CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
UNIDADE ARAXÁ

DANILO MAGONE DUARTE MARTINS

AQUISIÇÃO DE DADOS DE FERIADOS E EVENTOS PÚBLICOS RECORRENTES PARA APLICAÇÕES EM SISTEMAS DE PREVISÃO DE ACIDENTES

ARAXÁ/MG

2023

DANILO MAGONE DUARTE MARTINS

AQUISIÇÃO DE DADOS DE FERIADOS E EVENTOS PÚBLICOS RECORRENTES PARA APLICAÇÕES EM SISTEMAS DE PREVISÃO DE ACIDENTES

Projeto de Pesquisa apresentado ao Curso de Engenharia de Automação Industrial, do Centro Federal de Educação Tecnológica de Minas Gerais - CEFET/MG, como requisito parcial para obtenção do grau de Bacharel em Engenharia de Automação Industrial.

Orientador: Prof. Leandro Resende Mattioli
Coorientadoras: Profª. Maria Clara Rodrigues Bento Vaz Fernandes e Profª. Ana Isabel Pinheiro Nunes Pereira

ARAXÁ/MG

2023

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
COORDENAÇÃO DO CURSO DE ENGENHARIA DE AUTOMAÇÃO INDUSTRIAL / ARAXÁ

TRABALHO DE CONCLUSÃO DE CURSO – TCC – ATA DE DEFESA

Ata de Defesa do Trabalho de Conclusão de Curso de Engenharia de Automação Industrial do Aluno Danilo Magone Duarte Martins

Às **quatorze horas** do dia **seis de dezembro de dois mil e vinte e três**, reuniu-se, no Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG/ Campus Araxá, a Comissão Examinadora de Trabalho de Conclusão de Curso para julgar, em exame final, o trabalho intitulado **Aquisição de dados de feriados e eventos públicos recorrentes para aplicações em sistemas de previsão de acidentes**, como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Automação Industrial. Abrindo a sessão, o Presidente da Comissão, **Prof. Leandro Resende Mattioli**, após dar a conhecer aos presentes o teor das Normas Regulamentares do Trabalho Final, concedeu a palavra ao candidato, **Danilo Magone Duarte Martins**, para a exposição de seu trabalho. Após a apresentação, seguiu-se a arguição pelos examinadores, com a respectiva defesa do candidato. Ultimada a arguição, a Comissão se reuniu, sem a presença do candidato e do público, para julgamento e expedição do resultado final. Após a reunião da Comissão Examinadora, o candidato foi considerado: **APROVADO**, obtendo nota final de: **94**/100 **(noventa e quatro)**. O resultado final foi comunicado publicamente ao candidato pelo Presidente da Comissão. O aluno, abaixo assinado, declara que o trabalho ora identificado é da sua autoria material e intelectual, excetuando-se eventuais elementos, tais como passagens de texto, citações, figuras e datas, desde que devidamente identificada a fonte original. Declara ainda, neste âmbito, não violar direitos de terceiros. Nada mais havendo a tratar, o Presidente encerrou os trabalhos. O Prof. Leandro Resende Mattioli, responsável pela disciplina “Trabalho de Conclusão de Curso II”, lavrou a presente ATA, que, após lida e aprovada, será assinada por todos os membros participantes da Comissão Examinadora e pelo candidato.

Araxá, 06 de dezembro de 2023.

Prof. Leandro Resende Mattioli
CEFET-MG / Campus Araxá
Presidente e Orientador

Prof. Mateus Antunes Oliveira Leite
CEFET-MG / Campus Araxá
Membro Titular

Assinado por: **Maria Clara Rodrigues Bento Vaz Fernandes**
Data: 2023.12.08 18:16:10+00'00'

Profª. Clara Vaz
Instituto Politécnico de Bragança
Coorientadora

Profª. Joice Kamila Alves
CEFET-MG / Campus Araxá
Membro Titular

Assinado por: **Ana Isabel Pinheiro Nunes Pereira**
Num. de Identificação: 10145766
Data: 2023.12.08 16:31:16+00'00'

Profª. Ana Pereira
Instituto Politécnico de Bragança
Coorientadora

Danilo Magone Duarte Martins
Discente

Dedico este trabalho aos meus pais, meus irmãos
Arthur e Bárbara, minhas tias Amélia e Janaina e aos
amigos e colegas que acompanharam a minha trajetória
no curso

Agradecimentos

Aos meus pais que tanto se esforçaram para que fosse possível terminar o curso Ao tio Zezé, a tia Maria, Luciana e Luciene que me receberam de braços abertos em sua casa, e que me ajudaram tanto em todos estes anos e a quem serei eternamente grato A todos da minha família que apoiaram e ajudaram de diversas maneiras neste sonho Ao amigos que deram palavras acolhedoras nos momentos difíceis e com quem compartilhei experiências ao longo do curso Ao orientadores que me guiaram durante essa jornada, especialmente as professoras Ana e Clara, que me possibilitaram estender o período de intercâmbio em Portugal

Resumo

A imprevisibilidade dos acidentes de trabalho destaca a importância dos modelos de previsão como ferramentas essenciais para evitar esse tipo de incidente. Embora diversas áreas já tenham adotado esses modelos, o setor de varejo ainda não desfrutou plenamente desses avanços. Para aprimorar a precisão e a exatidão desses modelos, no entanto, é fundamental contar com parâmetros de entrada diversificados. Diante desse cenário, este trabalho trata da coleta e do pré-processamento de dados relacionados a eventos e feriados, considerando que tais ocasiões podem impactar o nível de estresse e a carga mental no setor varejista, tornando-se então candidatos valiosos a parâmetros de entrada em modelos de previsão de acidentes ocupacionais nesse segmento. Para alcançar esse propósito, foram desenvolvidos scripts que usam técnicas de *Web Scraping* para buscar e construir listas de eventos e feriados de 5 cidades portuguesas. Os dados obtidos passam por um processo de agrupamento hierárquico para que eventos repetidos sejam filtrados das listas. No total, foram coletados 260 dados de feriados e 4963 dados de eventos em 5 cidades portuguesas.

Palavras-chave: Web Scraping, Serviços Web, Coleta de dados

Abstract

The unpredictability of workplace accidents underscores the importance of predictive models as essential tools to prevent such incidents. While various sectors have already adopted these models, the retail industry has not fully embraced these advancements. To enhance the precision and accuracy of these models, however, it is crucial to have robust and diverse input parameters. In this context, this work focuses on the collection and pre-processing of data related to events and holidays, considering that such occasions can impact stress levels and mental workload in the retail sector, thus becoming valuable candidates for input parameters in occupational accident prediction models in this segment. To achieve this purpose, scripts were developed using Web Scraping techniques to gather and construct lists of events and holidays from 5 Portuguese cities. The obtained data undergoes a hierarchical clustering process to filter out repeated events from the lists. In total, 260 holiday data and 4963 event data were collected.

Keywords: Web Scraping, Web Services, Data Extraction

Lista de Figuras

Lista de Quadros

Sumário

Capítulo 1

Introdução

Pesquisas abrangentes sobre acidentes de trabalho estão sendo conduzidas globalmente em diversas áreas visando compreender as causas e consequências desses eventos, além de contribuir para a elaboração de métodos preventivos eficazes (SANTURTÚN; SHAMAN, 2023; MANZOOR; OTHMAN; WAHEED, 2022; DYREBORG et al., 2022). Apesar do amplo conhecimento existente sobre muitos dos incidentes e riscos associados, assim como a compreensão da frequência e gravidade das lesões, a previsão precisa dos momentos, locais e causas exatas dos acidentes ainda representa um desafio significativo (JØRGENSEN, 2016).

Uma explicação para esse cenário reside no considerável número de combinações de causas e na inerente dificuldade em identificar perigos ocultos que frequentemente são reconhecidos tardiamente (JØRGENSEN, 2016). A falta de uma consciência sólida dos riscos presentes no ambiente de trabalho contribui para a negligência de precauções essenciais, tornando as pessoas mais suscetíveis a acidentes (JØRGENSEN, 2016).

Considerando essa realidade, modelos de previsão têm desempenhado um papel crucial no aprimoramento da segurança no ambiente de trabalho e na prevenção de incidentes (SARKAR; PATEL et al., 2016). Esses modelos foram implementados em diversas áreas, incluindo construção, mineração e aviação (SARKAR; MAITI, 2020). Outros segmentos, especialmente o varejista, ainda não foram amplamente investigados e apresentam oportunidades para estudos (SARKAR; MAITI, 2020).

Para que tais modelos sejam mais precisos e eficientes, são necessários bons e diversificados parâmetros de entrada. Desta forma, fatores como clima, treinamento e cultura corporativa foram utilizados como variável preditora em estudos anteriores(ATTWOOD; KHAN; VEITCH, 2006; KSENHUCK et al., 2019; LUZ POLA, 2018).

Além desses dados, estudos confirmaram que variáveis relacionadas a fatores de estresse têm uma relação importante com acidentes (BARKHORDARI; MALMIR; MALAKOUTIKHAH, 2019; KIM et al., 2017). Alguns desses fatores incluem conflito trabalho-família, desequilíbrio entre esforço e recompensa e *locus* de controle externo (BARKHORDARI; MALMIR; MALAKOUTIKHAH, 2019). Outros estudos sugerem a ligação

entre a demanda de trabalho e a vulnerabilidade a ocorrências, reforçando o efeito do estresse nos acidentes de trabalho (DAY; BRASHER; BRIDGER, 2012).

Assim, encontrar parâmetros de entrada relacionados a aspectos psicológicos é essencial para modelos que visem prever acidentes. Nesse sentido, este trabalho busca contribuir para a obtenção destes parâmetros.

No setor varejista, o estresse e a carga mental de trabalho durante eventos e feriados podem ser relevantes para avaliar fatores relacionados à segurança no trabalho, uma vez que, nestes períodos, há um aumento nas vendas deste setor. Dessa forma, dados sobre essas ocasiões podem ser bons candidatos a parâmetros de entrada em modelos de previsão de acidentes. Ao utilizar tais informações, os modelos passam a considerar fatores emocionais e mentais do trabalho, o que pode contribuir para obter melhores resultados. Além disso, as percepções advindas dessa pesquisa podem auxiliar na prevenção e avaliação de riscos no setor, incluindo medidas de segurança e treinamentos específicos. A literatura mostra que dados dessa natureza foram explorados com outra finalidade neste setor (JIANG; RUAN; SUN, 2021; WANG, F. et al., 2022).

Nesse contexto, o objetivo deste trabalho é criar um conjunto de dados sobre feriados e eventos públicos a partir de fontes online. Para alcançar esse objetivo, é necessário compreender os métodos de extração de dados da Web e implementar algoritmos para a sua coleta e pré-processamento. Dado que não foram encontradas fontes que fornecem os dados de maneira direta, como uma API, foi necessário empregar técnicas de Web Scraping e serviços web.

Com isso, busca-se ampliar as opções de parâmetros a serem utilizados em modelos de previsão de acidentes ocupacionais, aumentando assim as suas áreas de aplicação. Os dados de feriados provenientes desta pesquisa foram utilizados para avaliar quais fatores estão relacionados com os acidentes de trabalho da rede de supermercados Continente, em Portugal (SENA et al., 2022).

Capítulo 2

Revisão de bibliografia

No âmbito do desenvolvimento de códigos para a automatização da coleta de dados da Web, diversos processos devem ser considerados para que haja êxito em sua execução. Tais processos abrangem desde a comunicação eficiente com a fonte de dados, até o entendimento das fontes, ou seja, suas funcionalidades, estrutura e a forma de apresentação dos dados, e por último os métodos e abordagens de extração destes dados. Uma vez que esses dados tenham sido coletados adequadamente, a rotina de pré-processamento é aplicada para filtrar, transformar e organizar os dados brutos em informações significativas e úteis para as finalidades específicas da aplicação. Nesse contexto, este capítulo apresentará algumas tecnologias, conceitos e ferramentas associadas a cada um destes processos.

