Logotipo, nome da empresa

Descrição gerada automaticamente

**Universidade Paulista Campus Flamboyant Goiânia**

**Análise de sentimento através do resultado da API**

**e captura de protocolo de rede**

**Aluno: Matheus Ferreira De Araújo**

**RA: N910810**

**Goiânia – Goiás**

**2024**

Logotipo, nome da empresa

Descrição gerada automaticamente

**Análise de sentimento através do resultado da API**

**e captura de protocolo de rede**

**Aluno: Matheus Ferreira De Araújo**

**RA: N910810**

**Professores: Fernando Stela e André Campos**

**Disciplina: Atividades Práticas Supervisionadas**

**Curso: Ciências Da Computação**

**Goiânia – Goiás**

**2024**

**Índice**

**Objetivo do trabalho...............................................................................1.0**

**Introdução...............................................................................................2.0**

**Fundamentos da comunicação de dados em redes...........................4.0**

**Plano de desenvolvimento da aplicação.............................................6.0**

**Projeto (estrutura) do programa...........................................................8.0**

**- Arquivos CSV e JSON…………………………………...8.1**

**- Interfaces gráficas……………………………………….8.2**

**-Explicação do código...................................................8.3**

**Relatório com as linhas de código.....................................................20.0**

**Bibliografia.............................................................................................42**

**1.0 Objetivo do trabalho**

O objetivo em toda atividade prática supervisionada é fazer um projeto, por meio de um software, de todo o conteúdo ministrado em sala de aula ao longo de um período. No caso deste APS desse semestre não é diferente. Com o tema de busca de notícias através de uma API, tratamento dos dados para análise de sentimentos e captura dos protocolos de redes, podemos explorar diversas áreas do backend de um programa que realiza buscas em páginas e softwares, trazendo informações em formato JSON e possibilitando seu tratamento para oferecer ao usuário uma resposta satisfatória. O tratamento dos dados para análise de sentimentos é uma etapa crucial, pois envolve a utilização de técnicas de processamento de linguagem natural (PLN) para identificar padrões, emoções e opiniões expressas nas notícias.

Além disso, podemos compreender a complexidade do tratamento de dados de uma API, por exemplo, ao empregar uma IA para aprendizado e análise das informações fornecidas a ela, resultando na geração de um arquivo com os dados processados e analisados pela IA. Isso nos permite também entender melhor o fluxo de informações na rede onde no âmbito da busca de notícias usando uma API, é fundamental entender o funcionamento das requisições HTTP, a manipulação de dados em formato JSON e a integração desses dados em um sistema de análise de sentimentos. A API serve como uma ponte para acessar informações externas de maneira estruturada e eficiente, facilitando a obtenção de dados para análise e processamento posterior.

O trabalho tem como objetivo principal entender todo o processo de backend de um software, aplicativo ou site que utiliza requisições GET para obtenção de informações, dados ou notícias por meio de APIs públicas ou privadas, que muitas vezes são pagas para acesso a dados ilimitados. Através desse projeto, posso ter uma visão mais ampla do ambiente digital no contexto da busca por informações ou produtos, sejam eles físicos ou digitais. Também é possível aprender como são conduzidas vendas para clientes específicos com base nos dados de navegação dos próprios usuários, entre outras aplicações relacionadas ao tema do trabalho.

**1**

**2.0 Introdução**

Para entender um pouco mais sobre api e treinamento e analise de dados através de IA, vamos da uma breve introdução na história tecnológica. Em 1950

Alan Turing matemático e cientista informático já vinha estudando o possível comportamento de máquinas com o próprio nome dito inteligência artificial, nesse teste que avalia a capacidade de uma máquina exibir comportamento inteligente equivalente ou indistinguível do comportamento humano. A ideia por trás desse teste é verificar se uma máquina pode "pensar" de forma tão sofisticada que um observador humano não consiga distinguir suas respostas das de um humano. Basicamente o teste consiste em um cenário onde um observador humano interage com duas entidades, uma máquina e um ser humano, através de um sistema de comunicação como texto, por exemplo. Se o observador não puder distinguir qual é a máquina e qual é o humano com base nas respostas recebidas, então a máquina é considerada como tendo passado no Teste de Turing, demonstrando um comportamento inteligente comparável ao humano. Esse teste de Turing é muito importante para o meio acadêmico e tecnológico voltado para IA pois foi um ponta pé para que possamos estudar e aprimorar todo tipo de máquina que possa ser colocada uma IA foi um marco importante na história da inteligência artificial, pois na época o Alan conseguiu desafiar a comunidade científica colocando ela para pensar sobre o que realmente significa ser inteligente e se é possível que uma máquina alcance esse nível de inteligência. Sabendo disso hoje em 2024 podemos ver ao nosso redor diversos equipamentos eletrônicos, sites, app, softwares que tem o uso de inteligência artificial, segundo uma manchete da própria Microsoft feita em 4 de março de 2024 onde a IA estar presente 74% no dia a dia dos brasileiros e que 90% da população busca adotar IA no dia a dia. Mas o que seria basicamente a inteligência artificial (IA) é um campo da ciência da computação ou outras formações que visa desenvolver sistemas e máquinas capazes de realizar tarefas que exigiriam inteligência se realizadas por seres humanos. Isso inclui aprender, raciocinar, resolver problemas, perceber o ambiente e gerar linguagem. Os sistemas de IA podem ser baseados em regras, aprendizado de máquina, redes neurais e outras técnicas para simular a inteligência humana em diversas áreas, melhorando a automação, a produtividade e a capacidade de lidar com problemas complexos.

**2**

Sabendo um pouco sobre IA o que seria API, atualmente no meio de programação essa palavra tem ganha muita força mas não se engane as apis já existe a muito tempo, década de 1960 e 1970 nesse período inicial, as APIs eram mais relacionadas a interfaces de sistemas operacionais e linguagens de programação, permitindo que os desenvolvedores interagissem com hardware e software de forma mais flexível, já em 1980 e 1990 com o avanço da Internet e o surgimento da World Wide Web, as APIs foram fundamentais para permitir a comunicação entre diferentes sistemas e serviços online. Empresas como Microsoft, Apple e outras começaram a desenvolver APIs para seus produtos e plataformas. Na década de 2000 em diante até os tempos atuas com o crescimento das plataformas web e móveis, as APIs se tornaram ainda mais importantes, possibilitando a integração de aplicativos e serviços de terceiros. Empresas como Google, Facebook, Twitter e Amazon se destacaram por disponibilizar APIs poderosas para desenvolvedores. Explicando um pouco melhor uma API (Interface de Programação de Aplicações) é um conjunto de regras e protocolos que permite que diferentes softwares se comuniquem entre si de forma padronizada. Ela define como os programas devem interagir, facilitando a integração de funcionalidades e o acesso a recursos específicos de um sistema. As APIs são amplamente utilizadas para criar integrações, desenvolver aplicativos e automatizar processos, promovendo a interconexão e colaboração entre diferentes plataformas e serviços.

Após essa introdução, neste trabalho será explorado o uso de uma API específica para obtenção de informações sobre notícias ambientais, considerando aspectos como a fonte das notícias, a relevância dos dados e a precisão das informações coletadas. Além disso, será realizado um monitoramento detalhado dos protocolos de rede envolvidos nesse processo, destacando a segurança e a eficiência na transmissão e recebimento dos dados. Em seguida, será aplicada uma análise avançada das notícias obtidas, utilizando técnicas de processamento de linguagem natural e aprendizado de máquina. A IA será empregada para avaliar o sentimento de cada notícia, classificando-as em categorias como positivo, negativo ou neutro. Isso proporcionará insights valiosos sobre a percepção pública em relação a questões ambientais, auxiliando na compreensão das tendências e na identificação de áreas de preocupação ou otimismo.

**3**

**4.0 Fundamentos da comunicação de dados em redes**

Com o avanço da tecnologia e a popularização da internet, as redes se tornaram uma infraestrutura essencial para a sociedade como um todo. Além das redes locais e metropolitanas, surgiram as redes globais, conectando pessoas, organizações e dispositivos em escala mundial.O desenvolvimento das redes de longa distância, como as WANs, revolucionou a forma como empresas e instituições colaboram e compartilham informações em nível global. Isso possibilitou a criação de sistemas de comunicação mais robustos e eficientes, impactando diretamente setores como o comércio, a educação e a pesquisa científica.

Um exemplo emblemático dessa evolução é a criação da World Wide Web (WWW) por Tim Berners-Lee em 1989, que transformou a internet em uma plataforma acessível para o compartilhamento de informações em formato de páginas web. Isso impulsionou o surgimento de novas tecnologias, como as redes sociais, que se tornaram um meio fundamental de comunicação e interação social na era digital.

Além disso, as redes de computadores desempenham um papel crucial na era da Internet das Coisas (IoT), conectando dispositivos inteligentes e sensores em uma ampla rede de comunicação. Isso possibilita a automação de processos, a coleta de dados em tempo real e o desenvolvimento de soluções inovadoras em áreas como cidades inteligentes, saúde digital e indústria 4.0.

Com o advento das tecnologias de nuvem e computação distribuída, as redes se tornaram ainda mais flexíveis e escaláveis, permitindo o armazenamento e processamento de grandes volumes de dados de forma ágil e eficiente. Isso tem impulsionado o desenvolvimento de serviços em nuvem, plataformas de streaming, aplicativos móveis e muito mais, transformando a maneira como interagimos com a tecnologia no dia a dia.

Em resumo, as redes de computadores evoluíram de simples conexões locais para uma infraestrutura global que sustenta a era digital, possibilitando a troca de informações, o acesso a recursos compartilhados e a inovação tecnológica em diversas áreas da sociedade moderna. A interconexão global proporcionada pelas redes não só facilita a comunicação entre pessoas, mas também permite a colaboração entre empresas em diferentes partes do mundo. Essa conectividade é essencial para o comércio internacional, permitindo transações rápidas e seguras, bem como a gestão eficiente da cadeia de suprimentos em escala global.

