Classificação de Spam no Conjunto de Dados de Detecção de Spam de E-mail

Pedro Henrique Britto Aragão Andrade
Email: pedroa3@al.insper.edu.br

***Abstract*—Este paper tem como objetivo desenvolver um modelo de detecção de spam em e-mails utilizando técnicas de classificação baseadas em processamento de linguagem natural (NLP). O spam consiste em mensagens automáticas que podem incluir desde anúncios promocionais até tentativas de fraude. O projeto visa identificar padrões recorrentes nesses e-mails, permitindo sua classificação precisa e automática.**

***Index Terms*—classificação de spam de e-mail, processamento de linguagem natural**

I. Introdução

Os e-mails de spam trazem diversos prejuízos aos usuários, como o recebimento de uma grande quantidade de mensagens indesejadas, tanto semanalmente quanto diariamente, o que resulta em perda de tempo. Além disso, esses e-mails consomem uma quantidade significativa de memória do sistema operacional, impactando o desempenho [1]. Diante desse cenário, o objetivo deste trabalho é identificar, de forma eficaz, se uma mensagem é ou não spam. Para isso, foram realizados testes envolvendo diferentes técnicas de pré-processamento e modelos de classificação, com o intuito de otimizar a detecção de padrões em e-mails de spam.

II. Metodologia

	label	text	label_num
0	ham	Go until jurong point, crazy.. Available only ...	0
1	ham	Ok lar... Joking wif u oni...	0
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	ham	U dun say so early hor... U c already then say...	0
4	ham	Nah I don't think he goes to usf, he lives aro...	0
...	...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...	1
5568	ham	Will ì_ b going to esplanade fr home?	0
5569	ham	Pity, * was in mood for that. So...any other s...	0
5570	ham	The guy did some bitching but I acted like i'd...	0
5571	ham	Rofl. Its true to its name	0

Fig. 1. Dataset utilizado para o treinamento

A. Pré-processamento

Três técnicas de pré-processamento foram adotadas neste projeto: **Stemming**, **Lematização** (Lemmatization) e **Remoção de Stop Words**. Cada uma dessas técnicas apresenta características e estratégias distintas para manipulação dos dados textuais.

A primeira técnica, **Stemming**, tem como objetivo reduzir as palavras aos seus radicais, removendo sufixos e prefixos, de modo que diferentes formas de uma palavra com o mesmo significado central sejam unificadas. Por exemplo, em inglês, as palavras ”running”, ”ran” e ”runs” são transformadas em ”run”. Essa estratégia reduz a dimensionalidade dos dados, pois múltiplas variações de uma mesma palavra são condensadas em uma única forma, melhorando a eficiência no processamento e facilitando a generalização dos dados durante o treinamento do modelo. Neste projeto, utilizamos o Snowball Stemmer.

A segunda técnica, **Lematização**, é semelhante ao Stemming, mas é mais sofisticada. Em vez de simplesmente remover sufixos e prefixos, a lematização transforma as palavras em sua forma base ou ”lemma”, considerando o contexto gramatical da frase. Por exemplo, a palavra ”better” seria lematizada para ”good”, algo que não seria possível com o stemming. Essa técnica é, teoricamente, mais precisa do que o stemming, já que leva em conta o significado das palavras no contexto. Neste projeto, utilizamos o WordNetLemmatizer.

Por fim, a **Remoção de Stop Words** consiste em eliminar as palavras mais comuns em um idioma específico, que não agregam valor significativo à classificação do texto. No caso deste trabalho, foram removidas as stop words em inglês, como ”the”, ”is” e ”in”. Essas palavras, por serem frequentes em qualquer contexto, podem introduzir ruído e prejudicar a eficiência do modelo, tornando a classificação menos precisa.

B. Vetorizadores

No campo de processamento de linguagem natural (NLP), vetores são representações numéricas que convertem textos em formas compreensíveis por algoritmos de machine learning, permitindo que esses algoritmos identifiquem padrões e façam previsões. Neste paper, utilizamos dois dos vetorizadores mais comuns: CountVectorizer e TF-IDF Vectorizer.

