UNIVERSIDADE FEDERAL DO CEARÁ

TECHNICAL REPORT

Aluno: Carlos Eduardo Teles Alencar

1. Introdução

O presente relatório tem como objetivo analisar um dataset real de discurso de ódio em língua portuguesa, aplicado à tarefa de classificação supervisionada. O dataset utilizado é o Brazilian Portuguese Hate Speech Dataset, disponibilizado publicamente no Kaggle, contendo 5.670 textos curtos (comentários, frases e postagens de redes sociais) previamente anotados por especialistas.

Cada texto possui anotações indicando se contém ou não discurso de ódio, incluindo três níveis de granularidade (G1, G2 e G3) e uma label combinada, chamada hatespeech_comb, que consolida a classificação final em formato binário:

- 0 = texto sem discurso de ódio
- 1 = texto com discurso de ódio

Essa label foi escolhida como principal variável de análise do projeto, pois é a mais apropriada para modelagem supervisionada binária em tarefas de detecção de toxicidade e discurso ofensivo.

As principais características relevantes do dataset incluem:

- Textos reais e curtos, típicos de redes sociais.
- Linguagem informal, com erros ortográficos, gírias e abreviações.
- Presença de menções, hashtags, URLs e ruídos textuais naturais.
- Anotações em diferentes níveis de severidade de discurso de ódio.
- Balanceamento moderado entre classes (embora não perfeitamente equilibrado).

O objetivo deste trabalho é analisar o desempenho de diferentes estratégias de pré-processamento, extração de atributos e classificação, identificando quais combinações produzem a melhor acurácia para a tarefa de detecção de discurso de ódio.

2. Observações

Durante o desenvolvimento do projeto, não ocorreram problemas que impedissem a implementação ou execução das etapas propostas. Todos os módulos (pré-processamento, extração de atributos e classificação) funcionaram conforme o esperado. Apenas foram observados comportamentos esperados devido às características do dataset, como:

- O desempenho inferior do Word2Vec, resultado do corpus relativamente pequeno.
- A queda de acurácia ao utilizar lematização em alguns métodos, especialmente TF-IDF, possivelmente devido à perda de variações lexicais importantes em textos curtos e informais.

Fora esses pontos esperados, nenhum imprevisto relevante ocorreu.

3. Resultados e discussão

Nesta seção são apresentados, de forma organizada, os resultados obtidos ao longo das três etapas do projeto: pré-processamento (Questão 1), extração de atributos (Questão 2) e classificação (Questão 3). Cada subseção descreve as técnicas utilizadas, apresenta os gráficos correspondentes e discute os resultados, erro e acertos de cada etapa.

3.1. Fluxograma de aplicação:

3.1. Pré-processamento:

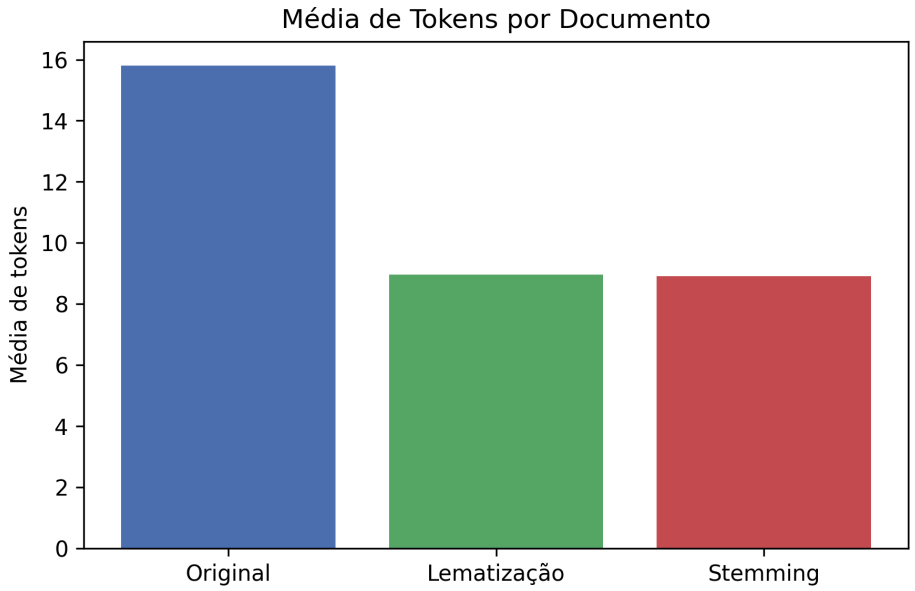
A etapa de pré-processamento teve como objetivo preparar os textos para às próximas fases de análise e classificação. Foram desenvolvidas e implementadas funções essenciais:

- remoção de URLs, menções e pontuação;
- conversão para minúsculas;
- tokenização;
- remoção de stopwords;
- stemming e lematização;
- limpeza numérica e espaços múltiplos;

Esses processamentos são importantes para reduzir ruído e padronizar o conteúdo textual, principalmente porque tweets apresentam informalidades, abreviações e símbolos.