**4**

Além do impacto econômico, as redes globais têm um papel fundamental na disseminação do conhecimento e da cultura. Plataformas de ensino online, museus virtuais e bibliotecas digitais são exemplos de como a internet e as redes conectam pessoas a informações e recursos educacionais em qualquer lugar do mundo.

No campo da saúde, as redes desempenham um papel vital na troca de informações médicas, permitindo a telemedicina, o compartilhamento de pesquisas e o acesso a especialistas em diferentes regiões. Isso é especialmente importante em emergências e para o avanço da medicina em áreas remotas.

A segurança cibernética tornou-se uma preocupação central com o aumento da dependência das redes globais. A proteção de dados, a prevenção de ataques cibernéticos e a garantia da privacidade dos usuários são desafios contínuos enfrentados pelas organizações e governos em todo o mundo.

O desenvolvimento futuro das redes está intimamente ligado a avanços em áreas como inteligência artificial, computação quântica e redes 5G. Essas tecnologias prometem redes mais rápidas, seguras e inteligentes, capazes de suportar uma gama ainda maior de aplicações e serviços inovadores.

À medida que a sociedade evolui, as redes continuam a desempenhar um papel fundamental na integração global, na disseminação do conhecimento e na promoção da colaboração e inovação em todos os aspectos da vida moderna.

**5**

**6.0 Plano de desenvolvimento da aplicação**

A API usada foi do GNews pois para esta fazendo get de informaçoes nela e gratuita e bem simples de ser usada. A escolha da linguagem Python foi baseada em sua recomendação amplamente aceita para projetos de Inteligência Artificial e manipulação de dados por meio de APIs. Além disso, escolhir o Python por ser uma linguagem onde tenho mais facilidade e conhecimento de trabalho.Ao utilizar Python para nossa APS, estamos aproveitando não apenas sua eficiência e versatilidade em projetos de IA, mas também a vasta quantidade de recursos e bibliotecas disponíveis que facilitam o desenvolvimento e a implementação de funcionalidades complexas. Isso me permitiu explorar não apenas os fundamentos da linguagem, mas também a aplicação prática de técnicas avançadas de análise de dados e automação de tarefas através de APIs.Adicionalmente, a escolha do Python reflete a tendência do mercado de tecnologia, onde a linguagem se tornou uma das mais populares e demandadas, oferecendo oportunidades de carreira e networking para aqueles que a dominam proficientemente. Dessa forma, nossa decisão vai além da simples escolha de uma linguagem para um projeto específico, mas também considera o impacto positivo que esse conhecimento pode ter em nosso desenvolvimento profissional a longo prazo.

Para o desenvolvimento o software foi utilizado a versão 3.12.3 do Python para executar os programas na linguagem Python, também vamos usar a IDE Visual Studio Code para criar o código fonte do programa.

Será implementada uma IA baseada em Machine Learning para realizar a análise de sentimentos das notícias buscadas, monitorar as estatísticas dos últimos 5 dias e apresentar os resultados por meio de números e gráficos. A IA utilizará técnicas avançadas de Machine Learning para classificar textos em positivos e negativos, contribuindo para uma análise mais precisa e eficiente dos sentimentos expressos.

Além disso, o programa incluirá o monitoramento dos pacotes de rede utilizando a biblioteca Scapy é após isso e necessário que seja baixada juntamente com o Python o Npcap para conseguir fazer o monitoramento de rede. A escolha do Scapy se alinha perfeitamente com os objetivos do programa, proporcionando uma ferramenta robusta e flexível para captura e análise de dados de rede em tempo real.

**6**

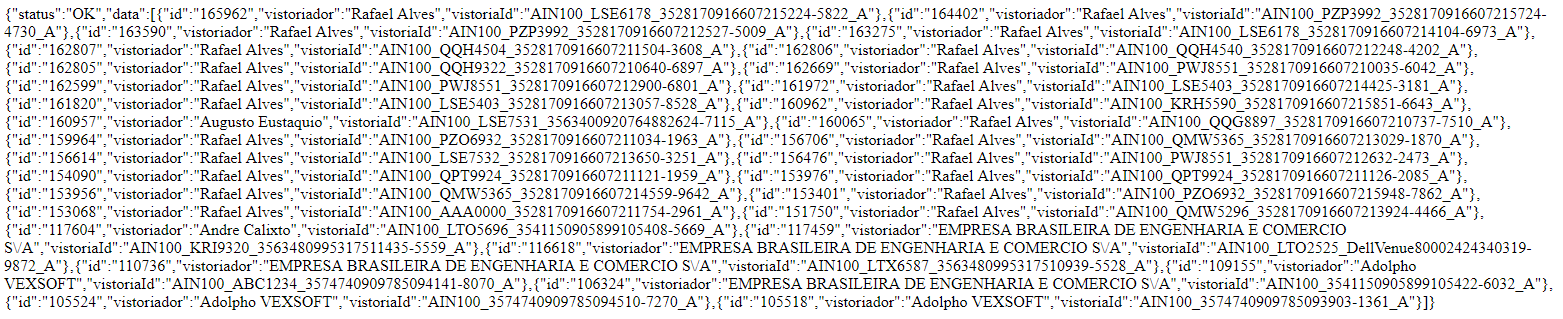
Com essas implementações, o projeto não apenas realizará análises avançadas de sentimentos textuais, mas também fornecerá insights valiosos sobre o comportamento da rede, permitindo uma visão abrangente e detalhada do ambiente de comunicação.

Foi usado de arquivos no formato CSV para treinar o sistema de inteligência artificial (IA) e também como uma forma de converter os resultados retornados pela API em formato JSON. Embora o JSON seja a forma mais comum de retorno de informações de uma API, não é tão agradável visualmente para o usuário final. Por esse motivo, optei pelo formato de planilha CSV.

Além das dependências mencionadas anteriormente, foi necessário utilizar arquivos para o treinamento da IA, denominados 'treinochuvas.csv', 'treinoenchentes.csv' e 'treinoqueimadas.csv'. Estes correspondem às três possíveis consultas que o usuário pode selecionar. Quando o usuário escolhe uma das opções (chuvas, enchentes ou queimadas), um condicional 'if' determina qual arquivo de treinamento será usado para a IA.

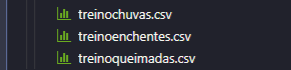
Abaixo estão imagens dos arquivos CSV mencionados, juntamente com um exemplo de arquivo JSON."

**Arquivo JSON**



*Autoria própria*

**Arquivos CSV**

****

*Autoria própria*

**7**

**8.0 Projeto (estrutura) do programa**

O projeto tem como objetivo principal o desenvolvimento de um software que realiza buscas na web relacionadas ao tema em que o usuário optar entre 3 possíveis no software do meio ambiente, empregando técnicas de Inteligência Artificial e monitoramento de protocolos de rede para uma abordagem abrangente e eficiente.

Inicialmente, implementamos a funcionalidade de busca utilizando a API do Gnews. Essa API nos proporciona acesso a qualquer tipo de notícia brasileira relacionada ao tema selecionado relacionados. Através dessa integração, e possível extrair informações relevantes dos últimos cinco dias, abrangendo tanto aspectos positivos quanto negativos.

Além da interface de pesquisa, a API também fornece dados sobre os assuntos em destaque no contexto ambiental, permitindo uma análise completa das discussões em curso. A imagem abaixo ilustra a interface do software, onde nela podemos ver a tela inicial que tem o nome de home.py, nela iremos ver 4 botes na parte superiora do frame onde o usuário terá a opção de selecionar o que deseja ser feito, fazer uma nova pesquisa de notícias, analisar um notícias já pesquisada com a IA, olhar a porcentagem de informações positivas e negativas de uma notícia especifica já pesquisadas é por fim ver uma comparação das notícias em gráfico.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

