Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de software

Modelos preditivos para categorizar notícias em português usando processamento de linguagem natural e transformadores

Autor: Ítalo Vinícius P. Guimarães
Orientador: Professor Dr. Nilton Correia da Silva

Brasília, DF
2023

Ítalo Vinícius P. Guimarães

Modelos preditivos para categorizar notícias em português usando processamento de linguagem natural e transformadores

Monografia submetida ao curso de graduação em Engenharia de software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Professor Dr. Nilton Correia da Silva

Brasília, DF

2023

Ítalo Vinícius P. Guimarães

Modelos preditivos para categorizar notícias em português usando processamento de linguagem natural e transformadores/ Ítalo Vinícius P. Guimarães. – Brasília, DF, 2023-

83 p. : il. (algumas color.) ; 30 cm.

Orientador: Professor Dr. Nilton Correia da Silva

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2023.

1. notícias. 2. categorias. I. Professor Dr. Nilton Correia da Silva. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Modelos preditivos para categorizar notícias em português usando processamento de linguagem natural e transformadores

CDU 02:141:005.6

Ítalo Vinícius P. Guimarães

Modelos preditivos para categorizar notícias em português usando processamento de linguagem natural e transformadores

Monografia submetida ao curso de graduação em Engenharia de software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de software.

Professor Dr. Nilton Correia da Silva
Orientador

Professor Dr. Fabrício Ataides Braz
Convidado 1

Professor Dr. Henrique Marra Taira Menegaz
Convidado 2

Brasília, DF
2023

Este trabalho é dedicado aos meus irmãos, que me fazem ver o mundo de um jeito diferente e tentar entendê-lo de uma maneira mais simples.

Apesar dos nossos defeitos, precisamos enxergar que somos pérolas únicas no teatro da vida e entender que não existem pessoas de sucesso ou pessoas fracassadas. O que existe são pessoas que lutam pelos seus sonhos ou desistem deles. (Augusto Cury)

Resumo

Atualmente vivemos a era da informação e diariamente grandes empresas e startups enfrentam problemas em relação à quantidade de dados processados, ou seja, têm grandes desafios em lidar com uma quantidade grande de dados e também não conseguem convertê-lo em informações relevantes de forma otimizada. Uma das áreas mais antigas que mais produzem dados é o jornalismo, onde desde o passado todas as informações eram transmitidas por papel e são atualmente feitas, em sua maioria, por meio da internet. Com a crescente quantidade de informações jornalisticas disponíveis online, é fundamental ter um sistema automatizado capaz de categorizar notícias com precisão e eficiência. Isso permitirá aos usuários acessarem conteúdo relevante de maneira mais rápida e aos portais de notícias aprimorarem sua organização e distribuição de informações. O objetivo deste trabalho é criar um categorizador de notícias fornecendo a categoria mais adequada àquele texto a partir das categorias mais abordadas atualmente pelos portais de notícias principais. Com o intuito de atingir este objetivo, serão realizadas as atividades de pré-processamento para maximizar a possibilidade de sucesso na tarefa de classificação. Isso pressupõe-se a necessidade de uma base de dados consolidadas e boas escolhas de métodos e técnicas que sejam efetivamente aplicadas e consequentemente tenhamos um bom desempenho nos resultados. Objetiva-se focar em técnicas atuais do estado da arte em aprendizado profundo para resolver e validar se são eficazes para alcançar os objetivos anteriormente apresentados.

Palavras-chave: classificação, categorias, noticias, dados, processamento de linguagem natural, modelo preditivo, transformadores

Abstract

We currently live in the information age and daily large companies and startups face problems in relation to the amount of data processed, that is, they have great challenges in dealing with a large amount of data and also fail to convert it into relevant information in an optimized way. One of the oldest areas that produce the most data is journalism, where since the past all information was transmitted by paper and is currently done mostly through the internet. With the growing amount of journalistic information available online, it is essential to have an automated system capable of categorizing news accurately and efficiently. This will allow users to access relevant content faster and news portals to improve their organization and distribution of information. The aim of this work is to create a news categorizer by providing the most suitable category for that text from the categories currently most addressed by the main news portals. In order to achieve this goal, pre-processing activities will be performed to maximize the possibility of success in the classification task. This presupposes the need for a consolidated database and good choices of methods and techniques that are effectively applied and consequently have a good performance in the results. The objective is to focus on current state-of-the-art techniques in deep learning to solve and validate whether they are effective in achieving the objectives previously presented.

Key-words: classification, categories, news, data, natural language processing, predictive model, transformers

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

BERT	EN-US: Bidirectional Encoder Representations for Transformers / PT-BR: Representações Bidirecionais de Codificadores para Transformadores
NLP	EN-US: Natural Language Processing / PT-BR: Processamento de Linguagem Natural
ELT	EN-US: Extract, Load, Transform / PT-BR: Extração, Carga e Transformação
ETL	EN-US: Extract, Transform, Load / PT-BR: Extração, Transformação, Carga
DML	EN-US: Data Manipulation Language / PT-BR: Linguagem de Manipulação de Dados
SQL	EN-US: Structured Query Language / PT-BR: Linguagem de Consulta Estruturada
API	EN-US: Application Programming Interface / PT-BR: Interface de Programação de Aplicações
CSS	EN-US: Cascading Style Sheets / PT-BR: Folhas de Estilo em Cascata
XPATH	EN-US: XML Path Language / PT-BR: Linguagem de Caminho XML
XML	EN-US: eXtensible Markup Language / PT-BR: Linguagem de Marcação Extensível
BS4	Beautiful Soup 4
CNN	EN-US: Convolutional Neural Network / PT-BR: Rede Neural Convolucional
RNN	EN-US: Recurrent Neural Network / PT-BR: Rede Neural Recorrente
LSTM	EN-US: Long Short-Term Memory / PT-BR: Memória de Curto Prazo de Longa Duração
GRU	EN-US: Gated Recurrent Unit / PT-BR: Unidade Recorrente Gateada
LLM	EN-US: Large Language Models / PT-BR: Modelos grandes de linguagem

HTML	EN-US: HyperText Markup Language / PT-BR: Linguagem de Marcação de Hipertexto
CSV	EN-US: Comma-Separated Values / PT-BR: Valores Separados por Vírgula
URL	EN-US: Uniform Resource Locator / PT-BR: Localizador Uniforme de Recursos
HF	Hugging Face
TF-IDF	EN-US: Term Frequency–Inverse Document Frequency / PT-BR: Frequência do Termo–Inverso da Frequência nos Documentos

Sumário

Parte I

Pré-texto

1 Introdução

Na atual era da informação, o consumo de notícias tornou-se parte integrante de nossa vida cotidiana (ROSCOE, 2021). Com o rápido crescimento das fontes de notícias online e o fluxo contínuo de artigos de notícias, tornou-se cada vez mais desafiador organizar, navegar e descobrir com eficiência o conteúdo relevante das notícias. A classificação de notícias oferece uma solução para esse problema, atribuindo automaticamente artigos de notícias a categorias predefinidas com base em seu conteúdo.

Esta pesquisa se concentra na interseção de três áreas importantes: aprendizado de máquina, processamento de linguagem natural e transformadores. Os algoritmos de aprendizado de máquina apresentaram avanços notáveis em vários domínios, inclusive em tarefas de processamento de linguagem natural. O processamento de linguagem natural, por outro lado, trata da interação entre computadores e linguagem humana, permitindo que as máquinas compreendam, processem e gerem linguagem humana.

Os transformadores, um tipo específico de modelo de aprendizagem profunda, surgiram como uma abordagem poderosa no processamento de linguagem natural. Eles revolucionaram o campo ao introduzir o conceito de mecanismos de autoatenção, que permite capturar relações contextuais entre palavras ou *tokens* em um texto. Os transformadores, como o BERT e o GPT, obtiveram resultados de ponta em várias tarefas relacionadas a idiomas, incluindo a classificação de textos.

A categorização de notícias serve como uma ferramenta fundamental para estruturar e organizar o volume cada vez maior de artigos de notícias disponíveis online. Ela permite que os usuários naveguem com eficiência por abundantes quantidades de informações e encontrem artigos relevantes para seus interesses. Os artigos de notícias categorizados também facilitam o desenvolvimento de sistemas de recomendação de conteúdo personalizado, oferecendo aos usuários sugestões personalizadas com base em suas preferências e histórico de navegação.

Além disso, a categorização de notícias contribui para a análise de tendências, permitindo visões sobre tópicos populares, análise de sentimentos e monitoramento da opinião pública. Ao agregar artigos em diferentes categorias, torna-se possível identificar tendências emergentes, analisar mudanças no sentimento do público e obter uma compreensão mais profunda da dinâmica da sociedade.

Na era do big data (LOHR, 2012), a categorização de notícias enfrenta novos desafios e oportunidades. O volume de artigos de notícias gerados diariamente atingiu níveis sem precedentes, exigindo técnicas de processamento escalonáveis e eficientes. A análise em tempo real torna-se crucial, pois os artigos de notícias são publicados continuamente,

exigindo classificação oportuna para acompanhar o cenário dinâmico das notícias. Além disso, a variedade de conteúdo de notícias, abrangendo tópicos como política, esportes, entretenimento, saúde e outros, exige modelos adaptáveis capazes de lidar com diversos tipos de dados textuais.

1.1 Objetivos

O objetivo deste trabalho é classificar notícias explorando aplicação de transformadores no contexto de aprendizagem profunda. Ao extrair recursos significativos de artigos de notícias e treinar modelos de aprendizado de máquina, busca-se automatizar o processo de categorização. As principais técnicas para processamento eficiente, adaptação de modelos e manipulação da velocidade, do volume e da variedade de dados de notícias serão investigadas.

Por meio do desenvolvimento de modelos avançados de categorização de notícias, esta pesquisa tem o potencial de aprimorar significativamente a organização das informações, a recomendação de conteúdo e a análise de tendências. Ao aproveitar efetivamente o poder da aprendizagem profunda, podem-se abrir novas possibilidades no consumo personalizado de notícias e obter *insights* mais profundos sobre o cenário de notícias em constante evolução. Pretende-se seguir os seguintes objetivos específicos:

- Selecionar sites de notícias
- Minerar dados das notícias
- Pré-processar os dados
- Rotular os dados
- Analisar os dados
- Aplicar modelos preditivos com transformadores
- Classificar as notícias
- Avaliar os resultados

1.2 Estrutura do trabalho

O trabalho segue a seguinte estrutura:

- Pré-texto

  - Introdução - Esta seção se concentra em apresentar o conteúdo inicial da pesquisa descrita no projeto. Ela define o contexto e fornece informações básicas sobre o tópico da pesquisa, destacando sua relevância e os objetivos do estudo, além de incluir uma declaração de problema e questões de pesquisa que serão abordadas.
  - Referencial teórico - Nesta seção, é fornecida uma análise abrangente da literatura e das teorias relevantes relacionadas ao contexto de processamento de linguagem natural e aprendizagem profunda. Este referencial resume e sintetiza estudos, estruturas e conceitos existentes que sejam pertinentes à pesquisa.

- Texto e Pós-texto

  - Materiais e métodos - Esta seção descreve os materiais, ferramentas e técnicas utilizados na pesquisa. Ela descreve as fontes de dados, os métodos de coleta de dados e quaisquer procedimentos ou protocolos experimentais que serão seguidos. Este capítulo também aborda as técnicas de análise de dados e os métodos estatísticos que serão aplicados para analisar os dados coletados.
  - Cronograma - A seção de cronograma fornece um cronograma ou plano detalhado das atividades e marcos da pesquisa. Ela descreve as tarefas específicas, suas durações estimadas e o cronograma proposto para a conclusão de cada tarefa. Esse cronograma ajuda a organizar e gerenciar o processo de pesquisa, garantindo que o projeto permaneça no caminho certo e seja concluído dentro do prazo determinado.

2 Referencial teórico

Decorrente do trabalho desenvolvido, algumas tecnologias, técnicas e projetos precisam ser estudados inicialmente, tais estudos são listados neste capítulo e são escolhidos a partir do conhecimento mais concreto e prático desse ferramental.

2.1 Trabalhos correlatos

Dentre os trabalhos correlatos, podem-se citar três que se identificam bastante com o trabalho que será desenvolvido, dentre eles serão observadas ferramentas que terão grande importância para o sucesso das tarefas que serão prestadas, como a questão do processamento, modelos preditivos e tentativas que foram feitas focados no contexto da monografia atual. Os artigos são listados abaixo e são descritos como foram desenvolvidos e como sustentam o trabalho de categorização de notícias.

A questão do processamento da linguagem natural e da aplicação dos modelos de aprendizado de máquina de uma maneira eficiente é uma das tarefas essenciais e mais árduas no contexto de trabalho de um cientista de dados, fazendo com que muitos destes desafios atrapalhem o processo de trabalho e progresso de entregas no contexto profissional. O artigo *Large-Scale News Classification using BERT Language Model: Spark NLP Approach* (NUGROHO; SUKMADEWA; YUDISTIRA, 2021) explicita como a aplicação da ferramenta Spark NLP (KOCAMAN; TALBY, 2021) otimizou o processo de treinamento do modelo BERT. O Spark é uma ferramenta focada no processamento de dados de maneira distribuída e paralela, já o Spark NLP é um pacote do Spark focado nas tarefas de processamento de linguagem natural e já possui bibliotecas importantes para essa tarefa, como os transformadores, que são tecnologias que serão melhor descritas nos próximos capítulos.

Alguns trabalhos já desenvolveram soluções parecidas com as que serão apresentadas neste trabalho, tais como o *NewsBERT: Distilling Pre-trained Language Model for Intelligent News Application* (WU et al., 2021) e o *A Complete Process of Text Classification System Using State-of-the-Art NLP Models* (DOGRA et al., 2022). Nos dois trabalhos a aplicação do modelo de rede neural chamado BERT é implementado, sendo o estado na arte no contexto de modelo de aprendizado de máquina focado em linguagem natural, porém serão abordados e testados outros novos, como o GPT e o LLaMA, os quais são utilizados em tecnologias atuais como o ChatGPT e o MetaAI respectivamente.

A partir dos trabalhos citados pode-se ter uma base de como iniciar o processo de desenvolvimento além de práticas ad-hoc de pesquisa que serão feitas visando diferentes

resultados. Existe outro contexto principal para o início do processo de pesquisa principal que é a necessidade da mineração de dados de notícias brasileira e rotulação dos dados. A partir do livro *Web Scraping with Python* (MITCHELL, 2015) observam-se ferramentas que podem otimizar o processo de extração de dados dos portais de notícias, além de já manipular os dados necessários para que os dados de interesse já sejam capturados. Será utilizada a biblioteca Scrapy do Python para desenvolver as tarefas de web *scrapping*, além de definir *crawlers* específicos para cada portal acessado. Este processo será melhor descrito nos próximos capítulos.

2.2 Distribuição de pesquisas na área

A Tabela 1 representa a distribuição de pesquisas em diferentes acervos demonstrando como o conteúdo apresentado nesta pesquisa reflete como ela é de grande valia para o desenvolvimento acadêmico das áreas de processamento de linguagem natural e da aplicação dos classificadores em diferentes contextos textuais.

Tabela 1 – Distribuição das pesquisas por área de pesquisa em diferentes acervos

Plataforma de busca	**Conteúdo**	**Quantidade**
ScienceDirect	Natural language processing	336909
Periódicos Capes	Text classification	825130
ScienceDirect	Text classification	295992
Periódicos Capes	Natural language processing	90526
IEEE Explore	Natural language processing	50222
ScienceDirect	News text classification	22931
IEEE Explore	Text classification	17179
IEEE Explore	News text classification	1084
Periódicos Capes	News text classification	897

2.3 Levantamento das ferramentas

As ferramentas utilizadas focam num processo de extração, transformação, armazenamentos e aplicação dos dados em modelos preditivos, onde em cada etapa será citada um dos processos aplicados, ou seja, inicialmente o processo será focando no contexto de ETL, já nos passo futuros, transformações serão necessárias para o processo analítico do dado e aplicações do dado nos modelos preditivos.

2.3.1 Python 3

O Python[1] é uma linguagem de programação utilizada em vários contextos, ela foi escolhida no contexto deste trabalho, pois supre as tarefas que serão demandadas, utilizada no processo de mineração, processamento, transformação dos dados e aplicação dos modelos preditivos.

2.3.2 Docker

O Docker[2] otimiza o processo de conteinerização, criando um ambiente segregado que facilita questões de instalação e testes de compatibilidade. O Docker foi utilizado para virtualizar a execução de *scripts* e o banco de dados que armazena os dados minerados, sendo possível armazenar os dados localmente e assim criar backups em volumes, instanciando outras imagens que utilizam o mesmo volume que foi armazenado localmente.

2.3.3 Postgres

O banco de dados Postgres[3] foi instanciado a partir de uma imagem pronta do repositório Docker e tem o objetivo de armazenar os dados minerados e fazer manipulações dos dados por meio apenas da sintaxe DML com o uso da linguagem SQL.

2.3.4 Kaggle

O Kaggle[4] é uma plataforma e comunidade *online* para competições de ciência de dados e aprendizado de máquina. Ele oferece um ambiente colaborativo onde cientistas de dados, pesquisadores e entusiastas podem participar de vários desafios de dados e resolver problemas do mundo real. O Kaggle oferece um ambiente de Jupyter Notebook baseado em nuvem chamado Kaggle Notebooks. Ele permite que os usuários criem, editem e executem códigos em um ambiente colaborativo. Os usuários podem compartilhar facilmente seus notebooks, que geralmente incluem análise de dados, visualização e modelos de aprendizado de máquina, com a comunidade Kaggle.

