Classificação de Notícias Utilizando Machine Learning

Luiza Gandolfi Barioto
Departamento de Computação
Universidade Federal de São Carlos
Sorocaba, Brasil
luizagandolfi@estudante.ufscar.br

***Abstract*—Com o aumento exponencial da disseminação de informações on-line, a identificação de notícias falsas tornou-se uma prioridade crucial. Este artigo propõe uma abordagem que explora as técnicas para a detecção de notícias falsas, utilizando desde formas mais simples, como algoritmos lineares a técnicas avançadas de aprendizado de máquina, como redes neurais, otimizando a capacidade do sistema em reconhecer padrões complexos e nuances presentes em informações enganosas. Além disso, destaca-se a importância da criação de conjuntos de dados representativos e equilibrados, bem como a validação rigorosa do modelo em cenários do mundo real. Os resultados experimentais demonstram a eficácia da abordagem proposta, revelando uma taxa significativa de detecção de notícias falsas, contribuindo assim para a mitigação dos impactos negativos associados à propagação de desinformação. Este estudo oferece uma contribuição valiosa para a área emergente de combate à desinformação, fornecendo uma base sólida para futuras pesquisas e desenvolvimentos práticos no campo da detecção de notícias falsas utilizando aprendizado de máquina.**

***Keywords*-fake news; machine learning; data preprocessing; vectorization;**

I. Introdução

No cenário contemporâneo da informação digital, a disseminação de notícias falsas tem emergido como um desafio crítico para a sociedade, comprometendo a integridade do espaço informativo e minando a confiança pública.[1] O advento das redes sociais e plataformas online proporcionou um ambiente propício para a proliferação dessas informações enganosas, ampliando sua influência de maneira significativa.[2] Nesse contexto, a aplicação de técnicas avançadas de aprendizado de máquina (ML) surge como uma ferramenta promissora na identificação eficiente e precisa de notícias falsas[3].

Este artigo propõe uma análise aprofundada do uso de técnicas de ML para detecção de notícias falsas, destacando a complexidade inerente ao fenômeno da desinformação e os desafios enfrentados pelos pesquisadores nesse campo. Ao explorar as possibilidades oferecidas por algoritmos de ML, pretende-se investigar como essas abordagens podem contribuir para a construção de sistemas robustos de detecção, considerando a rápida evolução das estratégias empregadas por propagadores de notícias falsas.

Ao longo deste artigo, serão examinadas diferentes abordagens de aprendizado de máquina, desde métodos tradicionais até abordagens mais recentes, como redes neurais profundas, para avaliar sua eficácia na identificação de padrões sutis associados a notícias falsas. Além disso, serão discutidas as limitações inerentes a esses métodos e as conclusões alcançadas pela exploração destes.

II. Pré-processamento dos Dados

Os dados utilizados para a confecção deste projeto estavam divididos entre os meses de coleta, indo de janeiro de 2019 a dezembro de 2020, além de dois arquivos (*train* e *test*) que irão compor as bases para o treinamento e teste dos modelos posteriormente propostos. O pré-processamento dos dados foi a fase mais custosa (em termos de tempo e capacidade computacional) deste projeto, sendo assim divididos em seções.

A. Bibliotecas Utilizadas

As bibliotecas utilizadas para o pré-processamento dos dados foram: *pandas*, utilizada para a manipulação de datasets e dataframes, *glob*, que encontra todos os caminhos de nomes que correspondem a um padrão especificado de acordo com certas regras usadas, *NLTK*, que serve neste contexto para remover *stopwords* (palavras comuns que geralmente são removidas durante a etapa de pré-processamento de texto, pois não contêm informações valiosas) e também foi utilizado para criar o lematizador baseado no WordNet, um banco de dados lexical da língua inglesa.

B. Classe e Funções Criadas

Para o pré-processamento dos dados, foi criada a classe *CreateDataset*, que possui todas as funções que serão utilizadas para o tratamento dos dados que serão utilizados para treino, teste e validação dos modelos. As funções são:

1) All_data_dataset: Recebe como parâmetro o caminho da pasta que possui todos os arquivos de dados utilizados para compor a base de dados do modelo e combina todos estes, cujos nomes começam com *"news data"*, em um arquivo *csv* chamado *"combined data"*.

2) All_data_train_test: Recebe como parâmetro os arquivos que possui *id* e *label* dos dados de treino e *id* dos dados de teste, além do dataframe criado na função anterior. Esta função filtra dados do dataframe completo para criar conjuntos de treino e teste, removendo entradas nulas e selecionando apenas dados do ano de 2020, uma vez que os dados de teste eram em sua grande maioria do ano de 2020 também.