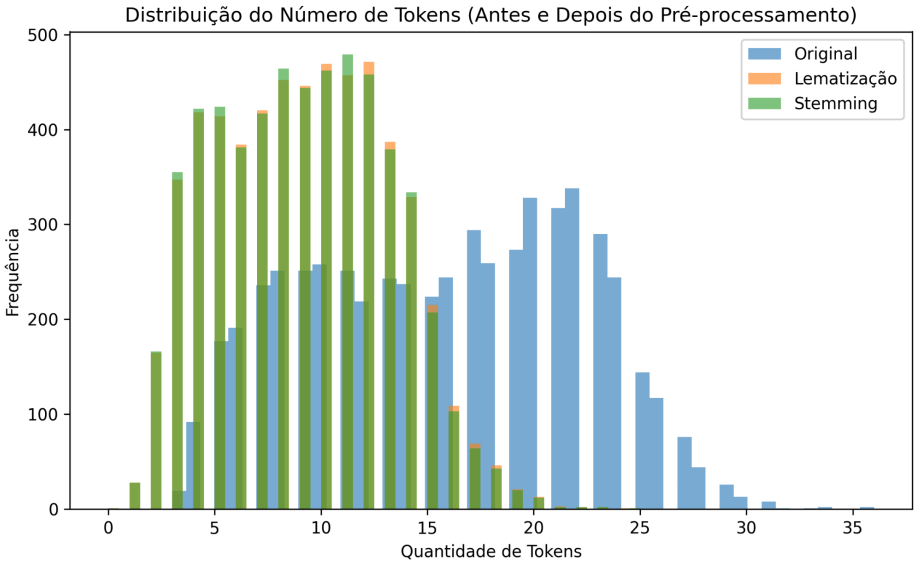
3.1.1. Impacto do Pré-processamento no tamanho do texto:

O primeiro gráfico mostra como o número médio de tokens por documento muda após os pré-processamentos.



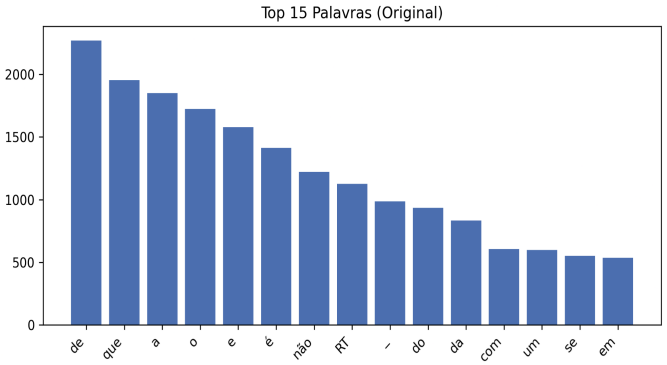
- O gráfico mostra que os textos originais têm quase o dobro de tokens quando comparados aos textos após lematização ou stemming.
- Tanto a lemmatização quanto o stemming reduzem drasticamente às variações morfológicas.