2.4 Levantamento das bibliotecas

Dentre as bibliotecas já citadas anteriormente, neste capítulo serão descritas de maneira mais detalhada as bibliotecas utilizadas para toda a pipeline da pesquisa atual, sendo separadas pelo contexto que foram ou serão aplicadas.

[1] <https://www.python.org/>
[2] <https://docs.docker.com/>
[3] <https://www.postgresql.org/>
[4] <https://www.kaggle.com/>

A linguagem principal utilizada neste projeto é o Python 3, sendo utilizando por meio de conteinerização Docker. A partir da escolha desta linguagem já havia sido consideradas quais bibliotecas seriam necessárias para execução do trabalho. As sessões abaixo descrevem estas bibliotecas.

2.4.1 Scrapy

Scrapy[5] é uma biblioteca Python em código aberto com objetivo principal de fazer *web-scraping*, extraindo dados de websites e APIs, automatizando e facilitando o processo de mineração de dados sem a intervenção manual.

No projeto, o Scrapy foi utilizado para minerar as notícias do G1[6], UOL[7], e CNN Brasil[8], onde foram feitos *scripts* chamados *spiders* que tinham o objetivo de percorrer as páginas de indexação das notícias e capturar as informações necessárias.

O uso desta biblioteca facilita também o pré-processamento e o pós-processamento dos dados, focando em utilizar seletores por meio de classes CSS ou tags XPATH para mapear o conteúdo de interesse e assim capturar esta informação, já no contexto do pós-processamento, existe a liberdade de escolher o que fazer com a informação selecionada.

Houve a necessidade de utilizar junto ao Scrapy as bibliotecas BeautifulSoup e LXML, pois as páginas de indexação dos sites escolhidos eram estruturadas em XML. Como o Scrapy apenas fazia o *parse* de páginas em HTML, então houve a necessidade de alterar o *parse* padrão usando o LXML como analisador sintático de XML na biblioteca BS4, que fazia a análise da página e repassava para o Scrapy a lista de sites que estavam listados na página de indexação, todos eles agora em HTML.

2.4.2 Psycopg2

O Psycopg2[9] é uma biblioteca que tem o objetivo de criar uma interação entre a linguagem Python e o banco de dados Postgres, facilitando o processo de comunicação entre as duas interfaces, fazendo o armazenamento e validação dos dados a serem inseridos na tabela de destino.

Houve a necessidade de utilizar a biblioteca "psycopg2-binary" ao invés da biblioteca padrão, pois como o banco de dados trata-se de uma instância conteinerizada, o psycopg2 não conseguia acessar os arquivos de instalação do banco de dados, já a biblioteca utilizada é uma distribuição binária pré-compilada do "psycopg2", o que significa

[5] <https://scrapy.org/>
[6] <https://g1.globo.com/>
[7] <https://www.uol.com.br/>
[8] <https://www.cnnbrasil.com.br/>
[9] <https://www.psycopg.org/>

que ele já está compilado e pronto para uso, sem a necessidade de compilar o código-fonte durante a instalação.

2.4.3 SQLAlchemy e Pandas

A biblioteca Pandas[10] é construída em Python e serve para manipulação e análise de dados. Ela fornece estruturas de dados e ferramentas de análise de dados de alto desempenho e fáceis de usar. Suas principais estruturas de dados são *DataFrames* (estruturas tabulares bidimensionais) e *Series* (matrizes unidimensionais). A biblioteca oferece funcionalidades para limpeza de dados, transformação, indexação, seleção, agrupamento, cálculos estatísticos, fusão de dados e visualização. Ele é amplamente utilizado em tarefas de ciência e análise de dados.

No contexto do projeto, a biblioteca Pandas foi utilizada como o conversor do banco de dados em um arquivo mais conciso e fácil de ser acessado, sem a necessidade de executar o contêiner Docker do banco de dados para interagir com os dados. Na questão da conversão a função "*read_sql_table*" foi utilizada para fazer a leitura dos dados da nossa tabela de notícias final.

Houve a necessidade da instalação da biblioteca SQLAlchemy[11] para que o Pandas conectasse com o banco de dados, esta é uma biblioteca Python de código aberto que fornece um conjunto de ferramentas e abstrações para trabalhar com bancos de dados. Ela funciona como uma estrutura de mapeamento objeto-relacional, o que significa que permite que os desenvolvedores interajam com bancos de dados usando objetos e métodos Python, abstraindo as consultas SQL subjacentes e os detalhes específicos do banco de dados.

Com o acesso à tabela, agora é possível consumir os dados e converter em um arquivo que separa os valores por vírgula (CSV) e podendo ser utilizado futuramente de uma maneira mais acessível em comparação a um banco de dados.

2.4.4 Jupyter Notebook

O Jupyter Notebook[12] é um aplicativo da Web de código aberto para criar e compartilhar documentos interativos. Ele é compatível com várias linguagens de programação e é popular na comunidade de ciência de dados e análise exploratória. Seus principais recursos incluem uma interface interativa, combinando células de código e texto, execução de código em tempo real, visualização de resultados, flexibilidade com várias bibliotecas e fácil compartilhamento e colaboração.

[10] <https://pandas.pydata.org/>
[11] <https://www.sqlalchemy.org/>
[12] <https://jupyter.org/>

O Jupyter oferece um ambiente de fácil utilização para análise de dados, prototipagem e compartilhamento de códigos e resultados e por esta razão será utilizado dentro do ambiente Kaggle para otimizar o desenvolvimento da análise dos dados, aplicação do modelo, prototipação e documentação conjunta das etapas desenvolvidas.

2.4.5 Pandarallel

O Pandarallel[13] é uma biblioteca cujo objetivo é processar datasets Pandas de maneira paralela entre os diferentes núcleos de processamento, funcionalidade a qual não é nativa da biblioteca Pandas.

Esta biblioteca serviu para efetuar o pré-processamento do texto nas tarefas de limpagem dos dados na atividade de exploração de dados.

2.4.6 Transformers

A biblioteca *Transformers*[14] do Hugging Face é uma biblioteca de software para processamento de linguagem natural que implementa modelos de transformadores. Os transformadores são uma arquitetura de rede neural que se tornou dominante na área de NLP nos últimos anos, alcançando resultados de estado da arte em uma ampla gama de tarefas, incluindo tradução automática, resumo de texto, geração de texto e perguntas e respostas.

A biblioteca fornece uma ampla variedade de modelos pré-treinados, que podem ser usados para tarefas de NLP específicas. Esses modelos são treinados em conjuntos de dados de grandes dimensões, o que lhes permite alcançar resultados de alta qualidade. Ela também fornece ferramentas para treinar e usar seus próprios modelos de transformadores. Essas ferramentas permitem que você personalize seus modelos para atender às suas necessidades específicas.

Esta biblioteca será utilizada neste trabalho exatamente para facilitar nos testes dos modelos e também para utilização dos modelos abertos de LLM que são compatíveis inteiramente com a biblioteca

2.4.7 Datasets

A biblioteca *datasets*[15] do Hugging Face é uma biblioteca de software para processamento de linguagem natural que fornece acesso a um grande catálogo de *datasets* de NLP. Os *datasets* são conjuntos de dados usados para treinar e avaliar modelos de

[13] <https://github.com/nalepae/pandarallel>
[14] <https://huggingface.co/docs/transformers/index>
[15] <https://huggingface.co/docs/datasets/index>

NLP. Ela fornece uma ampla variedade de recursos para facilitar o desenvolvimento de aplicações de NLP baseadas em *datasets*. Esses recursos incluem:

A biblioteca fornece um amplo catálogo de datasets de NLP, que cobrem uma ampla gama de tarefas e domínios. Esses *datasets* são coletados e preparados por uma equipe de especialistas em NLP, o que garante sua qualidade e confiabilidade. Além disto, ela fornece uma API simples e intuitiva que facilita o acesso e o uso de *datasets*. A API é baseada em Python e é compatível com as principais bibliotecas de NLP, como o *Transformers* e o PyTorch.

A biblioteca será utilizada neste projeto para armazenar os dados coletados em uma estrutura compatível com a biblioteca transformers, otimizando o treinamento, pois evita que transformações dos dados sejam feitas durante o processo de treinamento.

2.5 Modelos estudados e conceitos relevantes

2.5.1 Estado da arte - Processamento de linguagem natural

Classificar o texto em uma categoria conhecida é uma tarefa que faz parte do escopo do **processamento de linguagem natural** (VAJJALA et al., 2020), existem várias tarefas que o NLP exige antes para que a máquina compreenda o contexto do texto que está sendo processado, está é a tarefa principal do NLP, fazer com que a máquina tenha uma compreensão semelhante ao que o ser humano entende, esta atividade depende em converter de diferentes formas o texto em estruturas que podem ser reconhecidas por meio de um computador, otimizando o processo de entendimento da máquina sobre o texto que será processado.

Existem diferentes maneiras de executar esta tarefa de conversão e cada estratégia carrega objetivos e vantagens diferentes, mas o mais importante é que o contexto também seja compreendido, verificando o significado de cada palavra e sua respectiva ordem, a razão do contexto ser entendido é que ele faz com que problemas comuns do NLP sejam evitados, como a ambiguidade e diversidade da linguagem.

Diferentes modos de implementação do NLP foram desenvolvidos no decorrer do tempo, inicialmente com **modelos baseados em heurísticas**, sendo um processo guiado por meio de regras para definir o contexto das palavras e assim identificar possíveis características, como o sentimento de uma frase a partir da quantidade de palavras negativas nela. Um exemplo de ferramenta que utiliza esta estratégia é a Wordnet (MILLER, 1995), que possui um banco de dados que armazena as palavra e suas respectivas relações semânticas.

Decorrente da estratégia anterior, foi difundida a aplicação de **modelos de *machine learning*** para soluções de NLP, buscando agora meios mais autônomos de se pro-

cessar e analisar a linguagem natural, tanto por modelos supervisados ou não-supervisados, dependendo ou não de uma *label* que caracteriza o texto, respectivamente, o aprendizado de máquina neste contexto segue sempre três etapas comuns: extrair recursos do texto, usar a representação de recursos para treinar um modelo, principalmente usando tensores, e avaliar o resultado do modelo. Alguns exemplos de modelos de aprendizado de máquina são o *Naive Bayes* (ZHANG; LI, 2007), *Máquina de vetores de suporte* (JOACHIMS, 2002) e *Modelo oculto de Markov* (BAUM; PETRIE, 1966), estes modelos buscam uma descrição estatística de como os dados se comportam, sendo um trabalho do experimentador verificar quais são os parâmetros ideais para aquele modelo naquela situação.

Com a pesquisa de soluções mais modernas, houve a identificação dos modelos que utilizam **redes neurais**, que se inspiram no modelo de cognição a partir dos neurônios do ser humano, estruturando camadas de neurônios que lidam com as informações e assim compartilham seus resultados para outra camada. As redes neurais são aplicadas em diversas áreas, como as áreas de processamento de imagens, áudios e também textos.

Para cada contexto e dentre estes, cada tarefa, são pesquisadas redes neurais de aprendizado profundo ideais, como as **redes neurais convolucionais** para lidar com imagem e **redes neurais recorrentes** para tratar textos, áudio e séries temporais. As RNN mais conhecidas e pesquisadas são as de **memória de curto e longo prazo** e **unidades recorrentes com portas fechadas**, resolvendo as possíveis demandas relacionadas ao entendimento do contexto, pois utilizam um processo sequencial do processamento de texto ou de séries temporais para identificar como o passado interfere no contexto da frase, ponderando o sentido da frase e não apenas palavras quebradas sem a interpretação da frase em si.

2.5.2 Por que utilizar transformadores?

Percebe-se que as soluções de RNN resolvem quase todos os problemas em relação ao processamento de textos, porém existem problemas em utilizá-los, por exemplo, a solução de LSTM processa o texto de maneira linear, muito parecido com o comportamento do ser humano, isto é, exige que o processamento seja feito também de maneira linear, impedindo a aplicação de soluções paralelas e distribuídas para treinamento do modelo, além de existir uma capacidade cognitiva de palavras processadas, perdendo sentidos em textos muito grandes. Abaixo está um trecho mais descritivo do livro *deeplearningbook* em que o autor explica de maneira mais teórica a problemática das RNN que utilizam LSTM, lê-se *token* neste cenário como uma sequência de caracteres em algum documento específico agrupados como uma unidade semântica útil para processamento (MANNING; RAGHAVAN; SCHUTZE, 2008):

> Teoricamente, as informações de um *token* podem se propagar arbitrariamente ao longo da sequência, se em todos os pontos o estado continuar

> a codificar informações contextuais sobre o *token*. Mas, na prática, esse mecanismo é imperfeito: devido em parte ao problema do desaparecimento do gradiente, o estado do modelo no final de uma frase longa geralmente não contém informações precisas e extraíveis sobre os *tokens* anteriores. (DSA, 2022b)

A partir destas problemáticas, os pesquisadores da Google publicaram uma pesquisa chamada *Attention is all we need* (VASWANI et al., 2017) focada em resolver o problema do processamento linear e sequencial para compreender o contexto do texto todo em si e ser mais assertivo no entendimento geral, além de lidar com dependências de longo alcance com facilidade e não apenas focar no conteúdo mais próximo como o LSTM. A razão do título da pesquisa focar no contexto de atenção é principalmente pelo fato de que estes mecanismos permitem focar diretamente no estado de palavras anteriores e assim se baseiem para que o estado do contexto seja mantido, ponderando parcialmente os estados anteriores e validando seu sentido conforme o peso da palavra atual.

Em uma frase comum o peso de uma palavra reflete em outra, criando uma relação entre as palavras, onde cálculos matriciais são feitos por meio de tensores para verificar questões do peso, valor e outros atributos de cada palavra, sendo processado a partir de funções de ativação que reforçam um aprendizado de maneira não-linear, evitando que o comportamento de uma rede neural seja idêntico a uma regressão linear comum (DSA, 2022a).

Decorrente da otimização dos treinamentos de modelos, houve a carga em massa de grandes empresas com modelos treinados com muitas palavras, utilizando-se a própria base de dados web para o seu treinamento e criação do modelo, estes modelos são nomeados **modelos grandes de linguagem**, estes modelos atualmente são altamente aplicados em diferentes contextos e refinados para tarefas mais específicas, estas tarefas divergem para cada contexto, no entanto, são generalistas inicialmente, exemplos famosos são atualmente o BERT (*Bidirecional Encoder Representations from transformers*), GPT (*Generative Pretrained Transformer*) e LLaMa (*Large Language Model Meta AI*), respectivamente desenvolvidos pelas empresas Google, OpenAI e Meta. Estes modelos são customizações da ideia inicial dos transformadores e buscam otimizar interpretações e geração de texto.

Os modelos acima já foram customizados pela própria comunidade e são compartilhados através da internet, onde são retreinados com diferentes palavras e bases de dados, criando geradores de textos generalistas e especialistas, ou seja, são vários modelos compartilhados diariamente para sucesso de tarefas específicas sem a necessidade de treinamento inicial, usando estratégias de *fine-tuning*, solução a qual o modelo é treinado novamente para otimização dos resultados na tarefa específica que deseja resolver.

Estes três modelos serão aplicados no projeto e avaliados como se performam, comparando seus resultados, dificuldades, complexidade e velocidade de predição, para

consumir estes modelos serão utilizados os modelos hospedados na plataforma Hugging Face[16], onde serão baixados e estudados para implementação na tarefa de categorização das notícias mineradas.

[16] <https://huggingface.co/>

Parte II

Texto e Pós Texto

3 Materiais e métodos

3.1 Visão geral

Neste capítulo serão descritos todas as atividades que envolvem a metodologia da pesquisa atual, buscando sanar dúvidas de como é a execução do projeto e seus detalhes de construção, em cada passo será apresentado um diagrama que reflete como é a esteira de atividades. A Figura 1 é um diagrama inicial de como é a construção geral do projeto.

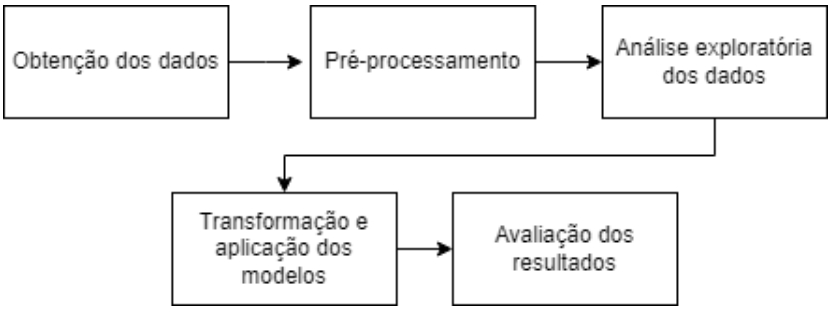

Figura 1 – Metodologia do projeto

3.2 Obtenção dos dados

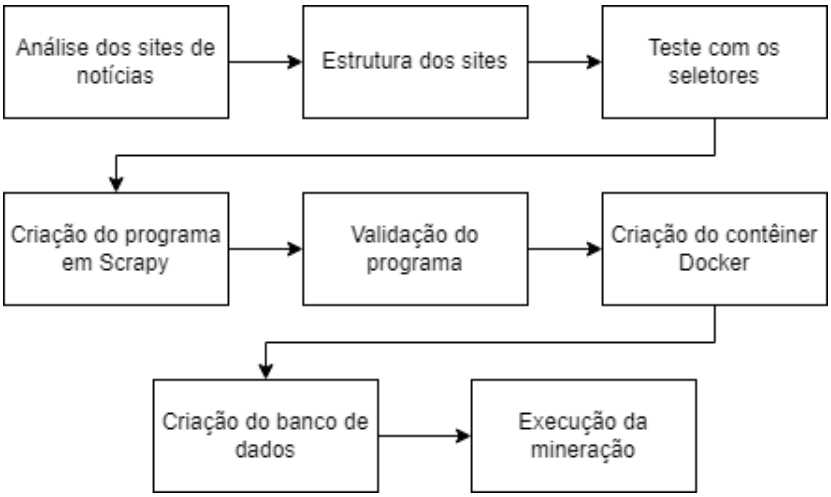

Figura 2 – Processo de obtenção dos dados

A Figura 2 se refere ao modo como foi feito o *crawler* de notícias, buscando soluções que sejam eficientes no processo de obtenção dos dados, testes foram feitos para verificar qual seria a forma ideal de se construir esta arquitetura, sendo um processo não-dinâmico, ou seja, o programa é executado uma vez e todas as páginas listadas nos sites de indexação

são mineradas, havendo uma necessidade de rodar o programa novamente caso alguma notícia seja adicionada.

