivos, navegou-se até o diretório extraído do Elasticsearch e executou-se o comando: $.\backslash bin\backslash elasticsearch.bat$.
- No final do processo de instalação surgiu uma mensagem com os recursos de segurança contendo uma senha gerada para o super usuário interno Elastic, um token para integração e configuração do Kibana, um certificado e chaves para TLS.
- Copiou-se a mensagem, a qual foi salva em um bloco de notas. Essa etapa é muito importante, pois, as informações que estão na mensagem são utilizadas para configurar o Kibana, que possui as credenciais de acesso ao Stack.

3.8.1.2 Instalação do Kibana

- Semelhante aos passos do Elasticsearch, após extrair o arquivo, navegou-se até o diretório extraído do Kibana e executou-se o comando: $.\backslash bin\backslash kibana.bat$.
- No final do processo de instalação foi gerado um link no output dos logs. Ao clicar nele, fomos direcionados ao browser para finalizar o processo de instalação do Kibana.
- A página que corresponde ao link gerado no output dos logs é a tela do Kibana para a configuração do token. Para realizar essa etapa é necessário abrir o bloco de notas que foi salvo durante a configuração do Elasticsearch (seção 3.5.1.1), copiar o token do Kibana e colar no espaço correspondente no browser. Essa etapa está representada na Figura 3.3.

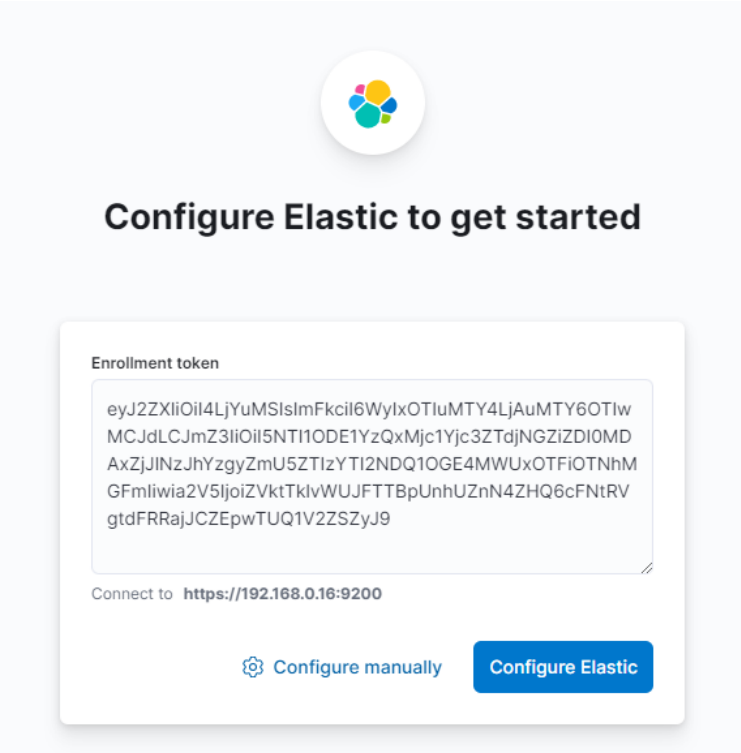

Figura 3.3 – Configuração do token no Kibana.

O token gerado tem duração de 30 minutos e, caso o tempo seja excedido, é necessário gerar um novo.

- Ao colocar o token, o Kibana faz automaticamente a integração com o Elasticsearch.
- Após a conclusão do processo de configuração, redirecionou-se à tela de login do Kibana, onde serão solicitados o usuário e a senha para acesso. O usuário e a senha para acessar o Kibana é o mesmo que foi gerado pelo Elasticsearch que também está salvo no bloco de notas.
- Após todo o processo ser concluído, acessou-se os recursos do Stack. A tela inicial do Kibana pode ser vista na Figura 3.4.

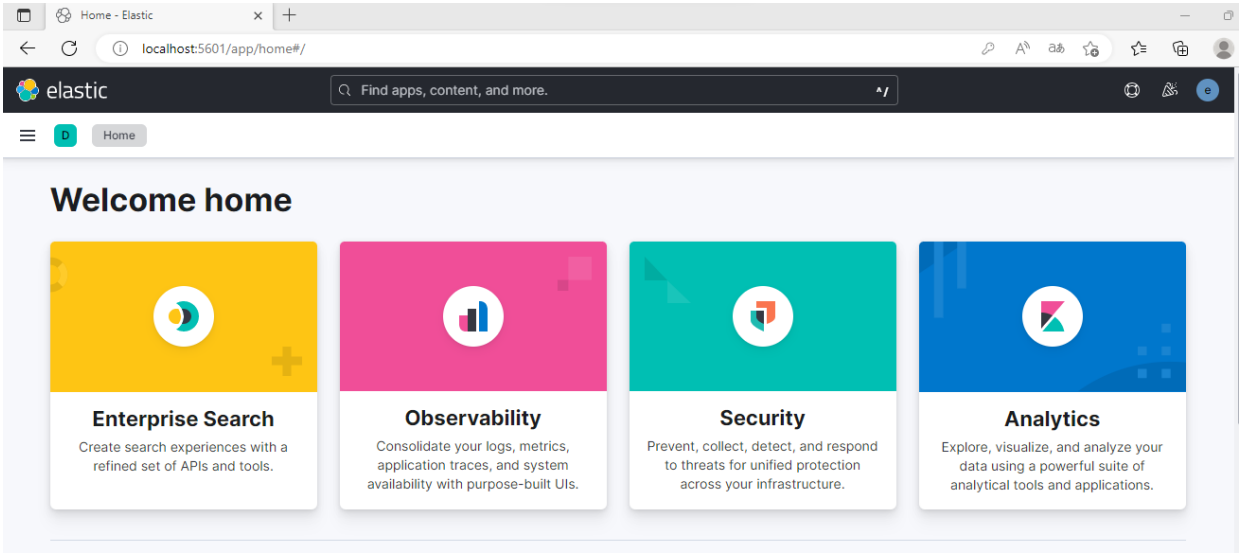

Figura 3.4 – Tela inicial do Kibana.

3.8.2 Implementação dos dados

Como falado anteriormente, o Elastic Stack é uma coleção de ferramentas de código aberto desenvolvidas pela Elastic para análise de dados em larga escala. A implementação de dados no Elastic Stack é o processo de enviar, armazenar, visualizar e analisar dados, portanto, será utilizada tal ferramenta para visualizar alguns dados do dataset criado.

3.8.2.1 Carregar o arquivo de dados

Por meio do Kibana, é possível carregar arquivos de log ou arquivos CSV, TSV ou JSON delimitados. Por padrão, o Kibana carrega arquivos de até 100 MB, mas esse valor pode ser configurado para até 1 GB. Como o arquivo do dataset está dentro dos critérios citados, optou-se por realizar o upload do arquivo pela própria ferramenta.

O arquivo do dataset é no formato CSV, o qual possui aproximadamente 3.200 KB e, para importá-lo, usou-se o *File Data Visualizer*. Ele pode ser encontrado no Kibana em *Machine Learning* > *Data Visualizer*. O usuário visualiza uma página que permite selecionar, arrastar e soltar arquivos, como mostrado na Figura 3.5.

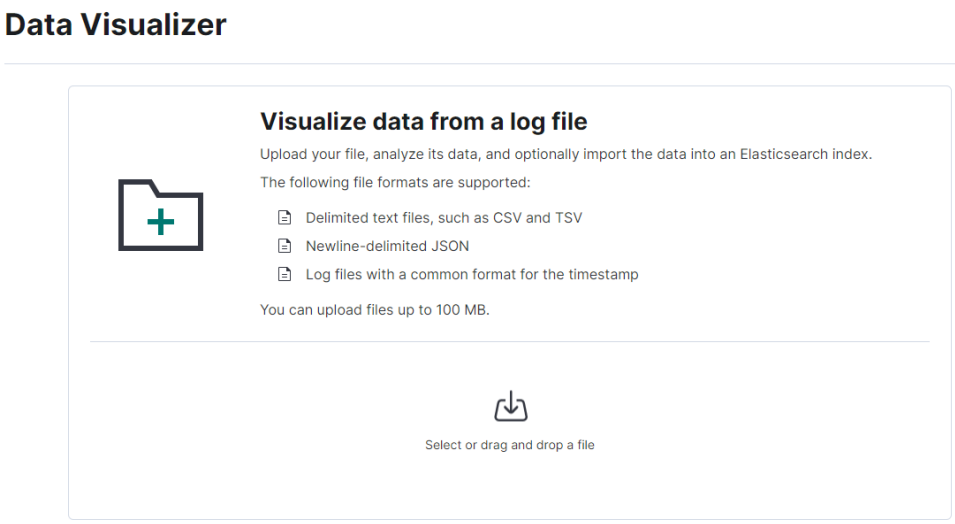

Figura 3.5 – Visualização do campo de importação dos arquivos.

Quando se seleciona um arquivo CSV, a página envia as primeiras 1.000 linhas do arquivo para o endpoint *find_file_structure* para que ele localize a estrutura do arquivo de texto, faça a análise e retorne os resultados.

Na seção *Summary*, mostrado na Figura 3.6, pode-se ver que foi detectado corretamente que os dados estão delimitados e que o delimitador é uma vírgula. Além disso, o recurso mostra que o arquivo CSV importado possui cabeçalho.

No campo *File Stats*, Figura 3.6, são mostrados os nomes detectados no cabeçalho e qual é o tipo. O primeiro nome fica no formato de data, a qual corresponde à coluna *"date"* do CSV. Já o segundo e terceiro estão no formato de número, que representam as colunas *"index"* e *"its_hate_speech"*, respectivamente e, por fim, o *text_clean*, correspondente à coluna que possui os tweets dispostos em formato de texto. Observa-se que o recurso detectou corretamente todos os campos.

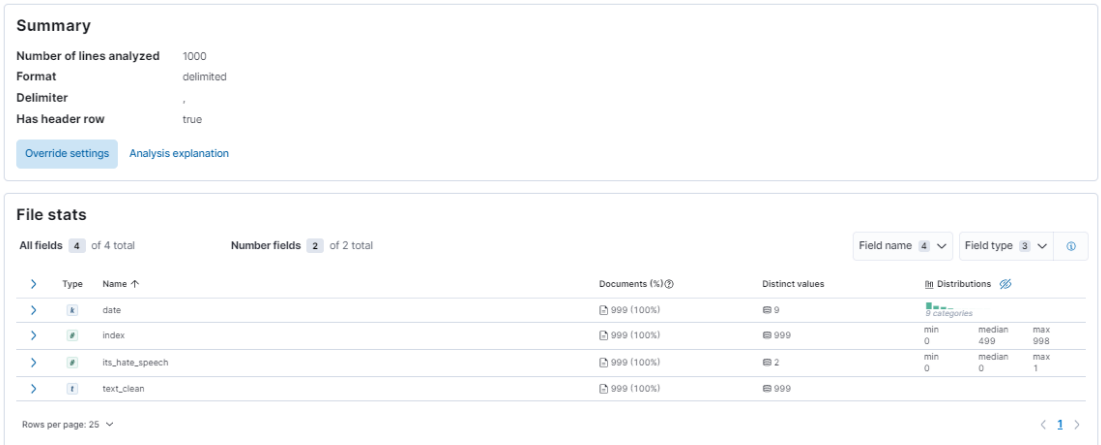

Figura 3.6 – Especificação do arquivo CSV importado.

3.8.2.2 Importar os dados CSV para o Elasticsearch

É necessário importar os dados para o Elasticsearch, visando realizar o armazenamento, pesquisas e análises do dataset. Para tanto, foi criado um índice com o nome *pfg_ge_ju* e clicou-se em *"import"*.

Após clicar no botão *"import"*, o processo de importação é iniciado. Ele consiste em 5 etapas: [72]

- 1. *File processed*: converte os dados em um documento NDJSON para que possa ser obtido usando a API de lote.
- 2. *Index created*: Cria o índice usando as configurações e os objetos de mapeamento.
- 3. *Ingest pipeline created*: Cria o pipeline de ingestão com o objeto ingest pipeline.
- 4. *Data uploaded*: Carrega os dados no índice do Elasticsearch.
- 5. *Data view created*: Essa opção acontece apenas se o usuário tiver optado por isso. Ela permite uma visualização de dados.

Após a conclusão da importação, o usuário verá um resumo com o nome do índice criado, o padrão do índice, um pipeline de ingestão e o número de documentos inseridos. Também haverá uma série de links do Kibana para explorar os novos dados importados. A Figura 3.7 mostra esse resumo.

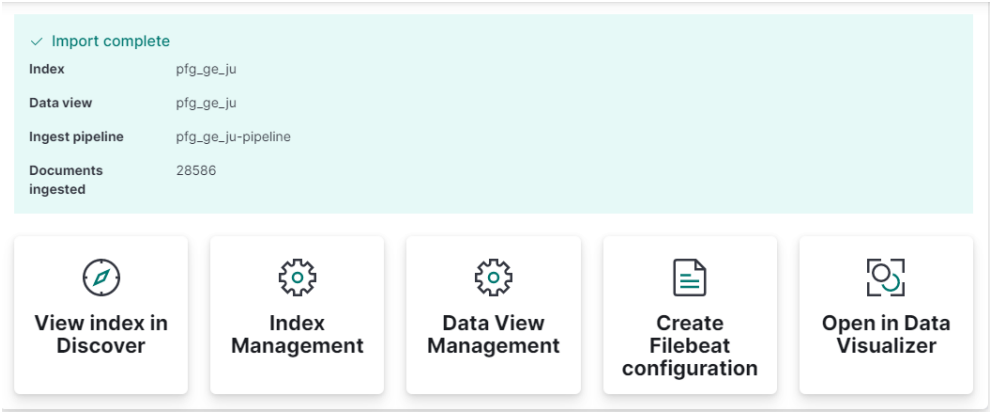

Figura 3.7 – Visualização do resumo da importação.

