Hate speech detection using feature vectors applied to a new comment base in Portuguese

Detecção de discurso de ódio utilizando vetores de *features* aplicados a uma base nova de comentários em português

Peter Dias Paiva
Universidade Federal do Piauí (UFPI)
peterdias130@gmail.com

Vanecy Matias da Silva
Universidade Federal do Piauí (UFPI)
vanecy@ufpi.edu.br

Raimundo Santos Moura
Universidade Federal do Piauí (UFPI)
rsm@ufpi.edu.br

ABSTRACT

Fighting hate speech on the Internet has become a big challenge. In this sense, we propose a solution to detect offensive comments that seek to spread hatred on the network. The proposal uses feature vectors, applied to a comment base in Portuguese generated by the work itself, being composed of posts taken from a news site and from Twitter. The results show the viability of the proposal and that it can be used as a basis for the development of new applications.

RESUMO

Combater discurso de ódio na Internet tem se tornado um grande desafio. Neste sentido, propõe-se uma solução para detectar comentários ofensivos que buscam disseminar ódio na rede. A proposta utiliza vetores de *features*, aplicados a uma base de comentários em português gerada pelo próprio trabalho, sendo composta de postagens retiradas de um site de notícias e do *Twitter*. Os resultados mostram a viabilidade da proposta e que ela pode ser utilizada como base para o desenvolvimento de novas aplicações.

CCS Concepts

•Computing methodologies→Artificial intelligence→Natural language processing→Speech recognition •Computing methodologies→Machine learning→Machine learning algorithms→Feature selection

Keywords

hate speech; natural language processing; artificial intelligence

Palavras-chave

discurso de ódio; processamento de linguagem natural; inteligência artificial

1. INTRODUÇÃO

A Internet tem sido palco de debates e de manifestações de opiniões pessoais sobre diversos assuntos. Mas como era de se esperar, nem sempre isso ocorre de maneira civilizada e saudável. Muitas pessoas insistem em disseminar ódio através de comentários ofensivos. Casos famosos de ataques em redes sociais, como o ocorrido recentemente com algumas celebridades, evidenciam isso cada vez mais [25]. A ineficiência de barrar atitudes desse tipo na Internet é bastante clara.

Discurso de ódio, segundo Brown [4] é qualquer forma de comunicação que tem o intuito de ofender uma pessoa ou grupo baseando-se em alguma de suas características, como raça, cor, etnia, gênero, nacionalidade, orientação sexual e outros. Detectar discurso de ódio na Web analisando comentários de usuários possibilitará que as redes sociais e os sites em geral alertem o que está sendo postado por seus usuários, registrando casos como o que foi citado anteriormente.

Determinar de forma automática se um comentário é ofensivo ou não, é um problema já conhecido e bastante estudado na Inteligência Artificial (IA), trata-se de uma tarefa de classificação. Algoritmos de Aprendizagem de Máquina têm obtido êxito em cenários desse tipo, devido à capacidade de "aprender" a partir de exemplos reais obtidos do ambiente em que irão atuar, de forma que padrões possam ser reconhecidos [12].

Utilizar classificação de textos torna possível elucidar problemas complexos, como o explorado neste trabalho, que busca determinar se o autor de um texto tem a intenção de ofender alguém. Dessa forma, tendo como base todos esses conceitos, propõe-se um algoritmo capaz de detectar discurso de ódio utilizando uma base de dados de comentários em português.

Além desta seção introdutória, o restante do trabalho está organizado como segue. A Seção 2 expõe a base teórica. A Seção 3 lista as principais tecnologias empregadas. A Seção 4 descreve alguns trabalhos sobre detecção de discurso de ódio em redes sociais. A Seção 5 explica a abordagem proposta, descrevendo cada passo, desde o pré-processamento até o uso de vetores de *features*. A Seção 6 exibe os experimentos realizados e os resultados obtidos. A Seção 7 termina com as considerações finais e conclusões.

2. REFERENCIAL TEÓRICO

2.1 Inteligência artificial

Definir o que é Inteligência Artificial (IA) é algo um tanto complexo, dada a sua imensa variedade de subcampos, que vão desde aplicações mais gerais capazes de aprender e perceber o mundo a sua volta, até as mais específicas, como sistemas especialistas que fazem diagnósticos de doenças. Em linhas gerais, a IA busca construir entidades inteligentes [23].

2.1.1 Aprendizagem de Máquina

A Aprendizagem de Máquina (Machine Learning) é uma subárea da IA que foca no desenvolvimento de algoritmos inteligentes que aprendem com a experiência, melhorando gradualmente seu desempenho a partir do contato com novas informações. Nesse âmbito, de acordo com Haykin [11], há duas categorias fundamentais: a Aprendizagem Supervisionada e a Aprendizagem Não-Supervisionada. Na primeira, o sistema tem conhecimento do ambiente em que irá atuar através de um conjunto de informações do tipo entrada-saída, aprendendo a mapear os padrões de entrada para as respectivas saídas. Na segunda, não há dados rotulados, a aprendizagem ocorre através da criação de representações internas baseadas nas características das entradas.

2.1.1.1 Técnicas de Aprendizagem de Máquina

2.1.1.1.1 Naive Bayes

Consiste em atribuir uma classe do problema a uma dada instância x a partir da análise do histórico dos dados, encontrando a classe com maior probabilidade de atribuição calculada através da fórmula de Bayes [22], definida na Figura 1.

$$P(classe|x) = \frac{P(x|classe) * P(classe)}{P(x)}$$

Figura 1. Equação de *Bayes.*

P(classe|x) é a probabilidade da classe dada a instância x, chamada de probabilidade posterior. P(x|classe) é a probabilidade de x dado que a classe é verdadeira. P(classe) é a probabilidade da classe ser verdadeira, chamada de probabilidade anterior. P(x) é a probabilidade de x.