2.1 Hypertext Transfer Protocol (HTTP)

O Hypertext Transfer Protocol (HTTP) é um protocolo de comunicação largamente utilizado na internet. Trata-se de um protocolo de rede reconhecido por sua simplicidade, utilizando texto como formato de mensagem, o que facilita a compreensão por parte dos usuários finais, ao contrário das mensagens binárias que não possuem uma estrutura textual definida (BROUCKE; BAESENS, 2018). O funcionamento do HTTP baseia-se em um esquema de comunicação simples de requisição e resposta. Durante esse processo, um cliente estabelece uma conexão com um servidor Web, enviando uma requisição e aguardando uma resposta em troca (FIELDING; TAYLOR, 2002).

Uma mensagem HTTP consiste nos seguintes elementos: uma linha inicial, cabeçalhos de solicitação, uma linha vazia e um corpo de mensagem opcional, que também pode ocupar várias linhas (FIELDING; RESCHKE, 2014a). As mensagens de solicitação têm o propósito de requisitar que os servidores executem uma ação em um recurso específico (FIELDING; TAYLOR, 2002). Nestas mensagens, a linha inicial é chamada linha de requisição. Esta linha inclui um método que especifica a ação que o servidor deve executar (FIELDING; RESCHKE, 2014a). O método é acompanhado de uma URL (Uniform Resource Locator, ou em português,

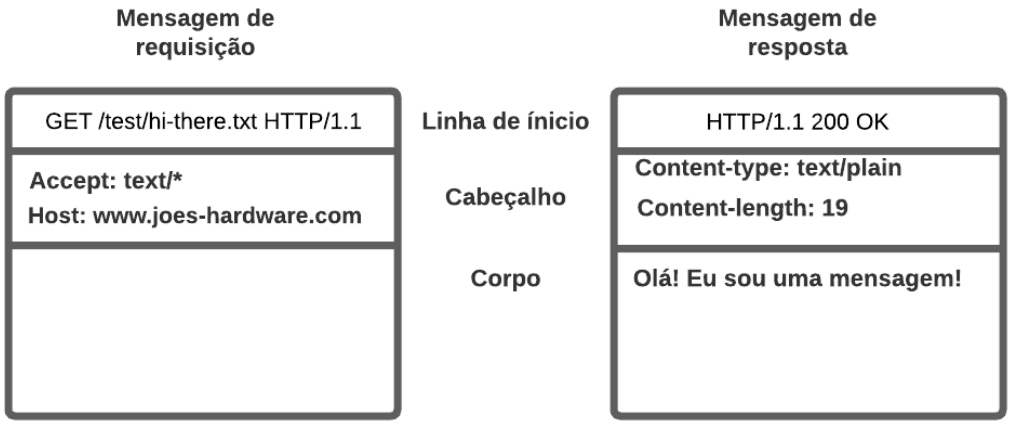
Localizador Uniforme de Recursos) que aponta para o recurso alvo dessa ação (GOURLEY et al., 2002; DIZDAREVIĆ et al., 2019). As especificações do protocolo HTTP estabelecem uma lista de métodos de requisição comumente utilizados. O Quadro 2.1 apresenta os métodos HTTP mais comuns. A Figura 2.1 mostra mensagens HTTP hipotéticas.

Quadro 2.1: Métodos HTTP mais comuns

Método	**Descrição**	**Corpo da mensagem?**
GET	Obtém um documento do servidor	Não
HEAD	Obtém apenas o cabeçalho de um documento do servidor	Não
POST	Envia dados para o servidor web para processamento	Sim
PUT	Armazena o corpo da requisição no servidor	Sim
DELETE	Remove um documento do servidor	Não
TRACE	Rastreia a mensagem por meio de servidores proxy até o servidor	Não
OPTIONS	Determina quais métodos podem operar em um servidor	Não

Fonte: Traduzido e adaptado de (GOURLEY et al., 2002)

Figura 2.1: Exemplo de mensagens de requisição e resposta



Fonte: Adaptado e traduzido de (GOURLEY et al., 2002).

Já nas mensagens de resposta, a linha inicial contém um código de status, que informa ao cliente da requisição o que aconteceu com a sua solicitação (FIELDING; RESCHKE, 2014b). O código de status é composto por um identificador numérico e uma descrição legível por humanos, separados por um espaço em branco. O Quadro 2.2 apresenta alguns dos códigos de status mais comuns.

Quadro 2.2: Códigos de status HTTP mais comuns

Código de status	**Significado**	**Frase razão**
200	Todos os dados solicitados estão no corpo da resposta	Ok
401	É necessário inserir um usuário e senha	*Unauthorized*
404	O servidor não pôde encontrar um recurso na URL fornecida	*Not Found*

Fonte: Adaptado e traduzido de (FIELDING; RESCHKE, 2014b).

O corpo da mensagem carrega o conteúdo, que pode ser tanto textos em alguma codificação como dados binários como imagens, vídeos e faixas de áudio (BROUCKE; BAESENS, 2018). O corpo contém exclusivamente o conteúdo puro e bruto, enquanto as informações descritivas estão nos cabeçalhos (GOURLEY et al., 2002).

Os cabeçalhos descrevem informações sobre o navegador, o aplicativo e o sistema (ZEGERS, 2015). O cabeçalho *Content-Type* desempenha o papel de indicar a natureza do conteúdo transportado, ou seja, o tipo de mídia do corpo, que segue o protocolo MIME (Multipurpose Internet Mail Extension) (FIELDING; RESCHKE, 2014b). O protocolo MIME padroniza, além de outras coisas, a extensão que deve ser usada para descrever o tipo de informação que está sendo transmitida, seja um arquivo nos formatos HTML ou XML apresentados nas próximas seções, uma imagem JPEG ou qualquer outro formato relevante (FREED; BORENSTEIN, 1996). Essa informação é utilizada pelo cliente para interpretar e processar adequadamente o conteúdo recebido. A Autoridade para Atribuição de Números da Internet, com a sigla em inglês IANA, é responsável por manter o registro oficial dos tipos de mídia MIME e por atualizar a lista com todos os tipos MIME reconhecidos oficialmente (IANA, 2023).

Além dos cabeçalhos citados, há ainda outros que são igualmente importantes para a comunicação HTTP. O cabeçalho *Accept-Charset* define a codificação dos caracteres aceitáveis para a resposta (CHANDRA; VARANASI, 2015). Por sua vez, o cabeçalho *Authentication* contém as credenciais de autenticação do usuário (FIELDING; RESCHKE, 2014b), no modelo mais básico. Já o cabeçalho *Host*, que é mandatório, identifica o hospedeiro e a porta do recurso solicitado. (FIELDING; RESCHKE, 2014b; BROUCKE; BAESENS, 2018). Por fim, o cabeçalho *User-Agent* fornece informações sobre o usuário, como sua identidade e a versão do navegador que está sendo utilizado, o que pode ser utilizado para resolver problemas de interoperabilidade (FIELDING; TAYLOR, 2002; BROUCKE; BAESENS, 2018).

2.1.1 Bibliotecas para comunicação HTTP

Achei uma descrição muito extensa das bibliotecas. Ficou cansativo

Nas linguagens de programação modernas, existe uma vasta coleção de bibliotecas que

possibilitam o envio de solicitações HTTP (RICHARDSON; RUBY, 2007). Em Python, a linguagem de programação usada neste trabalho, destacam-se: *httplib2*, *urllib3* e *requests* (CHANDRA; VARANASI, 2015).

O *requests* é uma biblioteca HTTP elegante e simples para Python, desenvolvida com o objetivo de fornecer uma interface intuitiva e amigável (BROUCKE; BAESENS, 2018). Além disso, permite a reutilização de uma conexão várias vezes, o que reduz a sobrecarga e a necessidade de manter uma nova conexão (CHANDRA; VARANASI, 2015). Devido a estes fatores, esta biblioteca é a mais utilizada e encontrada na literatura em aplicações que utilizam o Python. Especificamente, na extração de dados, ela foi extensivamente empregada para enviar solicitações GET a sites e recuperar o código-fonte HTML, conforme apresentado em diversos estudos acadêmicos (CENIĆ; STOJKOVIĆ, 2023; PENG et al., 2018; HO, 2020).

Por outro lado, embora a biblioteca *urllib* forneça funcionalidades HTTP confiáveis, sua utilização frequentemente exige a escrita de código repetitivo e extenso, o que pode resultar em um código menos agradável e elegante (BROUCKE; BAESENS, 2018). Ela disponibiliza funções e classes que auxiliam em diversas operações relacionadas a URLs, incluindo autenticação básica e avançada, redirecionamentos, cookies, entre outras (CHANDRA; VARANASI, 2015). Em comparação com o *urllib*, o *urllib3* estende o ecossistema do Python em relação ao HTTP com recursos avançados, porém não prioriza tanto a elegância ou a concisão (BROUCKE; BAESENS, 2018).

A biblioteca *httplib2* é uma solução abrangente para clientes HTTP, oferecendo suporte a uma ampla gama de recursos incomuns em outras bibliotecas do mesmo tipo (CHANDRA; VARANASI, 2015). Ela engloba funcionalidades como armazenamento em cache, manutenção de conexão, compressão, redirecionamentos e autenticação básica e avançada (CHANDRA; VARANASI, 2015).

2.2 Extensible Markup Language

Em relação as estruturas e formas de apresentação dos dados, a Extensible Markup Language (XML) é uma das linguagens utilizadas para este propósito. O XML, conhecido como Linguagem de Marcação Extensível, representa uma versão simplificada da Linguagem de Marcação Generalizada Padrão (SGML). Essa simplificação foi necessária devido à complexidade da SGML (NURSEITOV et al., 2009). O XML apresenta a capacidade distintiva de separar o conteúdo da formatação, proporcionando uma descrição clara da estrutura do texto contido em um documento (ZISMAN, 2000). Em termos simples, o XML estabelece regras explícitas e precisas para identificar o início e o fim das estruturas específicas dentro do documento (ZISMAN, 2000).

O XML foi projetado com o intuito de ser facilmente utilizável na Internet, com suporte a diversas aplicações (WORLD WIDE WEB CONSORTIUM, 2008). Além disso, foram

considerados aspectos como a legibilidade dos documentos XML pelos seres humanos e a facilidade de criação (WORLD WIDE WEB CONSORTIUM, 2008; NURSEITOV et al., 2009). A concisão na marcação XML, a formalidade e a clareza do design são aspectos essenciais. O XML também visou possibilitar a criação de programas que processam documentos XML de forma simples, como por exemplo as bibliotecas *libxml* e *ElementTree*
(WORLD WIDE WEB CONSORTIUM, 2008).

Os elementos presentes em XML podem possuir atributos, que são informações adicionais com o intuito de especificar propriedades ou características aos elementos (NURSEITOV et al., 2009; NIERMAN; JAGADISH, 2002). Em uma estrutura de árvore, cada nó representa um elemento do documento, enquanto os atributos são associados ao nó correspondente ao elemento pai (NIERMAN; JAGADISH, 2002). A Figura 2.2 apresenta um exemplo de documento XML de um catálogo de endereços.

Figura 2.2: Exemplo de documento XML

```
<?xml version="1.0" ?>
<catálogo-de-endereços>
    <instituição>
        <nome>CEFET-MG Campus Araxá</nome>
        <endereço>
            <logradouro>Av. Ministro Olavo Drummond</logradouro>
            <número>25</número>
            <estado>MG</estado>
            </país>Brasil</país>
        </endereço>
    </instituição>
</catálogo-de-endereços>
```

Fonte: Autoria própria.

Ao contrário do HTML, descrito na próxima seção, o XML não é uma linguagem de marcação pré-definida e permite que o autor do documento projete sua própria marcação (ALMEIDA, 2002; NURSEITOV et al., 2009). Isso significa que o XML permite que o autor defina suas próprias *tags*, o que resulta na simplificação do processo de disseminação de informações (ALMEIDA, 2002). Essa característica semântica do XML proporciona maior personalização e adaptação às necessidades específicas do autor e da aplicação em que o documento será utilizado (ALMEIDA, 2002).

Um documento XML pode ser representado como uma estrutura de árvore, onde cada elemento é associado a um nó com o nome de uma *tag* (NIERMAN; JAGADISH, 2002). As conexões na árvore indicam a inclusão de elementos filhos sob elementos pais no arquivo XML (NIERMAN; JAGADISH, 2002).