O **CountVectorizer** transforma o texto em uma matriz de contagem de palavras. Ele simplesmente contabiliza o número de vezes que cada palavra aparece em cada documento, resultando em um vetor para cada documento, onde cada elemento do vetor corresponde à contagem de uma palavra específica. Esse método é direto e eficaz para problemas em que a frequência de palavras é um indicador relevante.

Por outro lado, o **TF-IDF Vectorizer** (Term Frequency-Inverse Document Frequency) vai além da simples contagem de palavras. Ele leva em consideração a frequência de uma palavra em um documento, ponderada pela sua presença em

outros documentos. Assim, palavras que aparecem frequentemente em muitos documentos (como ”the” e ”is”) são consideradas menos informativas e recebem um peso menor. Esse método é útil para destacar termos que são mais relevantes em um documento específico, reduzindo a influência de palavras muito comuns que pouco contribuem para a diferenciação entre textos.

C. Transformadores

Os transformadores têm como objetivo reduzir ou converter a dimensionalidade dos dados, facilitando a identificação de padrões e características importantes para a classificação. Neste projeto, utilizamos dois tipos de abordagens: a aplicação de Non-Negative Matrix Factorization (NMF) e a não utilização de nenhum transformador (dados brutos).

O **Non-Negative Matrix Factorization** (NMF) é uma técnica que ajuda a identificar tópicos latentes presentes em documentos de texto. Ele decompõe a matriz de termos por documentos em duas matrizes menores, que representam os documentos em termos de tópicos e os tópicos em termos de palavras. No contexto da classificação de e-mails, o NMF permite encontrar características ou padrões que possam distinguir e-mails de spam e não spam, ao reduzir a dimensionalidade do texto e destacar informações relevantes.

Por outro lado, também avaliamos o impacto de não utilizar nenhum transformador, passando os dados textuais diretamente para os classificadores, sem qualquer redução de dimensionalidade. Essa abordagem serve como base de comparação, permitindo avaliar a eficácia do NMF na melhoria da performance do modelo.

D. Classificadores

Classificadores são algoritmos de aprendizado de máquina que aprendem a partir dos dados para realizar previsões. Existem diversos tipos de classificadores, cada um com suas características e aplicações. Neste projeto, os classificadores utilizados foram: BernoulliNB, Logistic Regression, e RandomForestClassifier.

O **BernoulliNB** é uma variante do algoritmo Naive Bayes, um modelo probabilístico simples, mas eficaz para tarefas de classificação. Ele assume que os dados seguem uma distribuição de Bernoulli, onde cada característica pode assumir apenas dois valores (1 ou 0). O algoritmo calcula a probabilidade de um documento pertencer a uma classe (por exemplo, spam ou não spam) com base na presença ou ausência de determinadas palavras. O Naive Bayes utiliza a regra de Bayes, fazendo a suposição de que todas as características (neste caso, palavras) são independentes umas das outras.

A **Regressão Logística** utiliza a função logística (sigmoide) para estimar a probabilidade de uma amostra pertencer a uma determinada classe. Ela aprende uma combinação linear das características para prever a classe alvo (por exemplo, spam ou não spam). Se a probabilidade calculada for maior que 0,5, o modelo prevê a classe positiva (por exemplo, spam); caso contrário, prevê a classe negativa (por exemplo, não spam).

Por fim, o **RandomForestClassifier** é um conjunto de árvores de decisão independentes, cada uma treinada com uma amostra aleatória dos dados e características. Essa diversidade entre as árvores cria um modelo mais robusto e menos suscetível ao overfitting. A previsão final é obtida pela agregação (votação) das previsões de todas as árvores, tornando-o um classificador poderoso para problemas complexos de classificação.

III. Resultados

A. Dataset

Para o treinamento de todas as 36 combinações possíveis, utilizamos o conjunto de dados disponível no Kaggle, denominado ”Email Spam Detection Dataset (Classification)” [2]. Este dataset foi selecionado devido à sua relevância e qualidade na área de detecção de spam em e-mails. O conjunto de dados tem como base o trabalho apresentado em [1], que explorou técnicas de classificação para identificar mensagens indesejadas.

B. Análise dos Resultados

A seguir são apresentados os gráficos que resumem a análise das combinações de pré-processamento e classificadores utilizados no projeto.