Observa-se no campo *"Documents Ingested"* que o número de dados importados é equivalente à quantidade de dados do arquivo CSV gerado após a preparação dos dados descritos na seção 4.3 do presente trabalho, portanto, percebe-se que a importação dos dados do *dataset* para o *Elasticsearch* foi realizada com sucesso.

3.8.3 Explorando os dados

Para explorar os dados, usou-se a ferramenta *Discover* do Kibana, pois ele permite visualizar, filtrar e explorar os dados armazenados no *Elasticsearch*. Para acessar a funcionalidade *Discover*, clicou-se na opção *"Discover"* no menu principal do Kibana.

Para explorar os dados é necessário escolher o índice que se deseja explorar, dessa maneira, os dados armazenados nesse índice serão exibidos em uma tabela, onde será possível visualizar as informações em cada registro.

O índice escolhido para realizar a análise foi o criado na seção 3.5.2.2 (índice *pfg_ge_ju*). Ao selecioná-lo, o Kibana retornou uma lista de resultados que correspondem aos parâmetros de consulta. É possível clicar em cada resultado para expandi-lo e visualizar todos os dados desse arquivo.

A Figura 3.8 apresenta a tela da funcionalidade *Discover* carregada com os dados indexados.

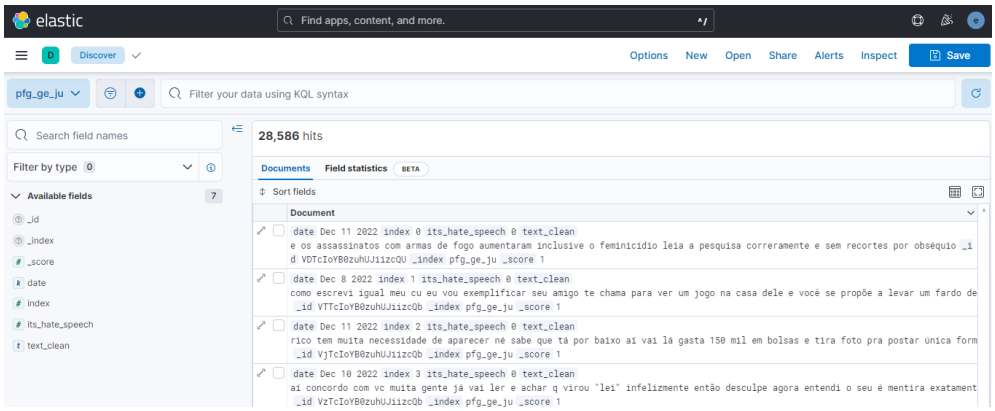

Figura 3.8 – Exploração de dados indexados.

3.9 Balanceamento do dataframe

Balancear o banco de dados antes do treinamento de *Machine Learning* é importante para garantir que o algoritmo de *Machine Learning* seja treinado de forma justa e que os resultados sejam precisos e confiáveis. Se isso não for feito, o algoritmo pode ter um viés, ou seja, ter resultados parciais, o que pode levar a decisões erradas. Balancear o banco de dados significa garantir que as classes diferentes tenham a mesma proporção de instâncias para que todas as classes sejam representadas de forma equilibrada. Para fazer o modelo supervisionado será preciso tratar os dados como *Undersampling* e *Oversampling*.

3.9.0.1 Undersampling

O código 3.7 faz parte de um processo de undersampling, que é uma técnica que consiste no balanceamento de dados no qual a classe majoritária (classe com mais exemplos) é reduzida ao tamanho da classe minoritária (classe com menos exemplos).

```
1 #undersampling - ele diminui a quantidade de exemplos da classe majoritaria
2
```

```
from sklearn.utils import resample

minoria = df.loc[df['its_hate_speech'] == 1]
maioria = df.loc[df['its_hate_speech'] == 0]

#criar outro dataset reorganizados

maior_menor = resample(maioria, replace=True, n_samples=len(minoria),
    random_state=123)

banco_undersampling = pd.concat([minoria, maior_menor])

banco_undersampling.to_csv('Data base/Dataframe_undersampling.csv')
```

Trecho de código 3.7 – Código para criar um banco de dados Oversampling

O código começa separando os dados em duas classes: a classe minoritária e a classe majoritária. Em seguida, usa-se a função *resample* do *scikit-learn* para reamostrar os dados da classe majoritária de forma aleatória. O argumento *replace* define que os dados são reamostrados com reposição, o argumento *n_samples* define o tamanho da nova amostra e o argumento *random_state* define o estado inicial da geração de números aleatórios.

Por fim, os dados da classe minoritária e da classe reamostrada são concatenados em um novo *dataframe*.

3.9.0.2 Oversampling

O código 3.8 faz parte de um processo de *Oversampling*, que é uma técnica de balanceamento de dados, no qual a classe minoritária (classe com menos exemplos) é expandida aleatoriamente ao tamanho da classe majoritária (classe com mais exemplos).

```
#Oversampling - basicamente aumenta o numero de exemplos da classe minoritaria

minoria = df.loc[df['its_hate_speech'] == 1]
maioria = df.loc[df['its_hate_speech'] == 0]

#criar outro dataset reorganizados

menor_maior = resample(minoria, replace=True, n_samples=len(maioria),
    random_state=123)

banco_oversampling = pd.concat([menor_maior, maioria])

banco_oversampling.to_csv('Data base/Dataframe_oversampling.csv')
```

Trecho de código 3.8 – Código para criar um banco de dados Oversampling

O código acima tem como objetivo aplicar a técnica de *oversampling* no *dataset*. Primeiro, foi separada a classe minoritária da classe maioritária para equalizar os tamanhos das classes. Em seguida, a função *resample* foi usada para aumentar o tamanho da classe minoritária com o

número de amostras da classe maioritária. Por fim, foi criado um novo *dataframe* com os dados reorganizados, o qual foi salvo em um arquivo CSV.

3.10 Treinamento do Machine Learning

Com base nos trabalhos [52] [53] [66] [64], o modelo escolhido para essa implementação é o modelo supervisionado para linguagem natural (NLP), que é a técnica de processamento de linguagem natural que utiliza modelos preditivos para descobrir padrões em dados textuais e usar esses padrões para prever resultados. Esta técnica é útil para encontrar informações úteis em dados de texto, como sentimento, tópicos, relações, intenções e muito mais. Ela é usada principalmente para análise de sentimento e classificação de texto.

A biblioteca *Scikit-learn* [73] é uma das principais bibliotecas de *Machine Learning* de código aberto disponíveis. Ela fornece ferramentas eficazes para executar tarefas de aprendizado de máquina como classificação, regressão, agrupamento e previsão. Além disso, ela é muito fácil de usar, tem um código limpo e documentação clara. A biblioteca também é muito versátil, pois pode ser usada com outras bibliotecas de *Machine Learning*, como o *TensorFlow* e o *Keras*.

No trabalho foi escolhido a biblioteca *sklearn.feature_ extraction*, onde consta vários algoritmos para *Machine Learning*. Esta biblioteca fornece ferramentas para a extração de características de dados de imagens, textos, sinais e outras fontes de dados. Possui diversas funções úteis para a preparação de dados, como normalização, transformação de dados, geração de modelos, etc. É usada para extrair características a partir de dados brutos, como no processamento de texto. Por exemplo, a biblioteca pode ser usada para converter texto em vetores de características que podem, então, ser usados para classificação de texto e tarefas de aprendizado de máquina relacionadas. Além disso, a biblioteca inclui algoritmos para análise de sentimento, classificação de imagens, redução de dimensionalidade, etc.

Melhores funções da biblioteca *sklearn.feature_ extraction* para linguagem natural:

- *CountVectorizer*: Esta função conta o número de vezes que uma palavra ou termo específico ocorre em um documento. É usado para criar vetores de características a partir de texto bruto.
- *tf-idf Vectorizer*: Esta função usa o conceito de frequência de termos *(tf)* e a medida de importância inversa dos documentos *(idf)* para criar vetores de características a partir de texto bruto.
- *N-gram Vectorizer*: Esta função cria vetores de características a partir de texto bruto usando sequências de palavras de tamanho variável *(n-gramas)*.
- *Word2Vec:* Esta função usa modelos de aprendizado profundo para criar vetores de características a partir de palavras individuais.
- *Doc2Vec:* Esta função usa modelos de aprendizado profundo para criar vetores de características a partir de documentos inteiros.

- *SGDClassifier:* Esse modelo é usado para classificar dados rotulados. Ele usa a técnica de otimização chamada Gradiente Descendente Estocástico para ajustar os parâmetros do modelo.

- *MultinomialNB:* Esse modelo é usado para classificação de documentos. Ele usa o Teorema de Bayes para classificar documentos em diferentes categorias.

Os modelos usados para treinamento na detecção de discurso de ódio nas mulheres serão os modelos *SGDClassifier, MultinomialNB, CountVectorizer* e *TfidfVectorizer*, que mais se encaixam para o que foi proposto neste trabalho, conforme a descrição acima.

Os modelos *SGDClassifier* e *MultinomialNB* são modelos de classificação de documentos. O primeiro é baseado em um algoritmo de aprendizado de máquina supervisionado chamado Gradiente Descendente Estocástico, que é usado para classificação de texto. O segundo é baseado no algoritmo de Naive Bayes para classificação de documentos.

As funções *CountVectorizer* e *TfidfVectorizer* são usadas para fazer a vetorização dos dados, ou seja, para transformar os dados em formato numérico. O primeiro é usado para contar a frequência de cada palavra em um documento. O segundo é usado para calcular o peso de cada palavra em um documento.

Com isso, serão propostos 4 modelos de treinamento para escolher o melhor modelo de acordo com a acurácia obtida.

```
def escolher_pipe(pipe):

  if(pipe == 1):
    modelo_Count_SGDC= Pipeline([
                        ('countVectorizer', CountVectorizer()),
                        ('modelo',  SGDClassifier())
                        ])
    return modelo_Count_SGDC
  elif(pipe == 2 ):
    modelo_Tfid_SGDC = Pipeline([
                        ('tfidfVectorizer', TfidfVectorizer()),
                        ('modelo',  SGDClassifier())
                        ])
    return modelo_Tfid_SGDC
  elif(pipe == 3):
    modelo_Tfid_Mult = Pipeline([
                        ('tfidfVectorizer', TfidfVectorizer()),
                        ('modelo',  MultinomialNB())
                        ])
    return modelo_Tfid_Mult
  elif(pipe == 4):
    modelo_Count_Mult = Pipeline([
                        ('countVectorizer', CountVectorizer()),
                        ('modelo',  MultinomialNB())
                        ])
```

```
26      return modelo_Count_Mult
```

Trecho de código 3.9 – Função para escolher o modelo de aprendizado

Escolher_pipe é uma função que recebe um número inteiro de 1 a 4 que representa um modelo de classificação de texto. A função retorna um objeto do tipo Pipeline, que é uma sequência de etapas de processamento que serão aplicadas a um conjunto de dados para obter um resultado desejado. Os 4 modelos possíveis incluem o *CountVectorizer* com *SGDClassifier*, *TfidfVectorizer* com *SGDClassifier*, *TfidfVectorizer* com *MultinomialNB* e *CountVectorizer* com *MultinomialNB*.

Os dados coletados e utilizados nos procedimentos acima foram distribuídos segundo a seguinte ordem: 25% para treino e 75% para validação cruzada. Com isto, foi possível validar a ferramenta de forma a dar continuidade ao seu desenvolvimento. Podemos ver isso no código 3.10. Ainda, é possível ver nessa parte de código que o modelo será treinado nas colunas *"text"* e *"its_hate_speech"* do dataset.

```
1 x_1 = df['text']
2 y_1 = df['its_hate_speech']
3
4 X_train_1, X_test_1, y_train_1, y_test_1 = train_test_split(x_1, y_1, test_size
      =0.25, random_state=123)
```

Trecho de código 3.10 – Código da separação de dados para teste e para treino

Finalmente será abordada a função *Treinarbanco*, onde é implementado o treinamento do ML. Essa função será implementada depois de todo preparo dos dados.

```
1  def Treinarbanco (Modelo, banco, imagem, imagem_Cross_validation, X_train_1,
       X_test_1, y_train_1, y_test_1, x_1, y_1):
2
3    treino = escolher_pipe(Modelo)
4
5    #treino do modelo
6    treino.fit(X_train_1, y_train_1 )
7    treino_pred = treino.predict(X_test_1)
8
9    #avaliando o modelo
10   print("A Acuracia do modelo", Modelo,   "(", banco, ") : " ,metrics.
       accuracy_score(y_test_1, treino_pred))
11
12   #matriz de confusao
13   print("Matriz de confusao modelo " ,Modelo, ":" )
14   print(pd.crosstab(y_test_1, treino_pred, rownames=['Classe esperada'], colnames
       =['Classe prevista'], margins=True), '' )
15
16   #matriz de confusao grafica
17   matriz_confusao = confusion_matrix(y_target=y_test_1,
18                                      y_predicted=treino_pred,
19                                      binary=False)
20
21   pp_matrix_from_data(y_test_1, treino_pred, imagem)
22
```

```
23  #Avaliando validacao cruzada com cross-validation
24  print("Cross-validation")
25
26  validacao_cruzada = cross_val_predict(treino, x_1, y_1, cv=10)
27
28  print("A Acuracia do modelo", Modelo, banco, "(validacao cruzada): " ,metrics.
      accuracy_score(y_1, validacao_cruzada))
29
30  print("Matriz de confusao modelo " ,Modelo, "(validacao cruzada):" )
31
32  print(pd.crosstab(y_1, validacao_cruzada, rownames=['Classe esperada'],
      colnames=['Classe prevista'], margins=True), '' )
33
34  pp_matrix_from_data(y_1, validacao_cruzada, imagem_Cross_validation )
35
36  return treino
```

Trecho de código 3.11 – Função de treinamento de aprendizado de maquinas

O código 3.11 é usado para treinar um modelo de *Machine Learning* em um banco de dados específico. Os parâmetros passados para a função incluem o nome do modelo a ser utilizado, o banco de dados a ser usado, a imagem a ser usada para treinar o modelo, a imagem de validação cruzada, X_train_1, X_test_1, y_train_1, y_test_1, x_1ey_1.