O XML possui uma ampla gama de aplicações em diversas áreas que envolvem a troca de dados, a interoperabilidade de bancos de dados e a publicação de documentos (ZISMAN,

2000). Ele atua como uma linguagem comum que permite a troca de informações complexas entre diferentes programas, funcionando como uma ponte entre sistemas heterogêneos (ZISMAN, 2000). Além disso, o XML é especialmente adequado para lidar com documentos semiestruturados, como manuais, catálogos, relatórios, patentes e demonstrações financeiras (ZISMAN, 2000; ALMEIDA, 2002).

2.3 HyperText Markup Language

O HyperText Markup Language (HTML) é a linguagem utilizada para criar páginas da web (WHATWG, 2023). Essa linguagem de marcação padrão estrutura o conteúdo e a apresentação das páginas na internet (WHATWG, 2023; LIE; SAARELA, 1999).

Os documentos HTML consistem em elementos, delimitados por *tags* de início e fim, e texto (WHATWG, 2023). As especificações do HTML estabelecem os significados, ou seja, a semântica dos elementos, atributos e os valores permitidos nestes (WHATWG, 2023). Por exemplo, o elemento "ol" é utilizado para representar uma lista ordenada, enquanto o atributo "lang" indica o idioma do conteúdo (WHATWG, 2023). Essas definições permitem que os navegadores web e mecanismos de busca interpretem e utilizem os documentos e aplicativos concebidos em HTML de diversas formas, abrangendo inclusive contextos não previstos pelo autor (WHATWG, 2023).

Os elementos HTML podem ter atributos, que são informações adicionais escritas dentro das tags de início (WHATWG, 2023). Esses atributos determinam particularidades e diretrizes para o comportamento do elemento, podendo incluir valores como texto ou números que os navegadores e outras ferramentas interpretam para apresentar o conteúdo de maneira adequada. Os atributos são compostos por um nome e um valor, os quais são separados por um caractere "=" (WHATWG, 2023). Um atributo de grande importância é o atributo "class" . Sua função é permitir a criação de diferentes classes para um elemento, em que cada classe pode ter suas próprias propriedades específicas (WHATWG, 2023). O atributo "class" pode ser utilizado para criar extensões de elementos existentes, possibilitando a criação de elementos personalizados (WHATWG, 2023; SRIVASTAVA; HAROON; BAJAJ, 2013). A Figura 2.3 mostra um exemplo de código HTML simples, com diferentes elementos e um atributo.

Figura 2.3: Exemplo de código HTML

```
<!DOCTYPE HTML>
<html lang="en">
    <head>
        <title>CEFET-MG Campus Araxá</title>
    </head>
    <body>
        <h1>Bem vindo ao meu TCC</h1>
        <p>Este TCC fala de Web Scraping e Web Services</p>
    </body>
</html>
```

Fonte: Elaboração própria.

O HTML5, a quinta e mais recente versão do padrão HTML (TABARÉS, 2021), foi proposta originalmente pela Opera Software. Esta versão expande consideravelmente o escopo de tecnologias, buscando aprimorar a linguagem com recursos avançados para multimídia, mantendo a legibilidade e a compatibilidade com dispositivos e navegadores (SHARMA; BHARDWAJ; BHARDWAJ, 2012; SHAHZAD et al., 2016). As diretrizes estabelecidas para o desenvolvimento do HTML5 incorporaram elementos essenciais do HTML, CSS, DOM e JavaScript, com o intuito de reduzir a dependência de plugins externos, melhorar o tratamento de erros e adotar uma abordagem independente de dispositivos (SHARMA; BHARDWAJ; BHARDWAJ, 2012; THEISEN, 2019; ANTTONEN et al., 2011). Isso resultou na introdução de funcionalidades e elementos que possibilitaram o suporte nativo a vídeos, armazenamento local, comunicação em tempo real, entre outras (SHARMA; BHARDWAJ; BHARDWAJ, 2012; THEISEN, 2019; ANTTONEN et al., 2011).

No entanto, as contribuições mais notáveis do HTML5 incluem uma série de API's e interfaces que impulsionaram o uso do JavaScript no desenvolvimento web (ANTTONEN et al., 2011; THEISEN, 2019). Duas delas se destacam: a API de Geolocalização, que permite o acesso à localização geográfica, e a API de Canvas, que viabiliza a criação de gráficos 2D diretamente no navegador. Essas inovações evidenciam a relevância significativa do HTML5 na evolução da web moderna.

2.4 Serviços Web

Para além de sites, dados e informações podem ser extraídos de Serviços Web. Serviços Web são aplicativos projetados para comunicação entre aplicações de software (WORLD WIDE WEB CONSORTIUM, 2004). Essa comunicação é realizada por meio do protocolo HTTP e de especificações como XML, JSON (Javascript Object Notation), entre outras, para os dados (ZANETTI; CAMARGO, 2012; SRINIVASAN; TREADWELL, 2005; TIHOMIROVS; GRABIS, 2016). Estes aplicativos desempenham diversas funções, desde solicitações simples

de informações até a execução de processos complexos (WANG, H. et al., 2004).

Um serviço web é composto por um módulo de software fornecido por um provedor, juntamente com uma descrição correspondente, acessível por meio da internet (TIHOMIROVS; GRABIS, 2016). A descrição de serviço contém os detalhes da interface e suas implementações, incluindo tipos de dados, metadados, informações de categorização e o local onde está exposto (TIHOMIROVS; GRABIS, 2016). A descrição é essencial para que os clientes entendam como acessare utilizar o serviço (CRASSO et al., 2010). Os detalhes de implementação, porém, ficam em sigilo (SRINIVASAN; TREADWELL, 2005). Essa abordagem permite a interação entre um cliente e o serviço, independentemente das plataformas e linguagens de programação utilizadas. Essa flexibilidade garante a interoperabilidade e torna os serviços web especialmente adequados para ambientes distribuídos e heterogêneos (BOZKURT; HARMAN; HASSOUN et al., 2010; SRINIVASAN; TREADWELL, 2005; TIHOMIROVS; GRABIS, 2016).

Em relação ao serviço, existem duas abordagens principais para a interface de serviços web: o SOAP e o REST (BARBAGLIA; MURZILLI; CUDINI, 2017). O Simple Object Acess Protocol (SOAP) é um protocolo baseado em XML que facilita a troca de dados por meio do Protocolo de Transferência de Hipertexto (HTTP). Ele é essencialmente um paradigma de troca de mensagens unidirecional e sem estado, permitindo interações complexas entre aplicações, como solicitações e respostas múltiplas, além de outros padrões de interação (WANG, H. et al., 2004). Quanto ao formato das mensagens, o SOAP estabelece um formato padrão para a comunicação, especificando como as informações devem ser organizadas em um documento XML (LEMOS; DANIEL; BENATALLAH, 2015). Essencialmente, uma mensagem SOAP possui uma estrutura extremamente simples: um elemento XML com dois elementos filhos, um para o cabeçalho e outro para o corpo. Tanto o conteúdo do cabeçalho quanto os elementos do corpo são igualmente representados em XML (WANG, H. et al., 2004).

A arquitetura REST (Representational State Transfer) surgiu como uma alternativa mais leve e simplificada em relação ao SOAP nos serviços web (TIHOMIROVS; GRABIS, 2016). Ao utilizar o protocolo HTTP e formatos de dados como XML e JSON (JavaScript Object Notation), o REST aproveita os padrões já estabelecidos e amplamente conhecidos, evitando camadas adicionais de processamento de dados na comunicação (LEMOS; DANIEL; BENATALLAH, 2015; TIHOMIROVS; GRABIS, 2016). Os serviços web REST são construídos em torno de recursos acessíveis por URLs específicas, utilizando os métodos HTTP para realizar operações nos dados do recurso (TIHOMIROVS; GRABIS, 2016; LEMOS; DANIEL; BENATALLAH, 2015).

Em relação à descrição do serviço, as diversas abordagens utilizam métodos distintos. Nos serviços SOAP, destaca-se a ampla adoção do Web Service Description Language (WSDL) (WANG, H. et al., 2004). Esta linguagem, baseada em XML, serve para descrever as características, interfaces e outras propriedades de um serviço web (SRINIVASAN; TREADWELL, 2005; LEMOS; DANIEL; BENATALLAH, 2015). Adicionalmente, ela detalha as

operações presentes no serviço e as mensagens trocadas por cada funcionalidade (LEMOS; DANIEL; BENATALLAH, 2015; SRINIVASAN; TREADWELL, 2005), estabelecendo os métodos, nomes dos parâmetros, tipos de dados dos parâmetros e tipos de dados de retorno para o serviço web (RATHOD, 2017).

Em relação à estrutura do documento WSDL, esta engloba agrupamentos de interfaces, denominados *port type*, que abrangem grupos de operações (WORLD WIDE WEB CONSORTIUM, 2007; CRASSO et al., 2010). As operações, por sua vez, contém uma sequência mensagens de entrada, saída e, em certos casos, de falha (WORLD WIDE WEB CONSORTIUM, 2007; CRASSO et al., 2010). Nas operações, as mensagens são constituídas por uma ou mais partes, destinadas a transportar dados no formato XML, cujas definições são realizadas por meio do XML Schema Definition (XSD) (CRASSO et al., 2010). Cada elemento presente na descrição do serviço é identificado por um nome e, adicionalmente, pode ser acompanhado por um comentário textual (CRASSO et al., 2010). A Figura 2.4 mostra um exemplo de código WSDL de um serviço web que retorna informações sobre as taxas de câmbio de dois países.

Figura 2.4: Exemplo de código WSDL

```
<types>
 <xsd:element name="GetRate">
 <xsd:complexType>
 <xsd:sequence>
 <xsd:element name="srcCurrency" type="xsd:string"/>
 <xsd:element name="destCurrency" type="xsd:string"/>
 </xsd:sequence>
 </xsd:complexType>
 </xsd:element>
</types>
<message name="GetRateSoapIn">
 <part name="parameters" element="s0:GetRate" />
</message>
<portType name="CurrencywsSoap">
 <operation name="GetRate">
 <documentation>
 Este método retorna a taxa de câmbio entre dois países
 </documentation>
 <input message="s0:GetRateSoapIn">
 <documentation>Os códigos dos dois países</documentation>
 </input>
 <output message="s0:GetRateSoapOut" />
 <fault name="nmtoken" message="s0:GetRateFault"/>
 </operation>
</portType>
```

Fonte: Traduzido de (RODRIGUEZ; ZUNINO SUAREZ; CRASSO, 2013)

No que diz respeito aos serviços web *RESTful*,como são chamados serviços com arquitetura REST, a linguagem utilizada para a descrição é o WADL (LEMOS; DANIEL; BENATALLAH, 2015).(LEMOS; DANIEL; BENATALLAH, 2015). O WADL (Web Application Description Language) também é baseado em XML e descreve o serviço como um conjunto de recursos e suas inter-relações (LEMOS; DANIEL; BENATALLAH, 2015). Essa abordagem, estreitamente relacionada ao WSDL, gera um único arquivo XML que contém todas as informações sobre a interface do serviço (LANTHALER; GRANITZER; GÜTL, 2010). Porém, diferentemente do WSDL que é baseado em operações e funcionalidades, o WADL modela os recursos disponibilizados pelo serviço e as relações entre eles (LANTHALER; GRANITZER; GÜTL, 2010). Cada recurso do serviço é descrito em uma solicitação no WADL, incluindo o método HTTP utilizado e os parâmetros necessários, além de uma resposta que descreve a representação esperada da resposta do serviço e o código de status HTTP (LANTHALER; GRANITZER; GÜTL, 2010). Uma crítica importante ao WADL é sua complexidade, que contrasta com a simplicidade dos serviços com arquitetura REST (LANTHALER; GRANITZER; GÜTL, 2010).

Há uma crescente dominância do REST em relação ao SOAP, impulsionada por suas vantagens em termos de eficiência, simplicidade e facilidade de integração. Os serviços web RESTful são mais leves, fáceis de consumir e autoexplicativos, tornando o desenvolvimento e a manutenção mais simples (HALILI; RAMADANI et al., 2018; TIHOMIROVS; GRABIS, 2016; LEMOS; DANIEL; BENATALLAH, 2015). Além disso, apresentam desempenho superior, sendo rápidos e consumindo menos largura de banda (HALILI; RAMADANI et al., 2018). Com isso, pode-se afirmar com segurança que nos próximos anos, a arquitetura REST continuará a ser amplamente adotada e dominante (HALILI; RAMADANI et al., 2018).