Um classificador *Multinomial Naive Bayes* representa as amostras observadas através de vetores de *features* inteiras, onde cada classe assume uma distribuição multinominal. O *Bernoulli Naive Bayes* é semelhante a ele, diferindo na representação dos dados, pois faz uso de vetores de *features* binárias, e as classes são vistas como uma distribuição binomial [13].

2.1.1.1.2 Regressão Logística

É um método estatístico que calcula a probabilidade de um dado evento ocorrer ou não. O modelo define uma fronteira, através de uma curva sigmoide, que separa dois conjuntos, como demonstrado na Figura 2. Os valores da variável chamada independente (eixo x), são utilizados para calcular a variável dita dependente (eixo y), que assume valores entre 0 e 1 referentes a probabilidade da amostra pertencer a uma das duas classes [15].

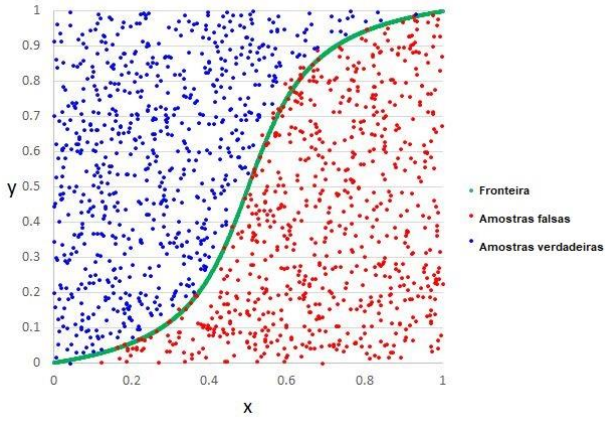

Figura 2. Exemplo de Regressão Logística.

2.1.1.1.3 SGD (Stochastic Gradient Descent)

O processo de classificação pode ser visto como a determinação de uma função f que mapeia amostras para uma determinada classe. Uma função de perda associa um custo às inferências erradas nesse tipo de atividade. Logo, surge a necessidade por um coeficiente de f que minimize a margem de custo. O Gradiente Descendente (GD) é um algoritmo que busca cumprir essa missão, testando diferentes valores para o coeficiente em questão e analisando o custo para cada classificação do conjunto de treinamento. Por ser uma tarefa dispendiosa, já que o valor do coeficiente só é atualizado depois de cada iteração, onde toda a base de treinamento foi percorrida, o SGD surge como uma solução para esse problema, atualizando o coeficiente para cada amostra do conjunto e não no fim das iterações [6].

2.1.1.1.4 Métodos de Agrupamento

Os métodos citados anteriormente, selecionam apenas um elemento do conjunto de hipóteses. Métodos de aprendizagem por agrupamento, selecionam um conjunto de hipóteses do espaço e combinam suas previsões [7]. São utilizados dados de treinamento diferentes ou algoritmos diferentes para gerar múltiplas definições de um conceito e assim gerar um modelo combinado.

Random Forest é um classificador de agrupamento que utiliza estruturas de dados do tipo árvore, que crescem de forma aleatória, onde seus nós internos representam testes dos atributos de entrada e as folhas contém as informações de retorno, que classificam a instância que está sendo testada. A randomização pode ser inserida no crescimento de cada árvore, dividindo o conjunto de treinamento [3]. Inferências são realizadas utilizando os caminhos da árvore, partindo da raiz até um nó-folha, ou seja, até encontrar uma classificação para a amostra observada.

Outro algoritmo de agrupamento é o *Gradient boosting,* que utiliza uma combinação de classificadores fracos, geralmente árvores de decisão, onde cada um deles é adequado para uma versão modificada, de maneira aleatória, do conjunto de dados original [9].

2.1.2 Processamento de Linguagem Natural (PLN)

Processamento de Linguagem Natural (PLN) é um ramo da IA que busca tratar a comunicação humana de forma computacional, seja ela escrita ou falada. De acordo com Gonzalez and Lima [10], PLN tem a intenção de utilizar o computador para se comunicar de forma humana, geralmente focando apenas em alguns níveis da

comunicação. Na linguagem escrita, a extração do significado de uma sentença particular, é obtido através da análise de sua estrutura, observando a construção das palavras, suas relações entre si e com seus significados. Ainda é possível realizar uma análise pragmática, que verifica o contexto em que a frase está inserida, a fim identificar seu significado de forma correta.

2.1.2.1 Análise de Sentimento

É uma área que busca analisar e identificar o comportamento humano, seja ele sentimentos, emoções, atitudes, etc [18].

A proliferação de opiniões, recomendações e outras formas de expressão, principalmente em redes sociais, têm atraído bastante atenção para esse âmbito, em especial de empresas interessadas em conhecer a opinião pública sobre seus produtos e serviços e com isso adaptar-se ao mercado.

A utilização de PLN na análise de sentimento de textos, possibilita reconhecer características psicológicas do autor, tendo inúmeras aplicações, desde sistemas reconhecedores de opiniões até programas que podem diagnosticar doenças mentais, como depressão e ansiedade.

3. Ferramentas

Aqui são listadas as principais ferramentas que foram utilizadas na pesquisa.

3.1 Python

Python é uma linguagem de programação de alto nível com sintaxe clara, o que facilita bastante o seu uso e o aumento da produtividade. Possui diversas estruturas de alto nível prontas para serem utilizadas, e caso haja necessidade por mais funcionalidades, novos módulos e *frameworks* desenvolvidos por terceiros podem ser facilmente instalados [2].