A função primeiro usa a função *escolher_pipe* para escolher o pipeline a ser usado para treinar o modelo. Em seguida, o modelo é treinado usando o método $fit()$ e as predições são feitas usando o método *predict*(). Por fim, a Acurácia do modelo é impressa usando o módulo de métricas do Python.

Em seguida, realiza uma previsão usando os dados de treino e teste e calcula a acurácia do modelo. Além disso, a função produz uma matriz de confusão para visualizar os resultados e avalia a validação cruzada com *cross-validation* para avaliar o modelo. Por fim, a função retorna o modelo treinado.

4 Resultados e Análises

Este capítulo tem o objetivo de descrever os resultados e as análises obtidos na aplicação da metodologia descrita na seção 3.

4.1 Categorização de tweets

A categorização de texto nas redes sociais consiste no processo de identificar e classificar o conteúdo de texto em categorias específicas, como notícias, opinião, propaganda, entre outras. Ela é frequentemente realizada por meio de aprendizado de máquina e análise de dados, usando algoritmos para analisar e categorizar automaticamente o conteúdo.

4.1.1 Coleta dos dados

A coleta de dados é uma etapa importante no processo de Machine Learning, pois a sua qualidade e quantidade determinam o desempenho do modelo de aprendizado de máquina. Com o intuito de realizar essa etapa fundamental, utilizou-se a biblioteca Twython [61] para fazer a comunicação com a API do Twitter.

Como descrito na seção 3.4, uma das etapas do aprendizado de máquina consiste em classificar um determinado texto em uma categoria específica, com base em suas características e conteúdo. Para realizar essa etapa, criou-se uma lista de discurso de ódio a fim de ser usada como parâmetro para ser empregada na coleta de *tweets*. A lista foi feita por meio do resultado de pesquisas sobre as principais palavras e frases com teor machista e de ódio direcionadas a mulheres. Ela está mostrada na Figura 4.1:

```
ofensas = ['Eu levaria você para casa', 'Mulher que se respeita não é estuprada',
           'Vai chamar muita atenção', 'Só queria te elogiar', 'Você come muito para uma menina',
           'Você come muito para uma mulher', 'isso é muito agressivo para mulheres','Mas é coisa de homem',
           'Você deve estar naqueles dias', 'Se você se esforçasse ficaria linda', 'Doméstica', 'malcomida',
           'mal comida', 'nega', 'magrela', 'gorda', 'feia', 'malamada', 'mal amada', 'vadia', 'v4dia', 'puta',
           'pu**', 'put*', 'put4', 'cachorra', 'sarnenta', 'mulher', 'menina', 'garota', 'estupida', 'ignorante',
           'burra', 'incapaz', 'gordona', 'mulher devia ficar em casa', 'mulher só serve para cuidar de casa',
           'lugar de mulher é em casa', 'chorona ', 'gostosa', 'delicia', 'buceta', 'bucet4', 'prostituta',
           'piranha', 'vaca', 'vagaba', 'vagabunda', ' dona encrenca', 'encrenca', 'encrenqueira', 'prostiranha',
           'cabelo ruim', 'cabelo de bombril', 'negona', 'macaca', 'peituda', 'cadela', 'jumenta', 'baranga',
           'baleia', 'feminicidio' ]
```

Figura 4.1 – Palavras utilizadas para categorizar os *tweets* no contexto de discurso de ódio.

A coleta dos dados para compor o *dataset* foi realizada entre o período de 20 de novembro a 4 de dezembro de 2022 e foram obtidos 29717 *tweets*. Por meio da Figura 4.2 pode-se ver os 5 dias que mais tiveram *tweets* coletados.

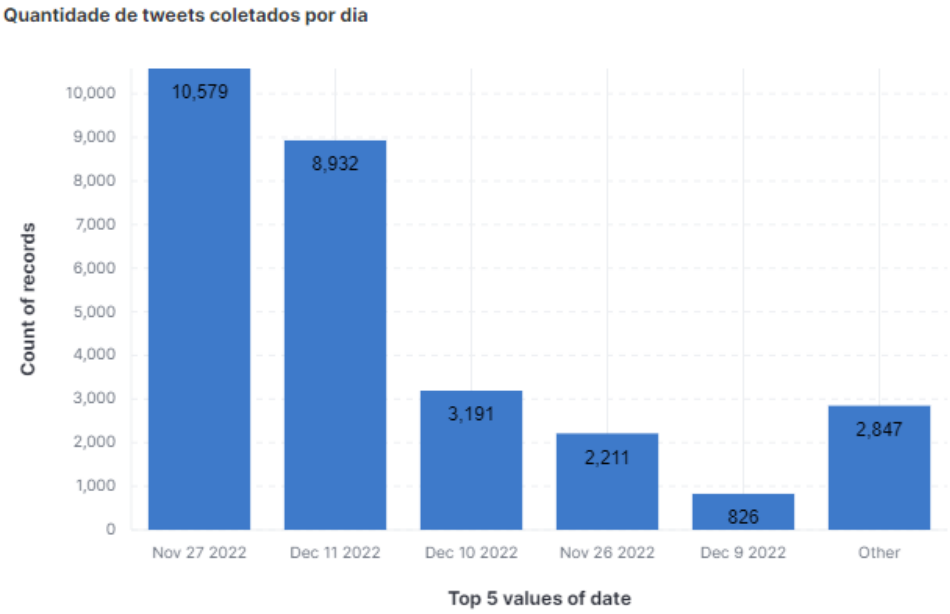

Figura 4.2 – Os 5 dias com maior quantidade de *tweets* coletados.

O gráfico foi realizado por intermédio da ferramenta Kibana. Para gerar esse gráfico, selecionou-se o campo “date” e utilizou-se o filtro com os 5 maiores valores de *tweets* coletados por dia para ser o eixo das abscissas e, para o eixo das ordenadas, a quantidade de dados. Por fim, o gráfico foi salvo e adicionado ao Dashboard.

4.2 Tratamento do banco de dados

4.2.1 Stopwords, Lemmatização e Tokenização

Stopwords são palavras comuns que são filtradas de um documento de texto, pois elas geralmente não têm significado real para o sentido do texto. As *stopwords* mais comuns incluem artigos, pronomes e preposições.

Lemmatização é o processo de classificar uma palavra de acordo com seu significado. Por exemplo, a palavra "melhor"pode ser classificada como "bom". A tokenização é o processo de dividir um texto em tokens (palavras, frases, símbolos, etc.). Por exemplo, um texto "Eu gosto de comer pipoca"pode ser dividido em tokens: "eu", "gosto", "de", "comer", "pipoca".

Tabela 4.1 – Exemplo de tweet após a aplicação de técnicas de stopwords e Lemmatization.

Frase original	Adão Negro pode não ter o melhor roteiro, mas ele entrega o que um filme de herói tem que ser. E com esse filme eu senti a mesma coisa quando vi Homem de Aço, olha que eu não gosto de Superman, mas apesar de ser do Snyder ele sabia como filme de herói tinha que andar.
Frase após stopwords e Lemmatization	Adão Negro poder ter bom roteiro entregar filme herói ter filme sentir coisa ver Homem Aço olhar gostar Superman apesar de o Snyder saber filme herói ter andar

Após a aplicação dos tratamentos de dataset descritos acima, a tabela 4.1 é um exemplo de uma frase do twitter sem modificações e como a frase ficaria depois da aplicação das técnicas de stopwords e Lemmatization. Para a preservação de dados foram alterados as citações e os links, para seguir a LGPD.

4.2.2 Remoção de caracteres indesejado

A remoção de caracteres indesejados no tratamento de banco de dados é uma etapa importante e necessária para garantir que os dados sejam tratados corretamente. Os caracteres indesejados podem incluir caracteres especiais, espaços em branco, caracteres de nova linha e outros caracteres que não sejam necessários para o processamento dos dados.

Tabela 4.2 – Exemplo de tweet após a aplicação de remoção de caracteres indesejado.

Frase original	@choquei Sim amg a mulher era feia msm https://t.co/jX03mA35i541654
Frase após remoção de caracteres indesejado	sim amg a mulher era feia msm

A tabela 4.2 é um exemplo de uma frase do Twitter, não modificada, após a aplicação do processamento do conjunto de dados acima. Para a preservação de dados foram alterados os links e as citações, para seguir a LGPD.

4.2.3 Remoção de linhas duplicadas na base de dados

O tratamento de dados é um conjunto de técnicas usadas para preparar, limpar e transformar conjuntos de dados para fins de análise, como modelagem e avaliação. Estas técnicas podem incluir limpar os dados de entradas incorretas, remover os desnecessários, converter dados entre formatos e normalizá-los. Estas tarefas são necessárias para garantir que os dados sejam consistentes e relevantes para o uso pretendido. Na tabela 4.3, pode-se observar exemplos de como foram retiradas as linhas duplicadas no banco, após a aplicação de todas as etapas.

As frases da tabela 4.3 foram retiradas do banco de dados, mas foram modificadas para proteção de informações das pessoas que postaram a frase no Twitter.

Tabela 4.3 – Exemplos de tweets após a aplicação de técnicas de tratamento dos dados.

Frases original	- @bkdjikjsmx Uma mulher feia como vc criticando os outros! -Incrível como gente feia e desconhecida quando fala besteira na internet e toma na cara sempre fala que tá sendo cancelada https://t.co/lmVH3GVm2z - @choquei Uma mulher feia como vc criticando os outros! - @blabla devia ter usado crack pra achar essa baranga linda https://t.co/J64545ONQ6jy4c - @CBF_Futebol @FIFAWorldCup Vida q segue oq seu bucet4! Isso é copa do mundo vcs entraram igual uma moç4 vai se fud3r bando de mulequ3 vcs só querem saber de dinheiro no bolso - @blabla Nada. Ela gordona mesmo. “Mulheres” de esquerda não querem ter filhos mas querem ter direito no filho das outras mulheres.
Frases após funções aplicadas.	- mulher feio vc criticar outro - incrível gente feio desconhecer falar besteira em o Internet tomar em o cara sempre falar tá cancela - dever ter usar crack pra achar baranga linda - vida q seguir oq bucet4 copa de o mundo vcs entrar igual moç4 ir fud3r bando mulequ3 vcs querer saber dinheiro em o bolso - nada gordona “ mulher ” esquerda querer ter filho querer ter direito em o filho de o outro mulher

4.3 Nuvem de Palavras

A Nuvem de palavras [69] é uma técnica de visualização de dados usada para exibir as palavras mais frequentemente usadas em um documento ou conjunto de documentos. As palavras são apresentadas como etiquetas de tamanhos diferentes, de acordo com a sua frequência de uso, criando assim uma "nuvem"visual que destaca as palavras mais importantes. Esta técnica pode ser usada para avaliar rapidamente o conteúdo de um documento, identificar temas e descobrir padrões.

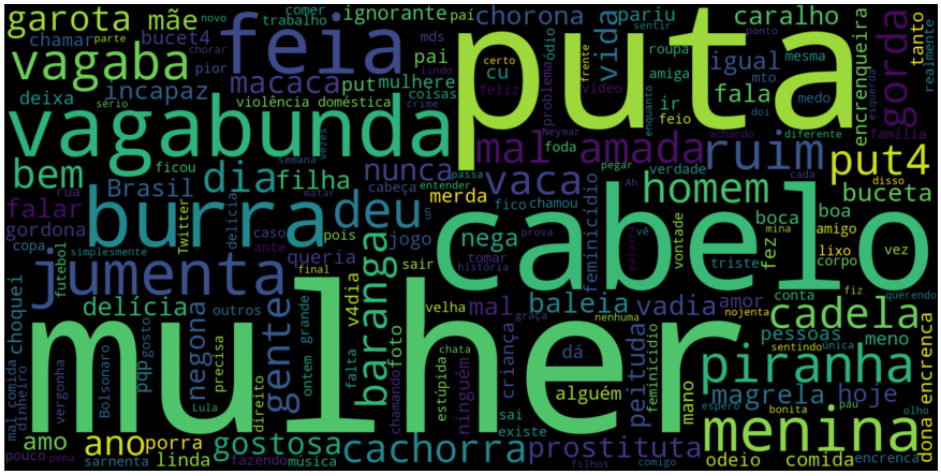

Figura 4.3 – Nuvem de palavras após o tratamento do Dataset.