2.5 Web Scraping

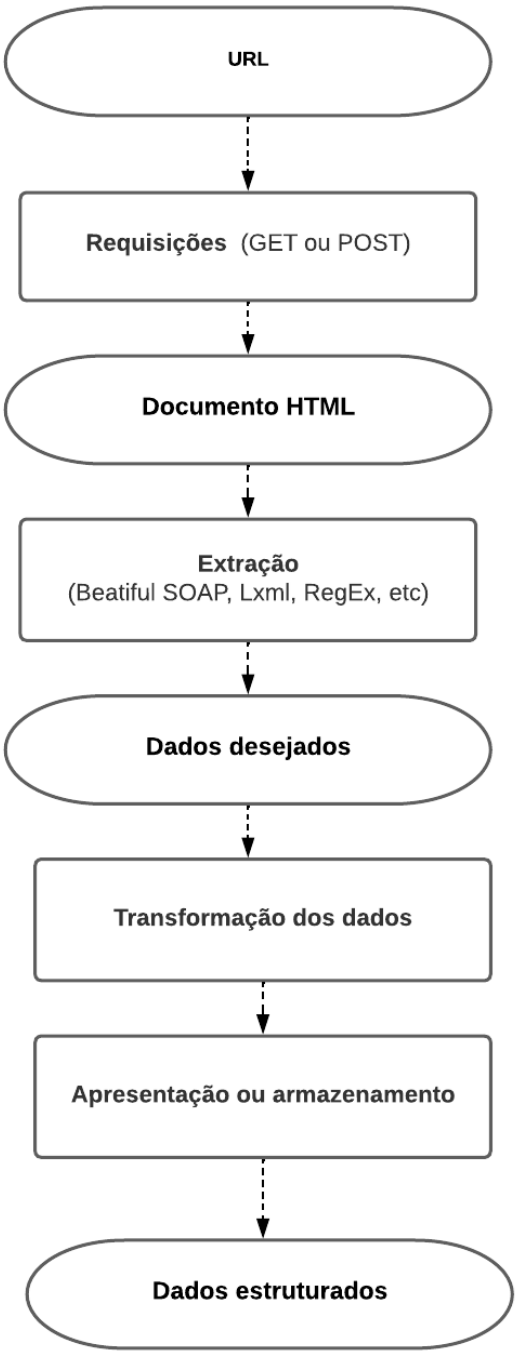
O Web Scraping, também conhecido como Web Harvesting, é uma técnica amplamente utilizada para extrair e armazenar dados coletados diretamente da internet, facilitando análises posteriores (ZHAO, 2017). O Web Scraping é definido como a prática de coletar dados por meio de métodos que não envolvam programas que interagem com uma API, simulando o comportamento humano (DIOUF et al., 2019; MITCHELL, 2018).

Um scraper inicia seu processo estabelecendo comunicação com o website alvo (GLEZ-PEÑA et al., 2013). Para isso, é feita uma requisição HTTP para obter recursos (GLEZ-PEÑA et al., 2013; ZHAO, 2017). Essa solicitação pode ocorrer de duas maneiras: por meio de uma URL que contém uma consulta GET ou por meio de uma consulta POST (GLEZ-PEÑA et al., 2013; ZHAO, 2017). Uma vez que a solicitação é enviada com sucesso e processada pelo site-alvo, o recurso solicitado é enviado de volta ao scraper (ZHAO, 2017).

Após o recebimento dos recursos, a etapa de extração continua com a análise dos dados.

Isso envolve a interpretação e o entendimento do conteúdo (ZHAO, 2017), a reformatação dos dados conforme necessário e a organização estruturada dos mesmos para análises futuras e armazenamento (GLEZ-PEÑA et al., 2013; ZHAO, 2017). O objetivo é extrair informações relevantes e significativas dos recursos adquiridos, garantindo que os dados estejam em um formato útil e coerente para análises e aplicações posteriores, normalmente em XML ou CSV (Comma Separated Values) (GLEZ-PEÑA et al., 2013). A Figura 2.5 resume o processo de Web Scraping descrito.

Figura 2.5: Fluxograma do processo de Web Scraping



Fonte: Adaptado e traduzido de (PERSSON, 2019)

De forma geral, é possível categorizar os métodos utilizados em Web Scraping em quatro abordagens de análise (DIOUF et al., 2019; CAMARGO-HENRÍQUEZ; NÚÑEZ-BERNAL, 2022), descritas nas próximas seções.

2.5.1 Mimetismo

No contexto de web scraping, o mimetismo refere-se a uma abordagem em que a estrutura da página web alvo é conhecida antecipadamente (DIOUF et al., 2019). Ao conhecer a estrutura da página, é possível acessar elementos HTML específicos para coletar os dados desejados, permitindo uma extração clara e direcionada (GUNAWAN et al., 2019; DIOUF et al., 2019).

O mimetismo é um método relativamente eficiente de web scraping, pois reduz a necessidade de exploração e busca de elementos na página web, economizando tempo e recursos de processamento (DIOUF et al., 2019).

No entanto, uma desvantagem do mimetismo é que ele depende da consistência na estrutura da página web de origem. Se o site modificar seu design, por exemplo, alterando a organização dos elementos HTML, renomeando classes ou IDs, ou realizando qualquer outra mudança significativa, o scraper precisará ser reprogramado para se adaptar às novas configurações (DIOUF et al., 2019; CAMARGO-HENRÍQUEZ; NÚÑEZ-BERNAL, 2022).

Além disso, o mimetismo pode enfrentar dificuldades ao lidar com sites heterogêneos, ou seja, páginas que possuem estruturas variadas entre si (DIOUF et al., 2019). Se um scraper for projetado com base em uma estrutura específica e for usado para coletar dados de diferentes sites com estruturas diferentes, ele pode não ser capaz de lidar adequadamente com essa variação. Isso pode resultar em erros de extração ou na incapacidade de coletar os dados corretamente (DIOUF et al., 2019).

2.5.2 Medição de pesos

Essa abordagem utiliza um algoritmo que analisa a estrutura de árvore DOM de uma página da web. O algoritmo realiza uma análise das palavras presentes em cada ramificação da árvore DOM, atribuindo um peso a cada uma delas. Através de uma inferência baseada nesses pesos, o algoritmo identifica o nó que possui o texto principal e extrai o conteúdo textual de todos os nós filhos desse nó (CAMARGO-HENRÍQUEZ; NÚÑEZ-BERNAL, 2022; DIOUF et al., 2019).

Uma das principais vantagens dessa abordagem é que ela não requer nenhum tipo de treinamento específico (DIOUF et al., 2019). O algoritmo é capaz de adaptar-se automaticamente às mudanças no design gráfico das páginas da web (DIOUF et al., 2019), o que o torna bastante flexível e robusto.

No entanto, é importante mencionar que os resultados obtidos por meio dessa abordagem costumam apresentar um nível de ruído relativamente alto (CAMARGO-HENRÍQUEZ; NÚÑEZ-BERNAL, 2022; DIOUF et al., 2019). Portanto, é necessário aplicar técnicas adicionais de limpeza e processamento dos dados extraídos para obter resultados mais precisos e confiáveis (CAMARGO-HENRÍQUEZ; NÚÑEZ-BERNAL,

2022).

2.5.3 Método diferencial

Essa abordagem parte da premissa de que as divergências entre duas páginas de um mesmo site estarão restritas ao conteúdo do corpo principal (DIOUF et al., 2019; CAMARGO-HENRÍQUEZ; NÚÑEZ-BERNAL, 2022). Assim, considera-se que os elementos de design, como barras de menu, colunas laterais e rodapés, serão totalmente idênticos nas páginas do site em que se deseja extrair os dados (DIOUF et al., 2019).

Para colocar essa abordagem em prática, utiliza-se um mecanismo que emprega um algoritmo de máscara (DIOUF et al., 2019). Esse algoritmo sobrepõe as duas páginas do site e identifica as regiões onde existem discrepâncias, removendo apenas essas diferenças (DIOUF et al., 2019). Dessa forma, é possível extrair os dados apenas das áreas em que ocorrem alterações no conteúdo.

2.5.4 Machine learning

Neste método, é realizado o treinamento de um algoritmo usando uma extensa seleção de páginas da web que foram analisadas previamente (DIOUF et al., 2019; AZIR; AHMAD, 2017). Essa abordagem se baseia em técnicas de aprendizado de máquina que utilizam indicadores geográficos dos blocos de texto encontrados nas páginas (DIOUF et al., 2019).

O objetivo é capacitar o algoritmo a deduzir, por si só, a localização mais comum do texto em uma página (DIOUF et al., 2019). Isso é alcançado ao observar padrões nos dados coletados durante o treinamento (DIOUF et al., 2019; AZIR; AHMAD, 2017). Tais padrões podem ser reconhecidos através de análise estatística, de modo a estabelecer a posição do bloco de texto principal em relação aos demais blocos (DIOUF et al., 2019). Quanto maior for a quantidade de páginas analisadas, mais precisa será a capacidade do algoritmo em identificar a localização típica do texto (DIOUF et al., 2019).

2.5.5 Bibliotecas e ferramentas para Web Scraping

Em relação às linguagens de programação para Web Scraping, o Python é a mais adequada e poderosa devido à sua vasta comunidade, às diversas bibliotecas disponíveis e maior velocidade se comparada com outras linguagens (THOMAS; MATHUR, 2019; R; S; M., 2023). Além das bibliotecas mencionadas anteriormente para comunicação HTTP, bibliotecas para interpretação de códigos HTML e bibliotecas que interajam com o JavaScript são essenciais para a prática do Web Scraping.

Uma dessas bibliotecas é a Beautiful Soup, que simplifica a extração de dados estruturados de páginas da web, especialmente para análise de HTML e XML (KHDER, 2021).

Comparada com a utilização de expressões regulares, a Beautiful Soup oferece uma abordagem mais simples, permitindo uma navegação, exame e atualização mais fácil da árvore de análise (KHDER, 2021). Dadas as características mencionadas, diversas pesquisas utilizaram essa biblioteca em operações de Web Scraping e extração de dados, sendo empregada para interpretar códigos-fonte em HTML e extrair tags específicas de uma variedade de sites e em diversas áreas.

Além do Beautiful Soup, o uso de recursos como as expressões regulares, também conhecidas como *regex*, é fundamental no desenvolvimento de algoritmos de web scraping. Esse método possibilita o processamento de texto de forma poderosa, flexível e eficiente (FRIEDL, 2006), permitindo a identificação de padrões específicos em vez de depender de correspondências exatas (CHAPMAN; STOLEE, 2016). As expressões regulares são compostas por caracteres especiais conhecidos como metacaracteres, os quais têm significados específicos, e caracteres normais de texto. Cada metacaractere possui uma função única e bem definida (FRIEDL, 2006). A combinação dos caracteres e dos metacaracteres forma a semântica da expressão regular, tornando possível a extração das informações desejadas (FRIEDL, 2006). No Python, a biblioteca "re" é empregada para tarefas relacionadas a essa técnica (THIVAHARAN; SRIVATSUN; SARATHAMBEKAI, 2020; KHDER, 2021).

Outra biblioteca importante para o Web Scraping é o *lxml*. Ela foi desenvolvida com o objetivo de analisar documentos em fragmentos (THIVAHARAN; SRIVATSUN; SARATHAMBEKAI, 2020; KHDER, 2021), permitindo a extração de informações de forma simplificada (THIVAHARAN; SRIVATSUN; SARATHAMBEKAI, 2020). Essa abordagem, conhecida como *chuncking*, busca identificar as conexões entre os componentes do conteúdo, facilitando a extração em casos de estruturas internas complexas (THIVAHARAN; SRIVATSUN; SARATHAMBEKAI, 2020). Documentos desse tipo geralmente contêm vários links recursivos, o que pode dificultar a extração com outras ferramentas.Lxml utiliza o XPath, uma árvore de extração de conteúdo (THIVAHARAN; SRIVATSUN; SARATHAMBEKAI, 2020; KHDER, 2021), que é especialmente útil em seu processo de análise (THIVAHARAN; SRIVATSUN; SARATHAMBEKAI, 2020). A Tabela 2.1 compara as três biblotecas citadas.