3) Clean_text: Recebe como parâmetro um dataset e realiza uma série de operações de limpeza nos textos do dataset, como converter para minúsculas, remover pontuações, números, URLs, espaços extras e *stopwords*.

4) Balance_df: Recebe como parâmetro um dataset e o balanceia, retornando um dataset que possua o mesmo número de dados presentes em cada rótulo utilizado. Isso dificulta que o modelo se torne enviesado pela distribuição desigual de classes na base de treinamento.

5) Concat_title_content: Recebe como parâmetro um dataset e combina as colunas ”title” e ”content” em uma única coluna chamada ”content title”. Isso serve para que haja uma nova forma de alimentar todos os dados textuais para o modelo.

6) Lemmatize_words: Recebe como parâmetro um texto ou conjunto de textos e aplica a lematização a eles, ou seja, reduz as palavras à sua forma base, o que promove uma maior acurácia nos resultados de treinos de Processamento de Linguagem Natural.

Além destas funções, também foi feita a ordenação das colunas dos datasets para uma ordem que facilite a visualização dos atributos mais importantes.

III. Análise Exploratória dos Dados

A. Classe e Funções Criadas

Para a análise exploratória dos dados, foi criada uma classe chamada *Analisys* que será utilizada para a contemplação dos dados, bem como a retirada de *insights* valiosos que podem ajudar a treinar um modelo mais robusto e mais preparado para cenários reais. Deve-se considerar que as análises foram feitas depois da limpeza dos dados, portanto, depois da retirada de *stopwords*, de dados em que os anos não eram 2019 ou 2020, ou em que os meses e anos não estavam entre os possíveis e também, após da retirada de linhas vazias do dataset, sejam de título, conteúdo ou rótulo. Deve-se ter em mente também que as análises foram feitas a partir do dataframe de treino, depois do balanceamento de rótulos. Todas essas considerações foram feitas para que a análise seja feita em cima dos dados que serão efetivamente alimentados para o modelo e os dados disponíveis em 2019 para efeito de comparação. Assim, a classe consiste das seguintes funções:

1) Pizza_graph: Recebe como parâmetro um dataframe e gera gráficos de pizza para a distribuição de suas classes ao longo de 2019 e 2020, com base na coluna *date* para data e na coluna *label*.

2) Wordcloud: Recebe como parâmetro um dataframe e gera uma nuvem de palavras contendo aquelas mais recorrentes, tanto entre notícias falsas como em notícias verdadeiras, o que promove um insight valioso das palavras que possuem mais ou menos peso em cada um dos rótulos.

3) Time_series: Recebe como parâmetro um dataframe e gera um gráfico de séries temporais comparando a contagem de notícias verdadeiras e falsas ao longo do tempo.

B. Análise dos Dados para Treinamento

A partir das imagens mostradas a seguir, pode-se fazer algumas inferências sobre o conjunto de dados usado.

Observa-se que houve um aumento significativo na frequência de notícias falsas em 2020, quando em comparação com 2019, devido à algumas situações como a pandemia da COVID-19, eleições presidenciais nos Estados Unidos, entre outros.

Figure 1.
Distribuição das Classes em 2019

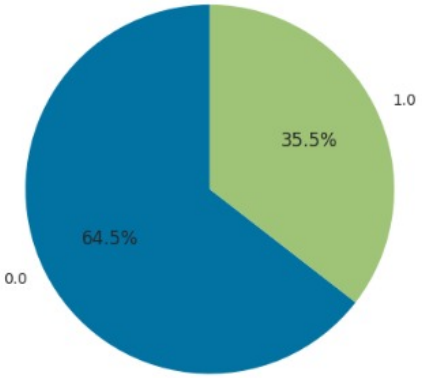
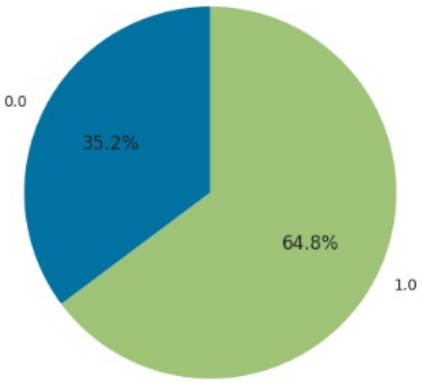


Figure 2.
Distribuição das Classes em 2020



Visualiza-se também as palavras que mais apareceram nas notícias falsas e verdadeiras, e, como visto abaixo, descon-

siderando palavras que aparecem frequentemente em ambos os tipos de notícias, podemos destacar que nas notícias falsas se faz mais presente termos como ”donald trump”, ”white house” e ”new york”, e nas notícias verdadeiras, pode-se observar a predominância de termos como ”work”, ”claim”, entre outros.