Após a aplicação de todas as etapas, descritas acima, para o tratamento do *dataset*, foi realizada uma nuvem com as palavras que mais se repetem no banco de dados para identificar temas e descobrir padrões. A nuvem pode ser encontrada na figura 4.3. E para fim de demonstração foi realizada uma nuvem de palavras antes do tratamento do banco, podendo ser visualizada na figura 4.4 .

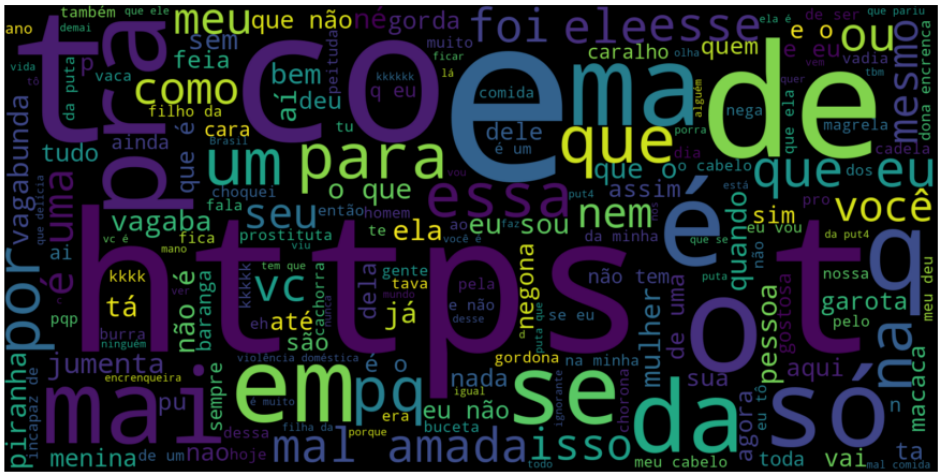

Figura 4.4 – Nuvem de palavras antes do tratamento do Dataset.

Comparando as nuvens de palavras, percebe-se que o processamento do conjunto de dados é importante. A primeira nuvem mostra as palavras que ocorrem com mais frequência após o processamento pelo banco de dados como, por exemplo, as palavras: "mulher", "prostituta", "cabelo", "puta". A segunda nuvem mostra palavras-chave no conjunto de dados antes do processamento de texto, como "https", "e", "de", "para", etc.

A segunda nuvem mostra que o tratamento do dataset foi importante para remover palavras irrelevantes, permitindo que as palavras-chave sejam mais claramente vistas. Além disso, o tratamento do dataset permitiu que palavras relevantes sejam destacadas, permitindo que os usuários entendam melhor o conteúdo do dataset.

4.3.1 Classificação dos dados

A classificação de dados é importante no aprendizado de máquina porque permite ao modelo identificar padrões e tendências nos dados. Isso é crucial para que o modelo possa fazer previsões precisas e tomar decisões sobre novos dados. Além disso, a classificação também ajuda a avaliar e aperfeiçoar o desempenho do modelo, pois permite a comparação entre as previsões do modelo e os resultados reais, portanto, a classificação é fundamental para a eficiência e a eficácia do aprendizado de máquina.

Após o tratamento dos bancos de dados, o *dataset* ficou com 28586 *tweets*, como pode ser verificado na Figura 4.5 onde foi utilizado o parâmetro índice no Kibana para mostrar a quantidade

de dados.

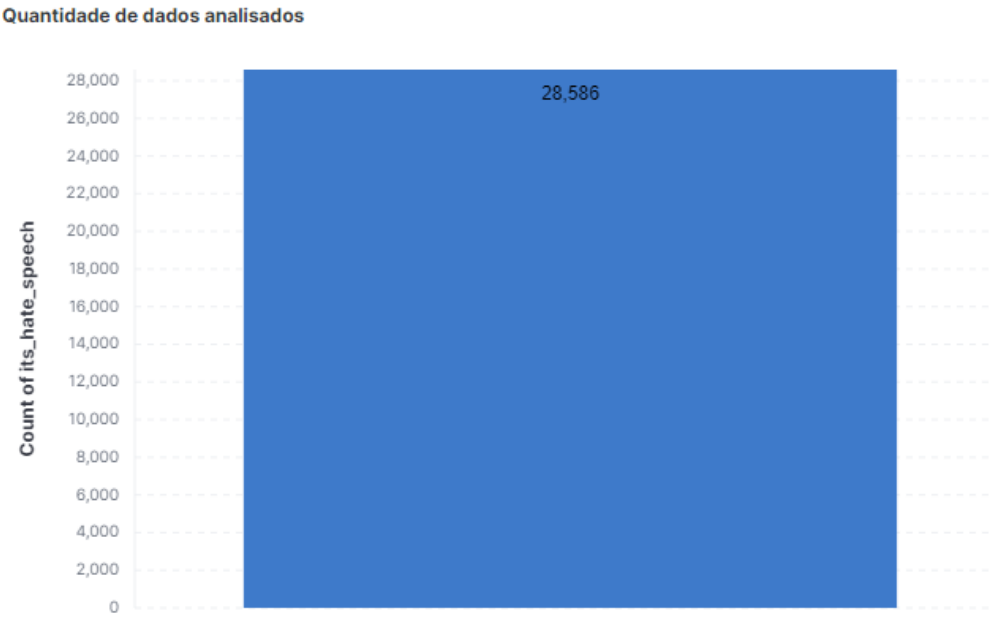

Figura 4.5 – Quantidade de *tweets* coletados.

A classificação dos dados foi realizada manualmente e, como critério, foram utilizadas as experiências pessoais das autoras do projeto e suas experiências com a rede social Twitter. Vale destacar que, pelo fato de as autoras serem mulheres, tendo ambas passado por algum tipo de constrangimento ou preconceito ao longo de suas vidas, foi mais fácil identificar os discursos de ódio praticados. Alguns exemplos dos *tweets* categorizados podem ser vistos na tabela a seguir.

Tabela 4.4 – Exemplos de tweets rotulados

Frase Coletada	Discurso de ódio ?
Isso, guarde p você filho de uma cadela sarnenta https://t.co/485kHSoBDd49z2	Sim
Lavar o cabelo é uma maravilha, o ruim é secar	Não
Não saio e não bebo e passo mal por causa de comida	Não
@bla Puta piranha vagabunda safada sem vergonha sarnenta sem futuro nojenta	Sim
A baleia era muito importante! A baleia é importante demais! Meu deus	Não
@fulana525 Amg se eu te contar meu drama por uma baleia que era XL.	Sim
A menina era balofo, bolota, gordo maldito, rolha de poço, baleia encalhada, hipopótamo, elefante, pata de dinossauro,bola da copa do mundo, pneu, peppa pig	Sim

Para a preservação de dados, foram alterados os links e as citações, para seguir a LGPD.

A partir do conjunto de dados foi constatado que, dos 28586 dados, 8383 estão dentro do contexto de discurso de ódio contra mulheres e 20203 não correspondem a esse cenário. A Figura 4.6 mostra o tipo de classificação e a porcentagem de tweets para cada categoria.

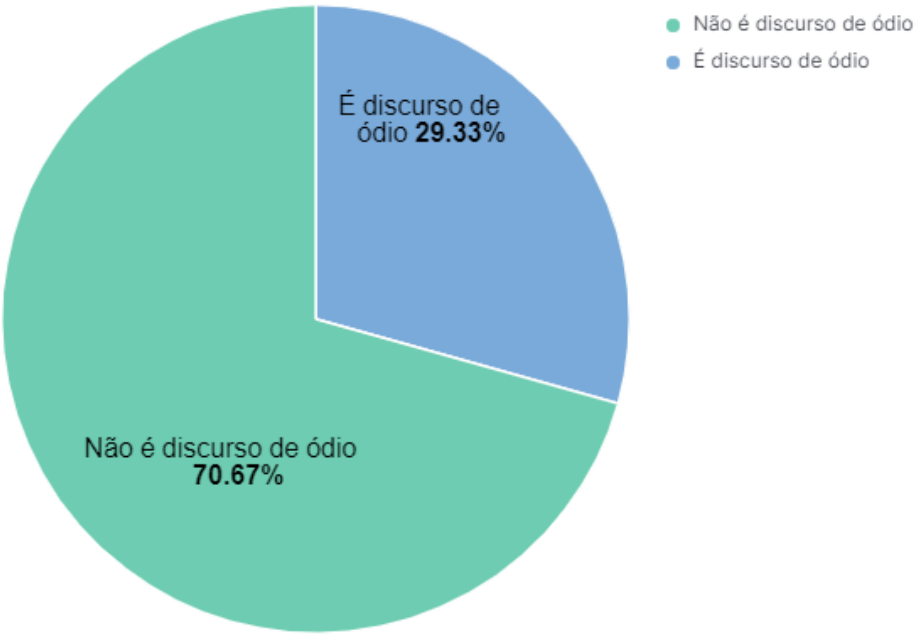

Figura 4.6 – Porcentagem da quantidade de *tweets* coletados.

A Figura 4.6 foi realizada no Kibana. Para ser possível visualizar os dados de maneira separada foi necessário, primeiramente, criar dois filtros para separar a quantidade de discursos de ódio classificados como 1 e como 0. A cor azul representa os *tweets* que são considerados ofensivos à mulher, já a cor verde são os *tweets* que não correspondem ao discurso de ódio.

Existem duas linguagens de consulta utilizadas pelo Kibana para pesquisar e filtrar dados, KQL e Lucene. A escolhida foi a KQL por ser uma linguagem simples e intuitiva que permite aos usuários escrever consultas para obter dados específicos do banco de dados. Por fim, escolheu-se visualizar o resultado por meio de porcentagem e no formato *"pie"*.

4.4 Balanceamento do dataframe

O balanceamento de dados é um processo importante para o treinamento de modelos de machine learning. É usado para garantir que os modelos não sejam treinados apenas com um conjunto de amostras favoráveis e que estes não sejam tendenciosos em relação a uma classe em particular. Na figura 4.7, pode-se notar que há um desequilíbrio entre o discurso de ódio com 20552 dados classificados e o não-discurso de ódio com 8563 dados classificados.

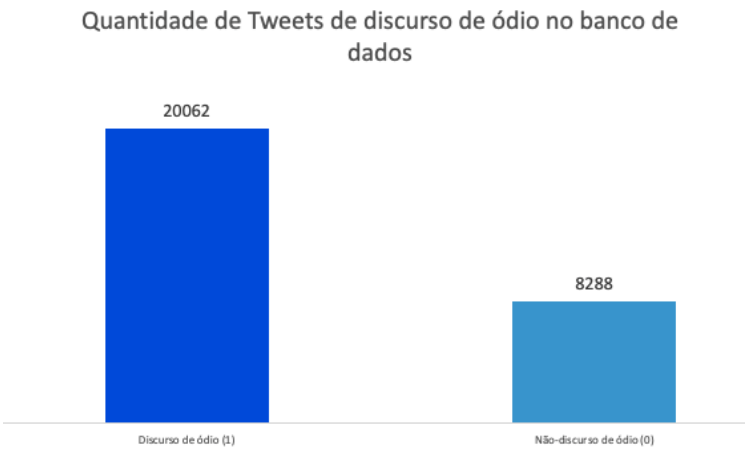

Figura 4.7 – Quantidade de Tweets de discurso de ódio no banco desbalanceado.

Para fazer o balanceamento dos dados, foram selecionadas amostras do conjunto de dados para equilibrar as classes, fazendo tanto o oversampling e undersampling.

Oversampling pode ser usado para gerar novos dados sintéticos da classe minoritária para tornar as classes mais balanceadas.

O undersampling pode ser usado para remover dados da classe majoritária, visando tornar as classes mais balanceadas.

Ambos os métodos podem ser usados para aumentar a precisão dos modelos de aprendizado de máquina.

Então, no final, três tipos de banco de dados serão gerados: banco desbalanceado, banco Undersampling e banco Oversampling. As quantidades de tweets por banco poderão ser vistas na figura 4.8.

