Modelos de classificação para detecção de discursos de ódio nas redes sociais com Machine Learning: Regressão Logistica

July Werneck
Pontifícia Universidade Católica de Minas Gerais Belo Horizonte, Minas Gerais Email: jfmwerneck@sga.pucminas.br

Sofia Bhering
Pontifícia Universidade Católica de Minas Gerais Belo Horizonte, Minas Gerais Email: sbhering@sga.pucminas.br

***Abstract*—**

I. Introdução

Com as redes sociais crescendo exponencialmente, alcançando novas camadas sociais e abrangendo público de todas as idades, aumenta cada vez mais a interatividade das pessoas. Desta forma, é imprescindível as plataformas de mídias sociais compreenderem até onde a execução da liberdade de expressão fere a liberdade do outro e se torna algo desrespeitoso. Sendo assim, devemos ensinar a máquina de maneira que ela realize atividades mais correlacionada com a mente humana, para isso ultilizamos a Inteligência Artificial, que significa a inteligência demonstrada por máquinas ao executar tarefas complexas associadas a seres inteligente e especificamente a área de Aprendizado Profunda, ou melhor conhecida mundialmente como *Deep Learning*. A Deep Learning é um tipo de machine learning que treina computadores para realizar tarefas como seres humanos, o que inclui reconhecimento de fala, identificação de imagem e previsões, sendo assim usadas. As plataformas das mídias sociais proíbem a maioria dos comentários de ódio e de assédio, de extrema importância o combate de pré-conceitos em mídias sociais e o respeito em qualquer âmbito, lembrando sempre que qualquer atitude tomada deve sim, ter suas consequências. No entanto, aplicar essas regras de proibição e romper todos comentários de ódio, ainda é um problema que demanda resolução.No mundo acadêmico, as pesquisas abordadas para o assunto são geralmente modeladas por classificação supervisionada, a data timing que baseia-se em prever a categoria de uma observação dada. Aqui, procura-se estimar um "classificador" que gere como saída a classificação qualitativa de um dado não observado com base em dados de entrada (que abrangem observações com classificações já definidas). Na aplicação de discurso de ódio, eles são treinados usando um conjunto de dados, que são dados já previamente disponíveis para ultilização global de determinado tema e para o trabalho escolhemos o *Hate Speech and Offensive Language Dataset* , aplicado na linguagem Python e disponível no site *https://www.kaggle.com/* contendo comentários postados, acompanhados dos respectivos rótulos que descrevem a natureza do comentário como sendo ofensivo ou não, representado por Labels, sendo divididas por três demandas, sendo elas o discurso de ódio e retorna para a classificação 0, quando é ofensiva e discurso de ódio a label é classificada como 1 e quando não são ambas é classificada como 2. A principal dificuldade encontrada é o dataset ser processado em texto, sendo assim, tivemos que aprender toda o processo por fora e ser autodidático na construção do projeto.

II. Trabalhos Relacionados

A pesquisa e o desenvolvimento tratando-se de detecção e classificação de discursos de ódio envolve diversas áreas do conhecimento tanto computacional quanto ético. Essa preocupação embora recente, foi se tornando algo gigantesco presente em toda comudidade acadêmica e com esse debate foi criada uma área nova, chamada de FAT, que visa estudar sobre Equidade (Fairness), Transparência (Transparency) e Responsabilidade (Accountability) na área de algoritmos e inteligência artificial. Nesse sentido, busca-se descrever brevemente nos parágrafos seguintes o percurso e abordagens do tema. [Spertus Ellen., 1997] descreve um sistema de classificação para reconhecimento de mensagens hostis, codificando-as com base em sintaxe e semântica. [Sindhu and Sarang., 2020] abordam a classificação de texto de forma comparativa, explicitando conceitos e, utilizando 8 classificadores diferentes NB [12], SVM [14], KNN [15], DT [16], RF [13], AdaBoost [17], MLP [18] and LR [19]. [Fabio Del Vigna., 2017] apresenta comparativos de desempenho utilizando CNN com wang2vec e word2vec, também abordados por Sindhu. Além disso, aponta vantagens no uso de word embeddings treinados com textos do domínio da aplicação. O objetivo do presente trabalho, é conciliar e comparar os resultados alcançados por [Fabio Del Vigna., 2017] na utilização do CNN e, também na implementação dos demais modelos pontuados por [Sindhu and Sarang., 2020]. Busca-se, dessa forma, entender para datasets de tamanhos variados, qual a melhor alternativa de classificaçõ considerando os dados providos. Spertus Ellen (1997). "Smokey:

Automatic Recognition of Hostile Messages". In: IAAI-97 Proceedings. https://www.aaai.org/Papers/IAAI/1997/IAAI97-209.pdf. Claver Pari, Gustavo M., e José Gabriel R. (2019). "Avaliação de técnicas de word embedding na tarefa de detecção de discurso de ódio"In: https://sol.sbc.org.br/index.php/eniac/article/view/9354/9256.