3.2 Natural Language Toolkit (NLTK)

Segundo seu site oficial, o NLTK é uma plataforma criada para trabalhar com PLN utilizando a linguagem de programação *Python*. São fornecidos diversos recursos como corpora e bibliotecas, para processamento da linguagem escrita, possibilitando classificação, tokenização, analise semântica, etc.

3.3 Scikit-learn

É uma biblioteca de aprendizagem de máquina de código aberto que contém implementações de algoritmos famosos de classificação, regressão e agrupamento. Possui uma interface amigável e é totalmente integrada ao *Python* [20]. Possibilita realizar a importação de conjuntos de dados para treinamento e a recuperação dos resultados da fase de testes, de maneira simples.

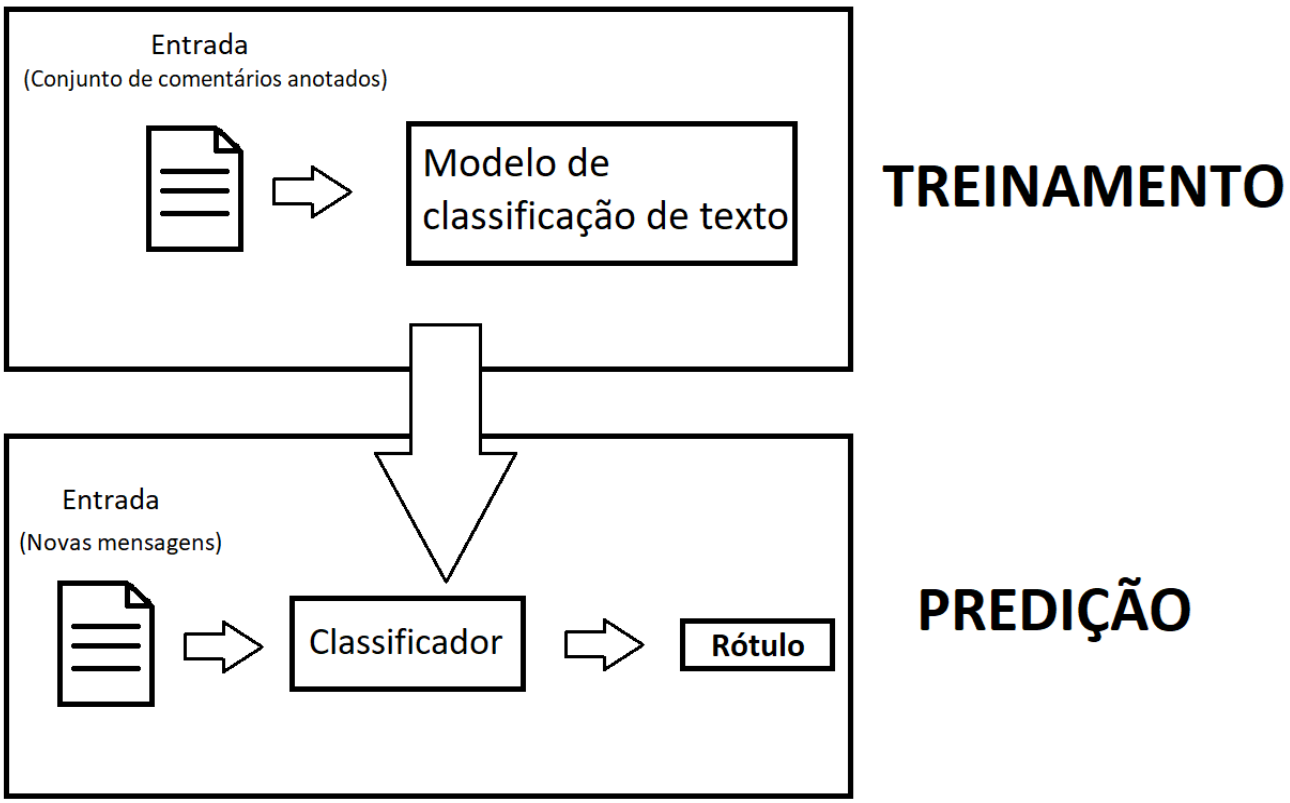

Figura 3. Esquema da abordagem proposta.

4. TRABALHOS RELACIONADOS

Dois trabalhos geraram sua própria base dados de comentários em português. Pelle and Moreira [21], recolheram postagens de usuários de um site de notícias brasileiro e geraram um conjunto de comentários chamado de OffComBr. Os autores utilizaram *Bag of Words* para extrair características dos textos e fizeram uso dos classificadores *Naive Bayes* e *SVM*. Já Fortuna [8], extraiu *tweets* da rede social *Twitter* e utilizou um sistema de classificação hierárquica com SVM Linear.

Silva e Serapião [24] detectaram discurso de ódio em português usando vetores de palavras com funções otimizadoras do gradiente de descida, combinados com CNN (Convolutional Neural network). Lima e Dal Bianco [17] utilizaram o algoritmo de classificação KNN (*K-Nearest Neighbors*) para extração de características, a partir de meta-atributos retirados da vizinhança de cada texto. Os dois trabalhos fizeram uso do OffComBR, sendo que o primeiro deles obteve melhores resultados.

Badjatiya et all [1] realizaram diversos testes em uma base com 16 mil *tweets* em inglês divididos em múltiplas classes (racismo, sexismo, etc). Na primeira parte dos experimentos foram utilizados n-gramas, *Bag of Words* e *TF-IDF*. Depois os autores utilizaram DNN (*Deep Neural Networks*) aliada a vetores de palavras.