Na atividade de análise dos sites de notícias, buscamos verificar sites que forneciam uma boa infraestrutura em relação à categorização das notícias, neste contexto, foram listados e definidos o **G1**, **CNN**, **UOL**, **Estadão** e **Terra**. O próximo passo foi estudar a estrutura dos sites.

No quesito de estrutura dos sites, alguns deles tinham um comportamento dinâmico, onde não conseguiríamos mapear por meio apenas de requisições, sendo necessárias ferramentas mais complexas, como o *Selenium*, em razão desta complexidade foram **removidos** os sites Terra e Estadão, no entanto, depois de várias buscas e experimentações, conseguimos encontrar sites de mapeamento de índices com todas as URLs que estavam nos sites escolhidos e assim poderíamos utilizar apenas o site de mapeamento para resgatar os dados das notícias, podemos ver na Fig. 3 como estes sites são estruturados em relação à plataforma do G1, por exemplo.

Figura 3 – Estrutura do site de mapeamento de índices

A partir do acesso às informações e também conhecimento melhor sobre a estrutura dos sites, pode-se iniciar a experimentação com os seletores das notícias, verificando o que e como era capturado, a base para o teste foi a documentação de seletores do Scrapy (SELECTORS. . . ,).

O próprio Scrapy fornece uma interface de comando que otimiza o processo de *scraping* sem a necessidade de executar um código, onde podemos utilizar o comando **scrapy shell <URL>** para interagir com o site e testar como cada seletor funciona e qual seria o seu resultado, a partir disto pôde-se estruturar como cada seletor seria em cada página de interesse.

A partir do sucesso nos testes dos seletores, foi desenvolvido o programa em Scrapy e também estruturado quais seriam os *spiders* - estrutura de captura de dados do Scrapy - de cada site de interesse, sendo eles o G1, UOL e CNN, com isso houve a primeira execução e observação dos dados coletados em um arquivo CSV para fins de teste. A Figura 4 representa um dos dados coletados no contexto dos testes iniciais.

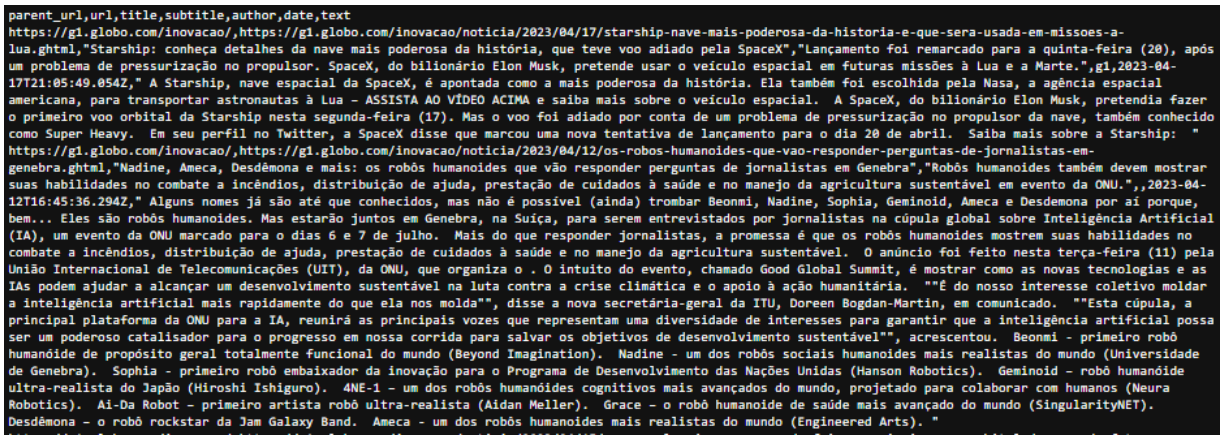

Figura 4 – Exemplo de dado dos testes iniciais

Decorrente da captura e observação dos dados, houve a interpretação de que os dados estavam corretos, além de validar quantidades dos dados e possíveis conteúdos que não seriam capturados, como o autor ou data e hora da última atualização da notícia, com isso houve uma comprovação de que a coleta estava executando corretamente e capturando os dados necessários de maneira correta.

Testes foram feitos em relação à etapa de extração, abaixo são listadas as tentativas e os problemas encontrados em cada situação:

- Em razão de uma falsa percepção da quantidade de dados, já que não se sabia quantos dados seriam minerados, houve uma escolha inicial de se trabalhar com uma arquitetura orientada a eventos, utilizando o Kafka, porém percebeu-se a dificuldade de se enviar conteúdos como textos, além de integrar esta ferramenta com o Scrapy, já existindo soluções, porém não se obteve sucesso com o uso delas, como a "scrapy-kafka"e a "os-scrapy-kafka-pipeline". É possível esta solução, porém no contexto de implantação da pesquisa, não é necessário criar uma arquitetura tão robusta que exige uma grande complexidade e robustez na configuração.
- Outro problema foi tentar identificar como evitar a redundância dos registros de cada notícia ao executar o programa de mineração de dados novamente. Neste contexto, foi pensado inicialmente em utilizar uma arquitetura Hadoop distribuída e armazenar os dados em Parquet, facilitando um consumo futuro destes dados por meio do Spark, porém nesta arquitetura seria muito complexo validar se o registro já foi armazenado ou não.

Com o sucesso da construção do programa depois da análise dos testes, agora é interessante que este programa execute e armazene os dados, para evitar a necessidade de executar o programa e o banco, foi criado um contêiner Docker com duas imagens, uma do **produtor** - representando o programa em Scrapy que captura as notícias - e outra da base de dados em Postgres que armazenava o volume dos dados de maneira externa, sem a necessidade de criar uma configuração do banco de dacos a cada vez que a imagem for instanciada, guardando em disco tanto os dados quanto sua respectiva configuração.

```
self.cur.execute(
    """
    CREATE TABLE IF NOT EXISTS news (
        parent_url VARCHAR(255),
        url VARCHAR(255) PRIMARY KEY,
        title VARCHAR(255),
        subtitle VARCHAR(255),
        author VARCHAR(255),
        date VARCHAR(255),
        text TEXT
    )
    """
)
```

Figura 5 – Criação da tabela para inserção dos dados das notícias

O Docker facilitou abstrações do banco de dados e a captura de dados, na Figura 5 temos exemplos do código do programa Scrapy que resumem como a tabela foi criada e o modo como a inserção dos registros ocorria enquanto a notícias eram capturadas (Fig. 6). O armazenamento fornecido pela biblioteca não resolvia as demandas que a tarefa requeria, por isso foi utilizado um banco de dados para armazenar os dados e evitar a redundância dos registros, optando por utilizar uma chave primária na URL e verificamos se houve algum conflito, caso houvesse não armazenaríamos aquele dado.

```
self.cur.execute(
    """
    INSERT INTO news (parent_url, url, title, subtitle, author, date, text)
    VALUES (%s, %s, %s, %s, %s, %s, %s)
    ON CONFLICT (url) DO NOTHING
    """,
    (
        item["parent_url"],
        item["url"],
        item["title"],
        item["subtitle"],
        item["author"],
        item["date"],
        item["text"],
    ),
)
```

Figura 6 – Inserção das notícias e verificação da chave primária

A partir da estrutura feita, agora é necessário executar o contêiner para obter todas as notícias indexadas nas páginas de mapeamento, fazendo uma busca exaustiva até onde não obter mais resultados ao progredir de página. Para captura completa, cerca de **16 horas** contínuas foram necessárias e **720977** notícias foram capturadas e armazenadas.

3.3 Pré-processamento

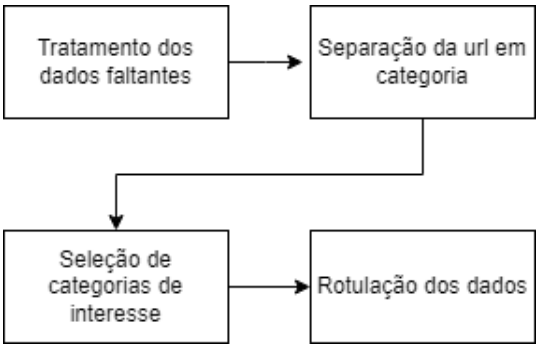

Figura 7 – Pré-processamento dos dados capturados

Os dados foram armazenados no banco de dados e agora podem ser estudados para uma melhor validação, exigindo que alguns pré-processamentos sejam feitos para que os dados inseridos no processo de aplicação do modelo esteja com uma boa qualidade.

A nossa variável de interesse no contexto deste trabalho é a categoria da notícia e neste caso esta categoria está incorporada na URL da notícia, sendo feito uma separação desta URL, criando uma nova coluna com as etiquetas de categoria de cada notícia.

Antes de começar a análise textual, foi observado que muitas notícias capturadas focam em outros tipos de mídias que não são as textuais, como vídeos e documentários, não possuindo um corpo de texto, por essa razão foi elaborado uma filtragem deste conteúdo por meio de *scripts* SQL, utilizando filtros que selecionavam apenas os textos que possuíam algum tipo de conteúdo.

A partir destes filtros, novas tabelas foram criadas para criar grupos de dados e um histórico das mudanças feitas na base de origem, mantendo a base de notícias original para ser possível utilizar em aplicações futuras, como, por exemplo, na predição de texto que não possuem categorias especificas. A quantidade de registros de notícias após o filtro de textos que não possuíam corpo foi para 609263, filtrando **111714** noticias.

Foi identificado na análise das etiquetas das categorias que elas possuíam problemas de rótulos muito específicos, com isso foram definidos categorias mais gerais e outras foram descartadas, pois não são identificadas com as categorias mais abordadas pelas notícias. As categorias foram definidas a partir da listagem abaixo, por exemplo, notícias que tinham a categoria “Mundo” foram definidas com a etiqueta “Internacional”.

- Economia
- Politica
- Internacional
  - Mundo
- Saúde
  - Bem-estar, Ciência e Saúde
- Nacional
- Entretenimento
  - Pop-arte, Carnaval, Turismo e Viagem, Estilo
- Ciência
  - Tecnologia, Inovação
- Educação
- Esportes
- Natureza
  - Meio ambiente
- Regional
  - SP, PE, PB, MG, RJ, RO, DF, BA, ES, PR, PI, RS, AL, CE, AP, SC, AC, SE, AM, PA, GO, SÃO-PAULO, PERNAMBUCO, RN, MA, MATO-GROSSO, DISTRITO-FEDERAL

A partir desta definição das categorias das notícias e da remoção das categorias que não se inseriam em nenhuma categoria, por exemplo, “últimas-noticias”, foi criado uma nova tabela que possui **261164** registros e a distribuição ocorre conforme a Tab. 2

Com uma base melhorada e as categorias definidas, houve a migração da base de dados para um arquivo CSV utilizando a biblioteca Pandas, esta base foi alocada ao serviço de processamento em nuvem chamado Kaggle, facilitando questões de processamento e ajudando nas questões de limpagem de dados. Com o conjunto de dados criado no Kaggle, foi utilizado o processamento do serviço para converter a base CSV para um arquivo Parquet, cujo objetivo é otimizar o processo de leitura, processamento dos dados e compactação dos dados, com este novo arquivo, foi feita a atualização do conjunto de dados, sendo utilizado nas tarefas futuras da análise dos dados, aplicação dos modelos

Tabela 2 – Quantidade de registros por categoria agrupada em ordem decrescente

Categoria	**Quantidade**
Economia	60537
Politica	47556
Saúde	32662
Internacional	25447
Regional	19987
Entretenimento	18445
Nacional	17327
Mundo	15083
Ciência	7675
Esportes	6083
Natureza	5198
Educação	5164

de transformação de dados em tensores para o processamento otimizado dos dados e treinamento do modelo.

3.4 Análise exploratória dos dados

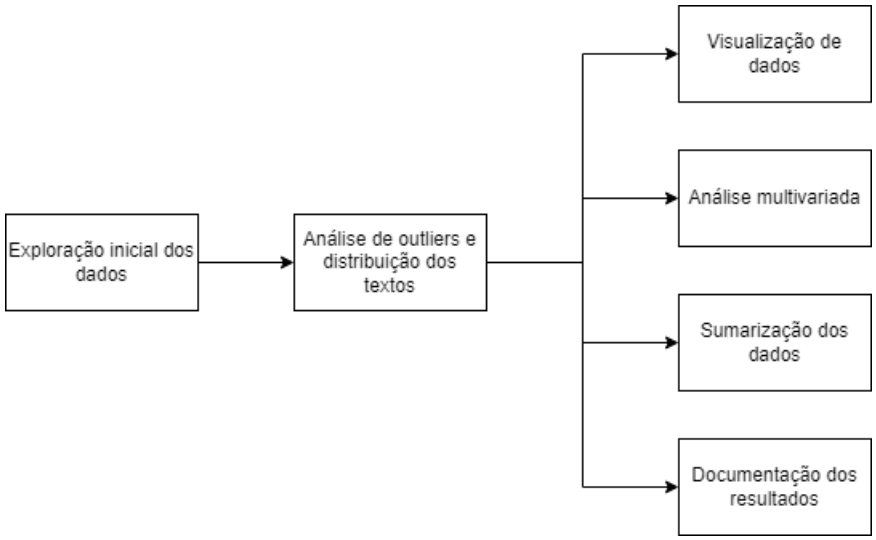

Figura 8 – Análise exploratória dos dados

A Figura 8 demonstra como é o processo de análise exploratória dos dados. Neste estágio, o foco está na análise exploratória de dados, visando descobrir padrões, relacionamentos e percepções no conjunto de dados. Os principais objetivos da análise incluem a compreensão da distribuição dos dados, a identificação de tendências e padrões, a avaliação da qualidade e da integridade dos dados, a exploração de relacionamentos e correlações e a descoberta de *outliers* e anomalias. Ao realizar uma análise exploratória completa dos dados, é possível obter *insights* valiosos que orientam outras análises, modelagens e pro-

cessos de tomada de decisão. Ela serve como uma etapa crucial para entender os dados e estabelecer a base para os estágios subsequentes do fluxo de trabalho da análise de dados.

Esta análise foi feita por meio do *Kaggle* e pode ser acessada através deste link, a partir deles temos o notebook com a leitura e manipulação dos dados, houve a separação dos contextos e eles serão inseridos abaixo com as informações encontradas.

3.4.1 Inspeção inicial dos dados

Neste tópico, validou-se a continuação das informações que foram abordadas acima, como quantidades de registros, campos que estavam nulos e a tipagem de cada dados, as colunas de interesse que são *text* e *grouped_category* mantém a quantidade de seus registros, no entanto, outros campos como *author* e *date* possuem alguns registros nulos, como pode ser visto na Fig. 9

```
<class 'pandas.core.frame.DataFrame'>
Index: 261164 entries, 0 to 23321
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   idx               261164 non-null  int64
 1   parent_url        261164 non-null  object
 2   url               261164 non-null  object
 3   title             246255 non-null  object
 4   subtitle          233719 non-null  object
 5   author            249272 non-null  object
 6   date              261116 non-null  object
 7   text              261164 non-null  object
 8   category          261164 non-null  object
 9   grouped_category  261164 non-null  object
dtypes: int64(1), object(9)
memory usage: 21.9+ MB
```

Figura 9 – Informações básicas do *dataset*

A partir das informações acima, os próximos passos foram feitos apenas com um *dataset* que possui as colunas de interesse, focando no texto e sua respectiva categoria e descartando as outras colunas.

3.4.2 Análise categórica

Aqui a primeira atividade foi validar quais categorias temos, as quais estão listadas abaixo:

- Economia
- Educacao

- Regional
- Ciencia
- Natureza
- Mundo
- Esportes
- Entretenimento
- Internacional
- Nacional
- Politica
- Saude

Como já apontado na seção de pré-processamento, percebe-se que já houve um erro na categorização, pois as categorias "Mundo" e "Internacional" identificam uma mesma categoria, por isso já houve a alteração da categoria mundo para internacional, então agora existem apenas 11 categorias.

Com essa modificação, já houve a verificação de como está a distribuição destas categorias, validando quantas amostras cada categoria possui, a qual pode-se comprovar por meio da Fig. 10. Podemos verificar que a quantidade de amostras é grande para cada categoria, onde a distribuição varia bastante para cada uma.