III. Metodologia

A. CONFIGURAÇÕES:

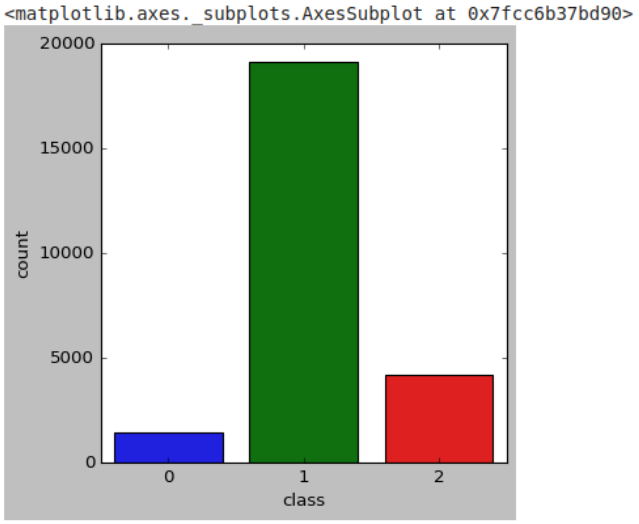
Nessa sessão do notebook importamos as dependencias para auxilio no desenvolvimento do projeto. Utilizamos a biblioteca pandas para compor a estrutura dos dados como DataFrames. As bibliotecas re, nltk e wordcloud foram especificamente para o que diz respeito ao processamento dos dados e visualização dos mesmos. Já a sklearn foi utilizada para construção do modelo escolhido pelo grupo.

B. PROCESSAMENTO DE DADOS:

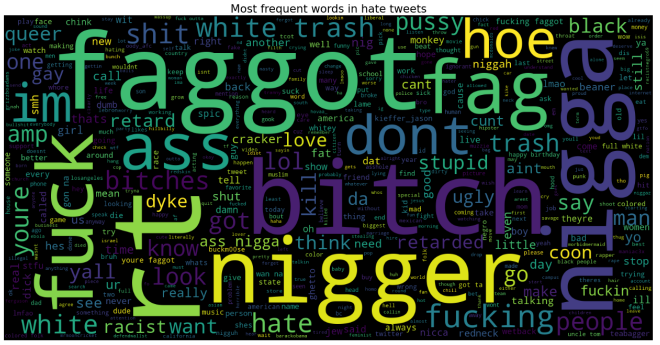
Nessa sessão do notebook focamos principalmente nos conceitos de NLP para processamento dos nossos dados. No primeiro momento carregamos os dados em DataFrame que nomeamos df. Em seguida, dividimos a estrutura para deixarmos apenas as colunas que são validas para nosso modelo, criando um novo dataframe que nomeamos de sub_df. O primeiro passo para processamento foi a limpeza de caracteres, visando considerar apenas palavras, retirando portanto de cada tweet os caracteres especiais, como @ e , além disso definimos todos os tweet como lowe case. O passo em especifico é construido atraves de um loop for com duas variaveis de interação, sendo uma i, o contador, e words, os tweets do dataframe. Ao final, atribuimos a coluna tweet do sub_df já tratada. Em seguida retiramos as linhas duplicadas através do metodo drop_duplicates e, para cada linha contida na coluna tweet do nosso dataframe, aplicamos a função criada tokenizationAndStopWords. A função tokenizationAndStopWords recebe o texto do tweet como parametro e utiliza a função word_tokenize da biblioteca nltk importada para transformar o tweet em tokens. Em seguida, utilizamos um laço for para retirar os tokens que são identificados como stop_words. A função retorna os textos separados por um espaço. O passo seguinte, utiliza a junção do metodo .apply na coluna tweet do sub_df e a lambda function para aplicar os conceitos de lemmatization, que consiste reduzir palavras semanticamente à sua raíz.