*Autoria própria*

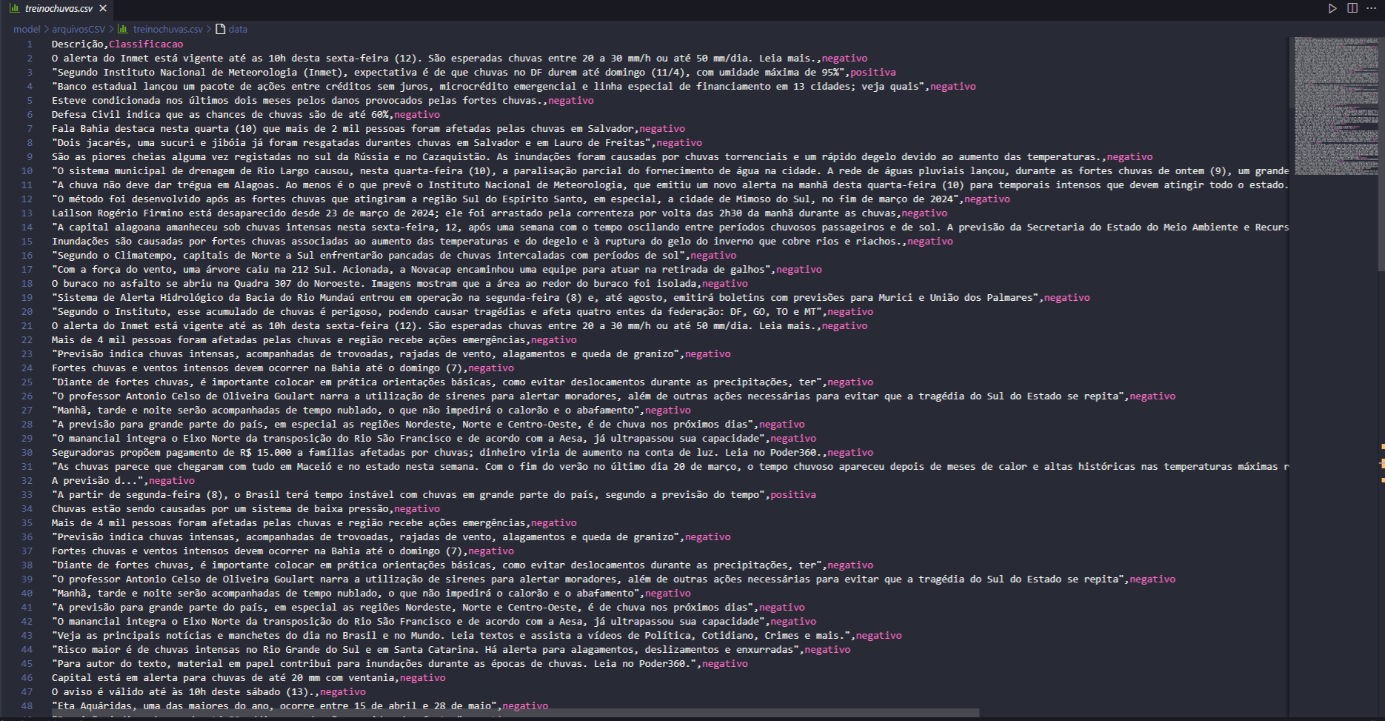
**8**

**8.1 Arquivos CSV e JSON**

Conforme mencionado anteriormente, utilizamos três arquivos de treinamento para aprimorar a IA: 'treinochuvas.csv', 'treinoenchentes.csv' e 'treinoqueimadas.csv'. Cada um desses arquivos representa pesquisas coletadas ao longo de uma semana sobre seus respectivos temas. Após essa fase inicial, compilamos todas as informações das pesquisas em um único arquivo CSV para cada tema. Por exemplo, todas as pesquisas sobre chuvas foram consolidadas no arquivo 'treinochuvas', e assim por diante para os demais temas. Essa organização foi realizada para garantir um treinamento mais preciso das análises das novas pesquisas que serão conduzidas por meio do software. Nestes arquivos, já analisados previamente pela própria IA, encontram-se duas tipos de informações: a descrição, ou seja, o texto analisado, e a classificação do texto, indicando se é positivo ou negativo. Essa abordagem contribui significativamente para aprimorar a precisão da IA, pois quanto mais dados de treinamento ela recebe, maior será sua capacidade de acerto nas análises futuras.

Abaixo, você encontrará uma imagem do conteúdo de cada um dos arquivos mencionados.