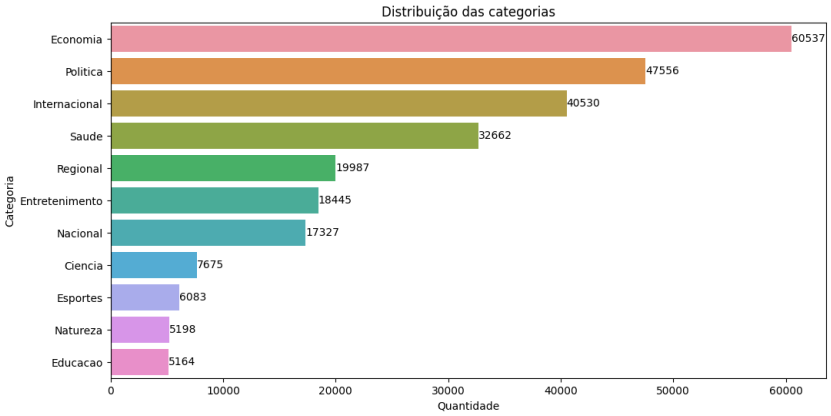

Figura 10 – Distribuição de amostras por categorias

Porém, precisa-se validar outro ponto: as amostras não são iguais, então a partir desta premissa, infere-se que cada categoria possui uma média de palavras, por isso a Fig. 11 reflete exatamente esta premissa, onde pode-se verificar que a média de palavras é relativa para categoria.

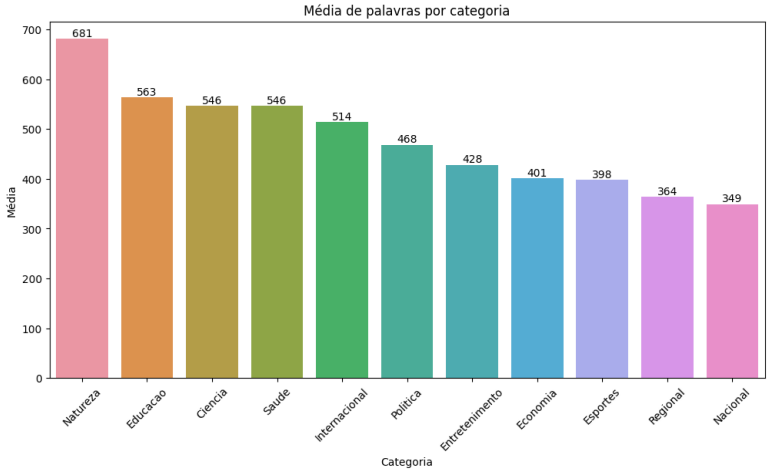

Figura 11 – Média de palavras por categorias

Com as validações acima, pôde-se seguir para a análise textual das notícias, porém esta etapa aqui comprovou que existiam categorias duplicadas e que a distribuição não é balanceada entre as categorias, sendo um fator importante para uma situação de treinamento de um modelo de classificação.

3.4.3 Análise textual

Inicialmente nesta etapa foram criadas novas colunas para armazenar outras informações sobre os textos, sendo elas: tamanho do texto, contagem de palavras e contagem de palavras únicas. Pode-se validar a distribuição destes dados no conjunto por meio das Fig. 12, 13, 14.

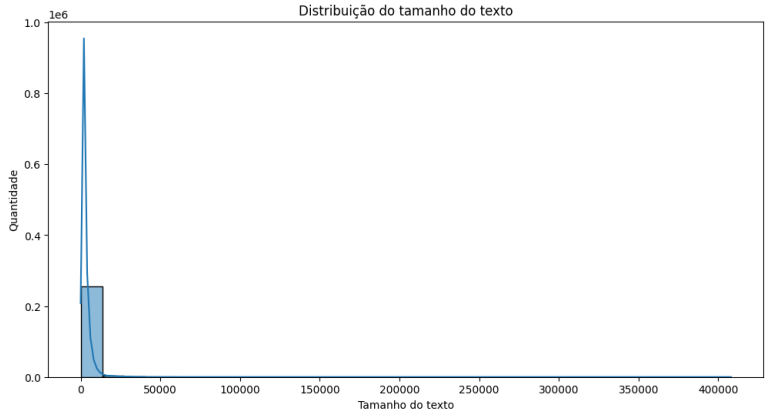

Figura 12 – Distribuição do tamanho do texto

Na Figura 15 pode-se observar que a distribuição é muito concentrada em um ponto e valida-se também a quantidade de *outliers*, que significa pontos fora da curva, ou seja, observa-se que existem textos muito longos que não se adéquam ao comportamento dos

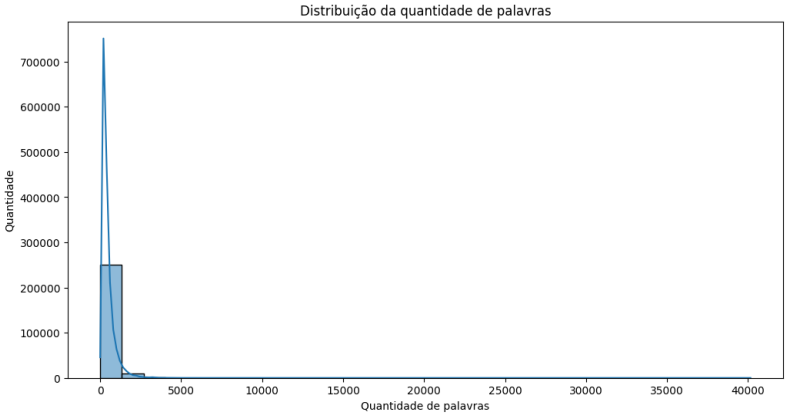

Figura 13 – Distribuição da quantidade de palavras

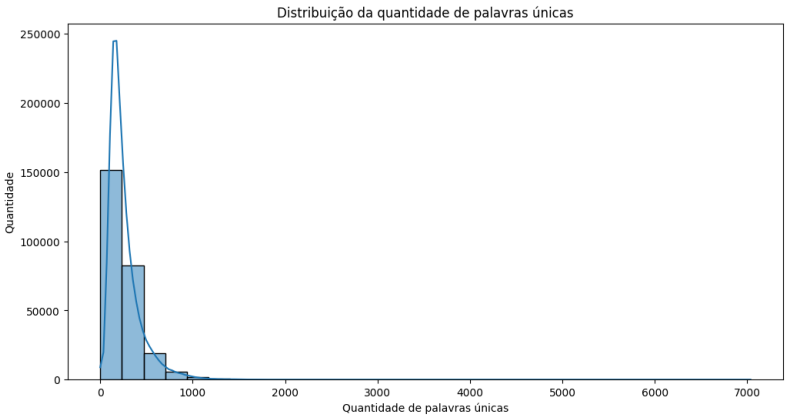

Figura 14 – Distribuição da quantidade de palavras únicas

textos em geral, podendo afetar negativamente a análise estatística futura e os resultados observados no contexto da análise exploratória inicial.

A razão para realizar esse processo está relacionada à qualidade dos dados. Textos muito curtos podem conter informações insuficientes para análises significativas, enquanto textos muito longos podem ser excessivamente detalhados ou até mesmo irrelevantes, o que pode complicar análises posteriores. Portanto, ao definir limites com base nos percentis 5 e 95 da distribuição de comprimento dos textos, removendo efetivamente *outliers*, ou seja, valores que estão significativamente abaixo ou acima da média, visando:

- Garantir que os dados usados nas análises sejam representativos e não estejam distortos por valores extremos.
- Simplificar análises posteriores, tornando os dados mais gerenciáveis e reduzindo a probabilidade de problemas causados por textos excessivamente curtos ou longos.
- Melhorar a eficácia de algoritmos e modelos que podem ser aplicados posteriormente, pois eles funcionam melhor com dados de qualidade e representativos.

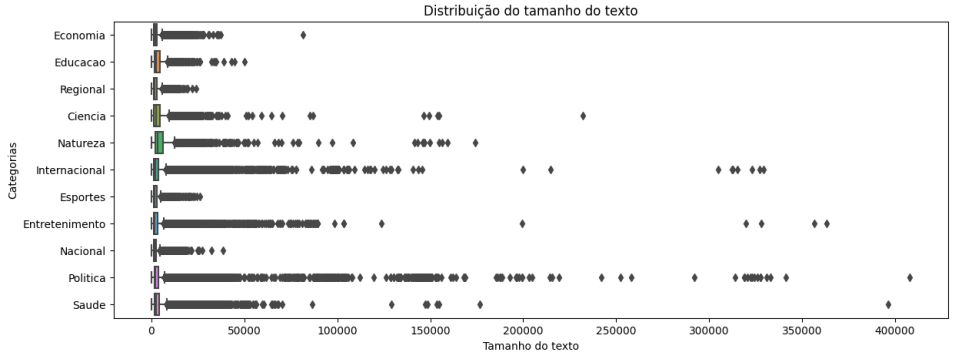

Figura 15 – Distribuição do tamanho do texto

Portanto, nessa etapa houve a filtragem dos dados com base nos percentis 5 e 95 visando criar um conjunto de dados mais coeso, removendo 26068 amostras, pronto para análises subsequentes, minimizando os desafios associados a valores extremos no comprimento dos textos, pode-se ver na Fig. 16 os resultados após a filtragem e na Fig. 17

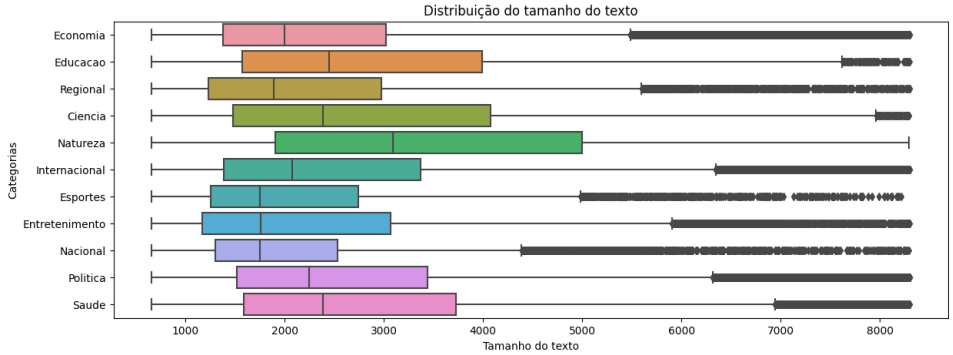

Figura 16 – Distribuição do tamanho do texto após filtragem

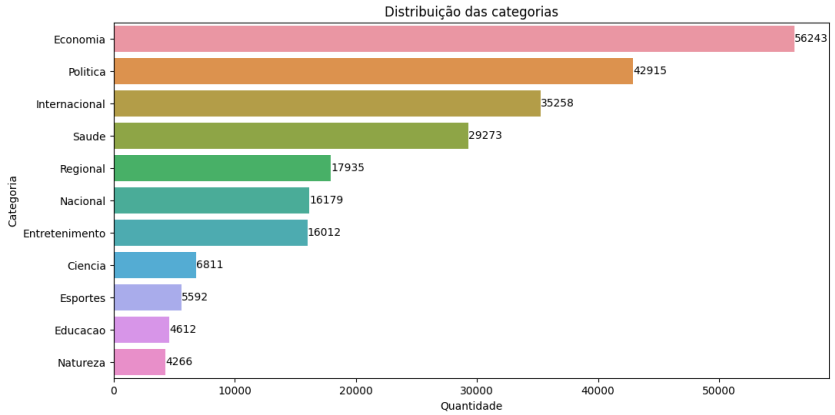

Figura 17 – Distribuição da quantidade de categorias após a filtragem

Com a filtragem dos dados e remoção dos textos mais destoantes, houve mudanças também nos histogramas da distribuição dos textos, se tornado mais normalizada, como

pode ser visto nas Fig. 18, Fig. 19 e Fig. 20.

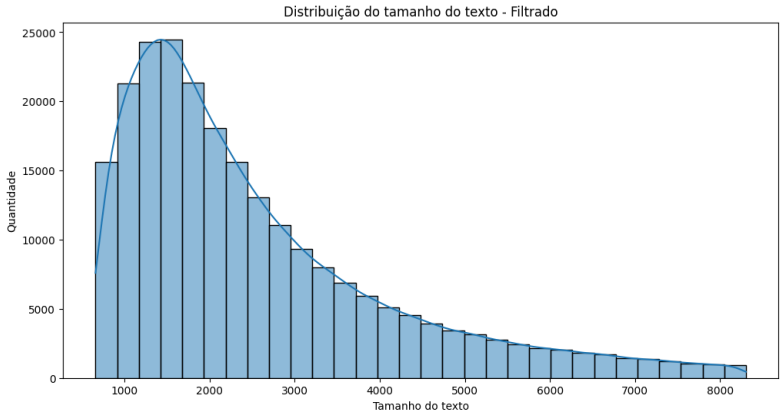

Figura 18 – Distribuição do tamanho do texto após a filtragem

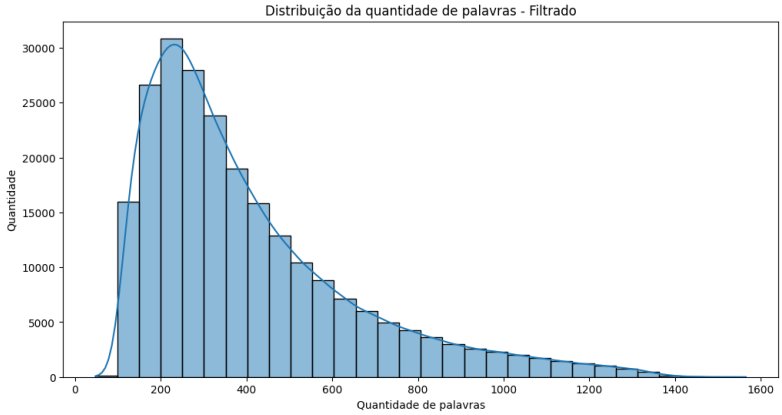

Figura 19 – Distribuição da quantidade de palavras após a filtragem

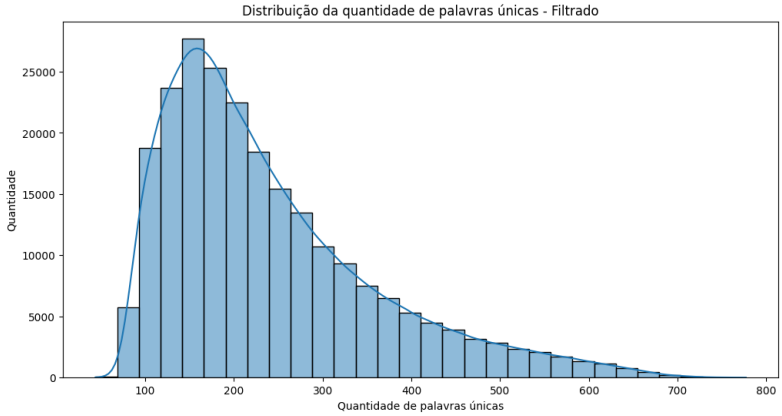

Figura 20 – Distribuição da quantidade de palavras únicas após a filtragem

3.4.4 Limpagem dos dados

A limpeza adequada do texto é uma etapa essencial no processamento de dados antes de realizar análises, especialmente em tarefas de Processamento de Linguagem Na-

tural (NLP). Esta etapa descreve a metodologia utilizada para limpar o texto no conjunto de dados em voga, visando garantir a qualidade e consistência dos resultados.

A limpeza do texto envolveu várias etapas, cada uma desenhada para abordar aspectos específicos do texto original. A função *clean_text* foi implementada para aplicar essas transformações. Abaixo estão as abordagens adotada:

- **Conversão para Minúsculas:** O texto foi convertido para minúsculas para garantir consistência nas análises, independentemente da formatação original.
- **Remoção de Acentuação:** Utilizando a biblioteca unidecode, a acentuação foi removida das palavras, normalizando caracteres acentuados para suas formas não acentuadas.
- **Remoção de Tags HTML:** Tags HTML foram removidas do texto para eliminar marcações ou formatações específicas da web.
- **Remoção de Símbolos de Moeda:** O símbolo da moeda brasileira 'R$' e o símbolo do dólar '$' foram removidos do texto.
- **Remoção de Pontuação:** Todos os caracteres de pontuação foram removidos do texto para simplificar a análise.
- **Remoção de Espaços em Branco:** Espaços em branco no início e no final do texto foram removidos, e espaços múltiplos foram substituídos por um único espaço.
- **Remoção de Palavras com Números:** Palavras contendo números foram removidas do texto.
- **Remoção de Números:** Todos os números foram removidos do texto.
- **Remoção de URLs:** URLs foram removidos do texto para evitar que links afetem indevidamente as análises.
- **Processamento com spaCy:** O texto foi processado usando o modelo *spaCy* em português ('pt_core_news_lg'). As etapas incluíram tokenização, lematização (redução das palavras às suas formas básicas), e remoção de *stopwords* (palavras comuns que não contribuem significativamente para o significado) e pontuação.
- **Recombinação das Palavras:** As palavras processadas foram recombinadas em uma única *string*, representando o texto limpo.

Os resultados da limpeza do texto foram armazenados na coluna *clean_text* do conjunto de dados. Essa coluna agora contém textos processados e prontos para análises subsequentes.

A limpeza eficaz do texto é crucial para garantir a validade e a precisão das análises de dados, especialmente em tarefas de NLP. A metodologia apresentada visa preservar a essência do conteúdo enquanto remove elementos indesejados.

3.4.5 Verificação da correlação entre as categorias

A análise de correlação entre categorias é uma etapa crucial para entender as relações semânticas e temáticas presentes nos textos associados a diferentes categorias. Esta tarefa é importante no contexto deste trabalho, pois identifica qual será o nível de dificuldade para a aplicação do modelo que categoriza os dados, ou seja, o quão distintos os textos são para cada categoria, dependendo dos resultados podemos observar se modelos simples podem aderir mais ao contexto ou é realmente necessário modelos de categorização mais complexos.