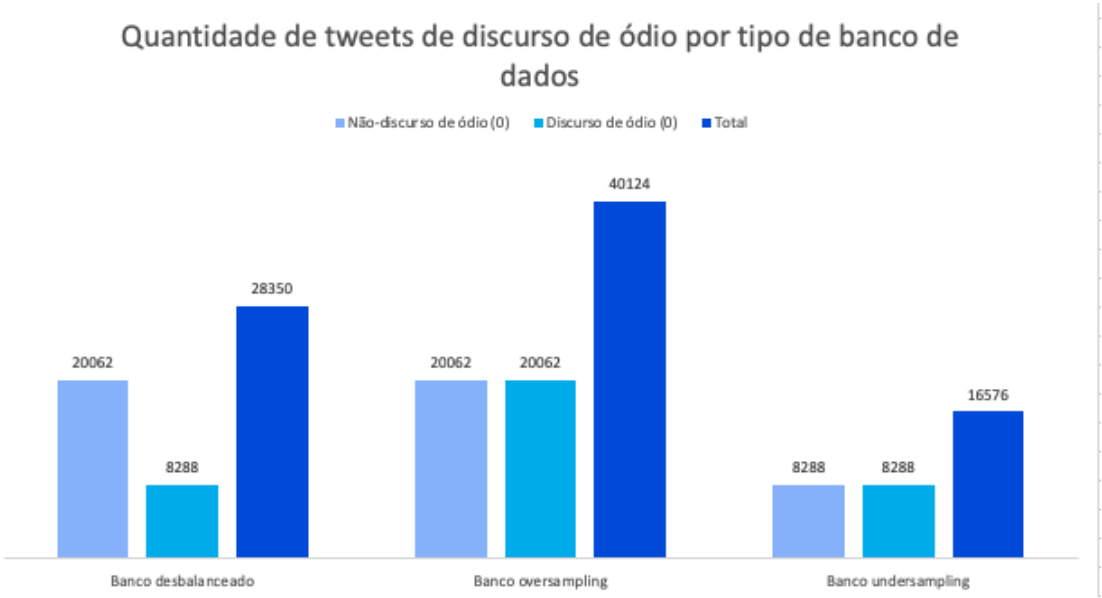

Figura 4.8 – Quantidade de twetts de discurso de ódio por tipo de banco de dados.

4.5 Cross Validation

A validação cruzada, ou *cross validation*, é um método usado para avaliar a precisão de um modelo de aprendizado de máquina. Ele consiste em dividir o conjunto de dados em várias partes (geralmente chamadas de *"folds"*), treinar o modelo em algumas dessas partes e testá-lo nas outras. Isso é repetido várias vezes, usando uma parte diferente como o conjunto de testes em cada interação. No final, as métricas de desempenho são agregadas para fornecer uma avaliação mais precisa do modelo. A validação cruzada é útil para evitar o *overfitting*, que ocorre quando um modelo é muito ajustado aos dados de treinamento e estes não generalizam bem para novos dados.

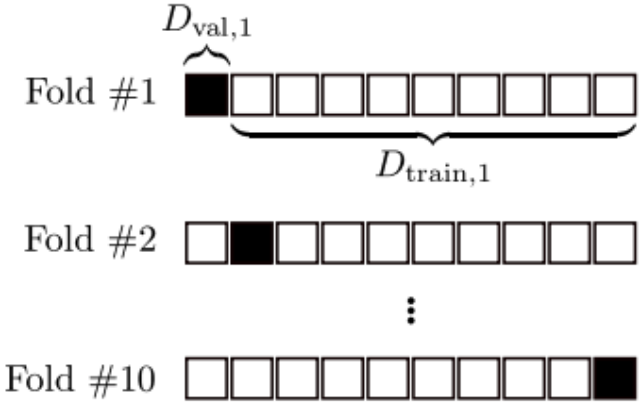

Figura 4.9 – Validação cruzada de 10-Folds [74].

A validação cruzada mais comum é chamada de validação cruzada *k-fold*, onde k é o número de partes *(folds)* em que o conjunto de dados é dividido. Usualmente, k é escolhido como 5 ou 10. Nesse caso, o conjunto de dados é dividido em k partes (ou *folds*) de tamanhos semelhantes. Em seguida, o modelo é treinado k vezes, sendo que, em cada vez, uma parte diferente é utilizada como o conjunto de testes e, as *k-1* partes restantes, como o conjunto de treinamento. No final, as métricas de desempenho são agregadas para fornecer uma avaliação mais precisa do modelo [74].

Para realizar a validação cruzada, o conjunto de dados foi dividido em 10 parte. Essa técnica é chamada de validação cruzada *10-fold*. A cada interação, um dos 10 *folds* é usado como conjunto de teste e os outros 9 como conjunto de treinamento. A métrica de desempenho é, então, calculada e armazenada. Ao final, das 10 interações, as métricas são agregadas para fornecer uma avaliação final do modelo.

4.5.1 Tratamento Undersampling

O *Cross validation* com *undersampling* é um método para avaliar a precisão de um modelo de aprendizado de máquina quando o conjunto de dados é desequilibrado. O *undersampling* é um método de tratamento de dados desequilibrados onde a classe majoritária é subamostrada para ser igual à classe minoritária. Isso é feito para equilibrar as classes no conjunto de dados e evitar o problema do viés de classe.

Para realizar a validação cruzada com *undersampling*, o conjunto de dados é dividido em k *folds* como descrito anteriormente. Em seguida, o *undersampling* é aplicado a cada *fold* de treinamento, de forma que a classe majoritária tenha o mesmo tamanho que a classe minoritária. O modelo é treinado usando esse conjunto de treinamento equilibrado e testado no conjunto de testes correspondente. As métricas de desempenho são, então, calculadas e agregadas ao final das k interações para fornecer uma avaliação final do modelo.

A validação cruzada com *undersampling* é útil quando o conjunto de dados é desequilibrado e o modelo tem dificuldade em aprender a classificar corretamente a classe minoritária. Ele ajuda a evitar o problema do viés de classe e fornece uma avaliação mais precisa do modelo.

4.5.1.1 Modelo 1: CountVectorizer com SGDClassifier

O *undersampling* é usado para equilibrar as classes, o *CountVectorizer* é usado para transformar o texto em dados numéricos e o *SGDClassifier* é usado para treinar e classificar os dados. Juntos, esses três componentes (*undersampling, CountVectorizer, SGDClassifier*) podem ser usados para criar um modelo de classificação de texto que lida com desequilíbrio de classes.

Tabela 4.5 – Tabela de Confusão: Undersampling modelo 1.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	5630 33,96%	1463 8,83%
	Não é discurso de ódio	2658 16,04%	6825 41,17%

A tabela 4.5 apresenta os resultados de uma validação cruzada do modelo de aprendizado de máquina. Ela mostra as quatro possíveis combinações de previsão e classe esperada: discurso de ódio previsto como discurso de ódio, discurso de ódio previsto como não-discurso de ódio, não-discurso de ódio previsto como discurso de ódio e não-discurso de ódio previsto como não-discurso de ódio.

A tabela 4.5 mostra que o modelo previu 5630 instâncias como discurso de ódio e elas foram realmente discurso de ódio, o que representa 33,96% do total de instâncias de discurso de ódio. Isso é conhecido como taxa de verdadeiros positivos (TP).

O modelo também previu 1463 instâncias como discurso de ódio, mas elas não eram realmente discurso de ódio, o que representa 8,33% do total de instâncias não-discurso de ódio. Isso é conhecido como taxa de falsos positivos (FP).

O modelo previu 2658 instâncias como não-discurso de ódio, mas elas eram realmente discurso de ódio, o que representa 16,04% do total de instâncias de discurso de ódio. Isso é conhecido como taxa de falsos negativos (FN).

Por fim, o modelo previu 6825 instâncias como não-discurso de ódio e elas foram realmente não-discurso de ódio, o que representa 41,17% do total de instâncias não-discurso de ódio. Isso é conhecido como taxa de verdadeiros negativos (TN).

4.5.1.2 Modelo 2: TfidfVectorizer com SGDClassifier

O *undersampling* é usado para equilibrar as classes, o *TfidfVectorizer* é utilizado para transformar o texto em dados numéricos e, o *SGDClassifier*, para treinar e classificar os dados. O uso de *TfidfVectorizer* em vez de *CountVectorizer* pode ajudar a dar mais peso a palavras menos frequentes, o que pode ser útil para lidar com problemas de desequilíbrio de classes.

Tabela 4.6 – Tabela de Confusão: Undersampling modelo 2.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	5612 33,86%	1521 9,18%
	Não é discurso de ódio	2676 16,14%	6767 40,82%

A tabela 4.6 representa uma tabela de validação cruzada, onde são mostrados os resultados de um teste de classificação, ou seja, se é ou não discurso de ódio. A primeira coluna mostra a classe esperada do discurso, que pode ser Discurso de Ódio ou Não-Discurso de Ódio. A segunda coluna mostra a classe prevista pelo modelo de classificação, que também pode ser Discurso de Ódio ou Não-Discurso de Ódio. Com base na tabela é possível observar os seguintes dados: Os 40,82% são os textos classificados corretamente como Não-Discurso de Ódio. O resultado 5612 decorreu dos itens que foram classificados corretamente como Discurso de Ódio, correspondendo a 33,86%.

4.5.1.3 Modelo 3: TfidfVectorizer com MultinomialNB

O *undersampling* é usado para equilibrar as classes, o *TfidfVectorizer* é usado para transformar o texto em dados numéricos e o *MultinomialNB* é usado para treinar e classificar os dados. Juntos, esses três componentes (*undersampling, TfidfVectorizer, MultinomialNB*) podem ser usados para criar um modelo de classificação de texto que lida com desequilíbrio de classes.

Tabela 4.7 – Tabela de Confusão: Undersampling modelo 3.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	5245 31,64%	1273 7,68%
	Não é discurso de ódio	3043 18,36%	7015 42,32%

A validação cruzada da tabela mostra o desempenho do classificador em relação às categorias "Discurso de ódio"e "Não é discurso de ódio".

A coluna "Classe esperada"indica a categoria real do texto, enquanto a coluna "Classe prevista"indica a categoria atribuída pelo classificador.

A linha "5245"mostra quantos dos textos foram corretamente classificados como "Discurso de ódio", representando a taxa de acerto do classificador para essa categoria, ou seja, "31,64%".

A mesma lógica se aplica à categoria "Não é discurso de ódio". A coluna "7015"indica o número de textos corretamente classificados como "Não é discurso de ódio", enquanto "42,32%"representam a taxa de acerto para essa categoria.

A partir da tabela 4.7, é possível avaliar o desempenho do classificador e identificar áreas de melhoria para aumentar sua eficácia.

4.5.1.4 Modelo 4: CountVectorizer com MultinomialNB

O modelo de *undersampling CountVectorizer* com o algoritmo *MultinomialNB* é usado para prever resultados binários com base na frequência de palavras nos documentos de texto, aplicando *undersampling* para reduzir o desequilíbrio de classes nos dados de treinamento.

Tabela 4.8 – Tabela de Confusão: Undersampling modelo 4.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	5610 33,84%	1607 9,69%
	Não é discurso de ódio	2678 16,16%	6681 40,31%

A tabela de validação cruzada, representada na tabela 4.8, mostra o desempenho de um classificador para diferentes classes. A tabela mostra duas classes: Discurso de Ódio e Não é Discurso de Ódio. A primeira coluna mostra a classe esperada, enquanto as outras quatro colunas mostram

a quantidade de observações classificadas corretamente para cada classe. A primeira linha mostra que 5610 observações foram classificadas corretamente como Discurso de Ódio, o que representa um desempenho de 33,84%. A segunda linha mostra que 1607 observações foram previstas corretamente como Não é Discurso de Ódio, o que representa um desempenho de 9,69%. A terceira linha mostra que 2678 observações foram classificadas incorretamente como Não é Discurso de Ódio, o que representa um desempenho de 16,16%. Por último, a quarta linha mostra que 6681 observações foram previstas incorretamente como Discurso de Ódio, o que representa um desempenho de 40,31%.

4.5.2 Dados desbalanceados

Em um conjunto de dados desbalanceado, o algoritmo de classificação pode ter dificuldade em reconhecer a classe minoritária devido à falta de exemplos suficientes para aprender.

A validação cruzada em conjuntos de dados desbalanceados é um processo semelhante à validação cruzada convencional, mas ele é adaptado para lidar com desequilíbrios de classe no conjunto de dados. A validação cruzada aplicada a conjuntos de dados desbalanceados é frequentemente usada em problemas de classificação, onde uma das classes é muito menor que as outras.

Existem várias abordagens para a validação cruzada em conjuntos de dados desbalanceados, como:

- *Stratified k-fold cross-validation*: este método garante que cada *"fold"* contenha uma proporção similar de exemplos de cada classe. Isso é útil para evitar problemas de viés de classe, onde o modelo pode se tornar muito preciso para a classe majoritária e menos preciso para a classe minoritária.
- Monte Carlo *Cross-Validation*: este método gera vários subconjuntos aleatórios do conjunto de dados e realiza a validação cruzada com cada um deles. Isso pode ajudar a lidar com desequilíbrios de classe mais severos.
- *Resampling*: O conjunto de dados é subamostrado ou sobre-amostrado para equilibrar a distribuição de classes antes da validação cruzada.

A escolha do método de validação cruzada a ser utilizado pode variar de acordo com o tamanho do conjunto de dados, a natureza do problema e a possibilidade de se ter mais recursos computacionais. A validação cruzada em conjuntos de dados desbalanceados é importante para garantir que o modelo seja capaz de lidar com classes minoritárias e possa generalizar bem para novos exemplos.

4.5.2.1 Modelo 1: CountVectorizer com SGDClassifier

O *CountVectorizer* é usado para transformar o texto em dados numéricos e o *SGDClassifier* é usado para treinar e classificar os dados. Juntos, esses dois componentes (*CountVectorizer, SGDClassifier*) podem ser usados para criar um modelo de classificação de texto que lida com desequilíbrio de classes.