Como contribuição do presente trabalho, pode-se destacar: a geração de um novo conjunto de comentários em português, que poderá auxiliar a comunidade científica em pesquisas futuras, pois será disponibilizado para uso público; a investigação do uso de vetores de *features*, levando em conta a corretude das palavras e a presença de palavrões e xingamentos em cada comentário.

5. ABORDAGEM PROPOSTA

A abordagem consiste em analisar um comentário e classificá-lo como sendo discurso de ódio ou não. Para isso, iremos treinar o modelo com um conjunto de dados anotados e, depois, iremos classificar novos comentários de acordo com ele. A Figura 3 esquematiza a ideia geral da proposta.

5.1 Conjunto de Dados

O conjunto de dados utilizado para treinamento e validação, consistiu de uma versão modificada do OffComBR, um *dataset* criado por Pelle and Moreira [21] que contém comentários ofensivos e não-ofensivos, retirados de um portal de notícias brasileiro.

O OffComBR possui duas versões: o OffComBR-2 e o OffComBR-3. O primeiro possui ao todo 1250 comentários, sendo 831 não-ofensivos e 419 ofensivos, que segundo os autores, foram avaliados por pelo menos dois juízes. Já o OffComBR-3, possui 1033 comentários, com 831 não-ofensivos e 202 ofensivos, determinado por três juízes. Apesar de ter sido originalmente dividido em categorias (racismo, sexismo, homofobia, xenofobia, intolerância religiosa, entre outras), o conjunto disponibilizado pelos autores está classificado apenas de forma binária (ofensivo e não-ofensivo).

Como é possível perceber, os conjuntos apresentados acima contêm um problema, a quantidade de comentários não-ofensivos supera, e muito, a de ofensivos. Em tarefas de classificação, essa situação pode gerar resultados controversos, já que a classe com menor número de amostras é mais suscetível a ser classificada de forma incorreta [19]. Por esse motivo, e por já ter sido explorado em três trabalhos relacionados, optou-se por gerar um novo conjunto de dados, tendo como base o OffComBr.

Inicialmente, as duas versões do *dataset* foram unidas em um só arquivo, excluindo-se comentários repetidos. Realizou-se, então, uma reclassificação manual com três juízes, resultando em 822 documentos não-ofensivos e 428 ofensivos.

Novos comentários foram retirados do *Twitter*, obtidos principalmente, de *hashtags* e de perfis de cunho político. Ao todo, um pouco mais de 1 mil *tweets* foram extraídos e classificados manualmente por dois juízes, determinando 391 como ofensivos. Como o objetivo era balancear o número de amostras das duas classes, os outros *tweets* foram descartados e os classificados como ódio, agregados ao arquivo descrito no parágrafo anterior, tendo agora 822 textos não-ofensivos e 819 ofensivos.

5.2 Pré-processamento

Para extrair informações mais precisas dos dados, foi necessário prepará-los, excluindo ruídos e informações consideradas desnecessárias, tudo isso feito de forma algorítmica. A Figura 4 abaixo ilustra o passo-a-passo desta fase através de um exemplo.

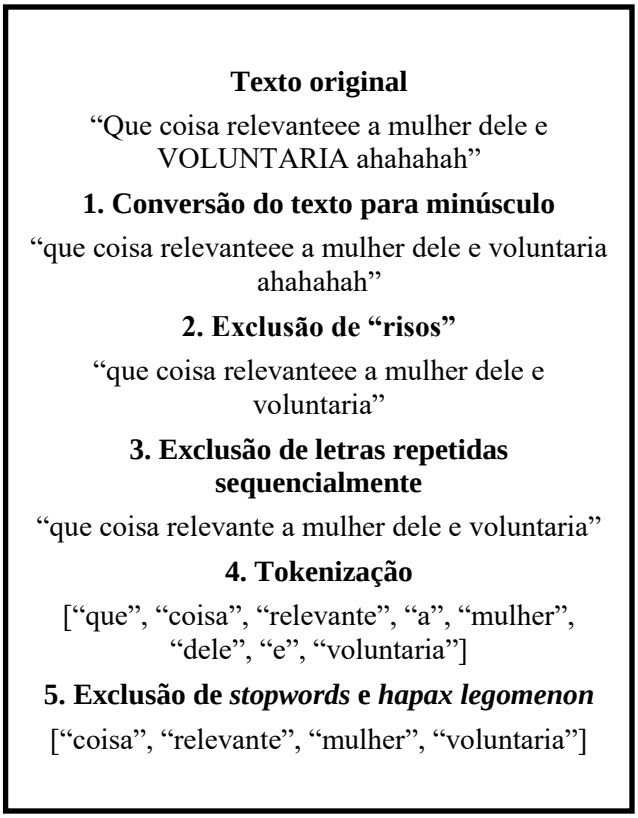

Figura 4. Exemplo de pré-processamento aplicado em um comentário

Inicialmente, o texto é convertido para minúsculo. Em seguida, palavras que simbolizam "risadas" são excluídas tendo como base uma pequena lista (construída de forma empírica) de termos comuns do linguajar da Internet, como por exemplo, "kkk" e

"hahah". Depois, são deletadas letras repetidas em palavras, tendo o cuidado de verificar, através do uso de um dicionário [16], se realmente se trata de um erro de grafia, já que o português possui termos com letras duplicadas. Após isso, é realizada a quebra da sentença em uma lista de palavras, chamada de tokenização. Por fim, *stopwords* e *hapax legomenon* são detectados e removidos, já que palavras com frequência muito alta e muito baixa não são úteis em uma tarefa de classificação [14].

5.3 Vetores de *Features*

Nesta etapa, dividida em duas fases, as informações pertinentes foram extraídas, cada comentário passou a ser representado através de um vetor de *features*, buscando elencar as informações que descrevem o conteúdo de um comentário e baseado nesta representação determinar se ele é ofensivo ou não.