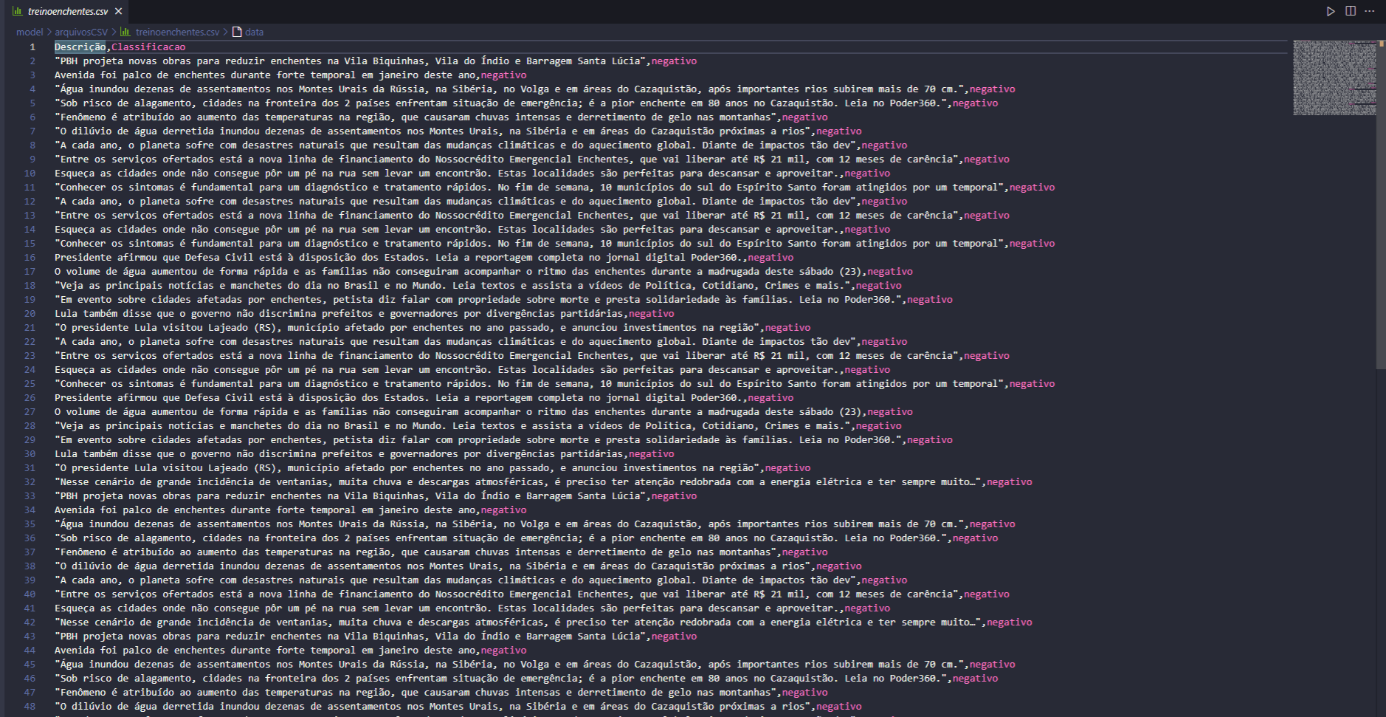
**Treinochuvas.csv**

****

*Autoria própria*

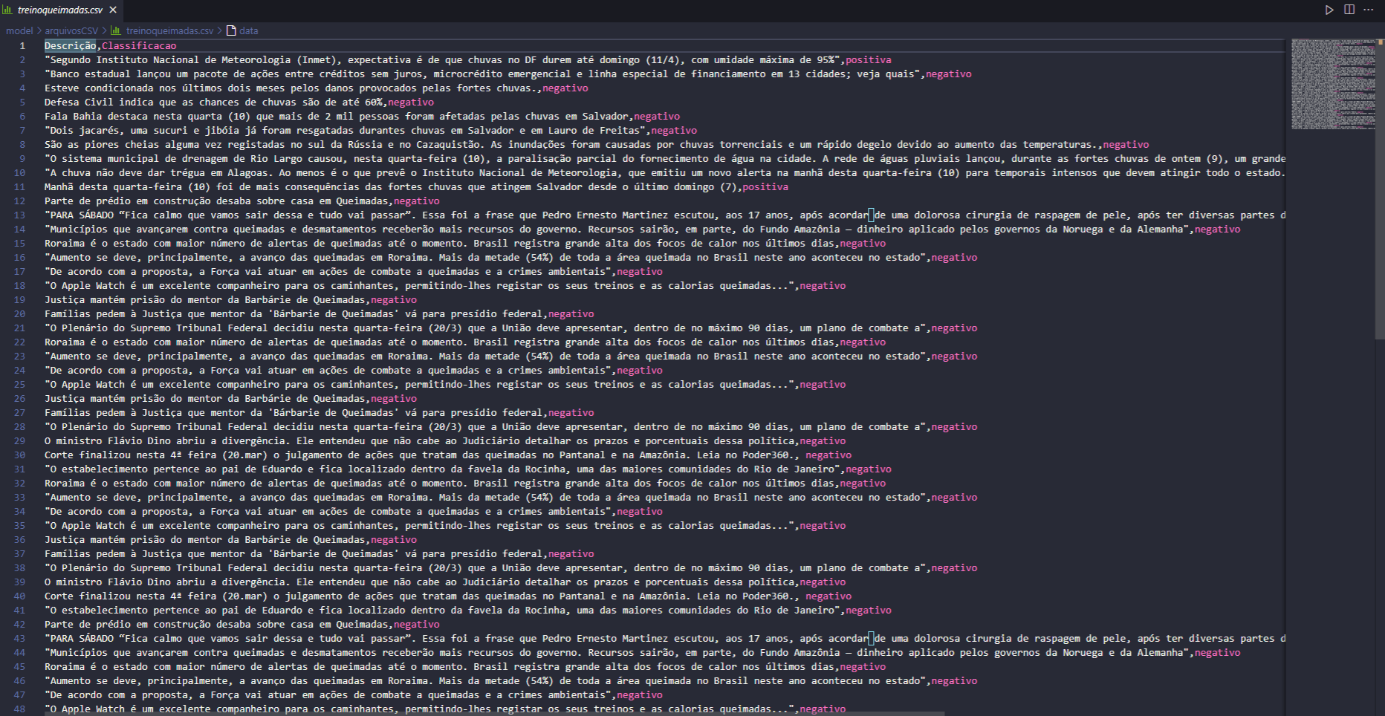
**9**

**Treinoenchentes.csv**



*Autoria própria*

**Treinoqueimadas.csv**



*Autoria própria*

*.*

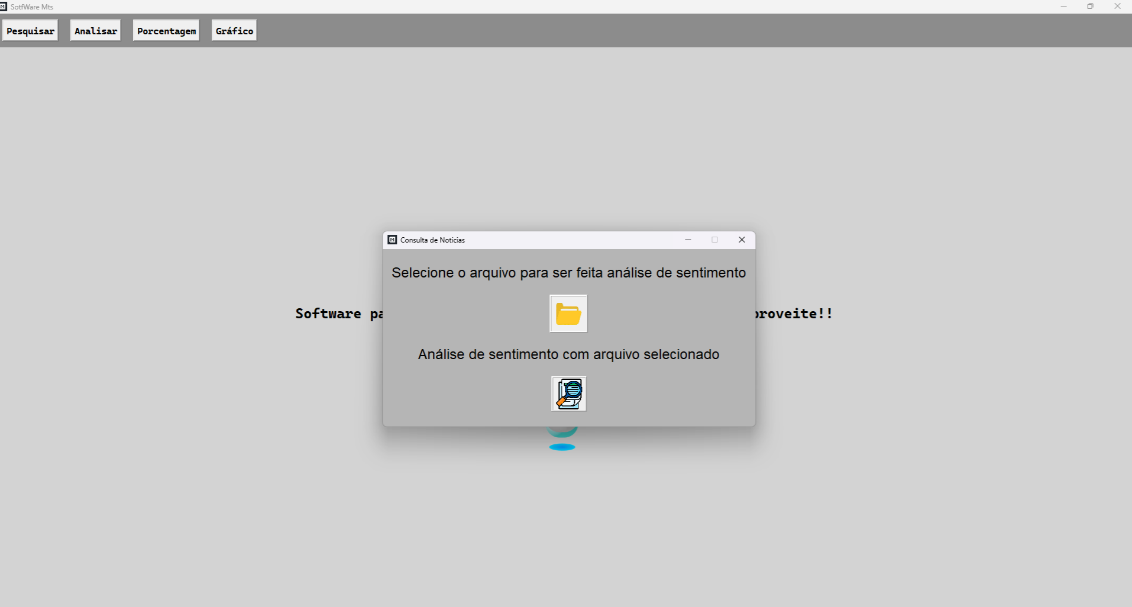
**10**

**8.2 Interfaces graficas**

No projeto, há 5 frames que apresentam opções para os usuários ao utilizar o software (frontend). A tela principal já foi mostrada na figura 1, e as outras 4 telas visíveis para o usuário são Janela.py, Janela2.py, Janela3.py e Janela4.py. Cada uma delas oferece opções diferentes para o usuário do programa. Além disso, há um código no backend chamado busca.py, responsável por realizar a comunicação com a API. Embora não seja visível para o usuário, ele tem grande importância, pois é por meio dele que toda a busca de dados é realizada.

Abaixo esta exemplo dos frames citados.

**Janela4.py** *Autoria própria*



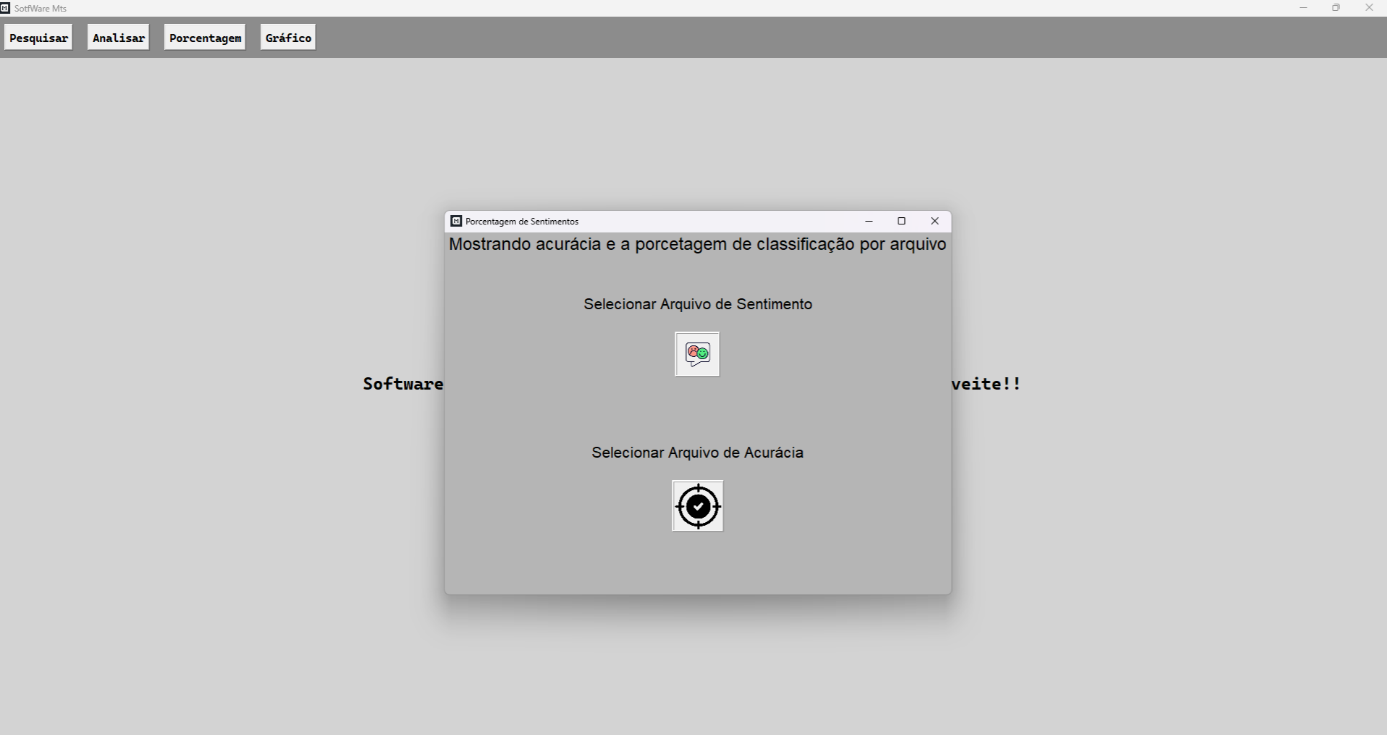
**Janela.py** *Autoria própria*

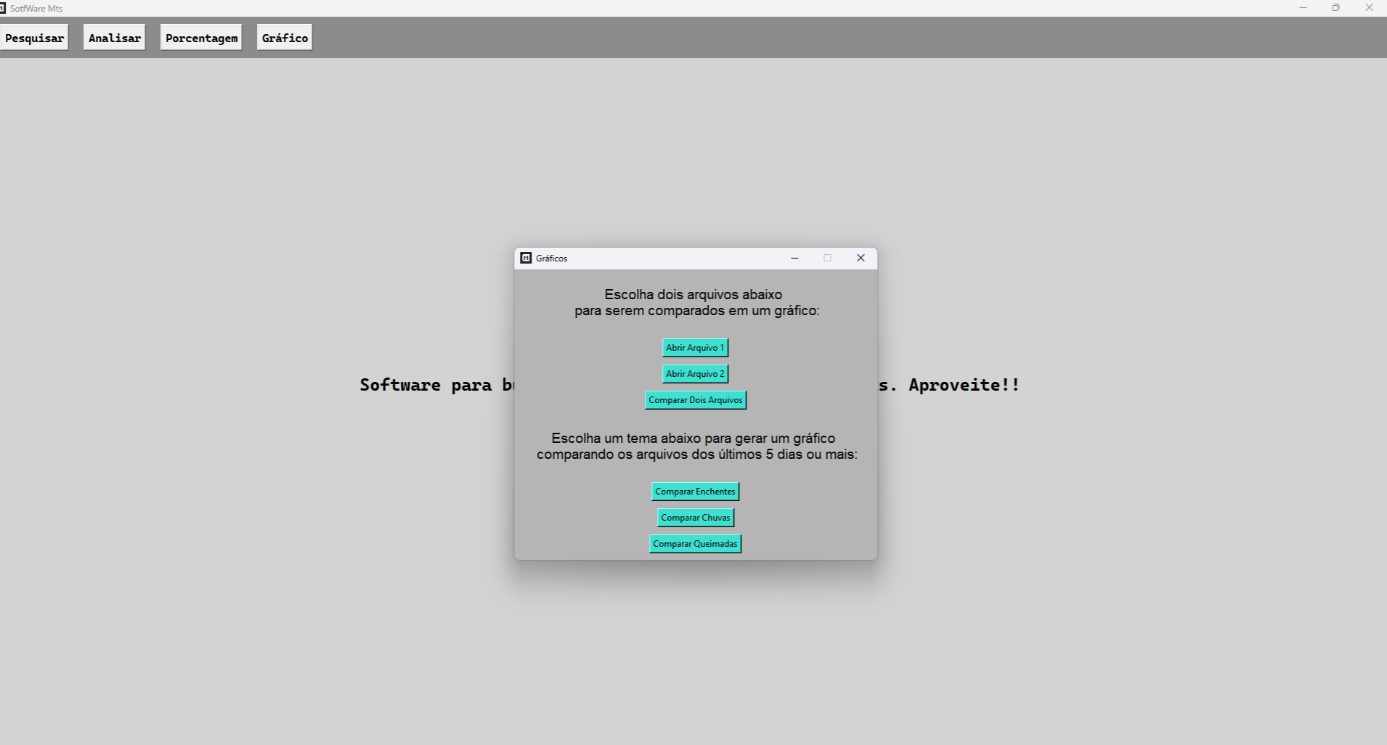


**11**

**Janela2.py**

*Autoria própria*



**Janela3.py** *Autoria própria*

**12**

**8.3 Explicação do código**

As buscas são realizadas utilizando a API de busca disponibilizada pelo GNews. Para acessar essa API, é necessário criar uma conta no site do Gnews https://gnews.io/. Após criar a conta o próximo passo e pegar sua chave API é após não e necessário mais o uso do site só para conseguir monitorar o quanto e usado da sua chave API. Com esses dados em mãos, podemos implementar o código.

O programa a seguir é uma aplicação de interface gráfica em Python usando a biblioteca Tkinter, que é comumente utilizada para desenvolver GUIs (Graphical User Interfaces).

**Imports de Bibliotecas:**

No início do código, são feitos diversos imports de bibliotecas essenciais para diferentes funcionalidades. Por exemplo, tkinter é a principal biblioteca para construção de interfaces gráficas em Python, enquanto subprocess é utilizada para executar comandos do sistema operacional, e pandas é empregada para manipulação de dados em formato tabular.

**Funções Definidas:**

Logo após os imports, são definidas algumas funções que serão utilizadas posteriormente no programa. Por exemplo, a função load\_image é responsável por carregar e redimensionar imagens para exibi-las na interface gráfica, enquanto pesquisar\_noticias é chamada para realizar a busca de notícias com base no tema selecionado pelo usuário.

**Configuração da Janela Principal:**

A seguir, o código configura a janela principal da interface gráfica. Isso inclui definir o título da janela, o ícone a ser exibido e o tamanho da janela, além de centralizá-la na tela do usuário.