Figure 3. Nuvem de palavras para notícias falsas

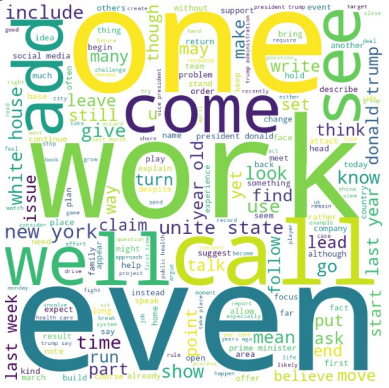
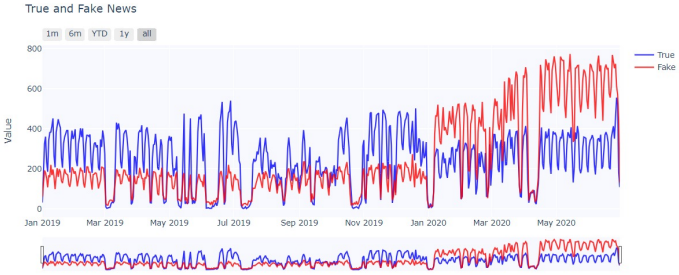


Figure 4. Nuvem de palavras para notícias verdadeiras

Com a análise da série temporal abaixo, pode-se perceber novamente que houve um aumento significativo na quantidade de notícias falsas quando comparado com a quantidade de notícias verdadeiras ao longo de 2019 e 2020.

Figure 5. Frequência de notícias falsas e verdadeiras durante 2019 e 2020



IV. Preparação para o treino

Antes de começar os treinamentos com os modelos propriamente ditos, se faz necessária a vetorização dos textos que serão alimentados ao modelo, ou seja, a conversão dos dados textuais em vetores numéricos que serão utilizados para a análise de dados. Para isso, foi utilizada a técnica de TF-IDF (term frequency-inverse document frequency) que é uma métrica que pode quantificar a importância ou relevância de palavras ou frases em um documento ou conjunto de documentos. Também há outras formas de concluir essa vetorização, como utilizando a classe *Count Vectorizer* da biblioteca *scikit-learn*, porém a técnica escolhida foi a TF-IDF por levar em conta também a importância da palavra ou termo no documento, não apenas a frequência em que ela aparece, o que cria a possibilidade da retirada de termos menos importantes, resultando numa complexidade menor na construção do modelo ao diminuir as dimensões das entradas. Além disso, também foi utilizada a técnica de divisão entre conjuntos de treino (90% de todos os dados) e teste (10% restantes), utilizando-se da função *train_test_split* da biblioteca *sklearn*.

Os dados utilizados para o modelo, após todos os processos de limpeza e calibragem, contém a seguinte distribuição:

Rótulo	Frequência	Porcentagem
Notícias verdadeiras - 2020	228.187	51,3%
Notícias falsas - 2020	216.504	48,7%
Total	444.691	100%

Table I
Distribuição dos dados utilizados.

V. Treino dos modelos e considerações sobre o melhor modelo

Foram utilizados vários tipos de modelos, sendo eles *K Nearest Neighbors* (KNN), *Logistic Regression* (LR), *Gaussian Naive Bayes* (GNB), *Linear Support Vector Machines* (SVM), *Random Forest* (RF), *Recurrent Neural Network* (RNN), *XGBoost* (XGB) e *Catboost* (CTB). Todos tiveram seus hiperparâmetros otimizados, utilizando-se da técnica de Randomized Search (para os modelos Logistic Regression, Gaussian Naive Bayes, Support Vector Machines, Random Forest, XGBoost e Catboost) ou o otimizador AdamW (para o modelo RNN). Além disso, foram testados *ensembles* entre os modelos, como entre LR e RF e SVM e RF e entre os modelos LR, RF, SVM e XGB.

Pode-se observar na tabela abaixo uma compilação das métricas de desempenho, incluindo precisão, recall e acurácia, para cada modelo individual e para os ensembles testados. Além disso, são apresentados para o melhor modelo testado com base na acurácia, o Catboost, a matriz de confusão e a curva ROC.