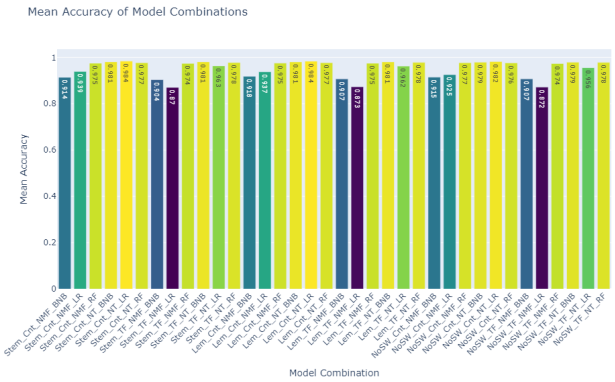

Fig. 2. Gráfico de barras da média das acurácias (100 execuções) vs. combinações dos modelos

A análise dos resultados revelou alguns padrões significativos:

- As combinações que **não utilizam transformadores** (NoTrans) apresentaram, em geral, as **melhores acurácias**. A combinação *Stemmed_Count_NoTrans_LogisticRegression* apresentou a **melhor acurácia média** de **0.983893**, com um desvio padrão de **0.003765**.
- O **RandomForestClassifier** apresentou-se como um classificador robusto, consistentemente alcançando **altas acurácias**, independentemente das técnicas de pré-processamento ou dos vetorizadores utilizados.
- As combinações que utilizaram o vetorizador **CountVectorizer** superaram, em média, as combinações que utilizaram o **TfidfVectorizer**, indicando que a contagem

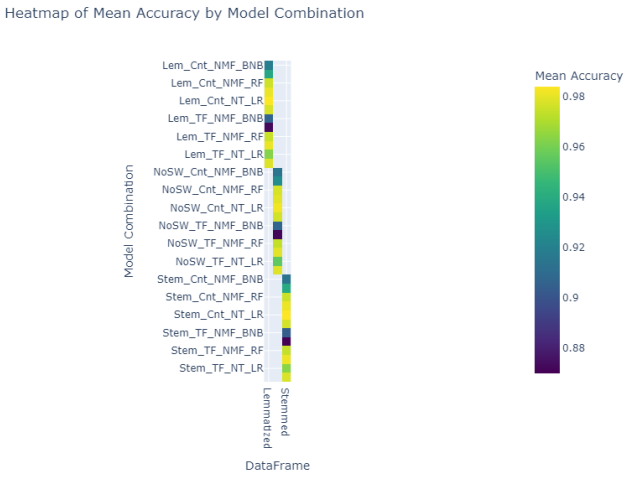

Fig. 3. Mapa de calor das médias das acurácias (100 execuções) vs. combinações dos modelos

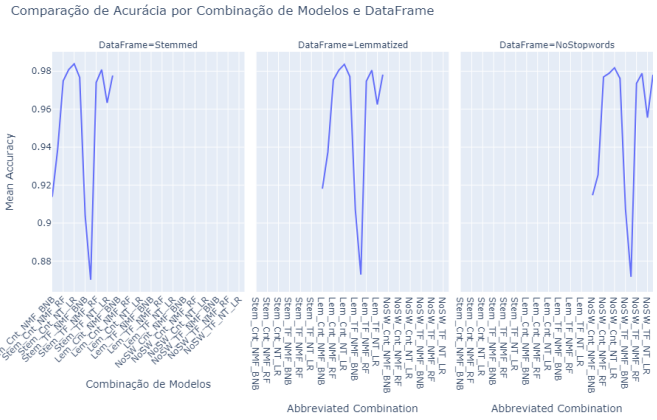

Fig. 4. Facetas das médias das acurácias (100 execuções) vs. combinações dos modelos