**Componentes da Interface Gráfica:**

Dentro da janela principal, são adicionados diversos componentes visuais, como labels (rótulos de texto) para exibir informações importantes, como o título da aplicação e instruções para o usuário. Além disso, são criados botões de rádio para permitir ao usuário escolher um tema de interesse.  **13**

**Funcionalidades dos Componentes:**

Cada componente visual adicionado à interface possui uma funcionalidade associada. Por exemplo, ao clicar no botão de pesquisa de notícias, a função pesquisar\_noticias é chamada para iniciar a busca de notícias relacionadas ao tema selecionado pelo usuário.

**Redirecionamento de Saída:**

Uma parte interessante do código é o uso da função redirect\_output, que redireciona a saída de um comando específico para a interface gráfica. Isso pode ser útil para exibir informações geradas por comandos ou scripts em tempo real na interface do usuário.

**Loop Principal da Interface Gráfica:**

Por fim, o código entra em um loop principal que mantém a interface gráfica aberta e interativa. Dentro desse loop, a aplicação aguarda as interações do usuário, como clicar em botões ou selecionar opções.

Em resumo, o código apresentado é um exemplo básico de como criar uma interface gráfica em Python usando Tkinter, incorporando funcionalidades interativas e elementos visuais para fornecer uma experiência amigável ao usuário.

**Janela2.py**

**Imports de Bibliotecas:**

O código inicia importando o módulo tkinter para criação da interface gráfica, filedialog para lidar com seleção de arquivos, csv para manipulação de arquivos CSV, subprocess para abrir uma nova janela, e PhotoImage para trabalhar com imagens na interface.

**Definição de Funções:**

Em seguida, são definidas várias funções que serão chamadas ao interagir com a interface:

selecionar\_sentimento(): Permite ao usuário selecionar um arquivo CSV de sentimento.

selecionar\_acuracia(): Permite ao usuário selecionar um arquivo CSV de acurácia.

exibir\_acuracia(): Exibe a acurácia do modelo selecionado a partir do arquivo CSV.

calcular\_porcentagens(): Calcula e exibe as porcentagens de classificação positiva e negativa a partir do arquivo CSV de sentimento.

**Criação da Janela Principal:**

O código cria a janela principal da interface com título e ícone definidos, além de configurar sua geometria para centralizá-la na tela do usuário.

**Componentes da Interface Gráfica:**

Dentro da janela principal, são adicionados diversos componentes visuais, como labels para exibir informações, botões para selecionar arquivos, e labels para exibir os resultados das análises.

**Funcionalidades dos Componentes:**

Cada componente visual adicionado à interface possui uma funcionalidade associada. Por exemplo, ao clicar nos botões para selecionar arquivos, são chamadas as funções selecionar\_sentimento() e selecionar\_acuracia() para lidar com a seleção dos arquivos correspondentes.

**Abrir uma Nova Janela:**

Há uma função abrir\_nova\_janela() definida para abrir uma nova janela quando necessário, utilizando o módulo subprocess.

**Loop Principal da Interface Gráfica:**

Por fim, o código entra em um loop principal que mantém a interface gráfica aberta e interativa. Dentro desse loop, a aplicação aguarda as interações do usuário, como clicar em botões ou selecionar arquivos.

Em resumo, o código é uma aplicação GUI que permite ao usuário selecionar arquivos CSV de sentimento e acurácia, calcular e exibir porcentagens de classificação, e abrir uma nova janela conforme necessário. É uma aplicação útil para análise de dados e exibição de resultados de forma visual na interface gráfica.

**15**

**Janela3.py**

**Funções:**

- `abrir\_arquivo1` e `abrir\_arquivo2`: Utilizadas para abrir arquivos CSV através de uma janela de seleção.

- `comparar\_dois\_arquivos`: Compara dois arquivos CSV selecionados e plota um gráfico com a comparação.

- Funções como `comparar\_enchentes`, `comparar\_chuvas`, e `comparar\_queimadas` são específicas para comparar arquivos relacionados a diferentes temas (enchentes, chuvas, queimadas) e plotar os gráficos correspondentes.

- `ler\_todos\_arquivos`: Lê todos os arquivos CSV em uma pasta e retorna os dados.

- `ler\_arquivo\_csv`: Lê um arquivo CSV, conta a quantidade de positivos e negativos e retorna esses valores em um dicionário.

- `plotar\_grafico`: Plota um gráfico de barras com os dados fornecidos.

- `adicionar\_valores\_barras`: Adiciona os valores nas barras do gráfico.

**Inicialização da Interface Gráfica:**

- O bloco principal (`if \_\_name\_\_ == "\_\_main\_\_":`) inicia a interface gráfica (`tk.Tk()`), define suas características (título, tamanho, ícone, cor de fundo), centraliza a janela na tela e cria e posiciona os elementos (rótulos, botões) na interface.

- Os botões têm comandos associados que realizam as ações de abrir arquivos, comparar arquivos, e plotar gráficos, chamando as funções definidas anteriormente.

**Loop Principal:**

- O loop principal (`janela3.mainloop()`) mantém a interface gráfica aberta e aguardando interações do usuário.Em resumo, o código cria uma interface gráfica interativa onde o usuário pode selecionar arquivos CSV, compará-los através de diferentes métodos (por tema ou selecionando diretamente dois arquivos), e visualizar as comparações em gráficos de barras.

**17**

**Janela4.py**

**Criação da Janela Principal:**

- A janela principal (`janela4`) é criada com título, ícone, tamanho e posição centralizada na tela.

**Criação do Frame Principal:**

- É criado um frame (`frame`) dentro da janela principal, com um background azul claro (`#B5B5B5`), e configuração para se expandir conforme o tamanho da janela.

**Elementos da Interface Gráfica:**

- Um label (`label\_estatus`) é criado para exibir o status da operação.

- São definidas funções para selecionar um arquivo (`selecionar\_arquivo`) e para analisar notícias (`analisar\_noticias`).

**Funcionalidades dos Botões:**

- O botão `btn\_selec` é criado com uma imagem de uma pasta para permitir ao usuário selecionar um arquivo.

- O botão `btn\_analisar` é criado com uma imagem para realizar a análise de sentimentos com o arquivo selecionado.

- Ambos os botões têm configurações de estilo e eventos para interação do usuário (hover).

**Loop Principal da Interface Gráfica:**

- O loop principal (`janela4.mainloop()`) é iniciado para manter a interface gráfica aberta e aguardando interações do usuário.Resumindo, esse código cria uma interface gráfica utilizando o Tkinter onde o usuário pode selecionar um arquivo CSV relacionado a notícias, analisar o sentimento dessas notícias utilizando um modelo de Perceptron multicamadas (MLP), e salvar os resultados da análise e a acurácia em arquivos CSV específicos.

**18**

**busca.py**

**Importação de Módulos:**

- `requests`: Para fazer solicitações HTTP.

- `csv`: Para manipular arquivos CSV.

- `os`: Para verificar se o arquivo CSV existe e criar diretórios.

- `scapy`: Importa todas as funções da biblioteca Scapy para manipulação de pacotes de rede.

- `sys`: Para acessar funcionalidades do sistema.

- `datetime`: Para manipulação de datas e horas.

**Função `pegar\_noticias(api\_key, query)`:**

- Utiliza a API da GNews para obter notícias com base em uma chave de API e um termo de pesquisa.

- Retorna os dados das notícias em formato JSON se a solicitação for bem-sucedida, senão retorna `None`.

**Função `salvar\_dados(news\_data, tema)`:**

- Salva os dados das notícias em um arquivo CSV com o tema e a data no nome do arquivo.

- Cria um diretório para salvar os arquivos de busca se não existir.

- Verifica se já existe um arquivo com a mesma data e adiciona um índice ao nome do arquivo se necessário.

- Escreve os dados das notícias no arquivo CSV, incluindo título, descrição, URL e data de publicação.

**Função `captura\_pacotes\_rede(count=10)`:**

- Utiliza a função `sniff` da biblioteca Scapy para capturar pacotes de rede.

- Exibe informações básicas sobre os pacotes capturados, incluindo o resumo do pacote e seus detalhes.

**Função `main()`:**

- Define a chave de API da GNews e o termo de pesquisa a partir do argumento da linha de comando.

- Obtém os dados das notícias chamando a função `pegar\_noticias(api\_key, query)`. **19**

- Salva os dados das notícias em um arquivo CSV chamando a função `salvar\_dados(news\_data, tema)`.

- Captura e exibe os pacotes de rede chamando a função `captura\_pacotes\_rede(count=10)`.

**Verificação do Programa Principal:**

- Verifica se o script está sendo executado como o programa principal.

- Chama a função `main()` se for o programa principal.

Resumindo, este código é um script em Python que utiliza a biblioteca `requests` para obter notícias da API da GNews com base em um termo de pesquisa e chave de API, salva essas notícias em arquivos CSV com tema e data, e também captura e exibe pacotes de rede usando a biblioteca Scapy.