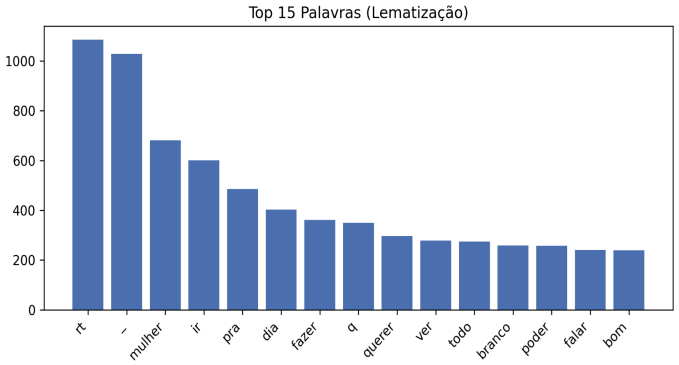
3.1.2. Distribuição do número de tokens

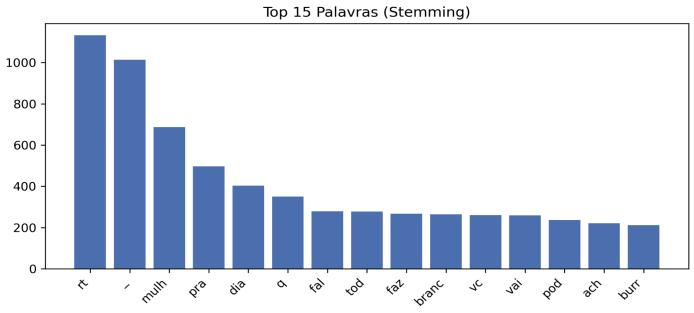


- O histograma reforça que a distribuição original é bem mais espalhada.
- Após a lematização e stemming, os tokens convergem para valores menores, evidenciando padronização.

3.1.3. Palavras mais frequentes antes e depois:







- No texto original aparecem muitas palavras funcionais (“de”, “que”, “a”, “o”), que não carregam significado discriminativo.

- Após remover stopwords e aplicar lematização e stemming, surgem palavras mais expressivas como: Mulher, fala, querer, pra, dia, poder.
- Evidencia que o pré-processamento conseguiu remover ruído e destacar conteúdo relevante.

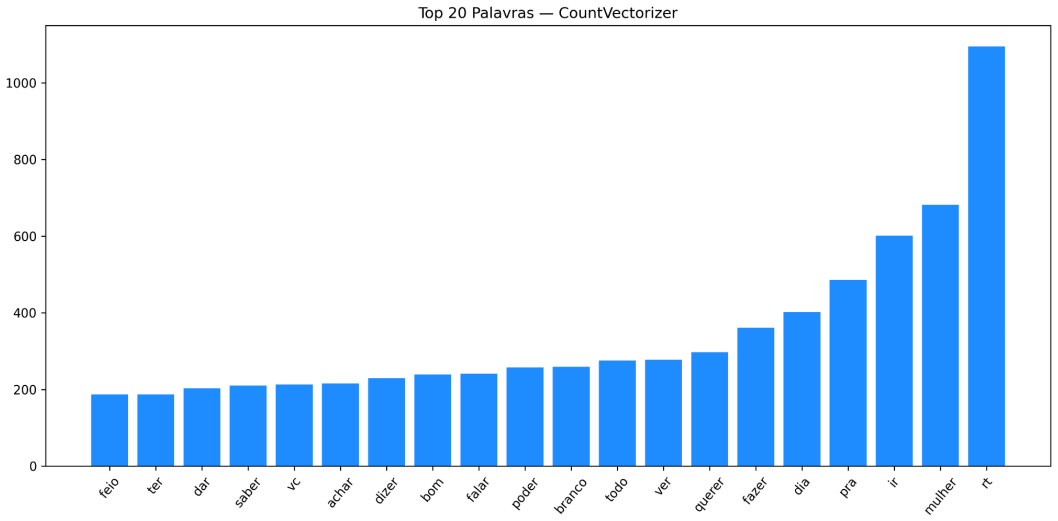
3.2. Extração de Atributos:

Nesta etapa os textos pré-processados passaram por diferentes técnicas de representação:

- CountVectorizer
- TF-IDF
- Matriz de Coocorrência
- Word2Vec

Cada método possui vantagens e limitações e foi avaliado quantitativamente e visualmente.