No contexto prático, os textos foram agregados por categoria, e a estratégia TF-IDF foi aplicada. O TF-IDF atribui pesos às palavras com base em sua frequência nos documentos (textos), destacando termos essenciais e diminuindo a importância de palavras comuns. Essa abordagem é valiosa porque foca nas características distintivas de cada categoria (BENGFORT; KIM, 2016).

A matriz de similaridade de cosseno foi calculada usando a matriz TF-IDF. A similaridade de cosseno mede a proximidade entre vetores e é frequentemente utilizada em análises de texto para determinar a semelhança entre documentos (ŽIžKA; DAřENA; SVOBODA, 2019).

A similaridade de cosseno entre as categorias foi visualizada em um mapa de calor. Esse tipo de representação gráfica facilita a interpretação, mostrando padrões de similaridade de uma maneira acessível.

A análise de correlação entre categorias, utilizando TF-IDF e similaridade de cosseno, oferece *insights* significativos para entender como diferentes temas se relacionam nos textos. Essa abordagem é fundamental em tarefas de processamento de linguagem natural, pois permite agrupar e compreender a distribuição semântica dos documentos, contribuindo para uma análise mais precisa e contextualizada. Na Figura 21 pode-se validar os resultados da correlação entre as categorias com os respectivos valores.

Na Figura 22 tem-se apenas uma melhoria visual no contexto de validar quais são as categorias que mais se correlacionam, validando que as categorias Nacional, Regional, Natureza e Ciência estão entre as categorias que mais se correlacionam com as outras.

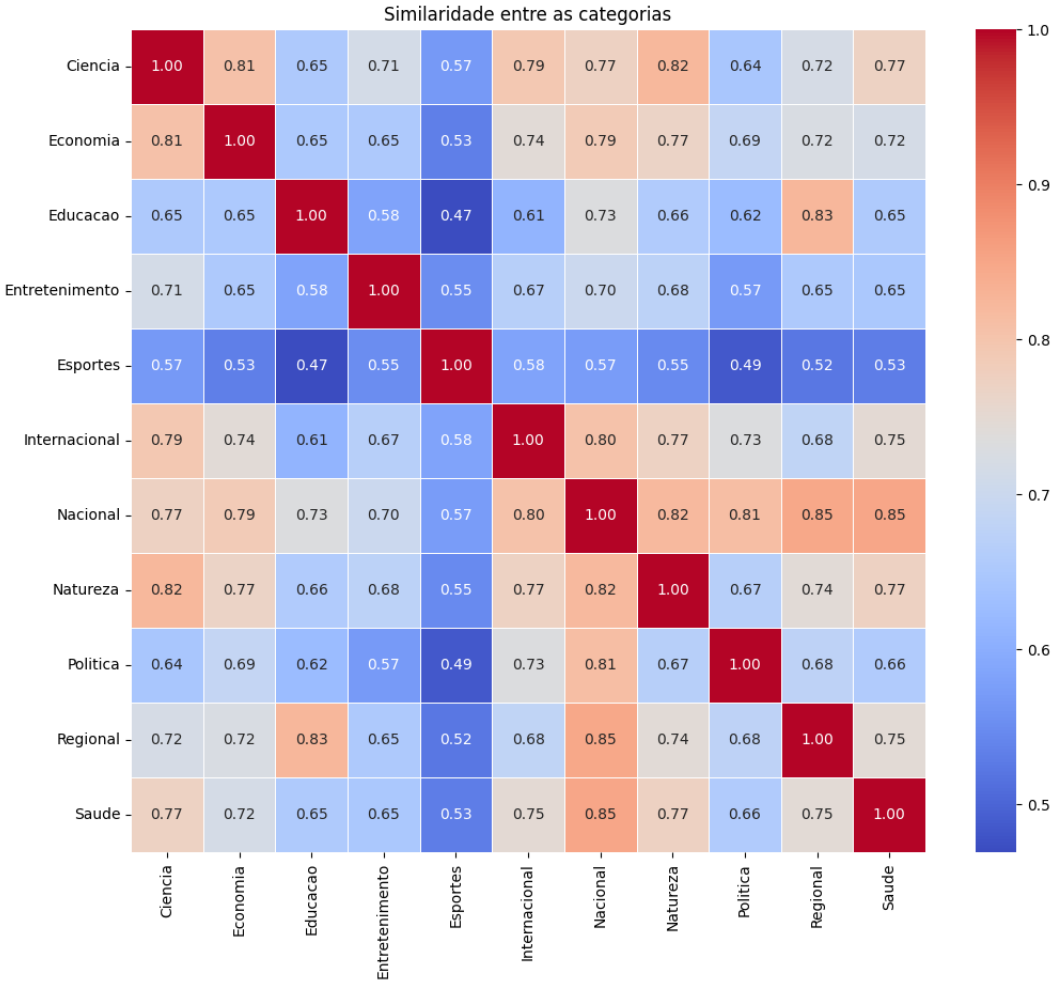

Figura 21 – Correlação entre as categorias usando TF-IDF

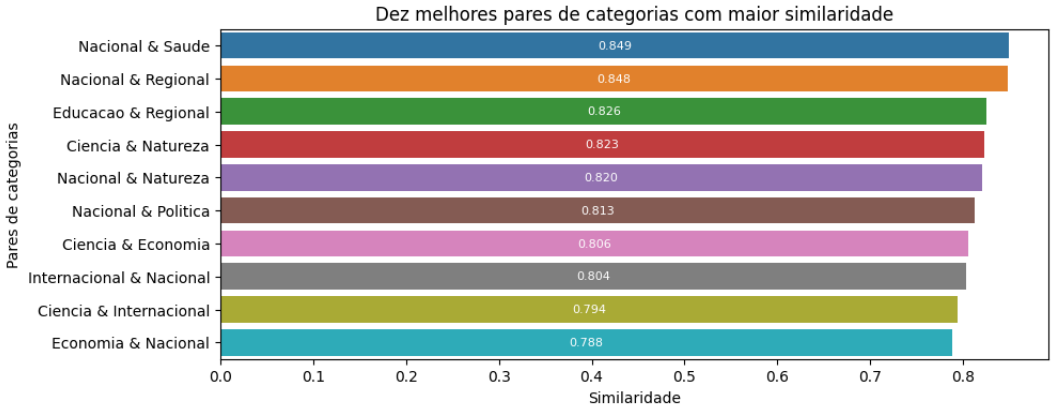

Figura 22 – Top 10 maiores correlações entre as categorias

3.5 Transformação e aplicação dos modelos

Nessa etapa, o foco será na transformação dos dados para a aplicação de modelos de transformadores, com foco específico nos tópicos de análise de modelos abertos, instalação de modelos, transformação de dados e previsão de categoria.

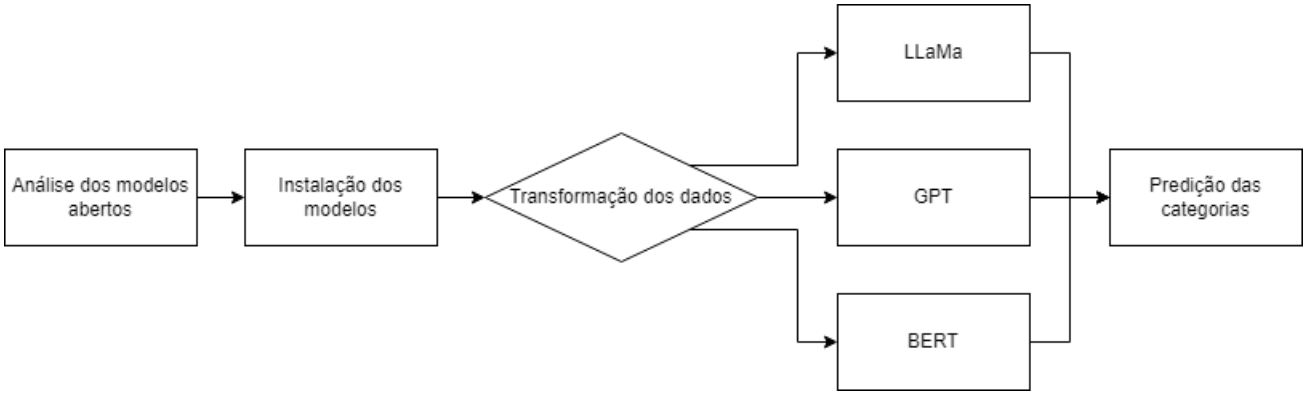

Figura 23 – Transformação e aplicação dos modelos

A primeira etapa envolverá a análise dos modelos de transformadores abertos, como BERT, LLaMA e GPT, para entender suas arquiteturas e recursos. A partir destes modelos serão selecionadas opções abertas que se baseiam nestes modelos para aplicação no projeto.

Em seguida, os modelos de transformadores selecionados são instalados para estabelecer o ambiente necessário para a transformação e a previsão de dados. Isso envolve a instalação das bibliotecas, dependências e estruturas necessárias para garantir a integração e a utilização compatível aos modelos.

Em seguida, os dados são transformados usando os modelos de transformação escolhidos para cada modelo. Esse processo inclui a tokenização do texto, a aplicação de todas as etapas de pré-processamento necessárias e a codificação dos dados em um formato adequado para análise e previsão. A transformação de dados garante que os dados de entrada estejam em um formato que os modelos possam processar e analisar com eficácia.

Por fim, os dados transformados são usados para fazer previsões das categorias usando os modelos de transformadores selecionados. Esses modelos aproveitam seus recursos pré-treinados para extrair representações significativas do texto e prever com precisão as categorias mais relevantes para os dados fornecidos.

Ao se concentrar nesses tópicos, os dados podem ser efetivamente preparados e transformados para previsão usando modelos de transformadores como BERT, LLaMA e GPT. Essa abordagem aproveita o poder das capacidades de representação de linguagem desses modelos, possibilitando previsões precisas e sensíveis ao contexto.

3.5.1 Pré-processamento

Houve a necessidade de realizar um pré-processamento nos dados para otimizar a estrutura deles, tornando-os compatíveis com a biblioteca *transformers*. Diferentemente

da etapa de exploração de dados, alguns passos neste pré-processamento divergem sutilmente, pois não é necessário realizar tarefas como remoção de pontuações, tokenização e lematização. Modelos como o DistilBERT já incorporam essas funcionalidades. Portanto, o foco aqui é replicar a remoção de *outliers* e limpar o texto, removendo apenas ruídos, como tags HTML e espaços duplicados. Os passos gerais realizados nessa fase são:

- Na etapa inicial, foram carregados dados de notícias em português em um DataFrame, consistindo principalmente de textos e categorias associadas.
- Para facilitar a análise, apenas as colunas relevantes (*text* e *grouped_category*) foram mantidas no DataFrame.
- A fim de otimizar a eficiência computacional e melhorar a qualidade dos dados, foi aplicada uma filtragem para manter apenas textos cujos comprimentos estivessem dentro do intervalo entre o 5º e o 95º percentil.
- Utilizando a biblioteca Pandarallel, o pré-processamento foi paralelizado para melhorar o desempenho. Cada texto foi submetido a uma função de pré-processamento, que incluiu a remoção de tags HTML, espaços em branco desnecessários e múltiplos espaços consecutivos.
- Os dados foram divididos em conjuntos de treinamento, validação e teste, usando uma estratégia estratificada para garantir representatividade de categorias em cada conjunto.
- Os conjuntos de treinamento, validação e teste foram convertidos para o formato de Dataset da Hugging Face para facilitar a integração com modelos baseados em transformers. As features "*text*", "*category*" e "*___null_dask_index___*" foram preservadas.
- O conjunto de dados foi salvo em disco para permitir uma fácil recuperação e compartilhamento.

A partir desta base pré-processada, ela foi utilizada como entrada para os *scripts* de *fine-tuning* de cada um dos modelos utilizados, sendo compatível com todas as arquiteturas, pois foi utilizada a biblioteca *transformers* para evitar este excesso de transformações para cada teste, sendo totalmente compatível com a nova estrutura de dados baseada na biblioteca *datasets*.

Fine-tuning é uma técnica de aprendizado de máquina que consiste em treinar um modelo pré-treinado em um novo conjunto de dados. Isso permite que o modelo seja ajustado para uma tarefa específica, mesmo que ele tenha sido originalmente treinado para outra tarefa (MARDESIC et al., 2017).

3.5.2 BERT

O primeiro modelo selecionado foi o DistilBERT, uma versão customizada do BERT, utilizado por meio da biblioteca *transformers* do Python, desenvolvida pela Hugging Face. O DistilBERT destaca-se por ser uma alternativa mais leve e eficiente em comparação com o BERT original, mantendo um desempenho satisfatório em tarefas de Processamento de Linguagem Natural (NLP) com uma exigência computacional reduzida.

Este modelo específico é disponibilizado pela biblioteca *transformers* da Hugging Face, que oferece uma extensa variedade de modelos pré-treinados e ferramentas para seu uso. A escolha pelo DistilBERT foi motivada pela sua capacidade de manter um desempenho robusto em tarefas de NLP, enquanto demanda recursos computacionais mais modestos em comparação com modelos mais complexos (SANH et al., 2019).

O processo de adaptação do modelo para a tarefa específica de classificação de categorias de notícias envolveu um procedimento de *fine-tuning*, utilizando o modelo previamente preparado e hospedado no *Hugging Face Hub*[1]. Nessa etapa, foram ajustados os pesos do modelo DistilBERT para otimizar seu desempenho na referida tarefa. A biblioteca *transformers* facilitou essa fase ao fornecer interfaces simplificadas para o *fine-tuning* e avaliação de modelos pré-treinados.

O processo para *fine-tuning* foi baseado no modelo de exemplo que se encontra no site do HF, exemplificando um caso que utiliza uma base de filmes e classifica as críticas dos filmes como positiva e negativa[2]. Todo o período de execução do treinamento do modelo para fine-tuning foi feito por meio do Kaggle utilizando a placa de vídeo Tesla P100 com 16 gigabytes de memória de vídeo.

No contexto deste trabalho, houve dificuldades para que o modelo fosse adaptado para o conjunto de dados de notícias, pois é necessário que a entrada do modelo tenha apenas as colunas desejadas, além disto houve a necessidade de dividir os dados, cuja separação foi feita em três partes, um para treino do modelo, com 207387 registros, um de teste com 126972 registros, um de validação com 88881 registros, sendo este último utilizado durante o treinamento do modelo para ajustar os hiperparâmetros e avaliar o desempenho do modelo em dados que não foram vistos durante o treinamento. A proporção de cada conjunto em relação à base original segue a tabela 3.

Além destes detalhes, outras alterações foram necessárias, por exemplo, no contexto de limitação de token e preenchimento (*padding*). No BERT há um limite de 512 *tokens*, então a estratégia foi remover tokens da parte final da sequência mais longa, ou seja, remove tokens da extremidade direita da sequência até que ela atinja o comprimento desejado. Em relação ao preenchimento (*padding*), houve a aplicação de uma

[1] <https://huggingface.co/docs/hub/models-the-hub>
[2] <https://huggingface.co/docs/transformers/v4.17.0/en/tasks/sequence_classification>

Tabela 3 – Distribuição Percentual dos Conjuntos de Dados

Conjunto de Dados	**Porcentagem**
Treinamento	49%
Validação	21%
Teste	30%

função chamada *DataCollatorWithPadding* que agrupa os tokens em lotes e aplica um preenchimento para este lote, criando uma matriz em que os vetores que armazenam os tokens possuam todos a mesma dimensão, isso é especialmente útil ao alimentar dados em modelos preditivos, pois muitos *frameworks* esperam lotes de dados com o mesmo comprimento.

Em resumo, *DataCollatorWithPadding* é usado para preparar e organizar dados durante o treinamento de modelos de linguagem natural, garantindo que os lotes tenham sequências uniformemente preenchidas e que o *tokenizer* seja aplicado consistentemente, evitando desperdícios computacionais.

Para a execução do treinamento do modelo, são utilizados dois componentes essenciais: *TrainingArguments* e *Trainer*. O *TrainingArguments* funciona como um guia de instruções, onde são definidos parâmetros como a quantidade de épocas de treinamento, o tamanho dos lotes de dados a serem processados de cada vez, entre outras configurações. Essencialmente, é nesse estágio que são especificadas as diretrizes sobre como o modelo deve se comportar durante o treinamento.

Já o *Trainer* atua como o executor dessas instruções. Ele recebe o modelo, os argumentos de treinamento e os dados, conduzindo efetivamente o processo de treinamento. Desempenha o papel de um maestro, garantindo que o modelo aprenda de maneira eficiente. Gerencia tarefas cruciais, como avaliações periódicas, salvamento dos melhores modelos e outras operações necessárias para assegurar um treinamento de qualidade.

Em conjunto, o *TrainingArguments* e o *Trainer* formam uma equipe de treinamento, onde as instruções são comunicadas e, posteriormente, executadas para garantir o aprendizado do modelo. O TrainingArguments indica ao treinador o que deve ser feito, enquanto o Trainer é o responsável por efetivar essas diretrizes.

Analogamente, podemos pensar nesse processo como instruir um treinador de cachorros (o Trainer) sobre como ensinar um novo truque a um cachorro (o modelo). O TrainingArguments seria como a lista de instruções detalhadas dadas ao treinador de cachorros, indicando, por exemplo, a quantidade de tempo diário dedicado ao treinamento e a abordagem a ser adotada. O treinador, então, assume a responsabilidade de garantir que o cachorro aprenda eficientemente, avaliando o progresso e realizando ajustes conforme necessário.