**14.Relatório com as linhas de código**

**busca.py**

import requests

import csv

import os

from scapy.all import \*

import sys

import datetime

def pegar\_noticias(api\_key, query):

url = f'https://gnews.io/api/v4/search?q={query}&token={api\_key}'

solicitacao = requests.get(url)

if solicitacao.status\_code == 200:

return solicitacao.json()

else:

print(f'Erro ao obter notícias: {solicitacao.status\_code}')

return None

**20**

def salvar\_dados(news\_data, tema):

if news\_data is None:

return

data\_atual = datetime.datetime.now().strftime("%d-%m-%Y")

busca\_dir = f'model/arquivosCSV/Busca/busca{tema.capitalize()}'

os.makedirs(busca\_dir, exist\_ok=True)

existing\_files = [f for f in os.listdir(busca\_dir) if f.startswith(f'noticias\_{tema}\_{data\_atual}')]

if existing\_files:

new\_index = len(existing\_files) + 1

filepath = f'{busca\_dir}/noticias\_{tema}\_{data\_atual}{new\_index}.csv'

else:

filepath = f'{busca\_dir}/noticias\_{tema}\_{data\_atual}.csv'

with open(filepath, 'a', newline='', encoding='utf-8') as csvfile:

colunas = ['Título', 'Descrição', 'URL', 'Publicado em']

escrito = csv.DictWriter(csvfile, fieldnames=colunas)

if csvfile.tell() == 0:

escrito.writeheader()

for article in news\_data['articles']:

escrito.writerow({

'Título': article['title'],

'Descrição': article['description'],

'URL': article['url'],

'Publicado em': article['publishedAt']

})

def captura\_pacotes\_rede(count=10):

packet = sniff(count=count)

for idx, packet in enumerate(packet):

print(f'Pacote {idx + 1}: {packet.summary()}')

print(packet.show()) **21**

def main():

api\_key = '3b3c5b02449c955a4bc05e9d20461d04'

query = sys.argv[1]

noticias = pegar\_noticias(api\_key, query)

if noticias:

salvar\_dados(noticias, query)

print(f'Notícias adicionadas ao arquivo "model/arquivosCSV/Busca/busca{query.capitalize()}/noticias\_{query}\_{datetime.datetime.now().strftime("%Y-%m-%d")}.csv"')

else:

print(f'Falha ao obter notícias sobre {query}.')

print('\nCapturando pacotes de rede:')

captura\_pacotes\_rede(count=10)

if \_\_name\_\_ == '\_\_main\_\_':

main()

**home.py**

import tkinter as tk

import subprocess

from PIL import Image, ImageTk

class MainFrame:

def \_\_init\_\_(self, master):

master.title("SotfWare Mts")

largura\_janela = 1920

altura\_janela = 1080

largura\_tela = master.winfo\_screenwidth()

altura\_tela = master.winfo\_screenheight()  **22**

x = (largura\_tela - largura\_janela) // 2

y = (altura\_tela - largura\_janela) // 2

master.geometry(f"{largura\_janela}x{altura\_janela}+{x}+{y}")

main\_frame = tk.Frame(master, bg="#D3D3D3")

main\_frame.pack(fill=tk.BOTH, expand=True)

message\_label = tk.Label(main\_frame, text="Software para busca e análise de notícias ambientais. Aproveite!!", font=("Cascadia Code", 18, "bold"), fg="black", bg="#D3D3D3")

message\_label.place(relx=0.5, rely=0.5, anchor=tk.CENTER)

gif\_path = "image/robo.png"

gif\_image = Image.open(gif\_path)

gif\_image = gif\_image.resize((200, 200))

gif\_photo = ImageTk.PhotoImage(gif\_image)

gif\_label = tk.Label(main\_frame, image=gif\_photo, bg="#D3D3D3")

gif\_label.image = gif\_photo

gif\_label.place(relx=0.5, rely=0.65, anchor=tk.CENTER)

top\_menu = tk.Frame(main\_frame, bg="#8c8c8c")

top\_menu.pack(side=tk.TOP, fill=tk.X)

btn\_pesquisar = tk.Button(top\_menu, text="Pesquisar", command=open\_janela, font=("Cascadia Code", 12, "bold"), borderwidth=2, relief=tk.RAISED)

btn\_pesquisar.pack(side=tk.LEFT, padx=10, pady=10)

btn\_pesquisar.bind("<Enter>", lambda event: btn\_pesquisar.config(bg="#40E0D0"))

btn\_pesquisar.bind("<Leave>", lambda event: btn\_pesquisar.config(bg="SystemButtonFace"))

**23**

btn\_porcentagem = tk.Button(top\_menu, text="Analisar", command=open\_janela4, font=("Cascadia Code", 12, "bold"), borderwidth=2, relief=tk.RAISED)

btn\_porcentagem.pack(side=tk.LEFT, padx=10, pady=10)

btn\_porcentagem.bind("<Enter>", lambda event: btn\_porcentagem.config(bg="#40E0D0"))

btn\_porcentagem.bind("<Leave>", lambda event: btn\_porcentagem.config(bg="SystemButtonFace"))

btn\_grafico = tk.Button(top\_menu, text="Porcentagem", command=open\_janela2, font=("Cascadia Code", 12, "bold"), borderwidth=2, relief=tk.RAISED)

btn\_grafico.pack(side=tk.LEFT, padx=10, pady=10)

btn\_grafico.bind("<Enter>", lambda event: btn\_grafico.config(bg="#40E0D0"))

btn\_grafico.bind("<Leave>", lambda event: btn\_grafico.config(bg="SystemButtonFace"))

btn\_analisar = tk.Button(top\_menu, text="Gráfico", command=open\_janela3, font=("Cascadia Code", 12, "bold"), borderwidth=2, relief=tk.RAISED)

btn\_analisar.pack(side=tk.LEFT, padx=10, pady=10)

btn\_analisar.bind("<Enter>", lambda event: btn\_analisar.config(bg="#40E0D0"))

btn\_analisar.bind("<Leave>", lambda event: btn\_analisar.config(bg="SystemButtonFace"))

def open\_janela():

subprocess.Popen(["python", "frontend/janelas/Janela.py"])

def open\_janela2():

subprocess.Popen(["python", "frontend/janelas/Janela2.py"])

def open\_janela3():

subprocess.Popen(["python", "frontend/janelas/Janela3.py"])

def open\_janela4(): **24**

subprocess.Popen(["python", "frontend/janelas/Janela4.py"])

def main():

root = tk.Tk()

root.iconbitmap('image/m.ico')

app = MainFrame(root)

root.mainloop()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Janela.py**

import tkinter as tk

import subprocess

import io

from contextlib import redirect\_stdout

import pandas as pd

from tkinter import filedialog

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.neural\_network import MLPClassifier

from sklearn.metrics import accuracy\_score

import datetime

import os

from tkinter import PhotoImage

from tkinter import ttk

from PIL import Image, ImageTk

escolha\_usuario = ""

def load\_image(path, size):

return PhotoImage(file=path).subsample(size)

def pesquisar\_noticias(): **25**

tema = var.get()

if tema == "Enchentes":

query = "enchentes"

elif tema == "Chuvas":

query = "chuvas"

elif tema == "Queimadas":

query = "queimadas"

else:

return

subprocess.run(["python", "backend/buscaAPI/busca.py", query])

def abrir\_nova\_janela():

janela1.destroy()

subprocess.run(["python", "frontend/janelas/Janela2.py"])

janela1 = tk.Tk()

janela1.title("Consulta de Noticias")

janela1.iconbitmap('image/m.ico')

largura\_janela = 800

altura\_janela = 580

largura\_tela = janela1.winfo\_screenwidth()

altura\_tela = janela1.winfo\_screenheight()

pos\_x = (largura\_tela - largura\_janela) // 2

pos\_y = (altura\_tela - altura\_janela) // 4

janela1.geometry(f"{largura\_janela}x{altura\_janela}+{pos\_x}+{pos\_y}")

janela1.resizable(False, False)

frame = tk.Frame(janela1, width=largura\_janela, height=altura\_janela, bg="#B5B5B5")

frame.pack(fill=tk.BOTH, expand=True)

**26**

label\_estatus = tk.Label(frame, text="", font=("Arial", 12), bg="#B5B5B5")

label\_estatus.pack()

label\_titulo = tk.Label(frame, text="Consulta de Notícias", font=("Arial", 25), bg="#B5B5B5")

label\_titulo.pack(pady=20)

label\_tema = tk.Label(frame, text="Escolha um dos temas:", font=("Arial", 18), bg="#B5B5B5")

label\_tema.pack()

var = tk.StringVar()

var.set("")

opcoes\_tema = ["Chuvas", "Enchentes", "Queimadas"]

imagens\_tema = {

"Chuvas": load\_image("image/chuvas.png", 5),

"Enchentes": load\_image("image/enchentes.png", 5),

"Queimadas": load\_image("image/queimadas.png", 5)

}

def show\_image(option):

return imagens\_tema[option]

for tema in opcoes\_tema:

radio\_button = tk.Radiobutton(

frame,

text=tema,

variable=var,

value=tema,

image=show\_image(tema),

compound=tk.LEFT,

font=("Arial", 15),  **27**

bg="#B5B5B5"

)

radio\_button.pack()

def pesquisa():

subprocess.run(["python", "backend/buscaAPI/busca.py"])

img\_pesquisa = PhotoImage(file="image/pesquisa.png").subsample(10)

btn\_pesquisa = tk.Button(frame, image=img\_pesquisa, command=pesquisar\_noticias, borderwidth=3, relief=tk.RIDGE)

btn\_pesquisa.pack(pady=20)

btn\_pesquisa.bind("<Enter>", lambda event: btn\_pesquisa.config(bg="#40E0D0"))

btn\_pesquisa.bind("<Leave>", lambda event: btn\_pesquisa.config(bg="SystemButtonFace"))

def redirect\_output():

output = io.StringIO()

with redirect\_stdout(output):

subprocess.run(["python", "backend/buscaAPI/busca.py"])

output\_str = output.getvalue()

print(output\_str)

janela1.mainloop()