5.3.1 Features básicas

Ao todo foram utilizadas três *features* básicas, sendo elas:

- A quantidade de palavras de baixo calão presentes no comentário. Para identificação de tais palavras utilizou-se uma lista que contém palavrões e xingamentos comuns da Internet, como por exemplo, "idiota", "burro", "vadia", entre outras. Vale ressaltar que a mesma foi criada de maneira empírica.
- A quantidade de palavras do comentário;
- A quantidade de palavras corretas do comentário, considerando o dicionário do projeto VERO [16].

Na figura 5, os números do vetor correspondem aos valores de cada uma das *features* citadas acima, respectivamente.

5.3.2 Bag of Words

A premissa do Bag of Words é representar cada comentário através de um vetor, onde cada um de seus elementos sinaliza se uma determinada palavra está presente ou não nele [5].

Dessa forma, um vocabulário (conjunto de palavras) é aprendido automaticamente dos próprios documentos e utilizado como *feature*. O intuito é determinar o grau de similaridade, já que textos parecidos terão um certo número de palavras em comum. A Figura 6 mostra o exemplo da Figura 5 após aplicação de *Bag of Words*.

6. EXPERIMENTOS E RESULTADOS

A Tabela 1, contém os melhores resultados dos trabalhos relacionados que utilizaram o OffComBr como base de dados.

Comentário pré-processado

["coisa", "relevante", "mulher", "voluntaria"]

Vetor de *features* básicas do comentário acima

[0, 4, 4]

Figura 5. Exemplo de vetor de *features* básicas.

Comentário representado por vetor de *features* básicas

[0, 4, 4]

Vetor de *features* do comentário acima com *Bag of Words*

[0, 4, 4, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, ...]

Figura 6. Exemplo de *Bag Of Words.*

Tabela 1. Resultados obtidos pelos trabalhos relacionados que utilizaram o OffComBr.

Autor	Conjunto de Dados	Algoritmo	F1-score	Accuracy
Pelle and Moreira (2017)	OffComBr-2	SVM	0,77	-
Pelle and Moreira (2017)	OffComBr-3	SVM	0,82	-
Silva e Serapião (2018)	OffComBr-2	CNN	0,89	0,83
Silva e Serapião (2018)	OffComBr-3	CNN	0,96	0,93
Lima e Dal Bianco (2019)	OffComBr-2	KNN	0,72	-
Lima e Dal Bianco (2019)	OffComBr-3	KNN	0,85	-

Pelle e Moreira [21] utilizaram apenas o *f1-score* como métrica, enquanto que Silva e Serapião [24] calcularam tanto o *f1-score* como a *accuracy*, obtendo valores superiores em relação aos do primeiro trabalho. É válido lembrar que tais métricas correspondem a classificação de um conjunto desbalanceado (o OffComBr original).

Para avaliar o modelo proposto, aplicado à base de dados gerada, foram utilizados seis algoritmos: *BernoulliNB, MultinomialNB, GradientBoosting, RandomForest, SGD* e *LogisticRegression*. Além do *f1-score* e da *accuracy*, *precision* e *recall* também foram calculadas. Os valores médios de cada uma delas encontram-se na Figura 7.

A configuração utilizada para gerar tais resultados foi a seguinte:

- Para compor o vocabulário do *Bag of Words*, foram considerados unigramas (palavras isoladas) e bigramas (pares de palavras), possibilitando analisar não apenas as palavras utilizadas, como também o contexto em que estão inseridas. Foi necessário desconsiderar os n-gramas (conjuntos de palavras) que estavam presentes em mais de 80% dos comentários, pois apesar de terem alta frequência, não foram excluídos na fase de pré-processamento pelo fato de não estarem contidas na lista de *stopwords* utilizada. Assim, o vocabulário foi composto de 2776 *features*.
- 80% dos comentários foram destinados para treinamento e 20% para testes.

O *BernoulliNB* obteve os melhores resultados, ficando empatado com o *MultinomialNB* nas métricas *recall, f1-score* e *accuracy*. Apesar do *GradientBoosting* ter tido o pior desempenho, seus valores não foram muito inferiores em relação aos mais altos. Em vista disso, é interessante analisar esses dois extremos do modelo, tanto o pior como o melhor.

A matriz de confusão da Figura 8, mostra o comportamento do *BernoulliNB* na etapa de testes. Olhando a imagem de baixo para cima, é possível perceber como as instâncias foram classificadas pelo algoritmo, sendo 187 como não-ofensivas e 142 como ofensivas. Visualizando da esquerda para a direita, verifica-se a classificação correta das amostras.

Constata-se que dos 187 comentários marcados como não-ofensivo, 150 realmente são, enquanto que 37 foram classificados erroneamente, pois são ofensivos. Do total de 142 considerados ofensivo, houve equívoco em apenas 15, o restante está na classe correta. Em resumo, houve acerto em 80% das inferências da classe não-ofensivo e em 90% da classe ofensivo.