Tabela 4.9 – Tabela de Confusão: Dados desbalanceados modelo 1.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	4140 14,60%	3714 13,10%
	Não é discurso de ódio	4148 14,63%	16348 57,66%

A tabela mostra que o modelo previu 4.140 ocorrências como discurso de ódio que, na verdade, eram discurso de ódio respondendo por 14,60% do total de ocorrências deste tipo de discurso. Isso é chamado de taxa de verdadeiro positivo (TP).

O modelo também previu 3.714 instâncias de discurso de ódio que não eram verdadeiros discursos de ódio, respondendo por 13,10% do número total de instâncias de não discurso de ódio. Isso é chamado de taxa de falso positivo (FP).

O modelo previu 4.148 instâncias como não discurso de ódio que, na verdade, correspondiam a discurso de ódio, respondendo por 14,63% do total de casos de discurso de ódio. Isso é chamado de taxa de falso negativo (FN).

No final, o modelo previu 16.348 instâncias como não discurso de ódio, que na verdade não eram discurso de ódio, respondendo por 57,66% do total de ocorrências de não discurso de ódio. Isso é chamado de taxa de verdadeiro negativo (TN).

4.5.2.2 Modelo 2: TfidfVectorizer com SGDClassifier

O *TfidfVectorizer* é usado para transformar o texto em dados numéricos e o *SGDClassifier* é usado para treinar e classificar os dados. Juntos, esses dois componentes (*TfidfVectorizer, SGDClassifier*) podem ser usados para criar um modelo de classificação de texto que lida com desequilíbrio de classes.

Tabela 4.10 – Tabela de Confusão: Dados desbalanceados modelo 2.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	3674 12,96%	2900 10,53%
	Não é discurso de ódio	4614 16,28%	17162 60,54%

A tabela 4.10 apresenta os resultados de uma validação cruzada feita para classificar textos como discurso de ódio ou não. Ela apresenta a comparação entre as classes previstas pelo modelo

e as classes esperadas, baseadas em um conjunto de dados previamente rotulado.

Por exemplo, o modelo previu 3674 ocorrências corretamente de discurso de ódio e essas ocorrências representavam 12,96% da classe esperada. Já para a classe "não é discurso de ódio", o modelo previu 17162 ocorrências e essas representavam 60,54% da classe esperada.

A tabela permite avaliar a acurácia do modelo e identificar eventuais problemas de desempenho, como a prevalência de falso positivo ou falso negativo, entre outros.

4.5.2.3 Modelo 3: TfidfVectorizer com MultinomialNB

O *TfidfVectorizer* é uma ferramenta de pré-processamento de texto que converte uma coleção de documentos em um conjunto de recursos de frequência de termo-documento inversa *(TF-IDF)* e o *MultinomialNB* é uma implementação do algoritmo Naive Bayes para classificação multiclasse. Juntos, esses dois componentes (*TfidfVectorizer, MultinomialNB*) podem ser usados para criar um modelo de classificação de texto que lida com desequilíbrio de classes.

Tabela 4.11 – Tabela de Confusão: Dados desbalanceados modelo 3.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	684 2,41%	230 0,81%
	Não é discurso de ódio	7604 28,82%	19832 69,95%

A tabela 4.11 apresenta os resultados de uma validação cruzada feita para classificar textos como discurso de ódio ou não. É importante ressaltar que o conjunto de dados utilizado é desbalanceado, o que significa que há uma diferença significativa entre a quantidade de exemplos de cada classe.

A tabela apresenta a comparação entre as classes previstas pelo modelo e as classes esperadas, baseadas em um conjunto de dados previamente rotulado.

Por exemplo, o modelo previu 684 ocorrências de discurso de ódio e essas ocorrências representavam 2,41% da classe esperada. Já para a classe "não é discurso de ódio", o modelo previu 7604 erroneamente e esses dados representavam 28,82% de falsos negativos.

A desproporção entre as classes pode afetar a precisão do modelo, tornando-o menos capaz de prever corretamente a classe minoritária. Isso pode ser evidenciado pelos baixos valores de acurácia para a classe "discurso de ódio".

A tabela permite avaliar a acurácia do modelo e identificar eventuais problemas de desempenho, como a prevalência de falso positivo ou falso negativo, entre outros.

4.5.2.4 Modelo 4: CountVectorizer com MultinomialNB

Neste modelo será implementado o *CountVectorizer* com *MultinomialNB*. Juntos, esses dois componentes podem ser usados para criar um modelo de classificação de texto que lida com desequilíbrio de classes. O *CountVectorizer* é uma ferramenta de pré-processamento de texto que converte uma coleção de documentos em uma representação numérica de frequência de termo *(bag-of-words)* e o *MultinomialNB* é uma implementação do algoritmo Naive Bayes para classificação multiclasse.

Tabela 4.12 – Tabela de Confusão: Dados desbalanceados modelo 4.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	3256 11,49%	1906 6,72%
	Não é discurso de ódio	5032 17,75%	18156 64,04%

A tabela 4.12 representa uma validação cruzada que mostra os resultados de um teste de classificação, ou seja, se são ou não discurso de ódio. A primeira coluna mostra a classe esperada do discurso, que pode ser Discurso de Ódio ou Não-Discurso de Ódio. A segunda mostra a classe prevista pelo modelo de classificação, que também pode ser Discurso de Ódio ou Não-Discurso de Ódio. Com base na tabela é possível observar os seguintes dados: 3256 foram os itens que foram classificados corretamente como Discurso de Ódio, correspondendo 11,49%. Os 64,04% são os textos classificados corretamente como Não-Discurso de Ódio.

4.5.3 Tratamento Oversampling

A validação cruzada com *oversampling* é um método para avaliar a precisão de um modelo de aprendizado de máquina quando o conjunto de dados é desequilibrado. O *oversampling* é um método de tratamento de dados desequilibrados onde a classe minoritária é sobre-amostrada para ser igual à classe majoritária. Isso é feito para equilibrar as classes no conjunto de dados e evitar o problema do viés de classe.

Para realizar a validação cruzada com *oversampling*, o conjunto de dados é dividido em *k folds* como descrito anteriormente. Em seguida, o *oversampling* é aplicado a cada *fold* de treinamento, de forma que a classe minoritária tenha o mesmo tamanho que a classe majoritária. O modelo é treinado usando esse conjunto de treinamento equilibrado e testado no conjunto de testes correspondente. As métricas de desempenho são, então, calculadas e agregadas ao final das k interações para fornecer uma avaliação final do modelo.

A validação cruzada com *oversampling* é útil quando o conjunto de dados é desequilibrado e o modelo tem dificuldade em aprender a classificar corretamente a classe minoritária. Ele ajuda a evitar o problema do viés de classe e fornece uma avaliação mais precisa do modelo.

4.5.3.1 Modelo 1: CountVectorizer com SGDClassifier

A técnica de *oversampling* deve ser aplicada antes de treinar o modelo para equilibrar as classes. O *CountVectorizer* é usado para transformar o texto em dados numéricos e o *SGDClassifier* é usado para treinar e classificar os dados. Juntos, esses dois componentes (*CountVectorizer, SGDClassifier*) podem ser usados com uma técnica de *oversampling* para criar um modelo de classificação de texto que lida com desequilíbrio de classes.

Tabela 4.13 – Tabela de Confusão: Oversampling modelo 1.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	18481 46,06%	5108 12,73%
	Não é discurso de ódio	1581 3,94%	14954 37,27%

A tabela 4.13 apresenta os resultados de uma validação cruzada do modelo de aprendizado de máquina. Ela mostra as quatro possíveis combinações de previsão e classe esperada: discurso de ódio previsto como discurso de ódio, discurso de ódio previsto como não-discurso de ódio, não-discurso de ódio previsto como discurso de ódio e não-discurso de ódio previsto como não-discurso de ódio.

A tabela 4.13 mostra que o modelo previu 18481 instâncias como discurso de ódio e elas foram realmente discurso de ódio, o que representa 46,06% do total de instâncias de discurso de ódio. Isso é conhecido como taxa de verdadeiros positivos (TP).

O modelo previu 14954 instâncias como não-discurso de ódio e elas foram realmente não-discurso de ódio, o que representa 37,27% do total de instâncias não-discurso de ódio. Isso é conhecido como taxa de verdadeiros negativos (TN).

O modelo também previu 5108 instâncias como discurso de ódio, mas elas não eram realmente discurso de ódio, o que representa 12,73% do total de instâncias não-discurso de ódio. Isso é conhecido como taxa de falsos positivos (FP).

Por fim, o modelo previu 1581 instâncias como não-discurso de ódio, mas elas eram realmente discurso de ódio, o que representa 3,94% do total de instâncias de discurso de ódio. Isso é conhecido como taxa de falsos negativos (FN).

4.5.3.2 Modelo 2: TfidfVectorizer com SGDClassifier

TfidfVectorizer é uma ferramenta de pré-processamento de texto que converte uma coleção de documentos em uma representação numérica baseada em *TF-IDF* e o *SGDClassifier* é uma implementação do algoritmo de descida de gradiente estocástico para classificação. A técnica de *oversampling* deve ser aplicada antes de treinar o modelo para equilibrar as classes.

Tabela 4.14 – Tabela de Confusão: Oversampling modelo 2.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	17173 42,80%	5744 14,32%
	Não é discurso de ódio	2889 10,19%	14318 35,68%

A tabela 4.14 mostra os resultados da validação cruzada para classificar o texto como discurso de ódio ou não discurso de ódio. Ele mostra como a classe prevista pelo modelo se compara à classe esperada, com base em um conjunto de dados previamente rotulado.

Por exemplo, o modelo previu corretamente 17.163 ocorrências de discurso de ódio, o que representou 42,80% da categoria esperada. Já para a classe "Isso não é discurso de ódio", o modelo prevê 14318 ocorrências, o que representa 38,68% da classe esperada.

Esta tabela permite avaliar a precisão do modelo e identificar possíveis problemas de desempenho, como a prevalência de falsos positivos ou falsos negativos, etc.

4.5.3.3 Modelo 3: TfidfVectorizer com MultinomialNB

O *TfidfVectorizer* é usado para converter um conjunto de documentos em vetores de características. Estes vetores são depois usados como entrada para o algoritmo de classificação, o *MultinomialNB*. O *MultinomialNB* é um classificador de Naive Bayes, o qual assume que as *features* devem ter uma distribuição de probabilidade de Bernoulli. Ele é útil para classificar texto, pois é capaz de lidar com grandes volumes de dados, utilizando um modelo probabilístico para classificar as instâncias.

O modelo *oversampling TfidfVectorizer* com *MultinomialNB* é particularmente útil quando se trata de classificar texto, pois permite que os dados sejam escalados automaticamente, enquanto também equilibra os dados. Isso permite que o classificador seja mais preciso e eficiente.

Tabela 4.15 – Tabela de Confusão: Oversampling modelo 3.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	17591 43,84%	4886 12,18%
	Não é discurso de ódio	2471 6,16%	15176 37,82%

A validação cruzada, representada na tabela 4.15, mostra o desempenho de um classificador

para diferentes classes. A tabela mostra duas classes: Discurso de Ódio e Não é Discurso de Ódio. A primeira coluna mostra a classe esperada, enquanto as outras quatro colunas mostram a quantidade de observações classificadas corretamente para cada classe. A primeira linha mostra que 17591 observações foram classificadas corretamente como Discurso de Ódio, o que representa um desempenho de 43,84%. A segunda linha mostra que 4886 observações foram previstas incorretamente como Discurso de Ódio, o que representa um desempenho de 12,18%. A terceira linha mostra que 2471 observações foram classificadas incorretamente como Não é Discurso de Ódio, o que representa um desempenho de 16,16%. Por último, a quarta linha mostra que 15176 observações foram previstas incorretamente como Discurso de Ódio, o que representa um desempenho de 37,82%.

4.5.3.4 Modelo 4: CountVectorizer com MultinomialNB

O modelo *Oversampling CountVectorizer* com *MultinomialNB* é um modelo de classificação binária que usa um mecanismo de sobre-amostragem para aumentar a quantidade de dados existentes. O *CountVectorizer* é usado para converter um conjunto de documentos de texto (texto bruto) em um vetor de contagens de palavras, enquanto o *MultinomialNB* é usado para aplicar a classificação binária. O resultado acerca da sobre-amostragem é usado para aumentar o número de exemplos do conjunto de dados que possuem um resultado positivo. Isso ajuda a equilibrar o conjunto de dados, tornando-o menos suscetível ao viés de classificação. Isso geralmente resulta em um modelo de classificação mais preciso.

Tabela 4.16 – Tabela de Confusão: Oversampling modelo 4.

		Classe esperada	
		Discurso de ódio	Não é discurso de ódio
Classe prevista	Discurso de ódio	17954 44,75%	5458 13,60%
	Não é discurso de ódio	2108 5,25%	14604 36,40%

A validação cruzada da tabela mostra o desempenho do classificador em relação às categorias "Discurso de ódio"e "Não é discurso de ódio".

A coluna "Classe esperada"indica a categoria real do texto, enquanto a coluna "Classe prevista"indica a categoria atribuída pelo classificador.

A linha "17954"mostra quantos dos textos foram corretamente classificados como "Discurso de ódio", representando a taxa de acerto do classificador para essa categoria "44,75%".

A mesma lógica se aplica à categoria "Não é discurso de ódio". A coluna "14604"indica o número de textos corretamente classificados como "Não é discurso de ódio", enquanto "36,40%"representa a taxa de acerto para essa categoria.