**Janela2.py**

import tkinter as tk

from tkinter import filedialog

import csv

import subprocess

from tkinter import PhotoImage

escolha\_sentimento = ""

escolha\_acuracia = "" **28**

def selecionar\_sentimento():

global escolha\_sentimento

arquivo\_sentimento = filedialog.askopenfilename(initialdir="./model/arquivosCSV/Sentimentos", title="Selecione o arquivo de sentimento",

filetypes=(("Arquivos CSV", "\*.csv"), ("Todos os arquivos", "\*.\*")))

if arquivo\_sentimento:

escolha\_sentimento = arquivo\_sentimento

label\_estatus\_sentimento.config(

text="Arquivo de sentimento selecionado", fg="green")

calcular\_porcentagens()

else:

label\_estatus\_sentimento.config(

text="Erro ao selecionar o arquivo de sentimento", fg="red")

def selecionar\_acuracia():

global escolha\_acuracia

arquivo\_acuracia = filedialog.askopenfilename(initialdir="./model/arquivosCSV/Acuracia", title="Selecione o arquivo de acurácia",

filetypes=(("Arquivos CSV", "\*.csv"), ("Todos os arquivos", "\*.\*")))

if arquivo\_acuracia:

escolha\_acuracia = arquivo\_acuracia

label\_estatus\_acuracia.config(

text="Arquivo de acurácia selecionado", fg="green")

exibir\_acuracia()

else:

label\_estatus\_acuracia.config(

text="Erro ao selecionar o arquivo de acurácia", fg="red")

def exibir\_acuracia():

global escolha\_acuracia **29**

if escolha\_acuracia:

with open(escolha\_acuracia, newline='', encoding='latin-1') as csvfile:

reader = csv.reader(csvfile)

acuracia = None

for row in reader:

if row:

acuracia = float(row[1])

break

if acuracia is not None:

label\_acuracia.config(text=f"Acurácia do modelo selecionado: {acuracia:.8f}")

else:

label\_acuracia.config(

text="Erro: Arquivo de acurácia vazio ou formato incorreto")

else:

label\_acuracia.config(

text="Erro: Nenhum arquivo de acurácia selecionado")

def calcular\_porcentagens():

global escolha\_sentimento

if escolha\_sentimento:

with open(escolha\_sentimento, newline='', encoding='utf-8') as csvfile:

reader = csv.reader(csvfile)

next(reader)

total = 0

positivos = 0

for row in reader:

total += 1

if row[1] == 'positivo':

positivos += 1

negativos = total - positivos

perc\_positivos = (positivos / total) \* 100

perc\_negativos = (negativos / total) \* 100

**30**

label\_positivos.config(text=f"Quantidade de sentimentos positivos: {perc\_positivos:.2f}%")

label\_negativos.config(text=f"Quantidade de sentimentos negativos: {perc\_negativos:.2f}%")

else:

label\_positivos.config(

text="Erro: Nenhum arquivo de sentimento selecionado")

def abrir\_nova\_janela():

janela2.destroy()

subprocess.run(["python", "frontend/janelas/Janela3.py"])

def abrir\_nova\_janela2():

janela2.destroy()

subprocess.run(["python", "frontend/janelas/Janela.py"])

janela2 = tk.Tk()

janela2.title("Porcentagem de Sentimentos")

janela2.iconbitmap('image/m.ico')

largura\_janela = 700

altura\_janela = 500

largura\_tela = janela2.winfo\_screenwidth()

altura\_tela = janela2.winfo\_screenheight()

pos\_x = (largura\_tela - largura\_janela) // 2

pos\_y = (altura\_tela - altura\_janela) // 2

janela2.geometry(f"{largura\_janela}x{altura\_janela}+{pos\_x}+{pos\_y}")

janela = tk.Frame(janela2, width=700, height=700, bg="#B5B5B5")

janela.pack(fill=tk.BOTH, expand=True)

label\_titulo = tk.Label(janela, text="Mostrando acurácia e a porcetagem de classificação por arquivo", font=(

"Arial", 18), bg="#B5B5B5") **31**

label\_titulo.pack()

label\_estatus\_sentimento = tk.Label(

janela, text="", font=("Arial", 12), bg="#B5B5B5")

label\_estatus\_sentimento.pack()

label\_estatus\_acuracia = tk.Label(

janela, text="", font=("Arial", 12), bg="#B5B5B5")

label\_estatus\_acuracia.pack()

label\_sentimento = tk.Label(janela, text="Selecionar Arquivo de Sentimento", font=(

"Arial", 16), bg="#B5B5B5", borderwidth=5)

label\_sentimento.pack()

img\_senti = PhotoImage(file="image/sentimentos.png").subsample(3)

btn\_senti = tk.Button(janela, image=img\_senti, command=selecionar\_sentimento, borderwidth=3, relief=tk.RIDGE)

btn\_senti.pack(pady=20)

btn\_senti.bind("<Enter>", lambda event: btn\_senti.config(bg="#40E0D0"))

btn\_senti.bind("<Leave>", lambda event: btn\_senti.config(bg="SystemButtonFace"))

label\_positivos = tk.Label(janela, text="", font=("Arial", 18), bg="#B5B5B5")

label\_positivos.pack()

label\_negativos = tk.Label(janela, text="", font=("Arial", 18), bg="#B5B5B5")

label\_negativos.pack()

label\_acu = tk.Label(janela, text="Selecionar Arquivo de Acurácia", font=(

"Arial", 16), bg="#B5B5B5", borderwidth=5)

label\_acu.pack()

img\_acu = PhotoImage(file="image/acuracia.png").subsample(8)

**32**

btn\_acu = tk.Button(janela, image=img\_acu, command=selecionar\_acuracia, borderwidth=3, relief=tk.RIDGE)

btn\_acu.pack(pady=20)

btn\_acu.bind("<Enter>", lambda event: btn\_acu.config(bg="#40E0D0"))

btn\_acu.bind("<Leave>", lambda event: btn\_acu.config(bg="SystemButtonFace"))

label\_acuracia = tk.Label(janela, text="", font=("Arial", 18), bg="#B5B5B5")

label\_acuracia.pack()

label\_distancia = tk.Label(janela, text="", font=("Arial", 18), bg="#B5B5B5")

label\_distancia.pack()

label\_distancia1 = tk.Label(janela, text="", font=("Arial", 18), bg="#B5B5B5")

label\_distancia1.pack()

janela2.mainloop()

**Janela3.py**

import os

import matplotlib.pyplot as plt

import tkinter as tk

from tkinter import filedialog, messagebox

import numpy as np

from tkinter import PhotoImage

import subprocess

def abrir\_arquivo1():

arquivo = filedialog.askopenfilename(title="Selecione o Arquivo 1", filetypes=[("Arquivos CSV", "\*.csv")])

if arquivo:

return arquivo

return None

def abrir\_arquivo2(): **33**

arquivo = filedialog.askopenfilename(title="Selecione o Arquivo 2", filetypes=[("Arquivos CSV", "\*.csv")])

if arquivo:

return arquivo

return None

def comparar\_dois\_arquivos(arquivo1, arquivo2):

if not arquivo1 or not arquivo2:

messagebox.showerror("Erro", "Selecione ambos os arquivos para comparar.")

return

dados = {}

dados[arquivo1] = ler\_arquivo\_csv(arquivo1)

dados[arquivo2] = ler\_arquivo\_csv(arquivo2)

plotar\_grafico(dados, "Comparação de Dois Arquivos")

def comparar\_enchentes():

dados\_enchentes = ler\_todos\_arquivos("model/arquivosCSV/Sentimentos/sentimentosEnchentes/")

plotar\_grafico(dados\_enchentes, "Enchentes")

def comparar\_chuvas():

dados\_chuvas = ler\_todos\_arquivos("model/arquivosCSV/Sentimentos/sentimentosChuvas/")

plotar\_grafico(dados\_chuvas, "Chuvas")

def comparar\_queimadas():

dados\_queimadas = ler\_todos\_arquivos("model/arquivosCSV/Sentimentos/sentimentosQueimadas/")

plotar\_grafico(dados\_queimadas, "Queimadas")

def ler\_todos\_arquivos(pasta):

dados = {} **34**

for arquivo in os.listdir(pasta):

if arquivo.endswith(".csv"):

dados[arquivo] = ler\_arquivo\_csv(os.path.join(pasta, arquivo))

return dados

def ler\_arquivo\_csv(arquivo):

with open(arquivo, 'r', encoding='utf-8') as file:

lines = file.readlines()

positivos = 0

negativos = 0

for line in lines:

if 'positivo' in line.lower():

positivos += 1

elif 'negativo' in line.lower():

negativos += 1

return {"positivos": positivos, "negativos": negativos}

def plotar\_grafico(dados, categoria):

arquivos = list(dados.keys())

positivos = [dados[arquivo]["positivos"] for arquivo in arquivos]

negativos = [dados[arquivo]["negativos"] for arquivo in arquivos]

largura\_barra = 0.20

posicoes = np.arange(len(arquivos))

fig, ax = plt.subplots()

bar\_positivos = ax.bar(posicoes - largura\_barra/2, positivos, largura\_barra, label='Positivos')

bar\_negativos = ax.bar(posicoes + largura\_barra/2, negativos, largura\_barra, label='Negativos')

ax.set\_ylabel('Quantidade') **35**

ax.set\_title(f'Comparação de {categoria}')

ax.set\_xticks(posicoes)

ax.set\_xticklabels(arquivos, rotation=45, ha='right')

ax.legend()

adicionar\_valores\_barras(bar\_positivos)

adicionar\_valores\_barras(bar\_negativos)

plt.show()

def adicionar\_valores\_barras(bars):

for bar in bars:

height = bar.get\_height()

plt.text(bar.get\_x() + bar.get\_width()/2, height, f'{height}', ha='center', va='bottom')

def abrir\_nova\_janela():

janela3.destroy()

subprocess.run(["python", "frontend/janelas/Janela2.py"])

if \_\_name\_\_ == "\_\_main\_\_":

janela3 = tk.Tk()

janela3.title("Gráficos")

janela3.geometry("500x400")

janela3.resizable(False, False)

janela3.iconbitmap('image/m.ico')

janela3.config(bg="#B5B5B5")

largura\_tela = janela3.winfo\_screenwidth()

altura\_tela = janela3.winfo\_screenheight()

pos\_x = (largura\_tela - 500) // 2

pos\_y = (altura\_tela - 400) // 2

janela3.geometry(f"500x400+{pos\_x}+{pos\_y}")