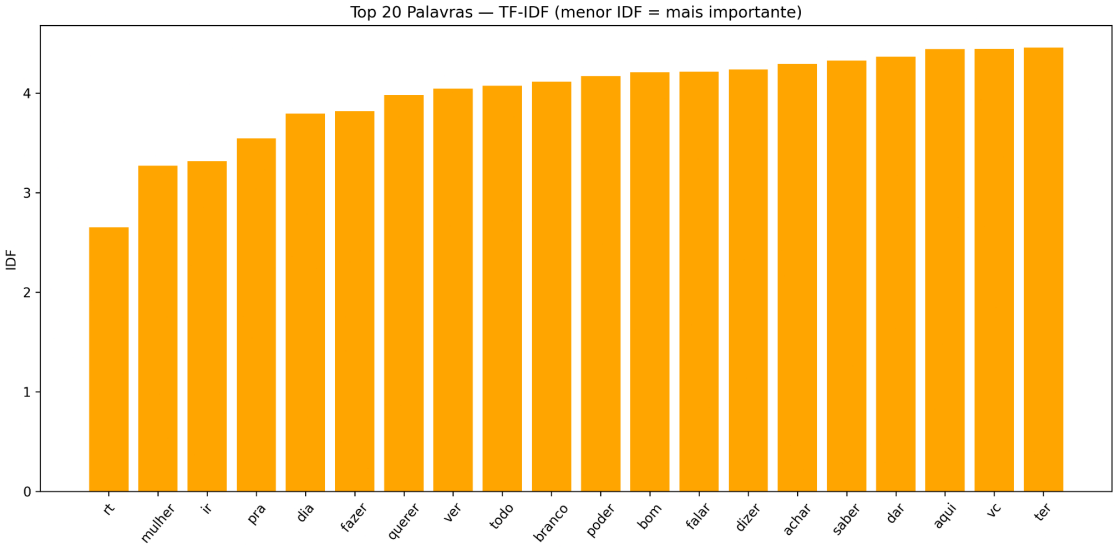
3.2.1. CountVectorizer - Palavras mais frequentes



- “rt”, “mulher”, “ir”, “pra”, “dia”, aparecem entre às mais frequentes.

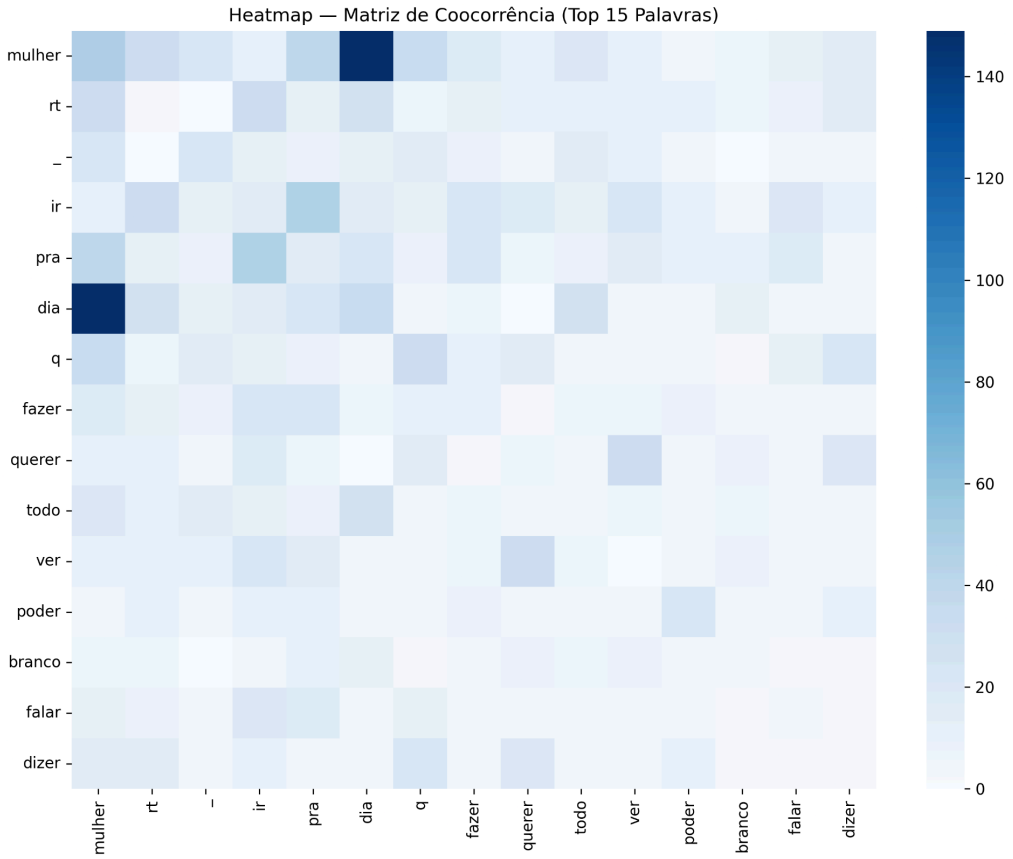
- CountVectorizer captura a frequência bruta, o que funciona bem para observar padrões lexicais, mas não diferencia palavras comuns de palavras realmente informativas.