Tabela 2.1: Comparação de bibliotecas utilizadas para análise de HTML

Biblioteca	**Desempenho**	**Facilidade de uso**
re	Rápido	Difícil
Beatiful Soup	Lento	Fácil
Lxml	Rápido	Fácil

Fonte: Adaptado e traduzido de (LAWSON, 2015)

Para sites que possuem conteúdo oculto ou interativo, a biblioteca Selenium é uma

ferramenta útil para a extração de dados. Diferentemente das outras bibliotecas citadas, o Selenium é capaz de lidar com elementos da página que são gerados ou controlados por JavaScript (TEOTIA et al., 2023; HAN; ANDERSON, 2021).

Em vez de simplesmente analisar o código HTML de um site, o Selenium permite simular o comportamento humano de navegação em um navegador real (HAN; ANDERSON, 2021). Isso significa que podemos automatizar a interação com o site, como por exemplo clicar em botões que revelem informações, antes escondidas, a serem extraídas (TEOTIA et al., 2023; HAN; ANDERSON, 2021). Diversas pesquisas aproveitaram dessa biblioteca para automação de da extração de códigos-fonte de sites com elementos controlados por JavaScript. Esses códigos-fonte são posteriormente interpretados e os dados extraídos por meio das bibliotecas mencionadas anteriormente (KHATTER; SHARMA; KUSHWAHA et al., 2022; LI; ZHOU; CAI, 2021).

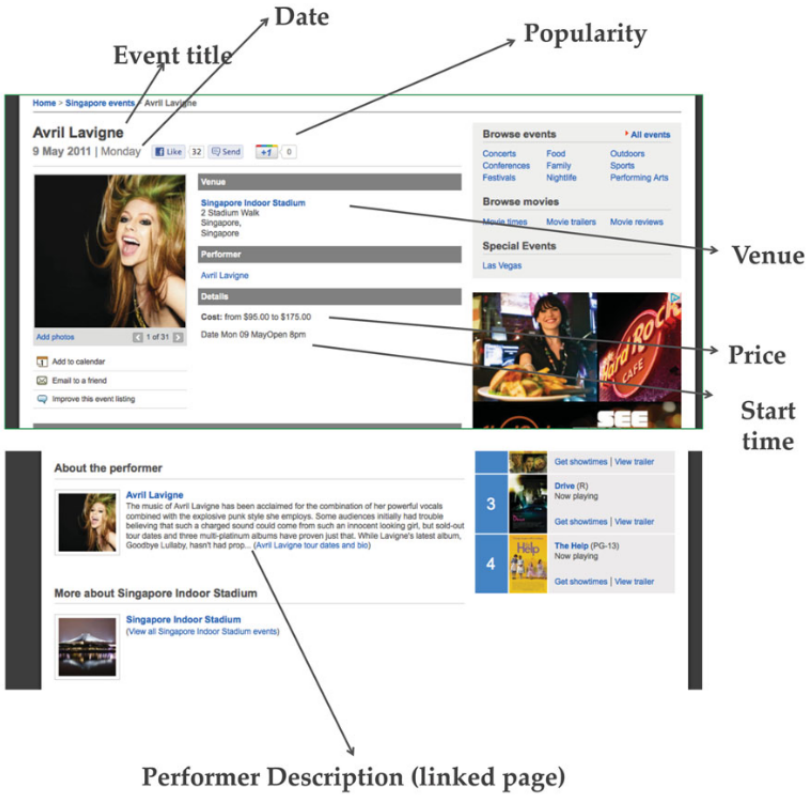
Fica evidente, portanto, que a seleção dos métodos e ferramentas de análise e extração a serem empregados em um algoritmo de web scraping está intrinsecamente ligada às finalidades e às características estruturais e de formato do site alvo. Essa escolha deve considerar diversos fatores, como a natureza dos dados desejados, a complexidade da página, a presença de elementos interativos, a compatibilidade com as tecnologias utilizadas no site e a necessidade de robustez e de velocidade.

2.6 Trabalhos Correlatos

Dados relacionados a feriados e eventos desempenham um papel crucial em diversos modelos de previsão, com variados objetivos, conforme destacado na literatura. Especificamente, no contexto da extração de informações online sobre eventos públicos, a pesquisa literária sublinha uma ampla gama de métodos de extração, escolhas de fontes e tipos específicos de eventos que são alvos de busca.

O estudo de Pereira, Rodrigues e Ben-Akiva (2015) concentra-se na obtenção de dados da internet relacionados a eventos públicos de grande porte, como shows, competições esportivas, desfiles e outros acontecimentos que reúnem uma grande quantidade de pessoas. Estes eventos costumam causar impactos consideráveis no sistema de transporte público. Dessa forma, o estudo propõe uma abordagem que utiliza a internet como fonte de informações contextuais sobre esses eventos especiais e desenvolve um modelo de previsão para estimar as chegadas do transporte público nas áreas onde esses eventos ocorrem. Para a coleta de dados sobre os eventos, foram utilizadas API's fornecidas por sites de locais que frequentemente hospedam eventos de grande envergadura. Com isto, o trabalho busca melhorar o planejamento e a gestão do transporte público em situações afetadas por eventos especiais. A Figura 2.6 mostra um exemplo de página web da qual os dados foram extraídos nesta pesquisa.

Figura 2.6: Exemplo de página de eventos



Fonte: (PEREIRA; RODRIGUES; BEN-AKIVA, 2015)

Adicionalmente, a pesquisa de Rodrigues, Markou e Pereira (2019) adotou uma estratégia abrangente, combinando técnicas de web scraping e a utilização de APIs para coletar informações sobre eventos provenientes de diversas fontes na web. Essa abordagem permitiu a obtenção de dados como títulos de eventos, datas, horários e descrições pormenorizadas. O objetivo primordial deste estudo centrou-se na melhoria da precisão das previsões de demanda de táxis durante estes eventos, alcançado através da integração de informações obtidas. Para concretizar essa meta, os pesquisadores coletaram dados de duas fontes principais: uma API fornecida pelo Facebook e o site de um grande centro de eventos situado em Nova York.

No contexto do setor varejista, informações relacionadas a feriados têm desempenhado um papel fundamental na melhoria da precisão dos modelos de previsão, especialmente na antecipação das vendas. O estudo de Wang Feng Wang et al. (2022) buscou aprimorar a previsão de vendas no varejo por meio de abordagens de aprendizado de máquina. Um dos métodos avaliados se destacou na previsão de vendas futuras, sugerindo sua utilidade geral em contextos varejistas. O estudo reconheceu a influência de fatores externos, como feriados, nas previsões.

No estudo de Meulstee Meulstee e Pechenizkiy (2008), foram utilizados dados relacionados a feriados, incluindo informações sobre tipos de feriados e datas específicas,

juntamente com dados sobre férias escolares semanais. O objetivo principal da pesquisa é aprimorar a previsão de vendas em empresas de alimentos, especialmente para produtos com prazo de validade curto e flutuações sazonais na demanda.

O modelo de (JIANG; RUAN; SUN, 2021) obteve resultados com um desvio-padrão inferior ao empregar o ajuste sazonal de feriados em comparação com o mesmo modelo sem tal ajuste. Isso ocorre devido à sazonalidade provocada pelos feriados.

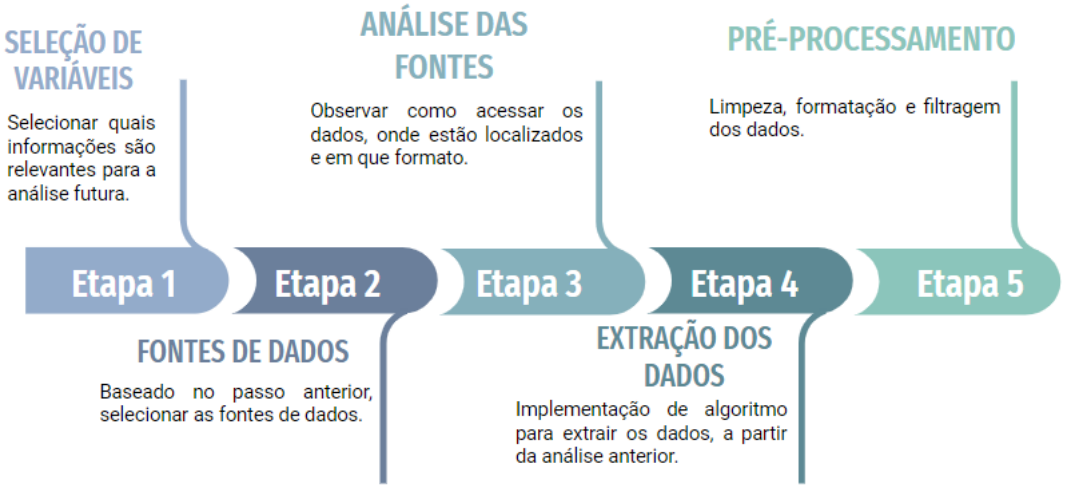
Assim, é possível perceber que a presença de eventos e feriados desempenha um papel crucial na movimentação de pessoas nas cidades e impacta diretamente as vendas do setor varejista. Durante esses períodos, é evidente um notável aumento na circulação de indivíduos, seja em busca de entretenimento, envolvimento em celebrações ou realização de compras relacionadas às festividades. Além disso, é importante destacar que dados relacionados a essas ocasiões especiais podem aprimorar significativamente a eficácia dos sistemas de previsão, permitindo uma melhor adaptação às flutuações sazonais.

Capítulo 3

Metodologia

Neste capítulo, são apresentadas todas as fases envolvidas no desenvolvimento e na validação das soluções propostas. Os passos aqui descritos visam estabelecer um processo robusto e consistente para a coleta. A Figura 3.1 apresenta um fluxograma com as etapas descritas neste capítulo.

Figura 3.1: Etapas do processo para coleta de dados



Fonte: Elaboração própria

3.1 Seleção de variáveis relevantes

Nessa etapa, as variáveis a serem coletadas foram selecionadas de forma a possibilitar que o conjunto final de dados possa responder questões como a incidência de acidentes em determinadas datas, a relação entre os tipos de acidentes e períodos de maior movimento, bem como possíveis diferenças significativas no comportamento dos funcionários durante eventos e feriados. Dessa forma, decidiu-se por coletar dados como as datas de início e fim dos feriados

e eventos, seu nome e o dia da semana em que ocorreram.

Além dos dados sobre feriados e eventos que serão coletados neste estudo, é fundamental obter informações relacionadas a acidentes de trabalho no setor em questão. Nesse sentido, nossa pesquisa está focalizada em um conjunto específico de cidades: Bragança, Matosinhos, Guimarães, Sobreda e Felgueiras. A coleta de dados abrangerá o período de 2016 a 2022. A escolha dessas cidades e desse intervalo temporal baseia-se nos requisitos do projeto iSafety (MORE, 2022), cujo objetivo é desenvolver um modelo de previsão de acidentes ocupacionais no setor varejista, contando com dados de acidentes da rede varejista portuguesa Continente nessas localidades.

Esse conjunto de cidades abrange locais de diversos tamanhos, incluindo grandes, médios e pequenos centros urbanos. Algumas delas são conhecidas pelo seu apelo turístico, enquanto outras desempenham papéis de destaque em nível regional, inclusive como capitais de seus respectivos distritos. Essa diversidade de características nos permitirá analisar a relação entre acidentes e eventos festivos em contextos urbanos variados, levando em consideração que a movimentação de pessoas durante essas ocasiões pode variar substancialmente de acordo com as particularidades de cada cidade.

3.2 Seleção das fontes de dados

Uma vez que as variáveis de interesse foram selecionadas, torna-se necessário localizar e identificar possíveis fontes de dados relevantes para o estudo. Dentre as opções consideradas estão páginas na Web, bancos de dados e serviços Web. É de extrema importância encontrar fontes confiáveis e abrangentes que possam fornecer a maior quantidade possível de informações relevantes.

Ao priorizar fontes de dados que ofereçam uma ampla gama de informações, é possível reduzir a necessidade de consultar várias fontes diferentes e, consequentemente, diminuir a complexidade do trabalho. Isso também pode contribuir para a eficiência e agilidade do processo de coleta.