A partir da tabela 4.16, é possível avaliar o desempenho do classificador e identificar áreas de melhoria para aumentar sua eficácia.

4.6 Análise comparativa do treinamento

A partir dos resultados do treinamento foi criada a tabela 4.17, que apresenta as taxas de precisão de quatro modelos diferentes de aprendizado de máquina em diferentes condições. As 4 condições possíveis incluem o modelo *CountVectorizer* com *SGDClassifier*, o modelo *TfidfVectorizer* com *SGDClassifier*, o modelo *TfidfVectorizer* com *MultinomialNB* e o modelo *CountVectorizer* com *MultinomialNB*. A taxa de acerto é uma métrica utilizada para avaliar o desempenho de um modelo, ela mede o número de previsões corretas feitas pelo modelo, apresentado o resultado como uma porcentagem do número total de previsões feitas.

Nesta tabela, os modelos estão sendo avaliados em três condições diferentes: banco *undersampling*, banco *oversampling* e o banco desbalanceado.

Tabela 4.17 – Acurácia dos bancos Undersampling, banco Oversampling e Banco desbalanceado e seus respectivos modelos.

	Undersampling		Banco desbalanceado		Oversampling	
	Ambiente de teste	Cross Validation	Ambiente de teste	Cross Validation	Ambiente de teste	Cross Validation
Modelo 1	80,42%	75,14%	82,06%	72,26%	87,76%	83,32%
Modelo 2	80,74%	74,68%	82,54%	73,49%	83,22%	78,48%
Modelo 3	77,48%	73,96%	74,11%	72,37%	84,03%	81,66%
Modelo 4	78,35%	74,15%	80,26%	75,52%	83,93%	81,14%

É possível observar que o melhor banco de dados para treinamento é o modelo de *Oversampling*. Para fins de comparação, serão comparados somente os dados em *cross validation*. O Modelo 1 tem a maior taxa de precisão em todas as condições, com um resultado percentual de 83,21%. O modelo 2 possui a menor taxa de precisão dos quatro modelos, com a taxa de precisão de 72,37%. O Modelo 3 tem uma taxa de precisão consistentemente menor do que o Modelo 1, com a taxa de precisão de 81,66%. O Modelo 4 tem uma taxa de precisão semelhante ao Modelo 3, com a maior taxa de precisão de 81,14%.

Em resumo, a tabela mostra o desempenho de quatro modelos em diferentes condições, a qual pode ser usada para comparar o desempenho de diferentes modelos e identificar o melhor desempenho entre eles.

```
def testando_treino(modelo1, modelo2, modelo3, modelo4):
  text = pd.read_csv('Data base/teste.csv')

  #gerando objeto com os texto modelado
  text_predict = pd.DataFrame(teste)

  tweets = text_predict['text'].apply(data_preparation)
  text_predict['clean'] = tweets
  print(text_predict)
```

```
10
11  #testando modelos
12
13  predicao_com_frase1 = modelo1.predict(text_predict['clean'])
14  print(predicao_com_frase1)
15  predicao_com_frase2 = modelo2.predict(text_predict['clean'])
16  print(predicao_com_frase2)
17  predicao_com_frase3 = modelo3.predict(text_predict['clean'])
18  print(predicao_com_frase3)
19  predicao_com_frase4 = modelo4.predict(text_predict['clean'])
20  print(predicao_com_frase4)
```

Trecho de código 4.1 – Função de exemplos para testar os modelos treinados

O trecho do código 4.1 mostra a função chamada *"testando_treino"* que recebe quatro parâmetros: "modelo1", "modelo2", "modelo3"e "modelo4", que são os critérios de realização de treinamento. A função usa um *dataframe* que foi criado para teste com as colunas *"text"* e *"its_hate_speech"*. A função aplica, então, outra função chamada *"data_preparation"*, que pode ser encontrada no código 3.6, para remover os dados desnecessários, criando uma nova coluna denominada *"clean"*. A função usa, a seguir, os quatro modelos passados como parâmetros para prever os dados de texto *"clean"*, imprimindo as previsões para cada modelo. Como exemplo de saída, será demonstrado na tabela 4.18 o resultado do teste. Para a preservação de dados foram alterados os links e as citações, para seguir a LGPD.

Tabela 4.18 – Saídas para novos exemplos pros bancos Undersampling, desbalanceado e Oversampling e modelos 1, 2, 3 e 4 respectivamente treinado.

		Undersampling				Banco desbalanceado				Oversampling			
	its hate speech	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
Agora ódio de aquele juíz filho baranga corno	1	1	1	1	1	1	1	0	1	1	1	1	1
Você devia pensa baranga escrota	1	1	1	1	1	1	1	0	1	1	1	1	1
vc é um put4 feioso	0	1	1	1	1	0	0	0	0	1	1	1	1
A ex da minha namorada	0	1	1	1	1	0	0	0	0	0	1	1	1
O neymar é vagabundo	0	0	1	1	0	0	0	0	0	0	1	1	1
macaca gosta de banana	0	0	1	0	1	0	0	0	0	0	1	0	0
rainha da minha vida	0	0	0	0	0	0	0	0	0	0	0	0	0
mulher gosta de banana	0	0	0	0	1	0	0	0	0	0	0	0	0
put4	1	1	1	1	1	0	0	0	0	1	1	1	1
vadi4	1	0	1	1	1	0	0	1	1	0	1	1	1
O restaurante tem uma comida gostosa mas tinha um cabelo	0	0	0	0	0	0	0	0	0	0	0	0	0
Eu não sou prostituta e nem vagabunda, eu preciso de respeito	0	0	1	0	1	0	0	0	0	1	1	0	1

5 Conclusão

O discurso de ódio contra as mulheres nas redes sociais é uma questão grave e preocupante que precisa ser combatida. As mulheres são frequentemente vítimas de comentários e mensagens degradantes, sexistas, misóginos e ameaçadores nas redes sociais. Além de ser uma forma de violência, esse tipo de discurso pode ter graves consequências psicológicas e sociais para as mulheres. As plataformas de mídia social devem tomar medidas eficazes para monitorar e combater o discurso de ódio contra as mulheres, incluindo a remoção de conteúdo inapropriado e a punição de usuários que violam as regras da comunidade. Além disso, é importante fomentar uma cultura de respeito e tolerância nas redes sociais e em toda a sociedade.

O *Machine Learning* pode ser uma ferramenta valiosa na luta contra o discurso de ódio contra as mulheres nas redes sociais, pois, através de algoritmos de *Machine Learning*, podem ser realizados treinamentos para prever se um comentário ou post foi ofensivo ou não, o que pode ajudar a prevenir a publicação de discurso de ódio nas redes sociais.

Neste trabalho foi feito um estudo da utilização de algoritmos de *Machine Learning* e ferramentas de processamento de linguagem natural, que podem ser usados na luta contra o discurso de ódio contra as mulheres nas redes sociais. A rede social escolhida foi o Twitter, por dispor de uma grande quantidade de dados, sendo uma ótima fonte para analisar dados de todos os tipos.

Por mais que a temática seja relevante, não foi possível encontrar um *dataset* com dados pré-rotulados que contemplasse o tema do trabalho. Então, foi necessário utilizar a API do Twitter para coletar os dados. Ela se mostrou muito eficiente e foi possível fazer a filtragem por palavras chaves e linguagem.

Com os dados coletados, iniciou-se a preparação destes, realizando-se seu processo de limpeza. Essa etapa é fundamental, pois, se os dados não estiverem corretamente formatados ou apresentarem *outliers*, valores ausentes ou duplicados, pode haver prejuízo à performance do modelo, levando-o a resultados ruins. Após o tratamento dos dados realizou-se a sua classificação.

A classificação dos dados é importante no aprendizado de máquina porque ajuda a determinar a categoria a que cada exemplo pertence. Isso é útil no cenário deste trabalho por se tratar de um aprendizado supervisionado, em que o objetivo é prever uma variável categórica a partir de outras variáveis. O maior desafio nessa etapa foi realizar a rotulagem manual dos dados. Por ter sido um processo manual e a classificação de forma subjetiva, podem ter ocorrido alguns erros durante o período de classificação dos dados.

Com o *dataset* criado, a próxima etapa consistiu no treinamento dos dados. Para isso, primeiramente realizou-se o balanceamento dos dados, pois a falta deste pode levar a uma representação distorcida da realidade, impactando a tomada de decisão. As técnicas utilizadas nesta etapa foram o *undersampling* e o *oversampling*.

Depois, iniciou-se a implementação do treinamento do *Machine Learning*. Para isso, foi uti-

lizada a biblioteca do *Scikit-Learn*, que se mostrou bem completa, sendo possível utilizá-la para a construção de diversos modelos de aprendizado de máquina. Dentre os modelos presentes no *Scikit-Learn*, foram escolhidos os modelos *SGDClassifier, MultinomialNB, CountVectorizer* e *TfidfVectorizer* para realizar o treinamento na detecção de discurso de ódio nas mulheres.

Para realizar o treinamento, foram realizados 4 modelos a partir da combinação dos algoritmos escolhidos da biblioteca *Scikit-Learn*. Os 4 modelos foram: *CountVectorizer* com *SGDClassifier, TfidfVectorizer* com *SGDClassifier, TfidfVectorizer* com *MultinomialNB* e *CountVectorizer* com *MultinomialNB*. Esses modelos foram usados em 3 cenários distintos: No cenário com os dados desbalanceados, no cenário *Undersampling* e no cenário *Oversampling*. Por meio da tabela 4.17 é possível verificar que o melhor banco de dados para treinamento foi o cenário de *Oversampling*, com precisão superior a 78,00%.

No trabalho também foi utilizado o Kibana para visualizar alguns dados. Foi escolhida essa ferramenta por oferecer ao usuário uma interface intuitiva para pesquisar e filtrar dados e descobrir *insights* sobre conjuntos de dados complexos, além de criar e salvar painéis personalizados que exibem gráficos, tabelas, mapas e outras visualizações de dados.

5.1 Trabalhos futuros

O trabalho futuro tem a pretensão de realizar a detecção de discurso de ódio no Twitter de forma automatizada no dia a dia da rede social. Também seria bom pensar em responder às pessoas que estão twittando como uma forma de ensinar a maneira correta de se expressar com as mulheres, usando o ChatGPT.

Um bot no Twitter é uma conta automatizada que pode realizar tarefas como publicar tweets, curtir, seguir outros usuários e outras ações sem intervenção humana. Os bots são criados usando programação e geralmente são utilizados para fins comerciais, de marketing, ou simplesmente para automatizar tarefas repetitivas. Alguns exemplos de uso incluem: responder automaticamente a mensagens diretas, enviar atualizações de notícias ou previsão do tempo, ou monitorar hashtags para rastrear tendências.

A implementação da detecção de discurso de ódio como um bot no Twitter é uma ideia interessante, mas também pode ser desafiadora. É importante levar em consideração as questões éticas e de privacidade envolvidas na monitoração de conversas nas mídias sociais, levando em consideração a Lei Geral de Proteção de Dados (LGPD). Além disso, será necessário considerar as políticas da plataforma do Twitter em relação a bots e à detecção de discurso de ódio.

Caso se decida seguir nessa linha de projeto, é importante trabalhar de forma cuidadosa e transparente, garantindo que a privacidade dos usuários seja protegida e que as respostas geradas pelo ChatGPT sejam apropriadas e precisas.

Outra abordagem desejada é utilizar o Kibana para explorar mais os dados e tentar identificar padrões, tendo em vista que, por meio dele, é possível visualizar e analisar os dados. O Kibana permite explorar, visualizar e compartilhar dados armazenados em bancos de dados Elasticsearch. Com seus recursos avançados de visualização e filtragem, o Kibana facilita a extração de insights valiosos, a partir de grandes quantidades de dados.

Bibliografia

[1] Silvana Mara de Morais dos Santos e Leidiane Oliveira. "Igualdade nas relações de gênero na sociedade do capital: limites, contradições e avanços". Em: *Revista Katálysis* 13 (2010), pp. 11–19.

[2] Rebeca Contrera Ávila. "Minha História das Mulheres. São Paulo: Editora Contexto, 2008, 190 p. Michelle Perrot". Em: *História Social* 16 (2009), pp. 249–253.

[3] Michele Berleze e Belinda Silva Pereira. "O racismo nas redes sociais: o preconceito real assumido na vida virtual". Em: *4º Congresso Internacional de Direito e Contemporaneidade: Mıdias e Direitos da Sociedade em Rede*. 2017, pp. 1–14.

[4] Renália Rafaela Cunha Silva, Marcelo Nicomedes dos Reis Silva Filho e Antonio Carlos Santana de Souza. "A representação da mulher no mundo virtual: Percepções acerca do preconceito machista nas redes sociais". Em: *Revista de Estudos Acadêmicos de Letras* 9.01 (2016), pp. 55–69.

[5] Censo IBGE. "Instituto Brasileiro de Geografia e Estatıstica-IBGE". Em: *<https://www.ibge.gov.br/>* (2017). Accessed: 2022-9-28.

[6] Crimes cibernéticos: evolução da legislação brasileira - Portal do Conhecimento - Tribunal de Justiça do Estado do Rio de Janeiro. *Crimes cibernéticos: Evolução da Legislação brasileira*. Accessed: 2022-9-28. Jun. de 2019. URL: <http://conhecimento.tjrj.jus.br/noticias/noticia/-/visualizar-conteudo/5736540/6447772>.