3.2.2. TF-IDF - Termos mais informativos



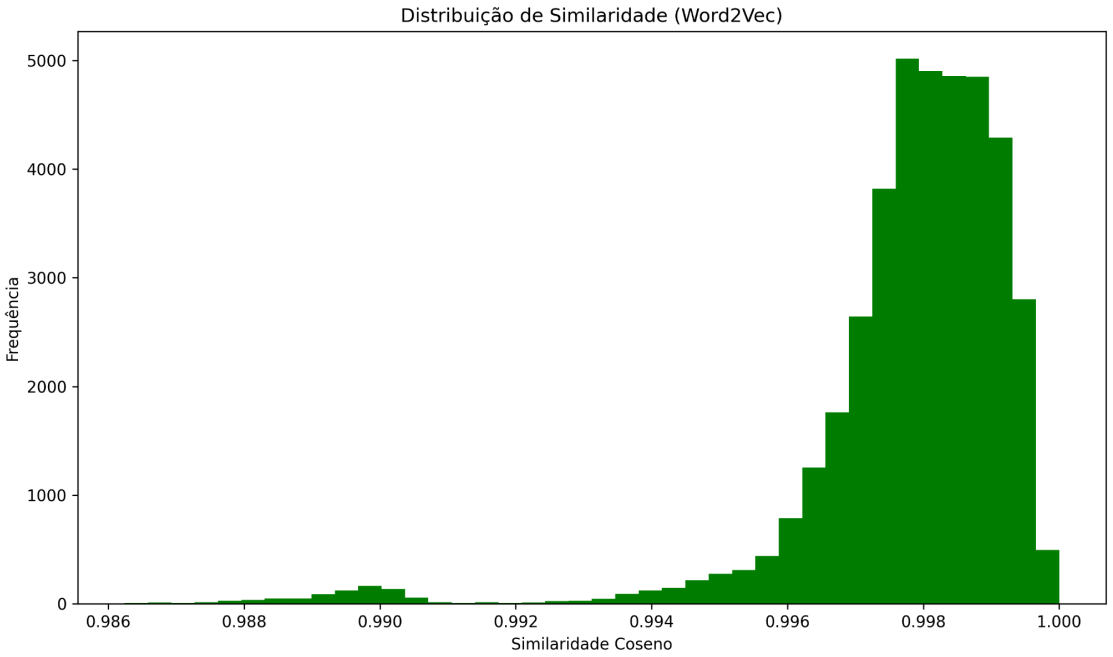
- Termos como “mulher”, “ir”, “pra”, “querer”, “ver”, “branco” têm os menores valores de IDF.
- Esses termos aparecem em muitos documentos e possuem importância discriminativa.
- TF-IDF se saiu melhor que CountVectorizer na classificação porque reduz o impacto de palavras genéricas.

3.2.3. Matriz de Coocorrência - Relação entre Termos



- O heatmap mostra quais palavras tendem a aparecer próximas entre si.
- “Mulher” e “dia” têm picos de coocorrência.
- A maioria das combinações têm valores baixos, indicando que os tweets são curtos e dispersos.
- Matriz de coocorrência reflete estrutura local do texto, mas não representou bem globalmente o dataset.