Dessa forma, após realizar uma pesquisa em busca de fontes adequadas para obter os dados sobre feriados, foi escolhido um *Web Service* da Sapo que fornece dados sobre feriados municipais, distritais e nacionais portugueses (SAPO, 2023). A escolha se justifica uma vez que os dados são abrangentes e estruturados, conforme estipulado acima. Além disso, as informações estão bem categorizadas e são de fácil filtragem. Este *Web Service* retorna à solicitação um documento XML.

No que diz respeito aos dados sobre eventos públicos, não foi encontrada uma única fonte que agregasse dados de todas as cidades de interesse. Como solução, decidiu-se coletar dados dos sites das Câmaras Municipais e Juntas de Freguesia, identificados como as fontes que fornecem dados mais confiáveis e abrangentes. Sendo assim, os dados incluídos nesses sites

foram coletados por meio de técnicas de *Web Scraping* (BRAGANÇA, 2023; MATOSINHOS, 2023; FREGUESIA DE CHARNECA DE CAPARICA E SOBREDA, 2023; GUIMARÃES, 2023; FELGUEIRAS, 2023).

Destacam-se como pontos positivos dessa abordagem a confiabilidade dos dados, uma vez que são informações oficiais dos governos municipais, e o nível de detalhamento que esses dados apresentam. No entanto, os dados nesses sites podem estar dispostos de forma não estruturada ou semi-estruturada, o que dificulta sua extração e análise. Além disso, essas fontes podem conter ruídos, ou seja, informações irrelevantes ou imprecisas que podem afetar a qualidade dos dados obtidos.

3.3 Análise das fontes de dados

Nesta etapa, foram analisadas as estruturas de cada uma das fontes de dados selecionadas. O objetivo foi compreender como essas fontes são estruturadas e organizadas, visando à preparação para a elaboração do algoritmo.

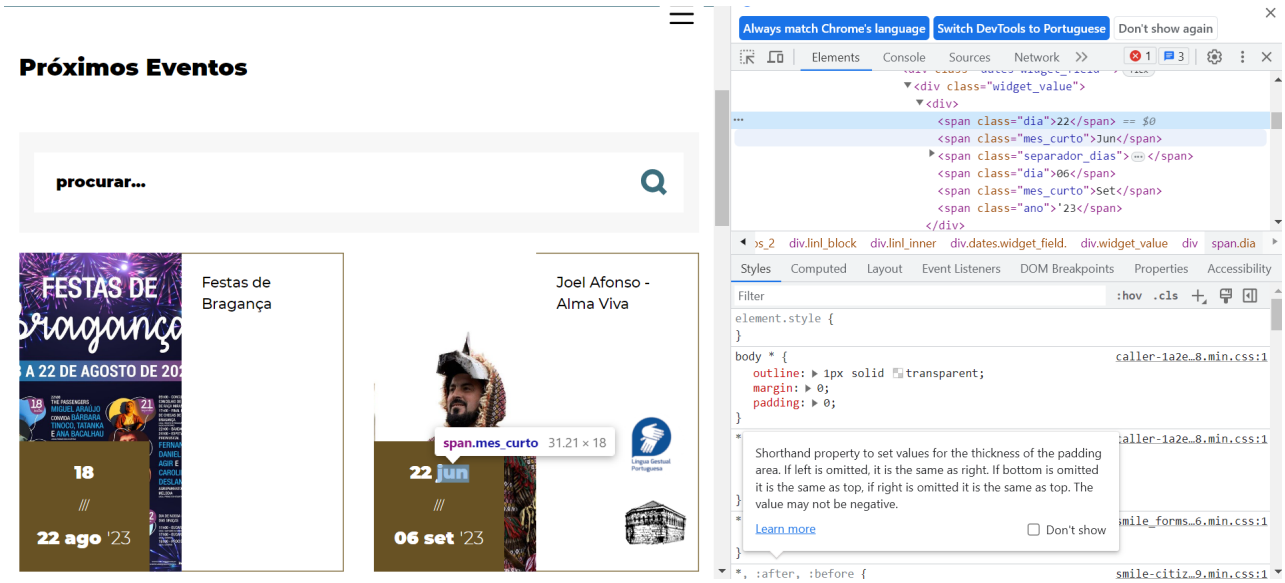
Nesse sentido, foram examinados aspectos relacionados à comunicação HTTP, como os parâmetros de rota das requisições e as URLs de interesse. Além disso, verificou-se a necessidade de preencher filtros, intervalos de datas, IDs ou outros critérios para restringir os resultados. Também foram observadas outras informações relevantes, como chaves de autenticação, tokens de acesso ou outros mecanismos necessários para autenticar e autorizar o acesso aos dados.

Além disso, esta análise visou apontar a localização dos dados nas fontes. Dessa forma, foram identificados os elementos, classes e atributos dos documentos XML e HTML que contêm os dados de interesse. A Figura 3.2 apresenta um exemplo da análise realizada, neste caso, no site da Câmara Municipal de Bragança. Na imagem, pode-se notar que a *tag* HTML `<span>` de classe `dia` possui informações sobre o dia de realização de determinado evento.

Alguns sites, entretanto, apresentam campos de busca, filtros de datas e botões que precisam ser acionados ou preenchidos a fim de visualizar os dados desejados. A Figura 3.3 ilustra um exemplo desses casos, onde é necessário preencher um filtro de datas para visualizar informações sobre eventos no site da Câmara Municipal de Felgueiras.

Em tais cenários, foi necessário empregar bibliotecas que interagem com navegadores em modo marionete e injetam scripts em JavaScript. Para essas rotinas, é crucial analisar os tipos de dados a serem inseridos, a localização dos campos no documento HTML e o funcionamento básico do site.

Figura 3.2: Exemplo de análise do código HTML de uma das fontes de dados



Fonte: Extraído de `https://www.cm-braganca.pt/visitar/agenda-de-eventos/todos-os-eventos`

Figura 3.3: Exemplo de análise de fonte de dados com interação em JavaScript

Fonte: Extraído de `https://cm-felgueiras.pt/venha-experimentar-saborear-e-sentir-felgueiras/agenda-de-eventos-municipal/`

3.4 Implementação do algoritmo de extração

Para coletar os dados das fontes identificadas, foram desenvolvidos scripts em Python com base nas técnicas de extração de dados encontradas na revisão de literatura e nas análises detalhadas das estruturas dessas fontes de dados. É importante destacar que, para cada fonte de dados, foi necessário desenvolver um script específico, considerando que as estruturas dos dados são distintas.

Utilizando os parâmetros de requisição e os métodos de acesso identificados durante a

análise das fontes de dados, os scripts interagem com o serviço Web e com os sites para obter os documentos HTML e XML relevantes.

Com os documentos em mãos, *tags* específicas são identificadas, utilizando classes e atributos previamente observados, que contêm os dados desejados. O texto principal dos elementos ou o valor dos atributos de interesse é então extraído. Os dados extraídos são adicionados a `DataFrames`, para facilitar o pré-processamento subsequente.

3.5 Pré-processamento de dados

Após a etapa de extração, o algoritmo passa para a fase de pré-processamento de dados. No cenário particular dos feriados, dada a estrutura bem organizada desses dados, o pré-processamento se limita à filtragem das informações. Enquanto o serviço Web disponibiliza os dados de todas as cidades portuguesas, apenas os feriados municipais das localidades de interesse são extraídos.

No contexto específico dos dados de eventos, o foco reside na identificação de conjuntos de eventos com base na similaridade de seus títulos. A meta principal é pré-agrupar os eventos de forma que aqueles de natureza semelhante sejam reunidos. Essa estratégia visa simplificar análises subsequentes sobre a correlação entre acidentes ocupacionais e eventos, uma vez que diferentes categorias de eventos têm o potencial de influenciar incidentes de maneiras diversas.

Para atingir esse objetivo, iniciamos formatando a coluna de títulos dos `DataFrames`, removendo acentos, espaços, letras que possam ser utilizadas como números romanos, caracteres especiais e dígitos. Essa etapa facilita a correspondência de títulos semelhantes. Adicionalmente, todas as letras maiúsculas são transformadas em minúsculas. Os títulos formatados são alocados em uma nova coluna do `DataFrame`.O Quadro 3.1 apresenta uma amostra de títulos de eventos em Bragança e o resultado após a formatação.

Quadro 3.1: Exemplo de dados obtidos após formatação de títulos de eventos

Título original	**Título formatado**
Festa Verão Bragança - Viver a Cidade - 2023	festaeraobragancaeracdade
Festa da História	festadahstora
PASSEIOS PEDESTRES BTT | 2019	passeospedestresbtt
Festival Literário de Bragança	festallterarodebraganca

Fonte: Elaboração própria

Em seguida, geramos uma lista de títulos formatados únicos, eliminando duplicatas.

Para realizar a categorização dos títulos com base em sua semelhança, empregou-se o agrupamento hierárquico, um tipo de algoritmo de aprendizado de máquina. Para tanto, foi utilizada a biblioteca *sklearn*, que já fornece uma implementação deste algoritmo. O método escolhido para o cálculo de similaridade foi o de Ward.

A partir do agrupamento realizado para cada localidade, foram construídos dendrogramas. Essas representações visuais foram elaboradas com o objetivo de restringir a hierarquia a, no máximo, cinco níveis, buscando uma apresentação mais eficaz. Essa abordagem implica na união de grupos menores em uma única entidade. Os valores entre parênteses na abcissa indicam a quantidade de dados contidos nessas entidades. Os valores fora dos parênteses representam o índice do dado em questão na lista de rótulos obtida do agrupamento. É importante observar que essa simplificação é exclusivamente visual e não compromete a integridade do agrupamento original.

Após o agrupamento, os títulos formatados são relacionados com seus respectivos títulos sem formatação para que seja possível visualizar o agrupamento em relação aos títulos originais.

Além disso, as datas dos eventos e feriados foram processadas para calcular o dia da semana correspondente a cada data.

A Figura 3.4 resume as etapas de pré-processamentos dos dados de eventos.

Figura 3.4: Etapas de pré-processamento dos dados de eventos

Fonte: Elaboração própria

Por fim, os dados foram armazenados em arquivos CSV a fim de facilitar o seu processamento futuro por modelos de previsão.

Capítulo 4

Resultados e Discussão

A coleta e o pré-processamento dos dados de interesse exigem uma análise da estrutura das fontes de dados e da forma de acesso a eles, bem como a aplicação de métodos estatísticos computacionais para a obtenção de informações pertinentes.

Nas seções seguintes, serão abordadas as implementações específicas para diferentes conjuntos de dados. Todo o desenvolvimento foi feito na linguagem *Python 3*. Posteriormente, os resultados obtidos são apresentados e discutidos.

4.1 Extração de dados de feriados

Para adquirir os dados de feriados do serviço Web escolhido, foi necessário, primeiramente, examinar a estrutura do serviço, compreendendo as funcionalidades oferecidas, as URLs de cada serviço, as respostas obtidas e os parâmetros de requisição necessários para obter as informações desejadas.

O serviço em questão fornece informações sobre feriados nacionais, regionais e municipais, com cada escopo disponível em uma URL distinta. Contudo, os dados regionais são restritos e não abrangem as regiões específicas consideradas neste trabalho, sendo, portanto, excluídos da coleta.

Com relação aos parâmetros de requisição, observou-se que, para os feriados nacionais, o único parâmetro necessário é o ano desejado. No entanto, no caso dos feriados municipais, é necessário obter o número de identificação do município, seguindo um padrão fornecido por outro serviço dentro do mesmo *Web Service*. Para atender a essa exigência, realizou-se a coleta prévia desses números de identificação antes de avançar para a obtenção dos dados propriamente ditos. A estratégia adotada foi coletar essas informações de todos os municípios disponíveis e, em seguida, filtrá-las para incluir apenas as cidades de interesse.

A estratégia para extrair os dados nacionais e municipais, bem como para obter os números de identificação dos respectivos locais, foi uniforme. Os códigos desenvolvidos realizam requisições HTTP usando o método GET, por meio da biblioteca *requests*, para cada

ano e, no caso dos feriados municipais, para cada município desejado no serviço em questão.

As respostas às requisições são apresentadas na forma de documentos XML. A Figura 4.1 detalha a estrutura retornada para os feriados nacionais.