Modelo	Precisão	Recall	Acurácia
KNN	0.694	0.716	0.700
LR	0.812	0.793	0.805
GNB	0.701	0.722	0.784
SVM	0.806	0.812	0.796
RF	0729	0.720	0.749
XGB	0.798	0.770	0.781
RNN	0.723	0.765	0.726
CTB	0.842	0.794	0.820
LR + RF	0.819	0.793	0.710
SVM + RF	0.815	0.824	0.772
LR + SVM + RF + XGB	0.830	0.827	0.844

Table II
MÉTRICAS OBTIDAS NO TREINO E VALIDAÇÃO DE CADA MODELO

Figure 6. Curva ROC do Melhor Modelo

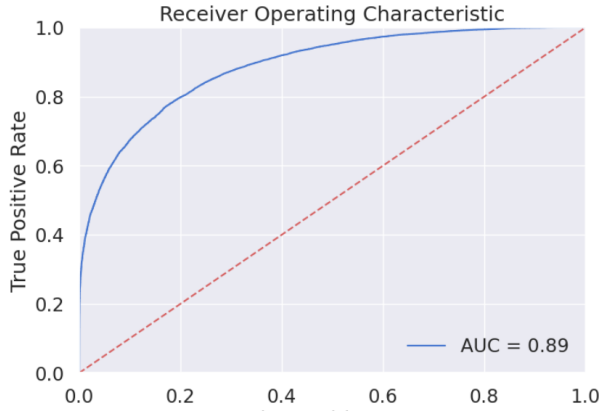
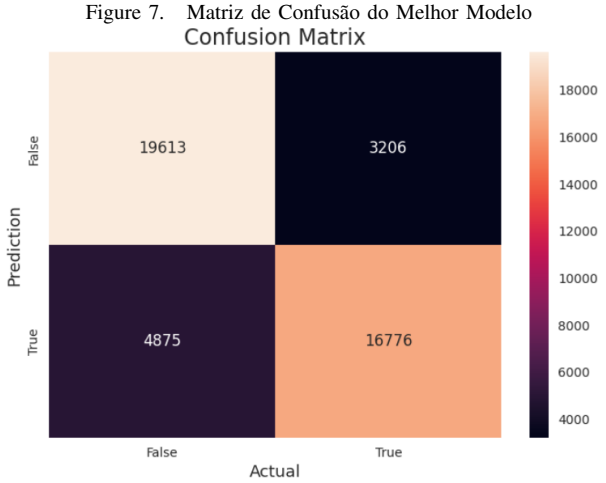


Figure 7. Matriz de Confusão do Melhor Modelo



Esta curva ROC mostra que o modelo se comporta de forma satisfatória, conseguindo atingir boa generalização e boa acurácia, o que é muito importante para este cenário de detecção de notícias falsas com a quantidade de atributos que estão disponíveis. A matriz de confusão também corrobora para esta afirmação, mostrando bons números em termos de classificação correta e incorreta de instâncias. .

VI. Conclusões e Considerações

O problema da classificação de notícias falsas no cenário dado apresenta algumas limitações, como entre elas, o tamanho do base de dados do modelo, que o torna impossível de ser utilizado completamente por limitações de memória e capacidade computacional da plataforma *Kaggle*. Os ruídos presentes na base de dados, como datas impossíveis, notícias com títulos e conteúdos idênticos, porém rótulos diferentes, entre outros, dificultam o treinamento de um modelo eficiente para a classificação de notícias falsas de forma mais acurada. Apesar disso, com o pré-processamento dos dados, divisão entre base de dados para treino e teste e certa filtragem entre aspectos que não seriam relevantes para o modelo, obteve-se um resultado satisfatório para a classificação de texto deste cenário.

Referências

[1] S. Mishra, P. Shukla, and R. Agarwal, "Analyzing machine learning enabled fake news detection techniques for diversified datasets," *Wireless communications and mobile computing*, vol. 2022, pp. 1–18, 2022.

[2] R. Varma, Y. Verma, P. Vijayvargiya, and P. P. Churi, "A systematic survey on deep learning and machine learning approaches of fake news detection in the pre- and post-covid-19 pandemic," *International journal of intelligent computing and cybernetics*, vol. 14, no. 4, pp. 617–646, 2021.

[3] I. Ahmad, M. Yousaf, S. Yousaf, and M. O. Ahmad, "Fake news detection using machine learning ensemble methods," *Complexity (New York, N.Y.)*, vol. 2020, pp. 1–11, 2020.