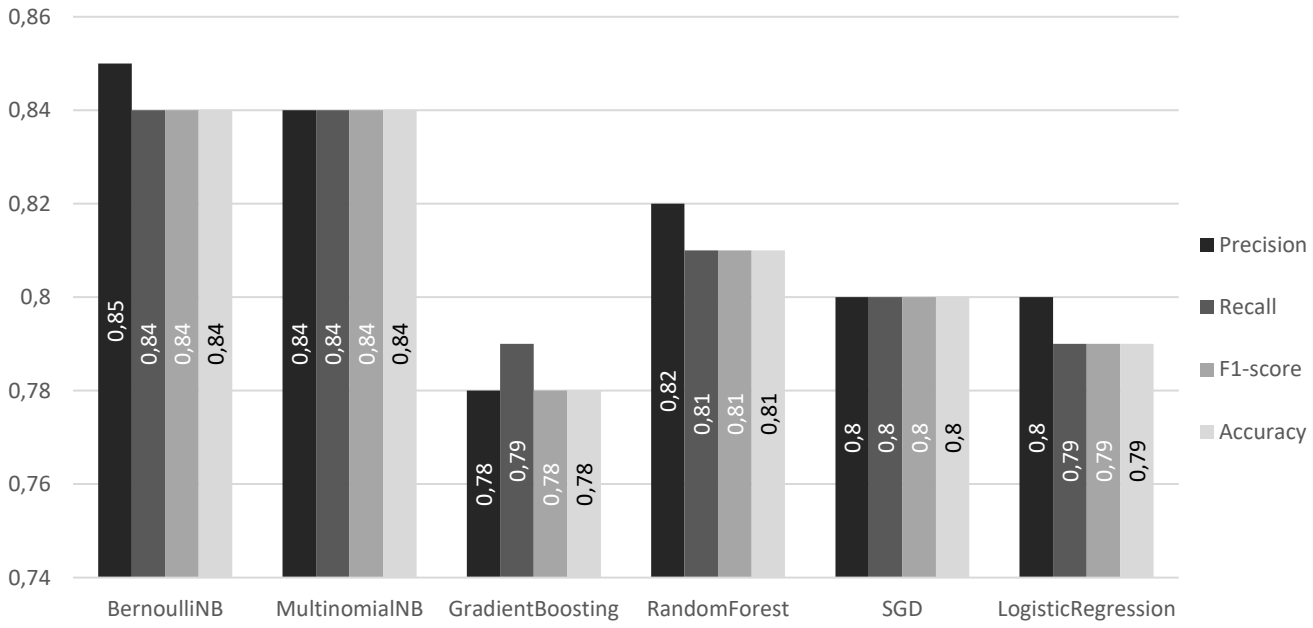

Figura 7. Resultados das métricas de cada um dos algoritmos utilizados.

Na matriz de confusão do *GradientBoosting* (Figura 9), para o mesmo conjunto de testes, percebe-se que houve uma queda na taxa de acertos, principalmente na classe ofensivo. 73% das amostras classificadas como não-ofensivo estavam corretas, divergindo 7% em relação ao *BernoulliNB*. Já no grupo de ofensivos, 85% foi inferido sem erros, gerando uma diferença de 5% do valor do *BernoulliNB*. Assim, o *GradientBoosting* obteve as piores métricas em meio aos algoritmos testados, principalmente, porque não conseguiu reconhecer 17 comentários como ofensivos.

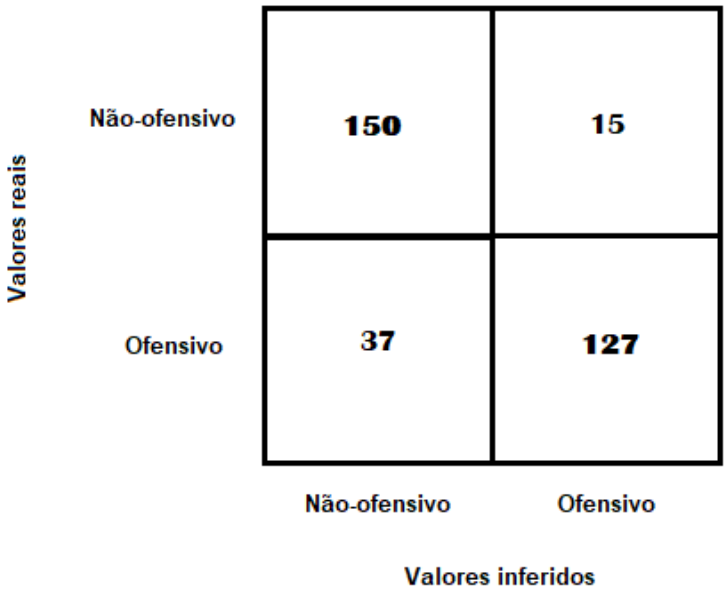

Figura 8. Matriz de confusão do *BernoulliNB*.

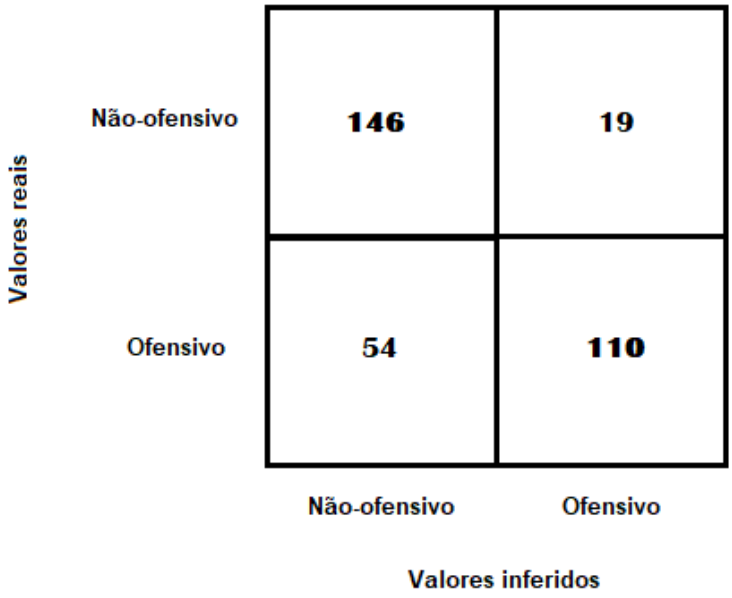

Figura 9. Matriz de confusão do *GradientBoosting*.