[7] Karlla Gavazzoni-Melo. ""Não é preconceito, é só a minha opinião": a contrução discursiva da intolerância no Facebook". Em: *Anais do VIII SAPPIL-Estudos de Linguagem* (2017).

[8] Sean MacAvaney et al. "Hate speech detection: Challenges and solutions". Em: *PloS one* 14.8 (2019), p. 3.

[9] John T Nockleby et al. "Encyclopedia of the American constitution". Em: *Detroit, MI: Macmillan Reference* 3.2 (2000).

[10] Álvaro Paúl Dıaz. "La penalización de la incitación al odio a la luz de la jurisprudencia comparada". Em: *Revista chilena de derecho* 38.3 (2011), pp. 573–609.

[11] Roger Raupp Rios. *Direito da antidiscriminação: discriminação direta, indireta e ações afirmativas*. Livraria do Advogado Editora, 2008.

[12] WP Malecki et al. "Defining Online Hating and Online Haters". Em: *Frontiers in Psychology* 12 (2021).

[13] Sergio Gomes da Silva. "Preconceito e discriminação: as bases da violência contra a mulher". Em: *Psicologia: ciência e profissão* 30 (2010), pp. 556–571.

[14] Tatyana Alcantara Fernandes Casarino, Elisama Romero Quevedo e Tássia A Gervasoni. "A discriminação contra a mulher: análise histórica e contemporânea". Em: *Semana Acadêmica Fadisma Entrementes* (2014).

[15] Guacira Lopes Louro. *Gênero, sexualidade e educação*. Petrópolis: vozes, 1997.

[16] Isabela Pires Villas Boas de Carvalho. "Gênero e Violência Contra as Mulheres no Brasil: preconceito e discriminação–implicações para a psicologia". Em: (2020).

[17] *Formas de violência contra a mulher I: violência física*. www.trt4.jus.br, abr. de 2016. URL: <https://www.trt4.jus.br/portais/trt4/modulos/noticias/98747#:~:text=Nos%5C%20termos%5C%20da%5C%20Lei%5C%20Maria> (acesso em 07/02/2023).

[18] *Formas de violência contra a mulher III: violência sexual*. www.trt4.jus.br, abr. de 2016. URL: <https://www.trt4.jus.br/portais/trt4/modulos/noticias/98727> (acesso em 07/02/2023).

[19] Érico Tlaija Ramos. *Formas de violência contra a mulher II: violência psicológica*. www.trt4.jus.br, mai. de 2016. URL: <https://www.trt4.jus.br/portais/trt4/modulos/noticias/98737> (acesso em 07/02/2023).

[20] Érico Tlaija Ramos. *Formas de violência contra a mulher V: violência moral*. www.trt4.jus.br, abr. de 2016. URL: <https://www.trt4.jus.br/portais/trt4/modulos/noticias/98703> (acesso em 07/02/2023).

[21] Vanessa Oliveira Carvalho e Fábio Guilherme Ronzelli Murback. "Estudo da utilização das redes sociais digitais nas empresas brasileiras". Em: *Gestão & Conhecimento, Poços de Caldas* (2014), pp. 1–50.

[22] André Lemos. *Cibercultura: alguns pontos para compreender a nossa época*. 2003.

[23] Arthur de Alvarenga Barros, Michelle Fernanda Alves do Carmo e Rafaela Luiza da Silva. "A influência das redes sociais e seu papel na sociedade". Em: *Anais do Congresso Nacional Universidade, EAD e Software Livre*. Vol. 1. 3. 2012.

[24] Andréia Oliveira Almeida e Geisla Aparecida Alves de Moraes. "A influência das redes sociais na vida cotidiana: análise dos perfis fakes em redes sociais como forma de sociabilidade e entretenimento". Em: *Fundação Educacional do Município de Assis - Assis* (2010), p. 31.

[25] Abhishek Akshay Chaudhri, SS Saranya e Sparsh Dubey. "Implementation paper on analyzing COVID-19 vaccines on twitter dataset using tweepy and text blob". Em: *Annals of the Romanian Society for Cell Biology* (2021), pp. 8393–8396.

[26] Youness Madani, Mohammed Erritali e Belaid Bouikhalene. "Using artificial intelligence techniques for detecting Covid-19 epidemic fake news in Moroccan tweets". Em: *Results in Physics* 25 (2021), p. 104266.

[27] Renália Rafaela Cunha Silva, Marcelo Nicomedes dos Reis Silva Filho e Antonio Carlos Santana de Souza. "A REPRESENTAÇÃO DA MULHER NO MUNDO VIRTUAL: PERCEPÇÕES ACERCA DO PRECONCEITO MACHISTA NAS REDES SOCIAIS". Em: *Revista de Estudos Acadêmicos de Letras* 9.01 (2016), pp. 55–69.

[28] Rosane Leal da Silva et al. "Discursos de ódio em redes sociais: jurisprudência brasileira". Em: *Revista direito GV* 7 (2011), pp. 445–468.

[29] Twitter. *Twitter*. Accessed: 2023-2-07. Twitter.com, 2019. URL: <https://twitter.com/home>.

[30] Marcelo Augusto Araújo dos Reis. "Predição de comentários em mıdias sociais sobre discursos racistas". Em: *Trabalho de conclusão de curso (Bacharelado em Engenharia de Software) — Universidade de Brasília, Brasília* (2021).

[31] Issam El Naqa e Martin J Murphy. "What is machine learning?" Em: *machine learning in radiation oncology*. Springer, 2015, pp. 3–11.

[32] Pat Langley e Herbert A Simon. "Applications of machine learning and rule induction". Em: *Communications of the ACM* 38.11 (1995), pp. 54–64.

[33] Eiman Alothali et al. "Real Time Detection of Social Bots on Twitter Using Machine Learning and Apache Kafka". Em: *2021 5th Cyber Security in Networking Conference (CSNet)*. 2021, pp. 98–102. DOI: <10.1109/CSNet52717.2021.9614282>.

[34] Emma Cueva et al. "Detecting Fake News on Twitter Using Machine Learning Models". Em: *2020 IEEE MIT Undergraduate Research Technology Conference (URTC)*. 2020, pp. 1–5. DOI: <10.1109/URTC51696.2020.9668872>.

[35] Vladimir Nasteski. "An overview of the supervised machine learning methods". Em: *Horizons. b* 4 (2017), pp. 51–62.

[36] H Jabbar e Rafiqul Zaman Khan. "Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)". Em: *Computer Science, Communication and Instrumentation Devices* 70 (2015).

[37] Wil MP Van der Aalst et al. "Process mining: a two-step approach to balance between underfitting and overfitting". Em: *Software & Systems Modeling* 9.1 (2010), pp. 87–111.

[38] Warren S Sarle et al. "Stopped training and other remedies for overfitting". Em: *Computing science and statistics* (1996), pp. 352–360.

[39] Xue Ying. "An overview of overfitting and its solutions". Em: *Journal of physics: Conference series*. Vol. 1168. 2. IOP Publishing. 2019, p. 022022.

[40] Muhammad Usama et al. "Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges". Em: *IEEE Access* 7 (2019), pp. 65579–65615. DOI: <10.1109/ACCESS.2019.2916648>.

[41] Xiaojin Jerry Zhu. "Semi-supervised learning literature survey". Em: (2005).

[42] Richard S Sutton, Andrew G Barto et al. "Introduction to reinforcement learning". Em: (1998).

[43] Leslie Pack Kaelbling, Michael L Littman e Andrew W Moore. "Reinforcement learning: A survey". Em: *Journal of artificial intelligence research* 4 (1996), pp. 237–285.

[44] Youngjoong Ko. "How to use negative class information for Naive Bayes classification". Em: *Information Processing & Management* 53.6 (2017), pp. 1255–1268.

[45] Fasihul Kabir et al. "Bangla text document categorization using stochastic gradient descent (sgd) classifier". Em: *2015 International Conference on Cognitive Computing and Information Processing (CCIP)*. IEEE. 2015, pp. 1–4.

[46] Basant Subba e Prakriti Gupta. “A tfidfvectorizer and singular value decomposition based host intrusion detection system framework for detecting anomalous system processes”. Em: *Computers & Security* 100 (2021), p. 102084.

[47] Turki Turki e Sanjiban Sekhar Roy. “Novel Hate Speech Detection Using Word Cloud Visualization and Ensemble Learning Coupled with Count Vectorizer”. Em: *Applied Sciences* 12.13 (2022), p. 6611.

[48] BV Elasticsearch. “Elasticsearch”. Em: *Internet: https://www. elastic. co/pt/,[Sep. 12, 2019]* (2018).

[49] James Turnbull. *The Logstash Book*. James Turnbull, 2013.

[50] Yuvraj Gupta. *Kibana essentials*. Packt Publishing Ltd, 2015.

[51] Shane Ducksbury. “Elastic Beats and Where They Fit With ELK Stack”. Em: *<https://www.instaclustr.com/blog/elastic-beats-and-where-they-fit-with-elk-stack/>* (2021). Accessed: 2023-1-15.

[52] Flávio Mosafi. *Machine Learning: metodologia de mineração automatizada com dados das redes sociais e processamento de linguagem natural*. Editora Dialética, 2022.

[53] Erikson Júlio De Aguiar et al. “Análise de sentimento em redes sociais para a lıngua portuguesa utilizando algoritmos de classificação”. Em: *Anais do XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos*. SBC. 2018, pp. 393–406.

[54] Geovana Melo e Júlia Jamile. “Detection of hate speech against women”. Em: *<https://github.com/juliajamile/detection-of-hate-speech>* (2022). Accessed: 2023-01-09.

[55] Pratiksha Y Pawar e SH Gawande. “A comparative study on different types of approaches to text categorization”. Em: *International Journal of Machine Learning and Computing* 2.4 (2012), p. 423.

[56] William B Cavnar, John M Trenkle et al. “N-gram-based text categorization”. Em: *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval*. Vol. 161175. Las Vegas, NV. 1994.

[57] Sarah Cardoso. “Discurso de ódio nas redes sociais.” Em: *<https://jus.com.br/artigos/71639/discurso-de-odio-nas-redes-sociais>* (2019). Accessed: 2022-11-20.

[58] Felipe Carvalho. “Não parece, mas é machismo: 20 frases para não repetir mais”. Em: *<https://jus.com.br/artigos/71639/discurso-de-odio-nas-redes-sociais>* (2019). Accessed: 2022-11-20.

[59] Thiago Guimarães. “Onze coisas que as mulheres não aguentam mais ouvir no Brasil (e por quê)”. Em: *<https://www.bbc.com/portuguese/brasil-36522791>* (2016). Accessed: 2022-11-20.

[60] Twitter. “Developer Platform”. Em: *<https://developer.twitter.com/>* (2023). Accessed: 2022-11-20.

[61] Ryan McGrath. “Twython”. Em: *<https://twython.readthedocs.io/en/latest/>* (2013). Accessed: 2022-11-20.

[62] Twitter. “Search Tweets: Standard v1.1”. Em: *<https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets>* (2023). Accessed: 2022-11-20.

[63] Hassan Saif et al. “On stopwords, filtering and data sparsity for sentiment analysis of twitter”. Em: (2014).

[64] Nicolas Nathan Maia e Giancarlo Dondoni Salton. “Utilização de machine learning para classificação de sentimentos no idioma Portguês-Brasil Using machine learning for sentiment classification in the Brazilian-Portuguese language”. Em: *Brazilian Journal of Development* 8.6 (2022), pp. 43568–43580.

[65] Steven Bird, Ewan Klein e Edward Loper. “Nltk documentation”. Em: *Online: accessed April* (2008).

[66] Augusto Weiand. “Análise de sentimentos do Twitter com Naıve Bayes e NLTK”. Em: *Trajetória Multicursos* 7.2 (2018), pp. 3–18.

[67] *string.punctuation in Python*. Accessed: 2022-12-28. GeeksforGeeks, mai. de 2019. URL: <https://www.geeksforgeeks.org/string-punctuation-in-python/>.

[68] Sairamvinay Vijayaraghavan et al. “Fake news detection with different models”. Em: *arXiv preprint arXiv:2003.04978* (2020).

[69] Tiago Dias. *NLP com scikit-learn - Dados ao Cubo - Machine Learning - Python*. URL: <https://dadosaocubo.com/nlp-com-scikit-learn/> (acesso em 13/01/2023).

[70] Julie Beth Lovins. “Development of a stemming algorithm.” Em: *Mech. Transl. Comput. Linguistics* 11.1-2 (1968), pp. 22–31.

[71] Elastic. “Baixar e implantar o Elastic”. Em: *<https://www.elastic.co/pt/downloads>* (2023). Accessed: 2023-1-20.

[72] James Gowdy. “Como importar dados de CSV e de log para o Elasticsearch com o File Data Visualizer”. Em: *<https://www.elastic.co/pt/blog/importing-csv-and-log-data-into-elasticsearch-with-file-data-visualizer>* (2019). Accessed: 2023-1-20.

[73] *scikit-learn: machine learning in Python — scikit-learn 0.20.3 documentation*. Accessed: 2023-01-10. Scikit-learn.org, 2019. URL: <https://scikit-learn.org/stable/index.html>.

[74] Daniel Berrar. *Cross-Validation*. 2019.