A métrica utilizada para avaliação do modelo é a acurácia. A acurácia representa a proporção de predições corretas em relação ao total de predições. Ela fornece uma medida geral do desempenho do modelo na tarefa de classificação de categorias de notícias. Durante o treinamento, o Trainer avalia o modelo periodicamente com base nessa métrica, permitindo ajustes e melhorias contínuas. A escolha da acurácia como métrica de avaliação para o treinamento do modelo no contexto de classificação de categorias de notícias é justificada pela natureza da tarefa e pela necessidade de uma métrica abrangente.

3.5.3 Outros modelos testados

Na tentativa de realizar o *fine-tuning* dos modelos GPT-Neo e MT5 para a tarefa específica de classificação de categorias de notícias, deparou-se com desafios significativos relacionados à demanda computacional desses modelos. GPT-Neo, desenvolvido pela EleutherAI, e MT5 (Multilingual Translation Transformer 5), desenvolvido pela Google, são conhecidos por suas arquiteturas robustas e complexas, o que pode resultar em requisitos elevados de memória durante o treinamento.

Exploraram-se plataformas, como Colab[3] e Paperspace[4], mesmo ao ajustar parâmetros como o tamanho do lote (*batch_size*) e o número de lotes (*num_batches*), foram encontradas limitações intransponíveis de alocação de memória. O processo de *fine-tuning* desses modelos, apesar das tentativas de otimização, mostrou-se impraticável em ambientes com recursos computacionais limitados.

Foram realizadas tentativas de reduzir a complexidade do modelo, diminuindo o tamanho do lote e ajustando outros hiperparâmetros, buscando um equilíbrio entre desempenho e utilização de recursos. Entretanto, mesmo com essas tratativas, a alocação extensiva de memória persistiu, impedindo o prosseguimento bem-sucedido do *fine-tuning*.

Essas dificuldades ressaltam a importância de considerar não apenas a arquitetura do modelo, mas também as restrições de *hardware* disponíveis durante o planejamento de projetos de *fine-tuning*. O entendimento da viabilidade computacional de modelos específicos é crucial para evitar desafios insuperáveis e garantir uma abordagem pragmática às tarefas de treinamento.

Essa experiência destaca a necessidade contínua de inovações em métodos de treinamento, bem como a importância de escolher modelos compatíveis com os recursos computacionais disponíveis, especialmente ao lidar com arquiteturas avançadas e de grande escala como GPT-Neo e MT5.

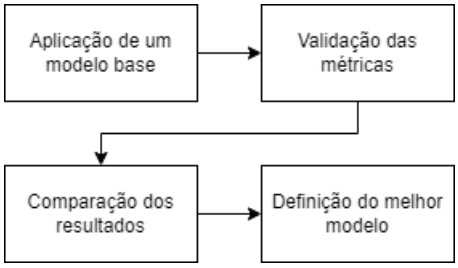

Figura 24 – Avaliação dos resultados

3.6 Avaliação dos resultados

A primeira etapa envolve a aplicação de um modelo básico para avaliar seu desempenho na previsão de categorias ou rótulos para os quais ele não foi explicitamente treinado.

Em seguida, as métricas usadas para avaliar o desempenho do modelo serão validadas. Isso garante que as métricas de avaliação escolhidas sejam confiáveis e eficazes para medir com precisão o desempenho do modelo. Métricas como precisão, revocação, exatidão ou pontuação F1 (*F-score* - Média harmônica entre duas métricas: precisão e revocação) podem ser empregadas para avaliar a qualidade das previsões do modelo.

Os resultados obtidos de diferentes modelos, incluindo o modelo básico como critério-base de classificação, será comparado com os resultados do modelo BERT. Essa comparação ajuda a identificar os pontos fortes e fracos relativos de cada modelo em termos de desempenho na tarefa específica. Ela fornece *insights* sobre qual modelo tem melhor desempenho na previsão das categorias desejadas.

Com base na avaliação e comparação dos resultados, será determinado o modelo de melhor desempenho para a tarefa em questão. Vários fatores, como precisão, consistência, eficiência computacional e adequação ao aplicativo pretendido, são considerados na tomada dessa decisão.

Será realizada uma avaliação abrangente dos resultados, permitindo a seleção do modelo mais eficaz para a tarefa em questão. Essa avaliação fornece percepções valiosas sobre o desempenho de diferentes modelos e ajuda a tomar decisões informadas com base em seus respectivos pontos fortes e fracos.

3.6.1 Modelo básico - XGBoost

Para critério comparativo e validação dos resultados do modelo BERT, foi criado um modelo básico utilizando XGBoost[5], uma biblioteca eficaz para implementação de

[3] <https://colab.research.google.com/>
[4] <https://www.paperspace.com/>
[5] <https://xgboost.readthedocs.io/en/stable/>

algoritmos de *gradient boosting*. O propósito dessa abordagem é avaliar como o desempenho do BERT se compara a um modelo tradicional de aprendizado de máquina. Houve a tentativa inicial de executar modelos *zero-shot* (XIAN et al., 2020) para servir de *baseline*, porém todos os modelos testados não tiveram resultados válidos para critério de comparação.

O XGBoost, abreviação de "Extreme Gradient Boosting", é uma técnica de ensemble learning que combina várias árvores de decisão para criar um modelo preditivo robusto. Sua capacidade de lidar com dados complexos e a habilidade de otimizar funções de perda não diferenciáveis fazem dele uma escolha popular em competições de ciência de dados.

O conjunto de dados foi carregado e pré-processado, seguindo etapas semelhantes às realizadas na exploração de dados. Remover *outliers* baseados no comprimento dos textos e aplicar uma função personalizada de pré-processamento garantiram que os dados estivessem prontos para serem alimentados no modelo.

A divisão dos conjuntos de treinamento, validação e teste foi mantida consistente com a metodologia utilizada durante o *fine-tuning* do modelo BERT. Isso assegura uma comparação justa entre os dois modelos, pois ambos são avaliados nos mesmos subconjuntos de dados.

Os textos foram convertidos para representações TF-IDF, uma técnica que avalia a importância de uma palavra em relação a um documento e ao corpus. Essas representações foram usadas como entrada para o modelo XGBoost, que foi treinado e avaliado ao longo de várias iterações.

Ao fornecer uma base sólida para a comparação com o modelo BERT, essa abordagem permite uma análise aprofundada de como métodos tradicionais de aprendizado de máquina se comparam a abordagens mais modernas baseadas em modelos de linguagem pré-treinados.

3.7 Resultados

Para avaliar objetivamente o desempenho dos modelos BERT e XGBoost, ambos foram submetidos ao conjunto de dados de teste. Nesta etapa, os modelos foram utilizados para realizar previsões sobre as amostras de teste, determinando os rótulos associados a cada entrada. Esses rótulos preditos foram então comparados com os rótulos reais do conjunto de teste para calcular diversas métricas de avaliação. Dessa forma, os resultados apresentados a seguir refletem a capacidade dos modelos em generalizar para dados não vistos durante o treinamento.

3.7.1 Acurácia

A acurácia, uma métrica amplamente utilizada na avaliação de modelos de classificação, oferece uma visão geral do quão preciso são os resultados preditos em relação aos rótulos reais. No contexto deste estudo, a acurácia foi escolhida como uma métrica central para avaliar o desempenho dos modelos BERT e XGBoost no conjunto de dados de teste. Vale ressaltar que a acurácia é calculada como a proporção de previsões corretas em relação ao total de previsões realizadas. A seguir, são apresentados os resultados de acurácia obtidos por ambos os modelos, proporcionando uma visão comparativa de sua eficácia na tarefa de classificação de categorias de notícias. Na Tabela 4 é encontrado os resultados de acurácia entre os dois modelos

Tabela 4 – Comparação de Acurácias entre Modelos

Modelo	**Acurácia (%)**
BERT	85.00
XGBoost	86.52

Os resultados apresentados na Tabela de Comparação de Acurácias entre os Modelos BERT e XGBoost revelam aspectos interessantes sobre o desempenho destes dois algoritmos de aprendizado de máquina no contexto de classificação de categorias de notícias.

A primeira observação é que ambos os modelos apresentam acurácias elevadas, com o BERT alcançando 85% e o XGBoost um pouco superior, com 86.52%. Isso indica que ambos os modelos são eficazes na tarefa de classificação de texto, sendo capazes de generalizar bem para os dados não vistos no conjunto de teste.

Com uma acurácia de 85%, o modelo BERT demonstra um alto grau de eficiência. Dada a natureza do BERT de entender o contexto das palavras no texto, é notável que ele tenha conseguido um desempenho tão sólido. Isso pode ser atribuído à sua capacidade de entender a semântica e a estrutura linguística das notícias.

O XGBoost supera ligeiramente o BERT, com uma acurácia de 86.52%. Este modelo é conhecido por sua eficiência em lidar com dados tabulares e características numéricas. Este resultado pode sugerir que as características transformadas do texto para a classificação de notícias são bem capturadas e utilizadas pelo modelo.

Embora a diferença de acurácia entre os dois modelos seja relativamente pequena (1.52%), ela pode ser significativa dependendo do contexto de aplicação. Em cenários onde até mesmo uma pequena melhoria na acurácia é crucial, o XGBoost pode ser a escolha preferida.

Em resumo, ambos os modelos demonstram competência na tarefa de classificação

de notícias, com o XGBoost apresentando uma leve vantagem em termos de acurácia. A escolha entre eles deve levar em conta não apenas o desempenho em termos de acurácia, mas também outros fatores relevantes à aplicação específica.

3.7.2 Precisão, Revocação e Pontuação F1

A precisão, revocação e pontuação F1 são métricas fundamentais para avaliar o desempenho de modelos de classificação (POWERS, 2020). Elas fornecem *insights* valiosos sobre a qualidade das previsões do modelo em diferentes aspectos.

A Precisão mede a proporção de instâncias positivas identificadas corretamente pelo modelo em relação ao total de instâncias identificadas como positivas. Em outras palavras, é a capacidade do modelo de evitar a classificação incorreta de instâncias negativas como positivas. Na Tabela 5 pode-se encontrar os resultados de precisão.

Tabela 5 – Resultados de Precisão

Categoria	XGBoost	BERT	Diferença
Ciencia	0.74	0.60	0.14
Economia	0.88	0.85	0.03
Educacao	0.86	0.81	0.05
Entretenimento	0.90	0.95	-0.05
Esportes	0.92	0.90	0.02
Internacional	0.86	0.86	0.00
Nacional	0.79	0.76	0.03
Natureza	0.79	0.70	0.09
Politica	0.87	0.86	0.01
Regional	0.92	0.84	0.08
Saude	0.84	0.86	-0.02

O XGBoost supera o BERT em várias categorias, notavelmente em “Ciencia”, “Educacao” e “Natureza”, com diferenças significativas. Por outro lado, BERT tem melhor precisão em “Entretenimento” e “Saude”, embora as diferenças sejam menores. A maior variação é observada em “Ciencia”, onde o XGBoost tem uma precisão notavelmente maior do que o BERT.

A Revocação, também conhecida como sensibilidade, mede a proporção de instâncias positivas identificadas corretamente em relação ao total de instâncias que realmente são positivas. É a capacidade do modelo de encontrar todas as instâncias positivas. Na Tabela 6 pode-se encontrar os resultados de revocação.

Em várias categorias, como “Economia”, “Educacao”, “Entretenimento”, entre outras, os modelos apresentam resultados idênticos. A maior diferença é vista em “Ciencia”, onde o XGBoost tem uma revocação significativamente maior do que o BERT.

Tabela 6 – Resultados de Revocação

Categoria	XGBoost	BERT	Diferença
Ciencia	0.56	0.39	0.17
Economia	0.92	0.92	0.00
Educacao	0.79	0.79	0.00
Entretenimento	0.86	0.86	0.00
Esportes	0.94	0.94	0.00
Internacional	0.88	0.88	0.00
Nacional	0.62	0.62	0.00
Natureza	0.60	0.60	0.00
Politica	0.91	0.91	0.00
Regional	0.83	0.83	0.00
Saude	0.88	0.85	0.03

A Pontuação F1 é a média harmônica entre precisão e revocação. Ela fornece um equilíbrio entre as duas métricas e é útil quando há um desequilíbrio entre as classes. Na Tabela 7 pode-se encontrar os resultados de pontuação F1.

Tabela 7 – Resultados de F1

Categoria	XGBoost	BERT	Diferença
Ciencia	0.64	0.47	0.17
Economia	0.90	0.89	0.01
Educacao	0.85	0.80	0.05
Entretenimento	0.90	0.90	0.00
Esportes	0.92	0.92	0.00
Internacional	0.87	0.87	0.00
Nacional	0.72	0.68	0.04
Natureza	0.73	0.65	0.08
Politica	0.89	0.89	0.00
Regional	0.86	0.84	0.02
Saude	0.86	0.85	0.01

Ambos os modelos apresentam a mesma pontuação F1 em categorias como “Entretenimento”, “Esportes” e “Internacional”. No entanto, há diferenças notáveis em categorias como “Ciência”, “Educação” e “Natureza”, com o XGBoost apresentando melhor desempenho.

Na Tabela 8 é encontrado os resultados relacionados a métricas relativas a este tópico, validando de forma generalista os resultados.

Tabela 8 – Resultados Gerais

Métrica	XGBoost	BERT	Diferença
Precisão	0.82	0.78	0.04
Recall	0.78	0.74	0.04
F1	0.80	0.77	0.03

O XGBoost tem uma precisão global ligeiramente superior (0.82) comparado ao BERT (0.78), indicando uma melhor habilidade geral em classificar corretamente as instâncias positivas. Similarmente, na revocação, o XGBoost (0.78) supera o BERT (0.74), mostrando uma capacidade superior em identificar todas as instâncias positivas relevantes. A pontuação F1, que equilibra precisão e revocação, também é ligeiramente superior no XGBoost (0.80) em comparação com o BERT (0.77). Isso indica um desempenho geral mais balanceado do XGBoost em diferentes cenários.

Em geral, o XGBoost apresenta um desempenho ligeiramente superior ao BERT em termos de precisão, revocação e pontuação F1. Isso sugere que o XGBoost pode ser mais adequado para tarefas onde a precisão e a capacidade de identificar corretamente todas as instâncias relevantes são cruciais. O BERT, no entanto, mostra uma consistência notável em várias categorias, especialmente onde as diferenças são mínimas ou inexistentes. Isso pode ser um indicativo de sua robustez em entender contextos complexos dentro do texto.

Em casos onde categorias específicas são de maior interesse (por exemplo, “Ciencia” ou “Natureza”), a escolha do modelo pode ser influenciada pelas diferenças observadas nas métricas para essas categorias específicas.

Em resumo, ambos os modelos demonstram competências notáveis na classificação de categorias de notícias, com o XGBoost apresentando uma vantagem geral em termos de precisão, revocação e pontuação F1. A escolha entre eles deve considerar as necessidades específicas da aplicação e as características dos dados em questão.

3.7.3 Matriz de confusão

A matriz de confusão é uma métrica fundamental na avaliação de modelos de classificação (FAWCETT, 2006). Ela apresenta uma visão abrangente do desempenho do modelo, permitindo a análise de seus acertos e erros para cada classe. A matriz é especialmente útil quando lidamos com problemas de classificação multi-classe, fornecendo uma representação quantitativa das previsões do modelo em comparação com os rótulos verdadeiros.

Uma matriz de confusão típica é organizada em linhas e colunas, onde as linhas correspondem aos rótulos verdadeiros e as colunas às previsões do modelo. Cada célula da matriz representa o número de instâncias que pertencem à interseção de uma classe verdadeira e uma classe prevista. Nas Figuras 25 e 26 pode-se validar o resultado dos dois modelos e a aderência a qual categoria eles mais se qualificam.

Na matriz de confusão do BERT, os valores na diagonal representam as previsões corretas para cada categoria, enquanto os outros valores indicam erros de classificação. Por exemplo:

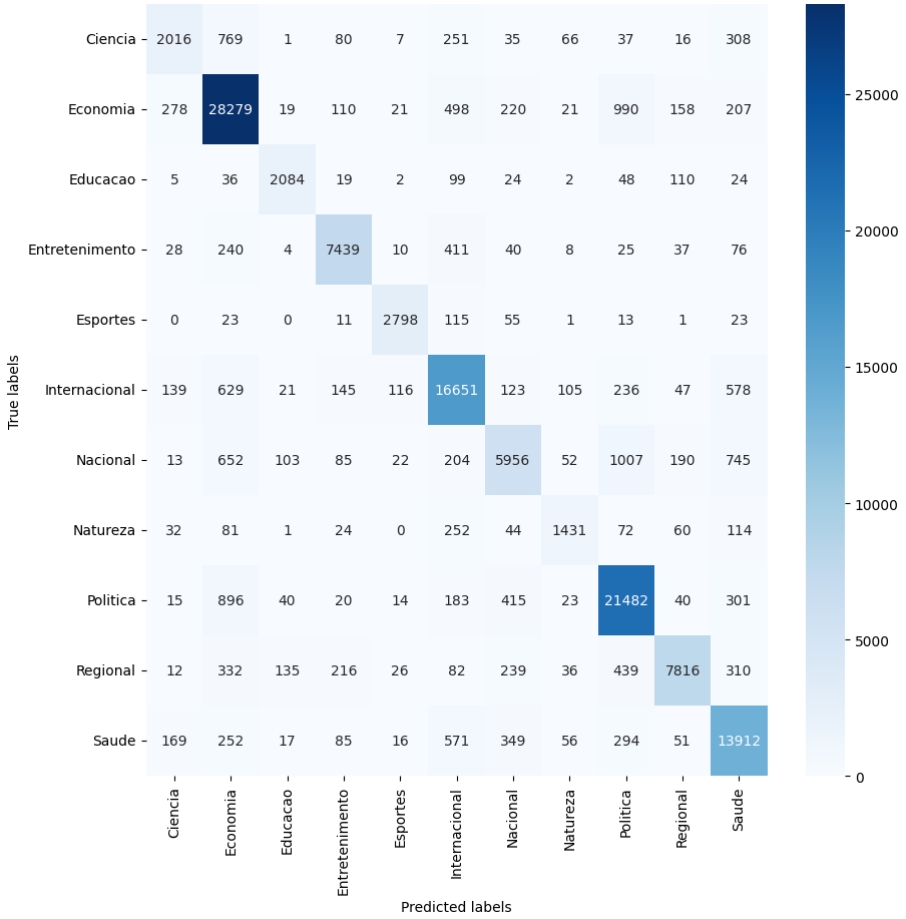

Figura 25 – Matriz de confusão - XGBoost

- Ciência: 1393 previsões corretas, mas também 1348 classificações incorretas como "Economia".