3.2.4. Word2Vec - Distribuição de Similares



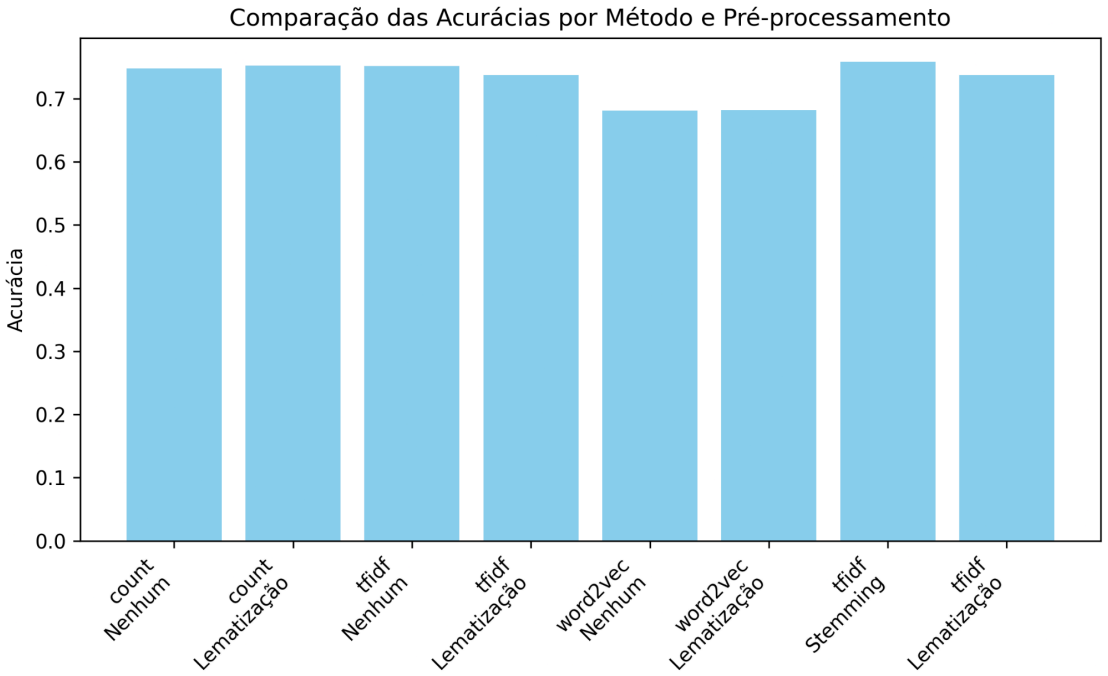
- Quase todas às similaridades ficaram entre 0.995 e 1.000, que significa que:
    - Os embeddings ficaram muito semelhantes entre si;
    - O vocabulário é pequeno e repetitivo;
    - A média de vetores perde o contexto.

3.3. Classificação:

Nesta fase foi utilizado um único classificador, Logistic Regression, e todos os métodos de extração foram testados com diferentes pré-processamentos:

- Sem pré-processamento;
- Com lematização;
- Com stemming.