C. VISUALIZAÇÃO DOS DADOS:

Nessa sessão, buscamos visualizar e entender a ditribuição dos dados, levando em consideração a classe e a composição dos textos para cada classe. Nesse sentido, utilizamos a biblioteca matplotlib para visualizarmos a quantidade de registros distribuídos para cada classe.



Além disso, utilizamos a biblioteca wordCloud para construção de uma nuvem de frequencia de palavras, tanto para as bases cuja classificação é de tweet ofensivo, quanto para base cuja clasificação é discurso de ódio.



Em seguida, utilizamos o método TfidfVectorizer para construção de ngramas com os tweets.

D. MODELO:

Nessa sessão é onde definimos a nossa base de teste e treino e treinamos o modelo utilizando Regressão Logistica. O primeiro passo é dividir o nosso sub_df atribuindo a uma variável X o tweets e a uma variável Y as classes. Em seguida, utilizamos a função transform do TfidfVectorizer, que basicamente transforma uma coleção data raw em uma matriz de TD-IDF, uma medida estatistica que avalia quão relevante uma palavra é para o documento. Em seguida, utilizamos o metodo train_teste_split para dividir as bases, como citado previamente, setamos o test_size para 0.2 e o random_state em 42. O próximo passo consiste no uso da classe LogisticRegression, que é referenciada através da variável logreg, em seguida utilizamos o metodo .fit para treinar o modelo com as variaveis, x_train e y_train, previamente definidas. A predição acontece no passo seguinte, utilizamos o metodo .predict passando a nossa base de teste e atribuimos o resultado a uma variavel que logo em seguida é utilizada para informar a acuracia do modelo, isso é feito atraveis da função accuracy_score, informando além da variavel de predição, o resultado esperado,

y_test. As etapas finais, utilizam bibliotecas e recursos para avaliação do modelo que serão detalhadas a seguir.

IV. Experimentos

A. *DATASET:*

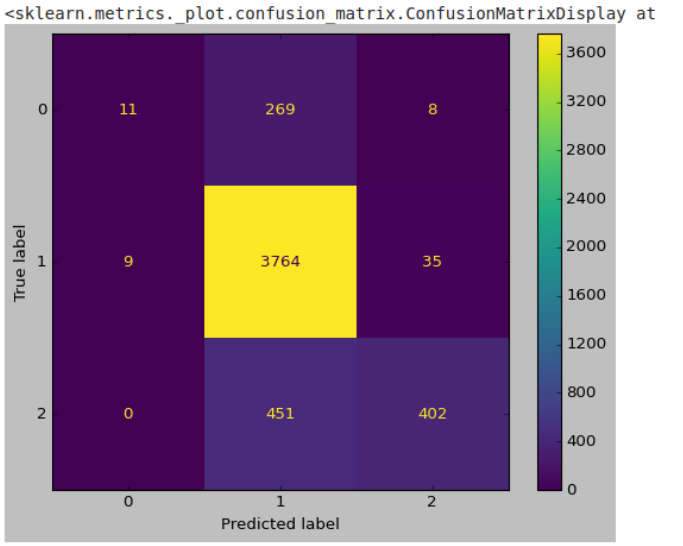
Os dataset escolhido para o trabalho foi o Hate Speech and Offensive Language Dataset disponibilizado na Kaggle (https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset). O dataset possui 24783 valores únicos. A base de dados é composta por uma coluna count que quantifica quantas pessoas usuárias do Crowd Flower classificaram o tweet, o valor minimo para esse campo é 3. As colunas seguintes são sublabels que quantificam os votos para cada uma das labels possíveis, sendo elas: - hate_speech: quantidade de usuários que julgaram o tweet como discurso de ódio; - offensive_language: quantidade de usuários que julgaram o tweet como ofensivo - neither: quantidade de usuários que julgaram o tweet como não sendo discurso de ódio e, nem ofensivo Finalmente, temos a coluna class que representa o label para a maioria dos votos computados. Sendo, 0 classificado como hate_speech, 1 como ofensa, e 2 como nenhum dos anteriores. A última coluna é o texto do tweet que foi avaliado nas colunas anteriores.