**36**

label = tk.Label(janela3, text="Escolha dois arquivos abaixo \n para serem comparados em um gráfico:", font=("Arial", 14), bg="#B5B5B5")

label.pack(pady=20)

btn\_arquivo1 = tk.Button(janela3, text="Abrir Arquivo 1", command=lambda: comparar\_dois\_arquivos(abrir\_arquivo1(), abrir\_arquivo2()), bg="#40E0D0")

btn\_arquivo1.pack(pady=5)

btn\_arquivo2 = tk.Button(janela3, text="Abrir Arquivo 2", command=abrir\_arquivo2, bg="#40E0D0")

btn\_arquivo2.pack(pady=5)

btn\_comparar\_dois\_arquivos = tk.Button(janela3, text="Comparar Dois Arquivos", command=lambda: comparar\_dois\_arquivos(abrir\_arquivo1(), abrir\_arquivo2()), bg="#40E0D0")

btn\_comparar\_dois\_arquivos.pack(pady=5)

label2 = tk.Label(janela3, text="Escolha um tema abaixo para gerar um gráfico \n comparando os arquivos dos últimos 5 dias ou mais:", font=("Arial", 14), bg="#B5B5B5")

label2.pack(pady=20)

btn\_enchentes = tk.Button(janela3, text="Comparar Enchentes", command=comparar\_enchentes, bg="#40E0D0")

btn\_enchentes.pack(pady=5)

btn\_chuvas = tk.Button(janela3, text="Comparar Chuvas", command=comparar\_chuvas, bg="#40E0D0")

btn\_chuvas.pack(pady=5)

btn\_queimadas = tk.Button(janela3, text="Comparar Queimadas", command=comparar\_queimadas, bg="#40E0D0")

btn\_queimadas.pack(pady=5)

**37**

janela3.mainloop()

**Janela4.py**

import tkinter as tk

import subprocess

import io

from contextlib import redirect\_stdout

import pandas as pd

from tkinter import filedialog

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.neural\_network import MLPClassifier

from sklearn.metrics import accuracy\_score

import datetime

import os

from tkinter import PhotoImage

from tkinter import ttk

from PIL import Image, ImageTk

janela4 = tk.Tk()

janela4.title("Consulta de Noticias")

janela4.iconbitmap('image/m.ico')

largura\_janela = 630

altura\_janela = 300

largura\_tela = janela4.winfo\_screenwidth()

altura\_tela = janela4.winfo\_screenheight()

pos\_x = (largura\_tela - largura\_janela) // 2

pos\_y = (altura\_tela - altura\_janela) // 2

janela4.geometry(f"{largura\_janela}x{altura\_janela}+{pos\_x}+{pos\_y}")

janela4.resizable(False, False)

**38**

frame = tk.Frame(janela4, width=largura\_janela, height=altura\_janela, bg="#B5B5B5")

frame.pack(fill=tk.BOTH, expand=True)

label\_estatus = tk.Label(frame, text="", font=("Arial", 12), bg="#B5B5B5")

label\_estatus.pack()

def selecionar\_arquivo():

global escolha\_usuario

arquivo = filedialog.askopenfilename(initialdir="./model/arquivosCSV/Busca", title="Selecione um arquivo",

filetypes=(("Arquivos CSV", "\*.csv"), ("Todos os arquivos", "\*.\*")))

if arquivo:

escolha\_usuario = arquivo

label\_estatus.config(text="Arquivo selecionado e armazenado", fg="green")

else:

label\_estatus.config(text="Erro ao selecionar o arquivo", fg="red")

def analisar\_noticias():

global escolha\_usuario

if escolha\_usuario:

tema = escolha\_usuario.split("/")[-1].split("\_")[1]

if tema == "chuvas":

arquivo\_treino = 'model/arquivosCSV/treinochuvas.csv'

elif tema == "enchentes":

arquivo\_treino = 'model/arquivosCSV/treinoenchentes.csv'

elif tema == "queimadas":

arquivo\_treino = 'model/arquivosCSV/treinoqueimadas.csv'

else:

label\_estatus.config(text="Erro: Arquivo de treino não encontrado", fg="red")

return **39**

paratreino\_df = pd.read\_csv(arquivo\_treino)

noticia\_df = pd.read\_csv(escolha\_usuario)

noticia\_df.drop(['Título', 'URL', 'Publicado em'], axis=1, inplace=True)

X\_traino, X\_teste, y\_traino, y\_teste = train\_test\_split(

paratreino\_df['Descrição'], paratreino\_df['Classificacao'], test\_size=0.2, random\_state=42)

vectorizer = CountVectorizer()

X\_trainn = vectorizer.fit\_transform(X\_traino)

X\_testt = vectorizer.transform(X\_teste)

model = MLPClassifier(hidden\_layer\_sizes=(100,), max\_iter=100, activation='relu', solver='adam', random\_state=42)

model.fit(X\_trainn, y\_traino)

acerto = model.predict(X\_testt)

accuracy = accuracy\_score(y\_teste, acerto)

sentimento\_dir = f'model/arquivosCSV/Sentimentos/sentimentos{tema.capitalize()}'

acuracia\_dir = f'model/arquivosCSV/Acuracia/acuracia{tema.capitalize()}'

os.makedirs(sentimento\_dir, exist\_ok=True)

os.makedirs(acuracia\_dir, exist\_ok=True)

num\_sentimento = 1

while os.path.exists(f'{sentimento\_dir}/sentimentos\_{tema}\_{datetime.datetime.now().strftime("%d-%m-%Y")}\_{num\_sentimento}.csv'):

num\_sentimento += 1

resultado\_file = f'{sentimento\_dir}/sentimentos\_{tema}\_{datetime.datetime.now().strftime("%d-%m-%Y")}\_{num\_sentimento}.csv' **40**

noticia\_counts = vectorizer.transform(noticia\_df['Descrição'])

noticia\_df['Classificacao'] = model.predict(noticia\_counts)

noticia\_df.to\_csv(resultado\_file, index=False)

num\_acuracia = 1

while os.path.exists(f'{acuracia\_dir}/acuracia\_{tema}\_{datetime.datetime.now().strftime("%d-%m-%Y")}\_{num\_acuracia}.csv'):

num\_acuracia += 1

acuracia\_file = f'{acuracia\_dir}/acuracia\_{tema}\_{datetime.datetime.now().strftime("%d-%m-%Y")}\_{num\_acuracia}.csv'

with open(acuracia\_file, 'w') as file:

file.write(f'Acurácia do modelo,{accuracy}\n')

label\_estatus.config(text="Notícias Analisadas", fg="green")

else:

label\_estatus.config(text="Erro: Nenhum arquivo selecionado", fg="red")

label\_selec = tk.Label(frame, text="Selecione o arquivo para ser feita análise de sentimento", font=("Arial", 18), bg="#B5B5B5")

label\_selec.pack()

img\_selec = PhotoImage(file="image/pasta.png").subsample(4)

btn\_selec = tk.Button(frame, image=img\_selec, command=selecionar\_arquivo, borderwidth=3, relief=tk.RIDGE)

btn\_selec.pack(pady=20)

btn\_selec.bind("<Enter>", lambda event: btn\_selec.config(bg="#40E0D0"))

btn\_selec.bind("<Leave>", lambda event: btn\_selec.config(bg="SystemButtonFace"))

label\_analise = tk.Label(frame, text="Análise de sentimento com arquivo selecionado", font=("Arial", 18), bg="#B5B5B5")  **41**

label\_analise.pack()

img\_analisar = PhotoImage(file="image/analise.png").subsample(10)

btn\_analisar = tk.Button(frame, image=img\_analisar, command=analisar\_noticias, borderwidth=3, relief=tk.RIDGE)

btn\_analisar.pack(pady=20)

btn\_analisar.bind("<Enter>", lambda event: btn\_analisar.config(bg="#40E0D0"))

btn\_analisar.bind("<Leave>", lambda event: btn\_analisar.config(bg="SystemButtonFace"))

janela4.mainloop()

**Bibliografia**

Neste documento foram usando documentação escritas encontradas em sites e em formato de vídeo, e também documentação fornecidas pelos professores ao longo do período, abaixo segue o link de todas as fontes usada para tal construção:

**Sites**

**Alan Turing**

* [**https://pt.wikipedia.org/wiki/Alan\_Turing**](https://pt.wikipedia.org/wiki/Alan_Turing)

**Teste de Turing**

* [**https://pt.wikipedia.org/wiki/Teste\_de\_Turing**](https://pt.wikipedia.org/wiki/Teste_de_Turing)

**Rede neural artificial**

* [**https://pt.wikipedia.org/wiki/Rede\_neural\_artificial**](https://pt.wikipedia.org/wiki/Rede_neural_artificial)
* [**https://aws.amazon.com/pt/what-is/neural-network/**](https://aws.amazon.com/pt/what-is/neural-network/)
* [**https://sites.icmc.usp.br/andre/research/neural/**](https://sites.icmc.usp.br/andre/research/neural/)

**Machine Learning**

* [**https://tecnoblog.net/responde/machine-learning-ia-o-que-e/**](https://tecnoblog.net/responde/machine-learning-ia-o-que-e/)

**42**

**API**

* [**https://aws.amazon.com/pt/what-is/api/**](https://aws.amazon.com/pt/what-is/api/)

**Protocolos de Redes**

* [**https://www.cloudflare.com/pt-br/learning/network-layer/what-is-a-protocol/**](https://www.cloudflare.com/pt-br/learning/network-layer/what-is-a-protocol/)

**Vídeos no youtuber**

**Machine Learning Vídeo**

* [**https://www.youtube.com/watch?v=MmSXHCxDwBs&list=PLyqOvdQmGdTR46HUxDA6Ymv4DGsIjvTQ-**](https://www.youtube.com/watch?v=MmSXHCxDwBs&list=PLyqOvdQmGdTR46HUxDA6Ymv4DGsIjvTQ-)

**API Vídeo**

* [**https://www.youtube.com/watch?v=eel1OVIdfUw**](https://www.youtube.com/watch?v=eel1OVIdfUw)

**Protocolos de Redes Vídeo**

* [**https://www.youtube.com/watch?v=h\_qeUwWGyTE**](https://www.youtube.com/watch?v=h_qeUwWGyTE)

**43**