- Economia: Alta taxa de acertos com 28413 previsões corretas, embora haja 392 erros classificados como "Internacional".

- Saúde: Excelente desempenho com 13439 acertos, mas com 237 classificações incorretas como "Ciência".

Estes resultados mostram que o BERT tem um bom desempenho em várias categorias, mas também sofre de confusões específicas, como "Ciência" sendo confundida com "Economia".

Da mesma forma, na matriz de confusão do XGBoost, os valores na diagonal representam acertos e os outros valores indicam erros. Por exemplo:

- Ciência: Excelente desempenho com 2016 acertos, mas com 769 erros classificados como "Economia".

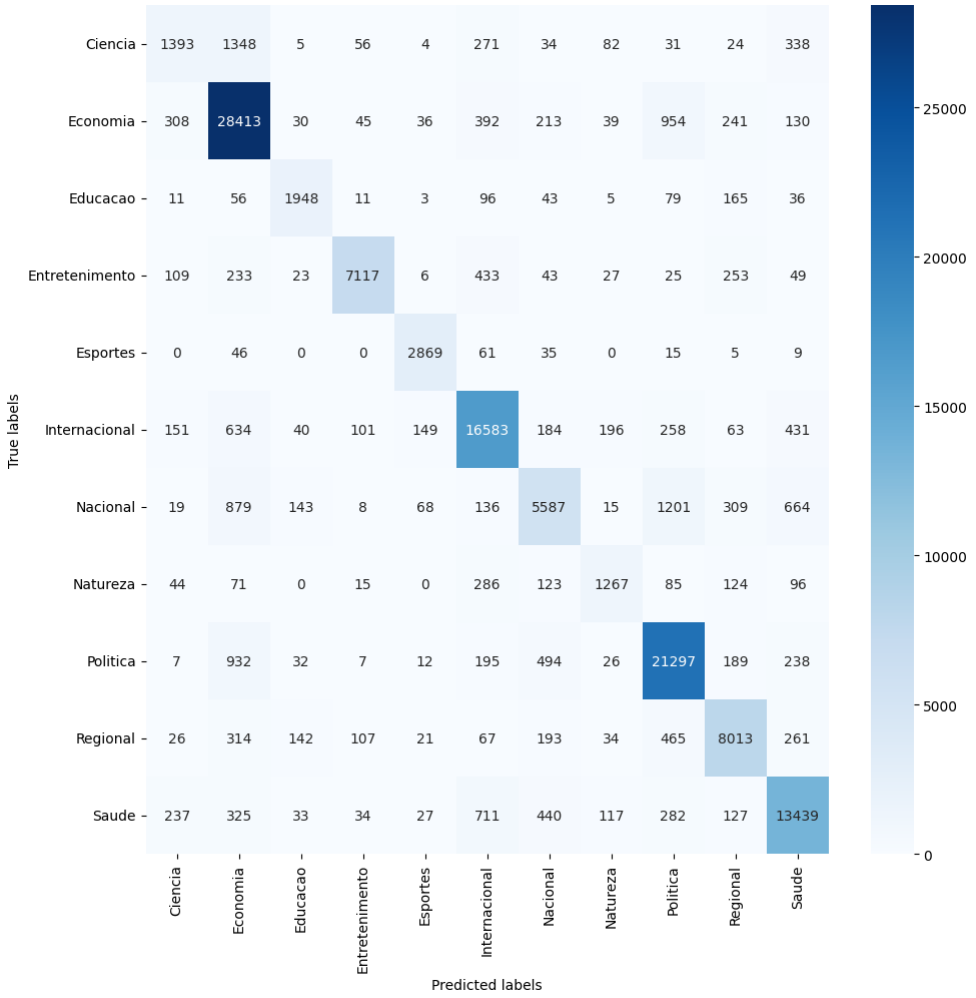

Figura 26 – Matriz de confusão - BERT

- Economia: Desempenho ainda melhor do que o BERT com 28279 acertos, embora haja 498 erros classificados como “Internacional”.
- Saúde: Muito bom desempenho com 13912 acertos, mas com 169 erros classificados como “Ciência”.

O XGBoost parece ter uma taxa de acerto geral maior, indicada pelo maior número de valores na diagonal principal em comparação com o BERT. Ambos os modelos compartilham padrões similares de erros de classificação, como confundir “Ciência” com “Economia” e vice-versa. Cada modelo tem suas forças e fraquezas em categorias específicas. Por exemplo, o XGBoost se sai melhor em “Ciência”, enquanto o BERT é mais equilibrado em “Entretenimento”.

Estes *insights* podem ser cruciais ao escolher um modelo para uma aplicação específica. Se a precisão em uma categoria particular é crítica, então um dos modelos pode ser preferido sobre o outro. Além disso, entender onde cada modelo erra pode ajudar a refinar e melhorar os algoritmos para futuras iterações.

3.8 Resultados - Base balanceada

Neste capítulo, são apresentados os resultados da avaliação dos modelos XGBoost e BERT após o pré-processamento e treinamento com uma base de dados balanceada. A base de dados foi ajustada de modo a garantir que todas as categorias tivessem o mesmo número de amostras, proporcionando uma visão equilibrada do desempenho dos modelos, esse fator é fundamental para validar se o resultados são relativos à quantidade de amostras.

Esta sessão foi elaborada visando validar se as informações obtidas na sessão de exploração de dados é aderente aos resultados dos modelos, principalmente pela obervação na correlação entre as categorias, já que aqui possuem-se os resultados numéricos para identificar quais foram as categorias que mais tiveram erros na previsão e comparar se é equivalente à taxa de correlação com estas categorias previstas incorretamente.

A estratégia empregada para equilibrar a base de dados consistiu em identificar a categoria com o menor número de amostras e, em seguida, ajustar as demais categorias para terem o mesmo número de amostras. Este processo foi realizado de forma aleatória para manter a representatividade dos dados. Conforme ilustrado na Figura 17, a categoria com o menor número de amostras tinha apenas 4266 registros. Portanto, esse número se tornou o padrão para todas as categorias, e exatamente 4266 amostras foram selecionadas aleatoriamente de cada uma das outras categorias para compor a base de dados balanceada.

O método de balanceamento utilizado assegura que cada categoria seja igualmente representada, eliminando o viés que poderia surgir devido ao desequilíbrio no número de amostras entre diferentes categorias. Os próximos tópicos são os resultados dos modelos utilizando as mesmas métricas apresentadas anteriormente, porém aplicados agora na base balanceada.

3.8.1 Acurácia

Tabela 9 – Comparação de Acurácias entre Modelos - Base balanceada

Modelo	**Acurácia (%)**
BERT	80.35
XGBoost	81.56

Com a base de dados agora balanceada, conforme a Tab. 9, a avaliação das acurácias dos modelos BERT e XGBoost revela novos *insights*.

Em comparação com os resultados anteriores (antes do balanceamento), observa-se uma redução na acurácia de ambos os modelos. Isso pode ser um indicativo de que os

modelos estavam anteriormente beneficiando-se de categorias com mais amostras, o que é uma situação comum em bases desbalanceadas. Com o balanceamento, os modelos agora têm que aprender padrões mais generalizados, o que pode ser mais desafiador.

O modelo BERT mostra uma acurácia de 80.35%. Embora reduzida em comparação com a medição anterior, essa acurácia ainda é considerável, indicando que o BERT mantém um bom desempenho mesmo em um cenário de dados mais equilibrado. Já o modelo XGBoost apresenta uma acurácia de 81.56%, também menor do que antes do balanceamento, mas ainda ligeiramente superior ao BERT. Isso sugere que o XGBoost é capaz de se adaptar um pouco melhor às mudanças na distribuição dos dados.

O balanceamento da base de dados teve um impacto claro na acurácia dos modelos, o que é um fenômeno esperado. Em muitos casos, o balanceamento dos dados pode expor fraquezas ou vieses nos modelos que não eram aparentes com os dados desbalanceados.

Em resumo, o balanceamento da base de dados proporcionou uma visão mais realista e desafiadora para os modelos, revelando a eficácia de cada um em um cenário mais equitativo. Ambos os modelos demonstram competência, mas com uma ligeira vantagem para o XGBoost em termos de acurácia.

3.8.2 Precisão, Revocação e Pontuação F1

Após o balanceamento da base de dados, uma nova avaliação das métricas de precisão, revocação e pontuação F1 para os modelos BERT e XGBoost revela informações valiosas sobre o desempenho de cada modelo.

Tabela 10 – Resultados de Precisão - Base balanceada

Categoria	XGBoost	BERT	Diferença
Ciencia	0.76	0.76	0.00
Economia	0.79	0.79	0.00
Educacao	0.86	0.86	0.00
Entretenimento	0.87	0.88	-0.01
Esportes	0.93	0.95	-0.02
Internacional	0.81	0.83	-0.02
Nacional	0.74	0.69	0.05
Natureza	0.81	0.79	0.02
Politica	0.77	0.76	0.01
Regional	0.87	0.73	0.14
Saude	0.76	0.81	-0.05

Na Tabela 10 nota-se uma convergência notável nas métricas de precisão entre os modelos BERT e XGBoost em várias categorias como “Ciência”, “Economia” e “Educação”. Isso sugere que o balanceamento da base de dados levou a uma similaridade maior no desempenho dos dois modelos nessas categorias.

Há variações notáveis em algumas categorias, como “Nacional”, “Regional” e “Saúde”, onde a diferença entre os modelos é mais pronunciada. Isso indica que, apesar do balanceamento, cada modelo ainda tem seus pontos fortes e fracos específicos.

Abaixo está a avaliação de cada categoria:

- Ciência, Economia, Educação: Ambos os modelos têm desempenho idêntico, com a mesma precisão. Isso pode indicar que essas categorias são bem definidas e mais fáceis de serem previstas por ambos os modelos.
- Entretenimento, Esportes, Internacional: O BERT apresenta uma ligeira vantagem nestas categorias, sugerindo uma melhor capacidade de identificar corretamente amostras positivas nestes tópicos.
- Nacional, Natureza, Política: O XGBoost tem uma vantagem nessas categorias. Especificamente, a diferença mais significativa é observada em “Nacional”, o que pode indicar uma melhor capacidade do XGBoost em capturar características distintas dessa categoria.
- Regional, Saúde: A maior diferença é vista em “Regional”, onde o XGBoost supera significativamente o BERT. Em contraste, o BERT tem um desempenho melhor em “Saúde”.

Tabela 11 – Resultados de Revocação - Base balanceada

Categoria	XGBoost	BERT	Diferença
Ciencia	0.77	0.80	-0.03
Economia	0.76	0.72	0.04
Educacao	0.92	0.87	0.05
Entretenimento	0.87	0.88	-0.01
Esportes	0.95	0.97	-0.02
Internacional	0.78	0.75	0.03
Nacional	0.64	0.63	0.01
Natureza	0.90	0.87	0.03
Politica	0.85	0.84	0.01
Regional	0.71	0.76	-0.05
Saude	0.81	0.75	0.06

Na Tabela 11, observa-se a análise dos resultados de revocação que, após o balanceamento da base de dados para os modelos XGBoost e BERTl, revela diferenças notáveis em seu desempenho por categoria. A revocação, sendo uma métrica que mede a capacidade dos modelos de identificar todas as instâncias positivas, fornece uma visão valiosa

sobre a sensibilidade de cada modelo em relação a diferentes categorias. Os dados apresentados na Tab. 11 permitem uma compreensão detalhada dessa sensibilidade em termos comparativos entre os dois modelos.

Abaixo está a avaliação de cada categoria:

- Ciência: Observa-se que o BERT supera o XGBoost, indicando uma maior capacidade do BERT em identificar instâncias positivas nesta categoria.
- Economia e Educação: O XGBoost apresenta uma revocação superior nessas categorias, sugerindo uma eficácia maior em capturar todas as instâncias relevantes em comparação com o BERT.
- Entretenimento e Esportes: As diferenças são mínimas, mas o BERT mostra uma leve vantagem em "Entretenimento" e em "Esportes".
- Internacional, Nacional, Natureza: O XGBoost demonstra uma revocação ligeiramente superior nestas categorias, o que pode ser atribuído à sua eficácia em classificar instâncias positivas nestes temas.
- Política, Regional, Saúde: Em "Regional" e "Saúde", o BERT tem uma revocação mais alta, o que indica uma maior sensibilidade do BERT a essas categorias.

A partir da análise dos resultados, conclui-se que o balanceamento da base de dados revelou diferenças específicas na revocação entre os modelos XGBoost e BERT. Tais diferenças sugerem que cada modelo possui características distintas em sua capacidade de identificar instâncias positivas em diferentes categorias. A escolha entre os modelos BERT e XGBoost para aplicações específicas deve considerar essas variações de desempenho. Em cenários onde categorias como "Ciencia", "Esporte" e "Regional" são de maior importância, o BERT pode ser mais adequado. Por outro lado, para categorias como "Economia" e "Educação", o XGBoost pode oferecer um desempenho superior.

Finalmente, os resultados destacam a importância do balanceamento de dados na avaliação de modelos de aprendizado de máquina, proporcionando uma base mais equitativa para comparar o desempenho dos modelos em diferentes categorias.

A análise dos resultados da Pontuação F1 para os modelos XGBoost e BERT, após o balanceamento da base de dados, consoante a Tab. 12, fornece uma perspectiva integrada sobre o desempenho de ambos em termos de precisão e revocação. A Pontuação F1, sendo a média harmônica entre precisão e revocação, é particularmente valiosa em situações onde é importante manter um equilíbrio entre as duas métricas.

Abaixo está a avaliação de cada categoria:

Tabela 12 – Resultados de F1 - Base balanceada

Categoria	XGBoost	BERT	Diferença
Ciencia	0.77	0.78	-0.01
Economia	0.77	0.75	0.02
Educacao	0.89	0.87	0.02
Entretenimento	0.87	0.88	-0.01
Esportes	0.94	0.96	-0.02
Internacional	0.80	0.78	0.02
Nacional	0.69	0.66	0.03
Natureza	0.86	0.83	0.03
Politica	0.81	0.80	0.01
Regional	0.78	0.74	0.04
Saude	0.78	0.78	0.00

- Ciência: Ambos os modelos apresentam desempenho semelhante, com uma ligeira vantagem para o BERT.
- Economia, Educação, Internacional: O XGBoost demonstra uma vantagem sobre o BERT nessas categorias, indicando um melhor equilíbrio entre precisão e revocação.
- Entretenimento, Esportes: O BERT apresenta uma pontuação F1 ligeiramente superior, sugerindo um desempenho mais balanceado nessas categorias.
- Nacional, Natureza, Regional: Nestas categorias, o XGBoost supera o BERT. Isso pode indicar uma habilidade superior do XGBoost em equilibrar precisão e revocação nessas áreas específicas.
- Política, Saúde: Desempenho similar entre os modelos em "Política", enquanto em "Saúde" eles apresentam a mesma pontuação F1.

Os resultados mostram que, embora haja algumas variações no desempenho entre os modelos por categoria, em geral, ambos os modelos apresentam competências comparáveis. As diferenças na Pontuação F1 são geralmente pequenas, sugerindo que ambos os modelos são capazes de manter um bom equilíbrio entre precisão e revocação após o balanceamento dos dados.

Em resumo, o balanceamento da base de dados proporcionou uma avaliação mais justa e reveladora das capacidades de cada modelo, com a Pontuação F1 oferecendo uma medida útil para comparar seu desempenho global em diferentes categorias.

A Tabela 13 apresenta os resultados gerais das métricas de Precisão, Revocação (Recall) e Pontuação F1 para os modelos XGBoost e BERT, oferecendo uma visão abrangente de seu desempenho global após o balanceamento da base de dados. A análise dessas

Tabela 13 – Resultados Gerais - Base balanceada

Métrica	XGBoost	BERT	Diferença
Precisão	0.82	0.80	0.02
Recall	0.82	0.80	0.02
F1	0.81	0.80	0.01

métricas fornece *insights* importantes sobre a eficácia geral de cada modelo em classificar as categorias de notícias.