As figuras 10 e 11 apresentam a curva ROC do *BernoulliNB* e do *GradientBoosting*, respectivamente. Esse tipo de gráfico mostra o quão bem um classificador consegue distinguir entre as duas classes do problema.

O AUC resume toda a informação visual em um número que corresponde a área sob a curva ROC e demonstra a probabilidade do modelo conseguir diferenciar entre os conjuntos.

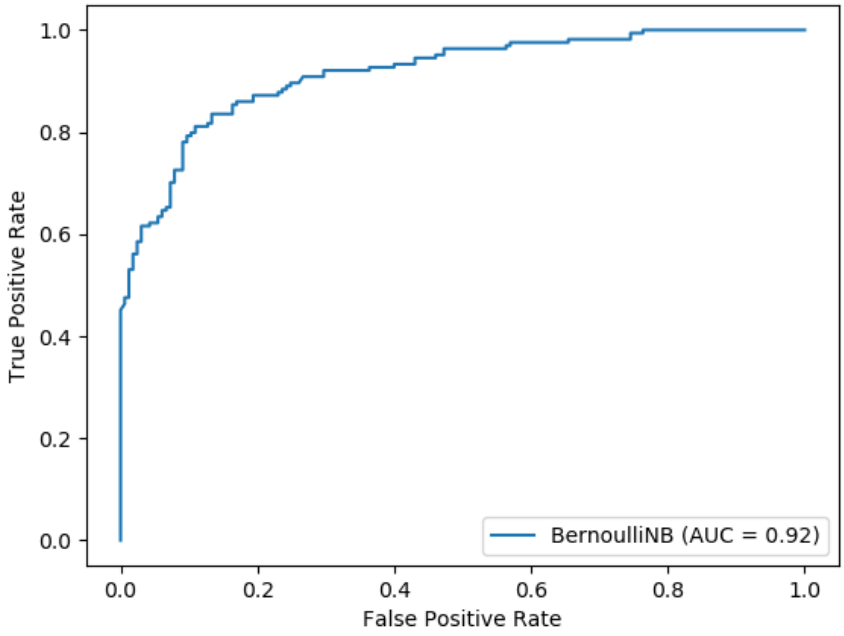

Figura 10. Curva ROC do *BernoulliNB*.

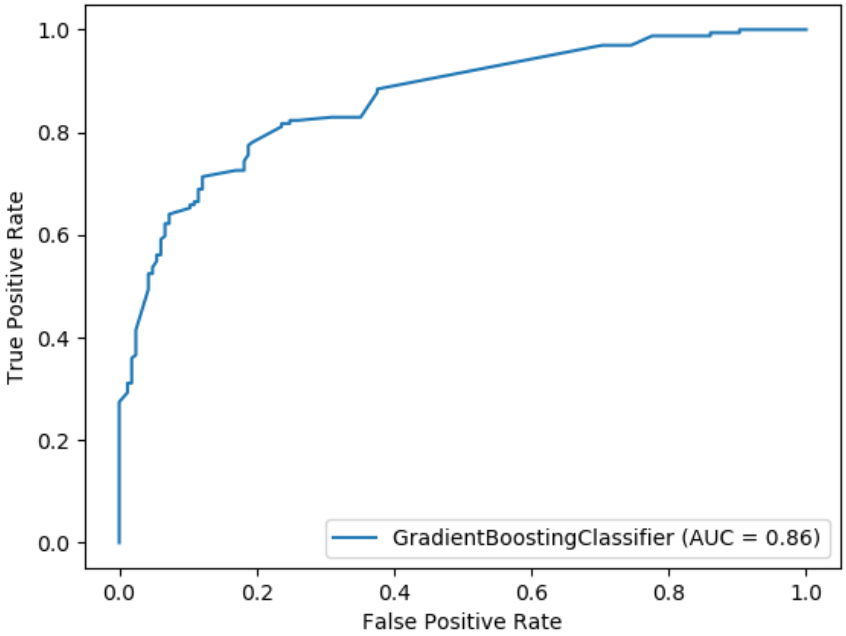

Figura 11. Curva ROC do *GradientBoosting*.

Quanto maior o AUC, mais preciso é o algoritmo, sendo desejáveis valores bem próximos ou até iguais a 1.

O *BernoulliNB* atingiu um AUC de 0,92, ao passo que o *GradientBoosting* alcançou 0,86, números que podem ser considerados bons e que comprovam a efetividade do método exposto.

7. CONCLUSÃO E TRABALHOS FUTUROS

Após analisar os resultados, fica evidente que a abordagem proposta obteve êxito. As métricas dos dois melhores classificadores, *BernoulliNB* e *MultinomialNB*, foram quase iguais, diferenciando-se apenas pela *precision* (o primeiro com 0.85 contra 0.84 do segundo). Os valores retornados pelo pior classificador observado, o *GradientBoosting*, não foram muito inferiores.

As matrizes de confusão do pior e do melhor algoritmo, mostraram uma certa variação na porcentagem de acertos entre as duas classes consideradas, sendo 10% (90% da classe ofensivo e 80% da não-ofensivo) para o *BernoulliNB* e 12% (85% da classe ofensivo e 73% da não-ofensivo) para o *GradientBoosting*. Isso pode ser um indicativo de que as *features* escolhidas não foram o bastante para determinar o que é um comentário não-ofensivo. Pode demonstrar também, que as instâncias dessa classe no conjunto de dados não foram muito bem classificadas pelos juízes, sendo necessária uma revisão, futuramente.