3.3.1. Comparação de acurácias

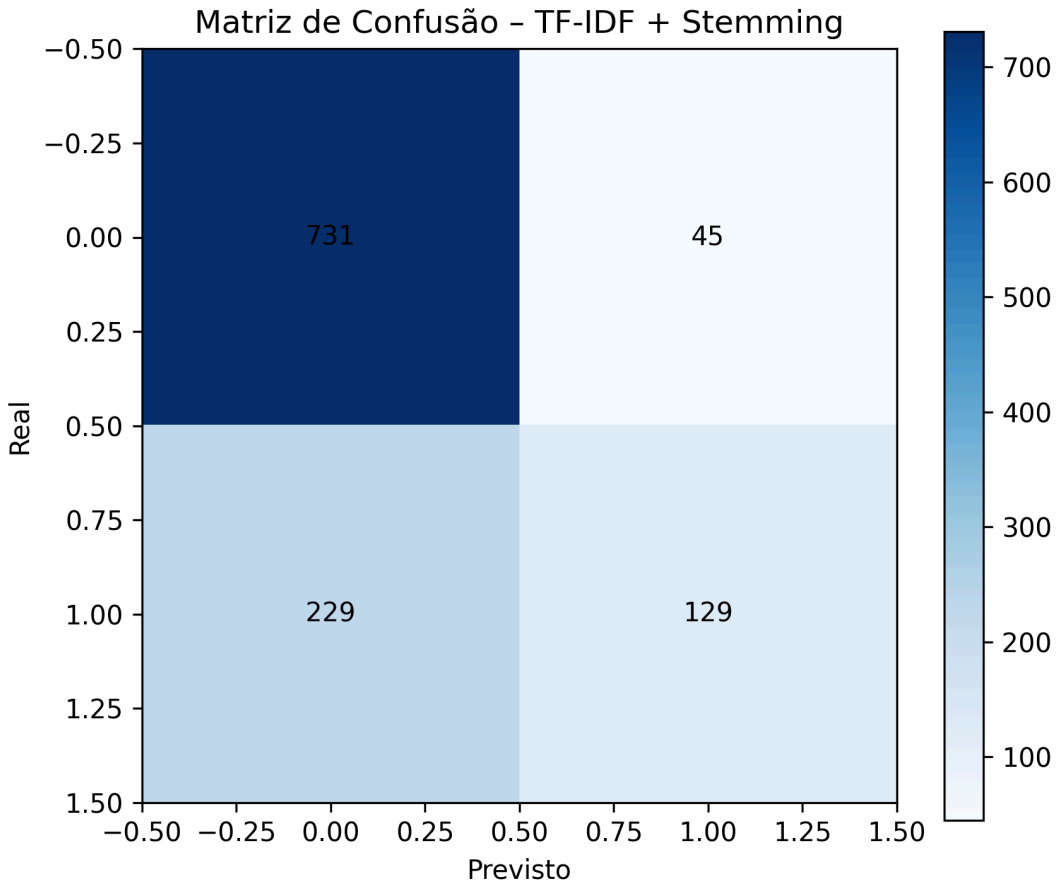


- TF-IDF + Stemming obteve a melhor acurácia: 0.7583.
- CountVectorizer também obteve um bom desempenho, porém inferior.
- Word2Vec ficou abaixo de todos.

Possíveis explicações para esses resultados podem ser:

- Tweets são curtos e o TF-IDF captura muito bem padrões de palavras.
- Stemming reduz o ruído de linguagem informal.
- Word2Vec médio perde contexto demais.

3.2.2. Matriz de confusão do melhor modelo



- A classe 0 (não hate speech) tem alta taxa de acertos.
- A classe 1 (hate speech) apresenta bastante confusão:
    - 229 textos ofensivos foram classificados como não ofensivos.
- Isso mostra que o modelo ainda tem dificuldade em identificar formas sutis de discurso de ódio, o que é comum no português informal.

3.4. Considerações Gerais:

Em geral, o que funcionou bem:

- Pré-processamento reduziu ruído e destacou termos relevantes.
- TF-IDF foi a técnica mais adequada para este dataset.
- Stemming melhora a padronização em textos informais.

4. Conclusões

Os resultados obtidos ao longo das três etapas do trabalho mostram que os objetivos esperados foram em grande parte satisfeitos. O pipeline construido desde o pré-processamento até a classificação, demonstrou capacidade de reduzir o ruído textual, representar os documentos de forma significativa e aplicar técnicas clássicas de classificação com desempenho coerente para o tipo de problema analisado.

A acurácia final de aproximadamente 0.75 obtida pelo modelo TF-IDF + Stemming + Logistic Regression está alinhada ao que se espera em tarefas reais de detecção de discurso de ódio em português, especialmente considerando características do dataset, como:

- Linguagem informal e cheia de abreviações;
- frases extremamente curtas;
- subjetividade no limite entre opinião agressiva e discurso de ódio;
- desequilíbrio relativo entre classes.

Embora o modelo tenha apresentado bom desempenho geral, alguns desafios ficaram evidentes:

1. Dificuldade na classe de hate speech

A matriz de confusão revela que a classe positiva (1) é mais difícil de identificar. Isso indica que:

- as fronteiras linguísticas entre “ofensa leve”, “ironia” e “hate speech” são tênues;
- há grande variação no modo como usuários expressam agressões;
- mesmo humanos discordariam em algumas classificações.

Isso explica por que a acurácia não foi maior, o problema é intrinsecamente complexo.

2. Word2Vec não apresentou desempenho satisfatório

O método baseado em média de embeddings teve desempenho inferior, reforçando que:

- os documentos são curtos demais para capturar semântica profunda;
- o vocabulário do dataset é pequeno;
- Word2Vec treinado localmente não extrai informações suficientes em ambientes ruidosos.

Isso está totalmente dentro do esperado.

3. Pré-processamento melhorou os resultados

Tanto a redução do vocabulário quanto a padronização linguística influenciaram positivamente o modelo. Stemming, em particular, lidou melhor com gírias e abreviações.

4. Resposta à pergunta central

Sim, **os resultados esperados foram satisfeitos**.

- O pipeline produziu resultados consistentes com o estado da arte para este tipo de dataset.
- O comportamento dos modelos foi coerente com as características das técnicas utilizadas.
- As comparações entre métodos demonstraram claramente como cada estratégia influencia o desempenho.

A única limitação foi a dificuldade em classificar corretamente textos mais sutis de hate speech, um desafio amplamente documentado na área, e que não invalida o experimento.

5. Próximos passos

Com base nos resultados obtidos, algumas opções de evolução surgem:

1. Testar modelos mais avançados:

Utilizar modelos baseados em transformers que capturam melhor ironia, contexto e nuances presentes no discurso de ódio.

2. Ampliar o pré-processamento:

Incluir normalização de gírias, emojis e abreviações comuns em redes sociais, que pode melhorar a separação entre às classes.