Figura 4.1: Parte de resposta à requisição feita ao serviço Web

```
<GetNationalHolidaysResponse
↪  xmlns="http://services.sapo.pt/Metadata/Holiday">
  <GetNationalHolidaysResult>
    <Holiday>
      <Name>Ano Novo</Name>
      <Date>2023-01-01T00:00:00</Date>
      <Description>O Ano-Novo é um evento que acontece quando uma cultura
      ↪  celebra o fim de um ano e o começo do próximo.</Description>
      <Type>National</Type>
    </Holiday>
    <!-- ... -->
  </GetNationalHolidaysResult>
</GetNationalHolidaysResponse>
```

Fonte: Elaboração própria

Há uma hierarquia onde as *tags* no nível superior representam o serviço específico do qual o documento foi obtido. Em um nível inferior, encontram-se as *tags* `<Holiday>`, que contêm dados referentes aos feriados. Os elementos filhos armazenam informações como nomes, datas, descrições e tipos dessas ocasiões. As datas são sempre representadas no padrão ISO 8601 sem fuso horário (ISO CENTRAL SECRETARY, 2016).

Em casos de feriados municipais, as respostas também incluem o nome do município correspondente e seu número de identificação, conforme ilustrado na Figura 4.2.

Figura 4.2: Parte de resposta à requisição feita ao serviço Web

```
<GetHolidaysByMunicipalityIdResponse
    xmlns="http://services.sapo.pt/Metadata/Holiday">
  <GetHolidaysByMunicipalityIdResult>
    <Holiday>
      <!-- ... -->
      <Type>Municipal</Type>
      <Municipality>
        <Id>1303</Id>
        <Name>Felgueiras</Name>
      </Municipality>
    </Holiday>
  </GetHolidaysByMunicipalityIdResult>
</GetHolidaysByMunicipalityIdResponse>
```

Fonte: Elaboração própria

O corpo das respostas às solicitações é armazenado em um *buffer*, cujo conteúdo é expandido a cada chamada ao *Web Service*.

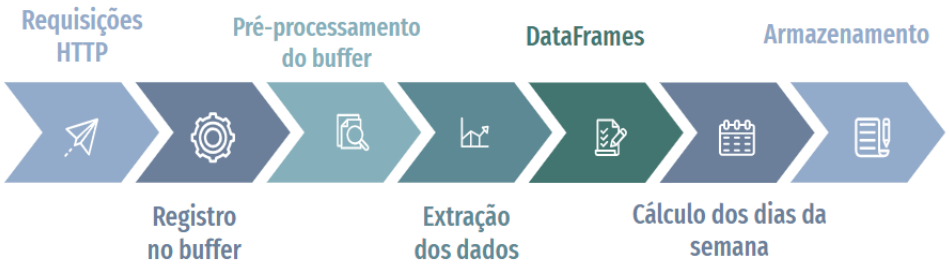
Realiza-se, então, o pré-processamento da estrutura resultante, ou seja, em um único documento contendo todas as respostas às requisições. Após uma adequação para a obtenção de um único documento XML, que envolve a remoção do elemento raiz de cada resposta, a *string* é processada por meio da biblioteca *ElementTree*.

Dentre os métodos usados, destaca-se o método `fromstring`, que realiza a leitura de um objeto a partir de texto. Tal objeto contempla mecanismos para identificar os elementos desejados, por meio dos métodos `find` e `find_all`. Eles recebem o nome da *tag* a ser buscada e a localizam na árvore. Os objetos retornados por esses métodos são do tipo `Element` e possuem o atributo `text`, que fornece o texto associado à *tag*, permitindo assim a obtenção dos dados desejados.

Dessa forma, o algoritmo constrói uma lista com todos os elementos `<Holiday>` do arquivo XML proveniente da etapa anterior. Para cada item dessa lista, os textos associados aos nós de nome e data são extraídos e incorporados a listas distintas. Com um cursor numérico comum, indexando cada uma das listas, é possível associar nomes a datas.

Os dados são incorporados como colunas em *DataFrames* da biblioteca *pandas*, aproveitando as capacidades do pacote para o pré-processamento. Nessa etapa, ocorre o cálculo dos dias da semana associados a cada feriado, utilizando o módulo *datetime* do Python. Com essas informações adicionadas, uma nova coluna de dias da semana é incluída. Posteriormente, o *script* gera tabelas CSV como resultado final. A Figura 4.3 resume o processo descrito. Os Quadros 4.1 e 4.2 apresentam uma amostra dos dados de feriados municipais e nacionais, respectivamente.

Figura 4.3: Etapas do processo de coleta dos dados de feriados



Fonte: Elaboração própria

Ao todo, foram coletados 176 dados de feriados municipais e 84 dados de feriados nacionais no período de 2018 a 2023. É importante notar que foram encontrados feriados municipais apenas em Felgueiras e Matosinhos. Os dados coletados nesta etapa são uma expansão dos dados coletados em outro trabalho já publicado e seguem a mesma metodologia (MARTINS et al., 2022).

Quadro 4.1: Amostra de feriados municipais coletados

Município	Data	Dia da semana
Felgueiras	29/06/2018	Sexta-feira
Matosinhos	2018/05/22	Terça-feira
Felgueiras	29/06/2023	Quinta-feira
Matosinhos	30/05/2023	Sexta-feira

Fonte: Elaboração própria

Quadro 4.2: Amostra de feriados nacionais coletados

Feriado	Data	Dia da semana
Ano Novo	01/01/2018	Terça-feira
Carnaval	13/02/2018	Segunda-feira
Dia da Liberdade	25/04/2019	Quinta-feira
Dia do Trabalhador	01/05/2021	Domingo
Imaculada Conceição	08/12/2023	Sexta-feira
Natal	25/12/2023	Sexta-feira

Fonte: Elaboração própria

4.2 Extração de dados de eventos

Para extrair os dados dos eventos públicos, foram desenvolvidos seis scripts, cada um adaptado às particularidades de sua fonte de dados. Todos seguem um processo semelhante, apresentado na Figura 4.4.

Figura 4.4: Etapas do processo de extração dos dados dos eventos

Fonte: Elaboração própria

O primeiro passo do algoritmo visa criar uma lista de URLs das páginas onde estão os dados de interesse. A metodologia para criar essa lista depende de como o site é apresentado.

Então, os algoritmos efetuam requisições GET a cada uma das URLs. As respostas a essas solicitações consistem nos códigos HTML correspondentes a cada página.

Para cada página, o contêiner principal onde estão armazenados os eventos é então identificado. A estratégia exata para essa localização também varia conforme a estrutura específica da página. Concluída a fase de identificação, o algoritmo avança para a etapa de extração de dados propriamente dita.

A biblioteca `BeautifulSoup` é empregada para a extração das datas e dos títulos dos eventos. Ela provê seletores que encontram elementos a partir do nome da tag e, opcionalmente, de seus atributos, notavelmente `class` e `name`. Os nós de interesse retornados são objetos contendo o código HTML do elemento e seus filhos, além de uma propriedade para retornar apenas o texto contido naquele elemento. Tal propriedade é utilizada para extrair os dados desejados.

Durante a extração, para cada evento encontrado, os algoritmos geram listas separadas para os dias, meses e anos envolvidos no evento. Com elas, as datas de início e término de cada evento ficam definidas. Eventos com uma única data são revelados por listas com comprimento 1. Nesses casos, por conveniência, o valor é duplicado, para que o mesmo estilo de processamento seja aproveitado. De posse dos dados, é gerado um `DataFrame` da biblioteca `pandas`, contendo, para cada evento, o nome e as datas de início e término, sendo estas armazenadas em colunas separadas para dia, mês e ano, conforme exemplificado na Tabela 4.1.

Tabela 4.1: Exemplo de `Dataframe` gerado a partir das listas de datas

título	**diaA**	**diaB**	**mesA**	**mesB**	**anoA**	**anoB**
XI Bienal da Máscara - Mascararte	23	25	Nov	Nov	23	23
Cinema Digital_Novembro 2023	1	29	Nov	Nov	23	23
INFORMAÇÃO_Mercado (···)	29	1	Out	Nov	23	23
Prémio Literário da Lusofonia (···)	13	13	Out	Out	23	23
Cinema Digital - Outubro 2023	4	27	Out	Out	23	23
Cinema Digital - Setembro 2023	6	29	Set	Set	23	23
Festas de Bragança	18	22	Ago	Ago	23	23

Fonte: Elaboração própria

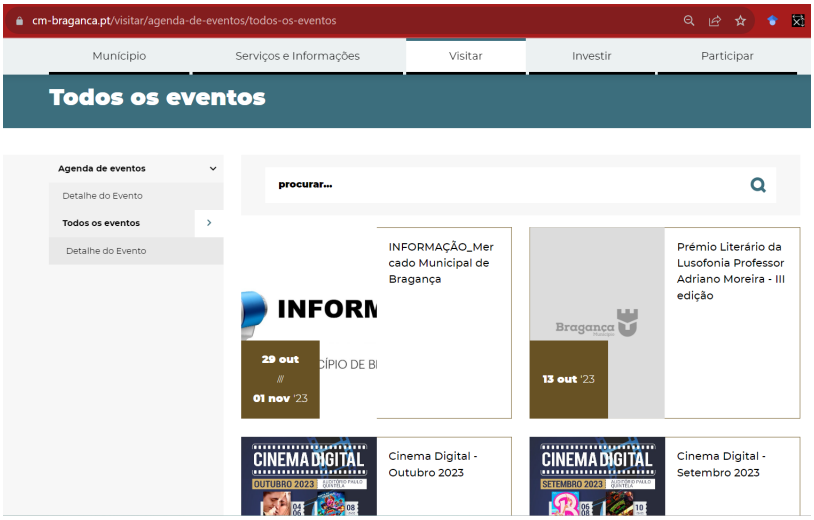
O processo abrangente de pré-processamento envolve várias etapas críticas, iniciando-se pela identificação de títulos semelhantes. Em seguida, procedemos à conversão das datas para formato numérico, fazendo uso do método `to_numeric`. Em determinadas situações, foi preciso não apenas converter o tipo das variáveis, mas sim traduzir as siglas dos meses para seus valores numéricos correspondentes. Na fase subsequente, que envolve o cálculo dos dias da semana em que os eventos ocorreram, recorremos ao módulo `datetime` do Python. Os seus métodos `strptime` e `weekday` foram utilizados para esta tarefa.

Nas subseções a seguir, detalharemos as características específicas das implementações dos algoritmos para cada cidade, bem como as análises das estruturas de cada fonte de dados.

4.2.1 Eventos públicos em Bragança

Para extrair os dados relacionados à cidade de Bragança, iniciamos com uma análise das páginas de eventos no site da Câmara Municipal (Figura 4.5).

Figura 4.5: Site da Câmara Municipal de Bragança

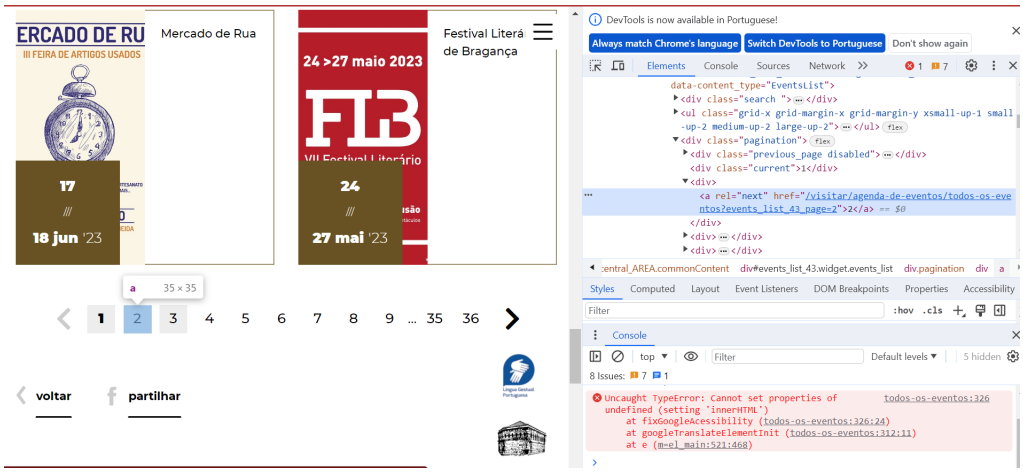


Fonte: Extraído de
`https://www.cm-braganca.pt/visitar/agenda-de-eventos/todos-os-eventos`

Ao examinar a imagem, constatamos que as informações sobre os eventos são organizadas em seções que incluem um cartaz do evento e a data. Os meses são abreviados nas três primeiras letras e os anos são precedidos por um apóstrofo. Esses detalhes devem ser levados em consideração no processamento.