Partindo das considerações do parágrafo anterior, espera-se que a base gerada seja utilizada em investigações futuras, e quem sabe, até modificada, caso realmente seja detectado algum problema na mesma. Outras *features* também podem ser extraídas dos documentos, com o intuito de balancear a taxa de acertos entre as duas classes.

8. REFERÊNCIAS

[1] Badjatiya, Pinkesh et al. Deep learning for hate speech detection in tweets. In: **Proceedings of the 26th International Conference on World Wide Web Companion**. International World Wide Web Conferences Steering Committee, 2017. p. 759-760. Disponível https://arxiv.org/abs/1706.00188

[2] Borges, L. E. (2014**). *Python para desenvolvedores: aborda Python 3.3*.** Novatec Editora. Pag 14. Disponível

[3] BoSCH, Anna; ZISSERMAN, Andrew; MUNOZ, Xavier. *Image classification using random forests and ferns*. In: **2007 IEEE 11th international conference on computer vision**. Ieee, 2007. p. 1-8.

[4] Brown, Alexander. (2017). **What is hate speech? Part 1: The Myth of Hate.** Law and Philosophy. 36. https://doi.org/10.1007/s10982-017-9297-1.

[5] Culotta, A., and Sorensen, J. (2004). **Dependency tree kernels for relation extraction.** In Proc. of the 42nd annual meeting on association for computational linguistics (ACL).

[6] Diab, Shadi. Optimizing Stochastic Gradient Descent in Text Classification Based on Fine-Tuning Hyper-Parameters Approach. A Case Study on Automatic Classification of Global Terrorist Attacks. **arXiv preprint arXiv:1902.06542**, 2019.

[7] DIETTERICH, T. G. (2000, June**). Ensemble methods in machine learning. In *International workshop on multiple classifier system*s** (pp. 1-15). Springer, Berlin, Heidelberg. Disponivel http://web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf

[8] Fortuna, Paula Cristina Teixeira. 2017. **Automatic detection of hate speech in text: an overview of the topic and dataset annotation with hierarchical classes**. Master's thesis, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal.Disponivel https://repositorio-aberto.up.pt/handle/10216/106028m

[9] Friedman, Jerome H. Stochastic gradient boosting. **Computational statistics & data analysis**, v. 38, n. 4, p. 367-378, 2002. Disponivel https://statweb.stanford.edu/~jhf/ftp/stobst.pdf

[10] Gonzalez, M., and Lima, V. L. (2003). **Recuperação de informação e processamento da linguagem natural.** In XXIII Congresso da Sociedade Brasileira de Computaçao (Vol. 3, pp. 347-395). Disponivel https://statweb.stanford.edu/~jhf/ftp/stobst.pdf

[11] Haykin, S. (2008). **Neural networks and learning machines** (Vol. 3). Upper Saddle River: Pearson education.Disponivel http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf

[12] Henke, M., Santos, C., Nunan, E., Feitosa, E., dos Santos, E., & Souto, E. (2011). *Aprendizagem de máquina para segurança em redes de computadores*: **Métodos e aplicações. In Livro dos Minicursos do XI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais** (SBC, ed.). Disponivel https://sbseg2011.redes.unb.br/resources/downloads/minicursos/91555.pdf

[13] Juan, Alfons; NEY, Hermann. **Reversing and Smoothing the Multinomial** Naive Bayes Text Classifier. In: **PRIS**. 2002. p. 200-212. Disponiveis : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.3018&rep=rep1&type=pdf

[14] Julia Hirschberg, and Christopher D. Manning. Processing , **Advances in Natural Language, página** 137, 17 JULY • VOL 349 ISSUE 6245 2015. Disponível em: https://cs224d.stanford.edu/papers/advances.pdf

[15] Kleinbaum, David G. et al. **Logistic regression**. New York: Springer-Verlag, 2002.

[16] LIBREOFFICE (2013) "**VERO".** Disponível em: https://pt-br.libreoffice.org/projetos/vero/ - pt-br.libreoffice.org.

[17] Lima, Cleiton e DAL BIANCO, Guilherme. Extração de característica para identificação de discurso de ódio em documentos. In: **Anais da XV Escola Regional de Banco de Dados**. SBC, 2019. p. 61-70.

[18] Liu, Bing. Sentiment analysis and opinion mining. **Synthesis lectures on human language technologies**, v. 5, n. 1, p. 1-167, 2012.

[19] Longadge, Rushi; DONGRE, Snehalata. *Class imbalance problem in data mining review.* **arXiv preprint arXiv:1305.1707**, 2013.

[20] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., & Vanderplas, J. (2011). Scikit-learn: **Machine learning in Python.** ***Journal of machine learning** research*, *12*(Oct), 2825-2830.

[21] Pelle, R., and Moreira, V. (2017). *Offensive Comments in the Brazilian* **Web: a dataset and baselines results. In Proc. of the 6th Brazilian Workshop on Social Network Analysis and Mining (**BraSNAM).

[22] Rish, I. (2001, August). **An empirical study of the naive Bayes classifier**. In *IJCAI 2001 workshop on empirical methods in artificial intelligence* (Vol. 3, No. 22, pp. 41-46).

[23] Russel, S. e Norvig, P. 2013. **Inteligência Artificial** (3ª ed.). Elsevier Editora, Rio de Janeiro.

[24] Silva, Samuel e Serapiao, Adriane. (2018). **Detecção de discurso de ódio em português usando CNN combinada a vetores de palavras.**

[25] UOL Notícias, "**De Xuxa a Madonna: Famosas sofrem ataques de ódio por envelhecerem",** (2019) Disponível em: *https://noticiasdatv.uol.com.br/noticia/celebridades/de-xuxa-madonna-artistas-sofrem-ataques-de-odio-por-envelhecerem-26946*.