Abaixo está a avaliação de cada categoria:

- Precisão e Revocação:
  - O XGBoost mostra uma ligeira vantagem tanto em Precisão quanto em Revocação, com valores de 0.82 em ambas as métricas, em comparação com 0.80 do BERT.
  - Esta diferença sugere que o XGBoost é marginalmente mais eficaz em identificar corretamente as instâncias positivas (Precisão) e também em capturar mais instâncias positivas relevantes (Revocação).
- Pontuação F1:
  - Na métrica de Pontuação F1, que equilibra Precisão e Revocação, o XGBoost mantém uma vantagem, embora pequena, sobre o BERT (0.81 contra 0.80).
  - Esta leve superioridade indica que o XGBoost apresenta um desempenho global ligeiramente mais balanceado em termos de precisão e capacidade de recuperação de instâncias positivas.

Os resultados indicam que ambos os modelos são eficazes na classificação de notícias, com o XGBoost apresentando um desempenho um pouco melhor nas métricas avaliadas. A diferença entre os modelos, embora pequena, pode ser significativa dependendo do contexto específico e da aplicação. Por exemplo, em cenários onde cada pequena melhoria na precisão ou na revocação é crítica, o XGBoost pode ser a escolha preferível.

No entanto, é importante considerar que a diferença de desempenho é relativamente modesta. Em resumo, o balanceamento da base de dados proporcionou um campo de teste mais equitativo para os modelos, revelando que ambos são competentes na tarefa de classificação, com o XGBoost exibindo uma ligeira vantagem nas métricas de desempenho geral.

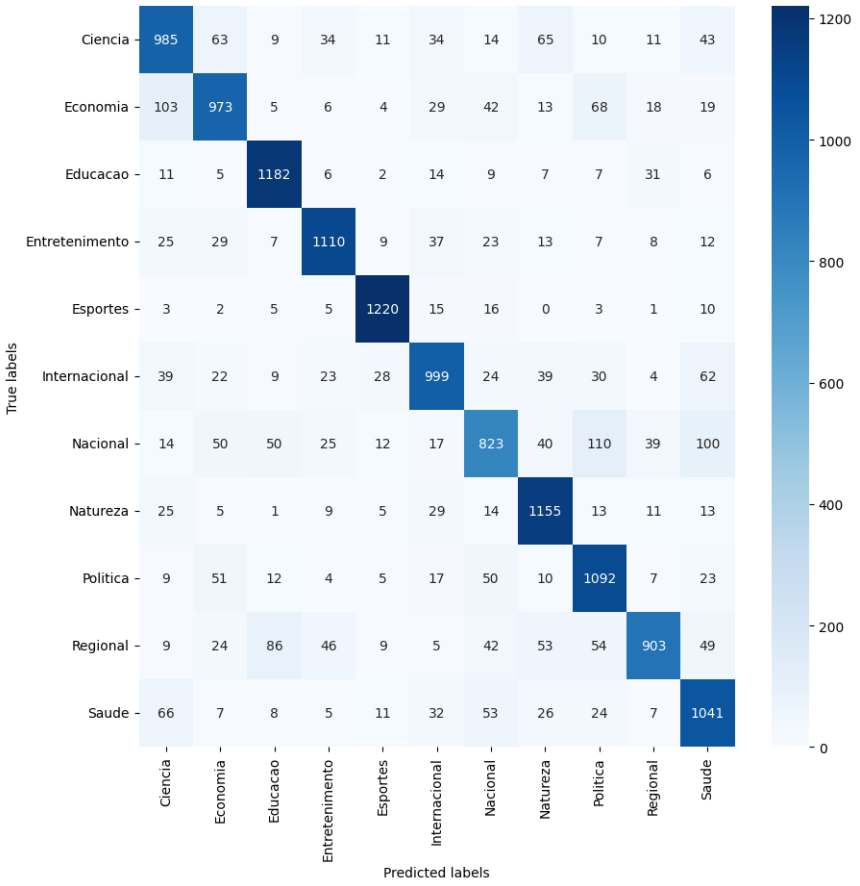

Figura 27 – Matriz de confusão - Base balanceada - XGBoost

3.8.3 Matriz de confusão

Com as matrizes de confusão fornecidas para os modelos XGBoost e BERT com os dados balanceados, a partir das figuras 27 e 28, há possibilidade de realizar uma análise detalhada do desempenho de cada modelo em classificar as categorias de notícias. O foco será na quantidade de previsões corretas (valores na diagonal principal) e nas classificações incorretas mais significativas (valores fora da diagonal).

O XGBoost mostra um bom desempenho em várias categorias, como “Ciência”, “Economia” e “Educação”, com altos números de previsões corretas. Abaixo estão descritos alguns erros significativos relativos à matriz de confusão do XGBoost:

- Em “Ciência”, houve uma confusão notável com “Natureza”.
- Em “Nacional”, observa-se uma quantidade significativa de erros em classificar como “Política”.
- “Regional” tem confusões notáveis em “Educação” e “Natureza”.

O BERT também apresenta um bom desempenho em várias categorias, com destaque para “Ciência”, “Educação” e “Entretenimento”. . Abaixo estão descritos alguns

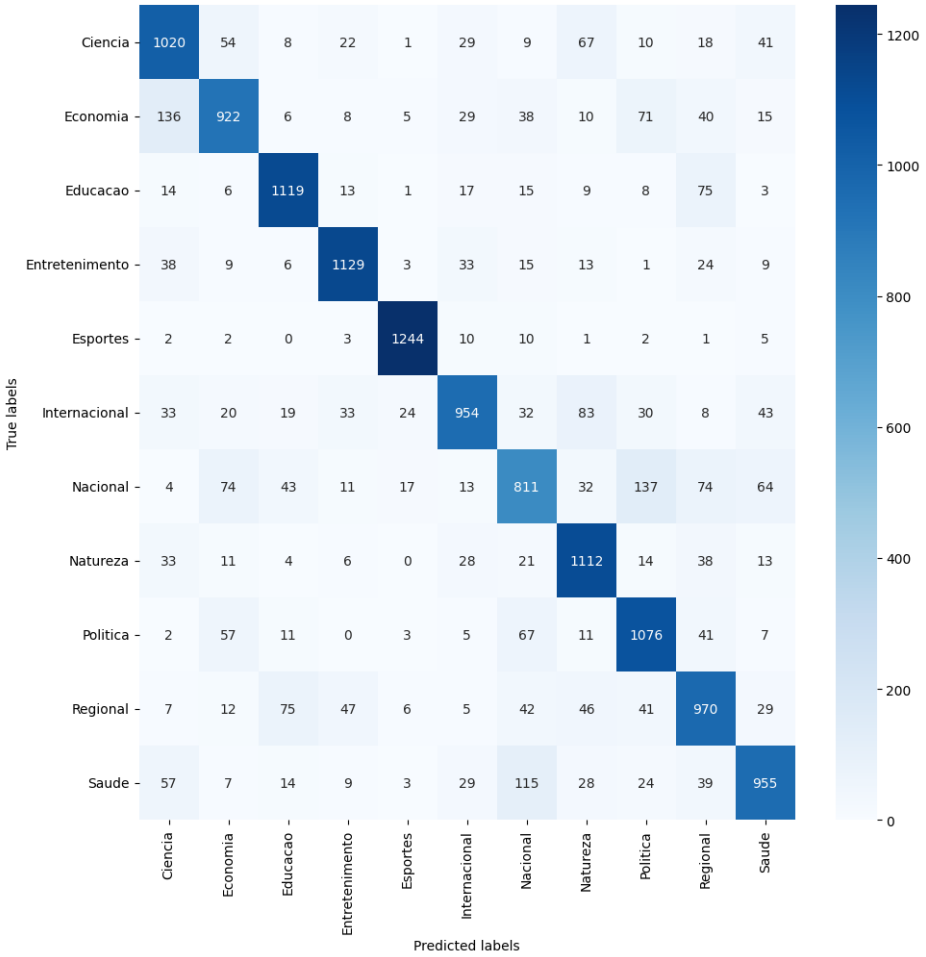

Figura 28 – Matriz de confusão - Base balanceada - BERT

erros significativos relativos à matriz de confusão do BERT:

- Em “Economia”, ocorreram erros ao classificar como “Regional”.
- “Nacional” foi frequentemente confundido com “Política”.
- Erros consideráveis em “Regional” ao classificar como “Educação”.

Ambos os modelos têm dificuldades similares em algumas categorias, como a confusão entre “Nacional” e “Política”. Isso pode indicar sobreposições temáticas ou características linguísticas semelhantes nessas categorias. Abaixo estão descritos alguns pontos de diferença válidos:

- O XGBoost parece ter mais desafios em distinguir “Ciência” de “Natureza”, enquanto o BERT tem mais dificuldade em distinguir “Economia” de “Regional”.
- O BERT mostra uma ligeira vantagem em “Ciência”, mas o XGBoost se destaca em “Esportes”.

A análise das matrizes de confusão com dados balanceados revela que ambos os modelos são eficazes, mas com desafios específicos em certas categorias. A compreensão dessas nuances é fundamental para otimizar o desempenho dos modelos em aplicações de classificação de notícias.

4 Considerações finais

Nas considerações finais desta monografia, observa-se que o XGBoost e o BERT, apesar de suas abordagens distintas em processamento de linguagem natural, apresentam desempenhos notáveis na classificação de notícias em português. O XGBoost, um modelo baseado em *gradient boosting*, demonstrou ser ligeiramente superior em métricas como precisão, revocação e pontuação F1, especialmente após o balanceamento da base de dados. Esta vantagem pode ser atribuída à sua eficiência em lidar com características numéricas e sua capacidade de operar com menos recursos computacionais.

Por outro lado, o BERT, um modelo de rede neural profunda, embora mais pesado e demandando mais recursos computacionais, mostrou-se eficaz em capturar nuances semânticas complexas do texto. Sua complexidade, no entanto, pode ser um desafio em ambientes com recursos limitados, tanto em termos de tempo de treinamento quanto de requisitos de hardware.

A escolha entre esses modelos, portanto, deve ser guiada por uma consideração cuidadosa das necessidades específicas da aplicação, equilibrando eficácia, recursos disponíveis e complexidade do modelo. Enquanto o XGBoost oferece uma solução mais prática e menos onerosa para cenários com restrições de recursos, o BERT é adequado para contextos onde a compreensão profunda do texto é crucial, apesar das demandas maiores de computação.

Esta pesquisa destaca a importância do balanceamento dos dados e fornece *insights* valiosos para a aplicação prática e pesquisa futura em classificação de textos em língua portuguesa, abrindo caminho para inovações contínuas na área de processamento de linguagem natural.

4.1 Trabalhos futuros

Várias direções promissoras podem ser exploradas para expandir a compreensão e aplicabilidade dos modelos de classificação de texto em português. Primeiramente, é recomendável a implementação e avaliação de uma gama mais ampla de modelos de aprendizado de máquina, incluindo redes neurais mais recentes e avançadas, para uma comparação mais abrangente de desempenhos.

Além disso, um estudo mais detalhado sobre o impacto da qualidade e do volume dos dados utilizados nos modelos, incluindo técnicas avançadas de pré-processamento, poderia proporcionar insights valiosos para a otimização dos modelos existentes. A análise da interpretabilidade dos modelos, especialmente do BERT, também é uma área que

merece atenção, visando compreender melhor as nuances do processamento de linguagem natural.

Outro aspecto interessante seria a aplicação dos modelos em contextos específicos, como a detecção de notícias falsas ou análise de sentimentos, para testar sua eficácia em cenários práticos. Por fim, explorar a integração desses modelos em sistemas mais amplos, como plataformas de recomendação ou sistemas de alerta, poderia oferecer uma nova perspectiva sobre sua aplicabilidade em situações reais.

Essas direções não apenas fortaleceriam o campo do processamento de linguagem natural em português, mas também contribuiriam para o avanço da inteligência artificial em aplicações práticas e relevantes.

Referências

BAUM, L. E.; PETRIE, T. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 37, n. 6, p. 1554 – 1563, 1966. Disponível em: <https://doi.org/10.1214/aoms/1177699147>. Citado na página 34.

BENGFORT, B.; KIM, J. *Analítica de dados com Hadoop: Uma introdução para cientistas de dados*. Novatec Editora, 2016. ISBN 978-85-7522-521-9. Disponível em: <https://books.google.com.br/books?id=IUsDDQAAQBAJ>. Citado na página 53.

DOGRA, V. et al. A Complete Process of Text Classification System Using State-of-the-Art NLP Models. *Computational Intelligence and Neuroscience*, v. 2022, p. 1883698, jun. 2022. ISSN 1687-5265. Publisher: Hindawi. Disponível em: <https://doi.org/10.1155/2022/1883698>. Citado na página 27.

DSA, E. *Capítulo 8 - Função de Ativação*. 2022. Disponível em: <https://www.deeplearningbook.com.br/funcao-de-ativacao/>. Citado na página 35.

DSA, E. *Capítulo 85 - Transformadores - O Estado da Arte em Processamento de Linguagem Natural*. 2022. Disponível em: <https://www.deeplearningbook.com.br/transformadores-o-estado-da-arte-em-processamento-de-linguagem-natural/>. Citado na página 35.

FAWCETT, T. An introduction to ROC analysis. *Pattern Recognition Letters*, v. 27, n. 8, p. 861–874, jun. 2006. ISSN 01678655. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S016786550500303X>. Citado na página 65.

JOACHIMS, T. *Learning to Classify Text Using Support Vector Machines*. Springer US, 2002. Disponível em: <https://doi.org/10.1007/978-1-4615-0907-3>. Citado na página 34.

KOCAMAN, V.; TALBY, D. Spark nlp: Natural language understanding at scale. *Software Impacts*, p. 100058, 2021. ISSN 2665-9638. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2665963.2.300063>. Citado na página 27.

LOHR, S. The Age of Big Data. *The New York Times*, fev. 2012. ISSN 0362-4331. Disponível em: <https://www.nytimes.com/2012/02/12/sunday-review/big-datas-impact-in-the-world.html>. Citado na página 23.

MANNING, C. D.; RAGHAVAN, P.; SCHUTZE, H. *Introduction to Information Retrieval*. Cambridge, England: Cambridge University Press, 2008. Citado na página 34.

MARDESIC, P. et al. Bounding the length of iterated integrals of the first nonzero melnikov function. arXiv, n. arXiv:1703.03837, mar. 2017. ArXiv:1703.03837 [math]. Disponível em: <http://arxiv.org/abs/1703.03837>. Citado na página 56.

MILLER, G. A. Wordnet: A lexical database for english. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 38, n. 11, p. 39–41, nov 1995. ISSN

0001-0782. Disponível em: <https://doi.org/10.1145/219717.219748>. Citado na página 33.

MITCHELL, R. *Web Scraping com Python: Coletando dados na Web moderna*. Novatec Editora, 2015. ISBN 978-85-7522-447-2. Disponível em: <https://books.google.com.br/books?id=x3tWCgAAQBAJ>. Citado na página 28.

NUGROHO, K. S.; SUKMADEWA, A. Y.; YUDISTIRA, N. Large-scale news classification using BERT language model: Spark NLP approach. In: *6th International Conference on Sustainable Information Engineering and Technology 2021*. ACM, 2021. Disponível em: <https://doi.org/10.1145%2F3479645.3479658>. Citado na página 27.

POWERS, D. M. W. *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*. arXiv, 2020. ArXiv:2010.16061 [cs, stat]. Disponível em: <http://arxiv.org/abs/2010.16061>. Citado na página 63.

ROSCOE, B. *Internet é principal meio de informação para 43%; TV é mais usada por 40%*. 2021. Disponível em: <https://www.poder360.com.br/midia/internet-e-principal-meio-de-informacao-para-43-tv-e-preferida-de-40/>. Citado na página 23.

SANH, V. et al. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In: *NeurIPS EMC²Workshop*.[*S.l.* : *s.n.*], 2019. *Citadonapágina*57.

SELECTORS — Scrapy 2.9.0 documentation. Disponível em: <https://docs.scrapy.org/en/latest/topics/selectors.html#using-selectors>. Citado na página 40.

VAJJALA, S. et al. *Practical natural language processing*. Sebastopol, CA: O'Reilly Media, 2020. Citado na página 33.

VASWANI, A. et al. Attention is all you need. In: . [s.n.], 2017. Disponível em: <https://arxiv.org/pdf/1706.03762.pdf>. Citado na página 35.

WU, C. et al. *NewsBERT: Distilling Pre-trained Language Model for Intelligent News Application*. 2021. Citado na página 27.

XIAN, Y. et al. Zero-shot learning – a comprehensive evaluation of the good, the bad and the ugly. arXiv, n. arXiv:1707.00600, set. 2020. ArXiv:1707.00600 [cs]. Disponível em: <http://arxiv.org/abs/1707.00600>. Citado na página 61.

ZHANG, H.; LI, D. Naïve bayes text classifier. In: *2007 IEEE International Conference on Granular Computing (GRC 2007)*. [S.l.: s.n.], 2007. p. 708–708. Citado na página 34.

ŽIžKA, J.; DAřENA, F.; SVOBODA, A. *Text Mining with Machine Learning: Principles and Techniques*. CRC Press, 2019. ISBN 978-0-429-89027-7. Disponível em: <https://books.google.com.br/books?id=avm7DwAAQBAJ>. Citado na página 53.

Apêndices

APÊNDICE A – Materiais de suporte

Abaixo estão os links dos materiais de suporte que referenciam os *notebooks* com os trabalhos feitos:

- Mineração dos dados
- Conversão de CSV para Parquet
- Exploração de dados
- Modelo XGBoost e avaliação dos resultados
- Conversão de *dataset* - Pandas para Hugging Face
- Modelo BERT
- Avaliação dos resultados do Modelo BERT
- Conversão de *dataset* balanceado - Pandas para Hugging Face
- Modelo XGBoost balanceado
- Modelo BERT balanceado