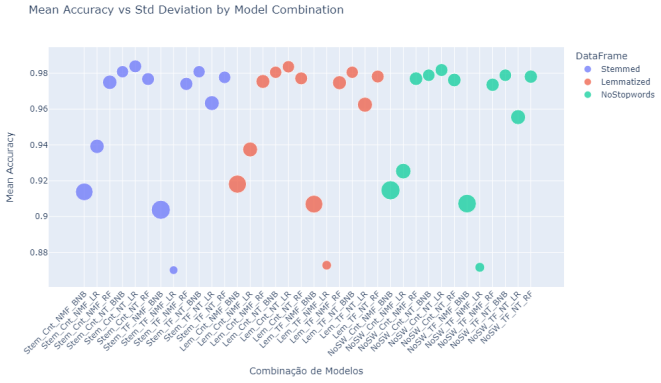

Fig. 5. Gráfico de dispersão com desvio padrão (tamanho dos pontos) das médias das acurácias (100 execuções) vs. combinações dos modelos

simples de palavras foi mais eficaz para o problema de classificação de spam.

- O uso do transformador **NMF** resultou, em geral, em **acurácias inferiores**, sugerindo que a redução de dimensionalidade proporcionada pelo NMF não capturou adequadamente as informações relevantes para a classificação de spam.
- As combinações de **TfidfVectorizer e Logistic Regression** apresentaram as **piores performances**, com destaque para a combinação *Stemmed_Tfidf_NMF_LogisticRegression*, que obteve uma acurácia de **0.8701**, evidenciando que essa abordagem não é adequada para este conjunto de dados.

Aqui está o texto sugerido para a seção Tamanho do Dataset com os dados fornecidos:

C. Tamanho do Dataset

Para avaliar o impacto do tamanho do dataset na performance do modelo, foi realizado um experimento de downsampling, testando diferentes frações do dataset original. O modelo utilizado foi a Regressão Logística com o pré-processamento de stemming, vetor de contagem (CountVectorizer) e sem aplicação de transformadores. A acurácia foi medida para cada fração do dataset, e os resultados foram os seguintes:

- Fraction 0.15: Acurácia = 83.33%
- Fraction 0.25: Acurácia = 85.90%
- Fraction 0.5: Acurácia = 92.26%
- Fraction 0.75: Acurácia = 94.53%
- Fraction 1.0: Acurácia = 93.16%

Os resultados mostram que, mesmo com apenas 15% dos dados, o modelo conseguiu uma acurácia de 83.33%, o que sugere que o modelo já consegue capturar padrões relevantes mesmo com uma fração reduzida do dataset. À medida que o tamanho do dataset aumenta, a acurácia melhora gradualmente, atingindo um pico de 94.53% com 75% dos dados. No entanto, com 100% dos dados, a acurácia cai ligeiramente para 93.16%, o que pode indicar que o aumento no tamanho do dataset não traz benefícios substanciais além de 75%, sugerindo que o modelo começa a estabilizar em termos de desempenho. Isso indica que, para este problema específico, a utilização de todo o dataset não é necessariamente essencial para alcançar uma performance ótima.

IV. Conclusão

Os resultados obtidos demonstram que a escolha das técnicas de pré-processamento e vetorização tem um impacto substancial na eficácia da detecção de spam. A combinação *Stemmed_Count_NoTrans_LogisticRegression* mostrou-se a mais eficiente, indicando que a contagem simples de palavras, sem a aplicação de transformações para redução de dimensionalidade, aliada a um classificador linear como a Regressão Logística, é suficiente para capturar os padrões característicos dos e-mails de spam.

Adicionalmente, o RandomForestClassifier destacou-se como uma solução robusta, apresentando resultados consistentes e confiáveis em diferentes cenários.

Por outro lado, técnicas como NMF e TF-IDF não se mostraram adequadas para este problema específico, sugerindo que transformações adicionais nos dados podem introduzir mais ruído do que benefícios. Esses achados fornecem direções claras para otimizar futuros modelos de detecção de spam, concentrando-se nas combinações que obtiveram os melhores resultados e evitando aquelas que apresentaram baixo desempenho.

REFERENCES

[1] R. Kumar, S. P. Sudarshan, and G. Mahalakshmi, "Email Spam Detection using Machine Learning Techniques," in Master of Computer Application, CEG Anna University, Chennai, Tamil Nadu.

[2] S. Dhakadd, "Email Spam Detection Dataset (Classification)," Kaggle, 2019. [Online]. Available: https://www.kaggle.com/datasets/shantanudhakadd/email-spam-detection-dataset-classification. [Accessed: 03-Oct-2024].