B. *MÉTRICAS DE AVALIAÇÃO:*

Para avaliar o modelo escolhido, utilizamos o metodo accuracy_score para medir a acuracia entre os dados preditos e o valor real. Além disso, utilizamos a biblioteca sklearn.metrics para gerar a matriz de confusão do modelo, com o metodo confusion_matrix, mostrando a quantidade de registros classificado para cada label. Ainda utilizando a biblioteca, usamos o metodo classification_report para obter as métricas de precisão, recall, f1-score e support, visando não a otimização mas a avaliação de diferentes aspectos de qualidade.

	precision	recall	f1-score	support
0	0.55	0.04	0.07	288
1	0.84	0.99	0.91	3808
2	0.90	0.47	0.62	853
accuracy			0.84	4949
macro avg	0.76	0.50	0.53	4949
weighted avg	0.83	0.84	0.81	4949

Utilizamos a função ConfusionMatrixDisplay para tornar a matriz de confusão mais visual e intuitiva.



V. Discussão

De forma geral, apesar do modelo ter apresentado uma acuracia relativamente satisfatória, quando analisamos o contexto e motivação do trabalho é evidente que, a criticidade é voltada para classificação de discursos de ódio. Nesse sentido, ao observar as métricas de avaliação para a classe que representava o hate_speech é possível observar que, a label está avaliada em 0.47 em recall e, analisando a matriz de confusão infere-se que mais da metade dos tweets classificados como discurso de ódio por usuários foi erroneamente classificado pelo modelo como ofensivos. O que evidencia uma das maiores limitações ao falarmos do uso de Machine Learning para detecção de discursos de ódio, principalmente em um contexto de redes sociais.

VI. Conclusão

A detecção de discursos de ódio em ambientes virtuais é ponto vital para manutenção e garantia de um ambiente seguro, nesse sentindo é inevitável a atribuição de responsabilidade aos cientistas da computação e a todos os profissionais relacionados a tecnologia na manutenção e dedicação para fomentar este ambiente. A dificuldade na detecção e curadoria de discursos de ódio em redes sociais é uma questão atual e pertinente, podendo impactar de diversas formas na construção da sociedade e até mesmo nas decisões politicas. Ao permitirmos um ambiente virtual que infringe a legislação e sai impune, estamos indiretamente fomentando e permitindo a crescente onda de discriminação e ódio. Acredito que um ponto de partida para futuras pesquisas no que se refere a classificação de discursos de ódio seria limitar a base à labels de ofensas e hate_speech, uma vez que a maior dificuldade do modelo apresentado foi na diferenciação das duas classificações. Além disso, acredito ser um próximo passo a otimização do modelo a escolha do algoritmo solver, utilizando por exemplo o 'newton-cg' disponível na biblioteca sklearn.

REFERENCES

[1] Spertus Ellen (1997). “Smokey: Automatic Recognition of Hostile Messages”. In: IAAI-97 Proceedings. https://www.aaai.org/Papers/IAAI/1997/IAAI97-209.pdf.

[2] Claver Pari, Gustavo M., e José Gabriel R. (2019). “Avaliação de técnicas de word embedding na tarefa de detecção de discurso de ódio” In: https://sol.sbc.org.br/index.php/eniac/article/view/9354/9256.

[3] Fabio Del Vigna, Andrea Cimino, Felice Dell’Orletta, Marinella Petrocchi, and Maurizio Tesconi (2017) In: Proceedings of the First Italian Conference on Cybersecurity (ITASEC17) https://www.researchgate.net/publication/316971988Hate_me_hate_me_not_Hate_speech_detection_on_Facebook

[4] Sindhu Abro, Sarang Shaikh, Zafar Ali, Sajid Khan, and Ghulam Mujtaba (2020). ”Automatic Hate Speech Detection using Machine Learning: A Comparative Study” In: (IJACSA) International Journal of Advanced Computer Science and Applications https://pdfs.semanticscholar.org/0445/07a2f4d0030c05434eceb0230c40f868804d.pdf

[5] Kavita Ganesan (2020). ”What are Stop Words?” In: https://kavitaganesan.com/what-are-stop-words/

[6] ”What is Tokenization?” In: https://www.tokenex.com/resource-center/what-is-tokenization

[7] Igor Nascimento Alves (2021). ”Lemmatization vs. stemming: quando usar cada uma?” In: https://www.alura.com.br/artigos/lemmatization-vs-stemming-quando-usar-cada-uma