Observada a organização do site, os parâmetros de rota que indicam a página específica estão ocultos para os usuários, fazendo com que as URLs aparentem ser idênticas no navegador. Assim, foi necessário extrair os links da barra de paginação, mostrada na Figura 4.6, juntamente com as *tags* e classes associadas.

Figura 4.6: Barra de paginação do site da Câmara Municipal de Bragança



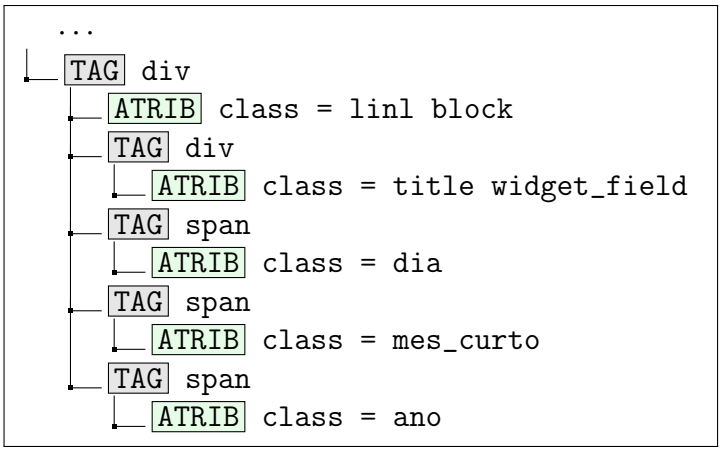
Fonte: Extraído de
`https://www.cm-braganca.pt/visitar/agenda-de-eventos/todos-os-eventos`

Na figura, percebe-se que a barra é codificada no HTML por meio de uma `<div>` com a

classe `pagination`. Nessa estrutura, encontram-se os elementos HTML `<a>`, que apresentam o número de cada página e possuem o atributo `href`, utilizado para a referência do link. Além disso, na imagem, pode-se perceber nem todos os números de página disponíveis são exibidos, substituindo alguns por reticências. Ao analisar os links, percebeu-se que as variações estavam centradas no último parâmetro da URL, mais precisamente no item `page`. Nesse cenário, o script inicia localizando o total de páginas de interesse, e com essa informação, é gerada a lista contendo os links para cada uma delas.

Após essa etapa, o script prossegue com a extração dos dados. Para isso, foi fundamental analisar a estrutura do código HTML do site. A Figura 4.7 ilustra a estrutura identificada em relação aos elementos que contêm os dados desejados.

Figura 4.7: Estrutura do site da Câmara Municipal de Bragança



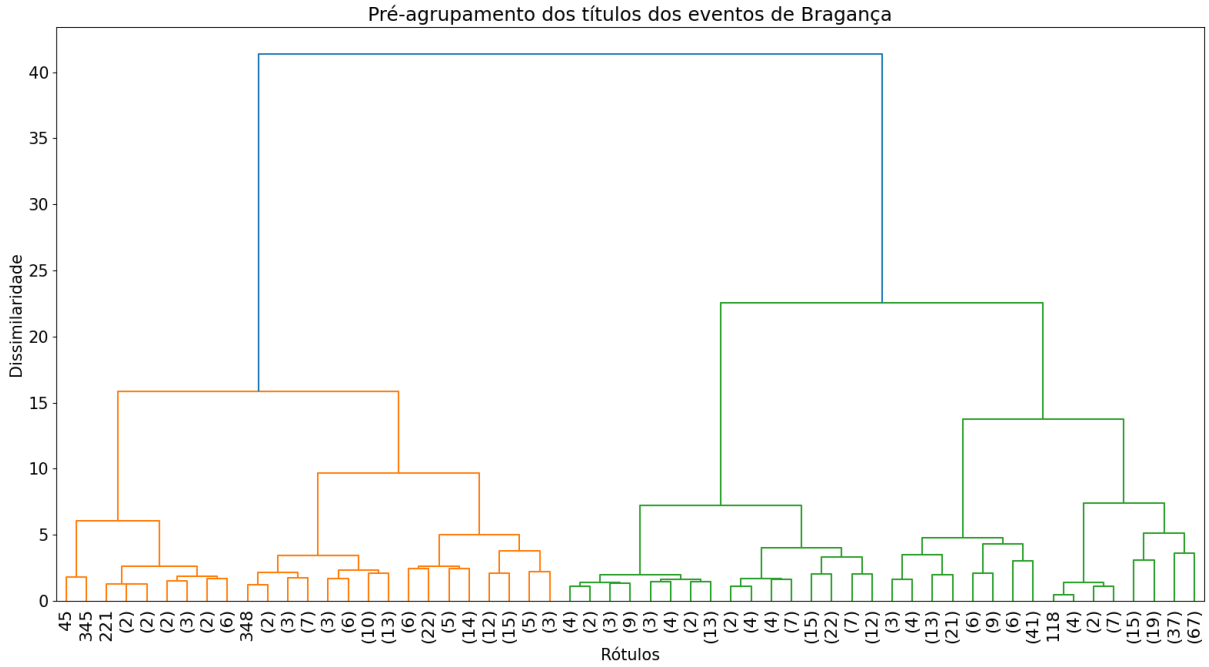
Fonte: Elaboração própria

Nesta estrutura, os elementos `<div>` de classe `link_block` são contêineres onde os eventos são exibidos. As informações para a data de dia, mês e ano são filhos destes contêineres principais, sendo usados para a extração dos dados.

Depois de gerar a tabela contendo os dados de todos os eventos, as colunas referentes as datas passaram por uma conversão para representações numéricas, substituindo as representações textuais anteriores, e os apóstrofos são removidos das representações dos anos. O resultado é um `DataFrame`, similar ao apresentado na Tabela 4.1, com 316 dados de eventos, excluindo aqueles com datas anteriores a 2016. A Figura 4.8 apresenta o dendrograma obtido após a classificação dos dados.

Em Bragança, a consolidação de 251 títulos únicos resultou em 76 *clusters*. A análise do dendrograma sugere um agrupamento inicial razoavelmente bem formado, considerando a natureza repetitiva dos dados, proveniente da grande recorrência de eventos.

Figura 4.8: Classificação obtida dos dados de eventos em Bragança

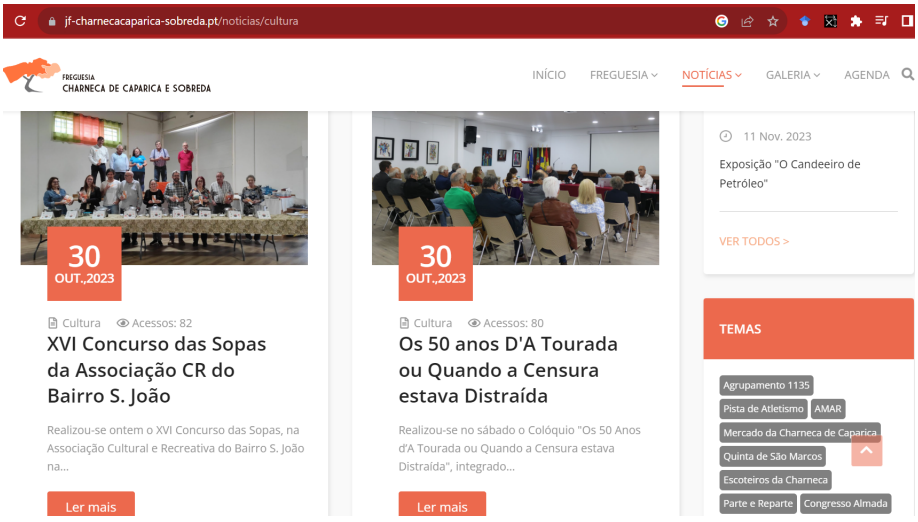


Fonte: Elaboração Própria

4.2.2 Eventos públicos em Charneca de Capernica e Sobreda

A análise da seção de eventos do site da Freguesia de Charneca de Caparica e Sobreda revelou um padrão de URLs, onde apenas um parâmetro de rota é modificado. A Figura 4.9 oferece uma visão geral do conteúdo acessível ao explorar a seção de eventos do site.

Figura 4.9: Seção de eventos do site da Freguesia de Charneca de Caparica e Sobreda



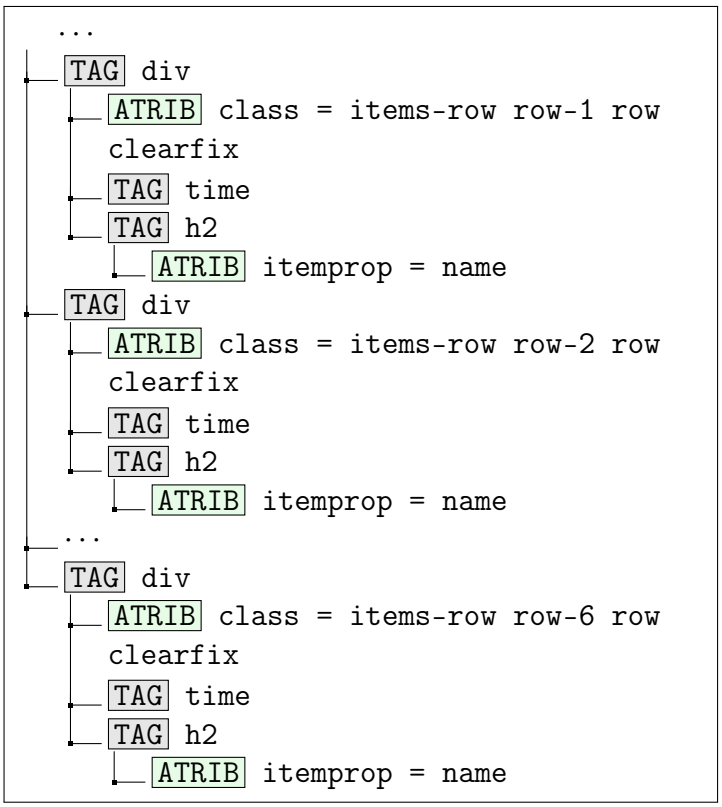
Fonte: Extraído de https://www.jf-charnecacaparica-sobreda.pt/noticias/cultura?start=6

Com base no padrão identificado, a compilação da lista de links inicia-se atribuindo a

cada elemento da lista uma URL distinta com o parâmetro de rota ajustado. Como a URL da primeira página difere desse padrão, ela foi adicionada separadamente à lista, enquanto as demais foram geradas por meio de um simples laço de repetição.

Para prosseguir com a extração dos dados, foi necessário analisar a organização do código-fonte das páginas. A Figura 4.10 exibe a estrutura observada.

Figura 4.10: Estrutura do site da Freguesia de Charneca de Carpenica e Sobreda



Fonte: Elaboração própria

Na estrutura analisada, os eventos seguem um padrão organizacional, sendo agrupados em elementos `<div>` de classes distintas. Cada página exibe precisamente seis eventos, com os nomes das classes dos elementos sequenciados numericamente de 1 a 6. Desta forma, o número 1 está associado ao evento mais recente em cada página, enquanto o número 6 está presente na classe do evento mais antigo. O laço de repetição ajusta dinamicamente a classe do contêiner de busca, identificando um evento a cada iteração.

Durante a análise, constatou-se que as componentes das datas dos eventos não estão contidas em elementos HTML distintos, sendo apresentadas em um único elemento do tipo `<time>`. Este elemento carrega todos os dados necessários para realizar a leitura da data do evento. Também foi constatado que nenhum evento possui uma data de término, tornando desnecessária a verificação inicialmente proposta para esse aspecto.

Quanto aos títulos, eles estão localizados nos elementos `<h2>`, sendo esses elementos filhos dos contêineres que apresentam as informações sobre os eventos.

O Quadro 4.3 apresenta uma amostra dos dados coletados a partir das informações apresentadas.

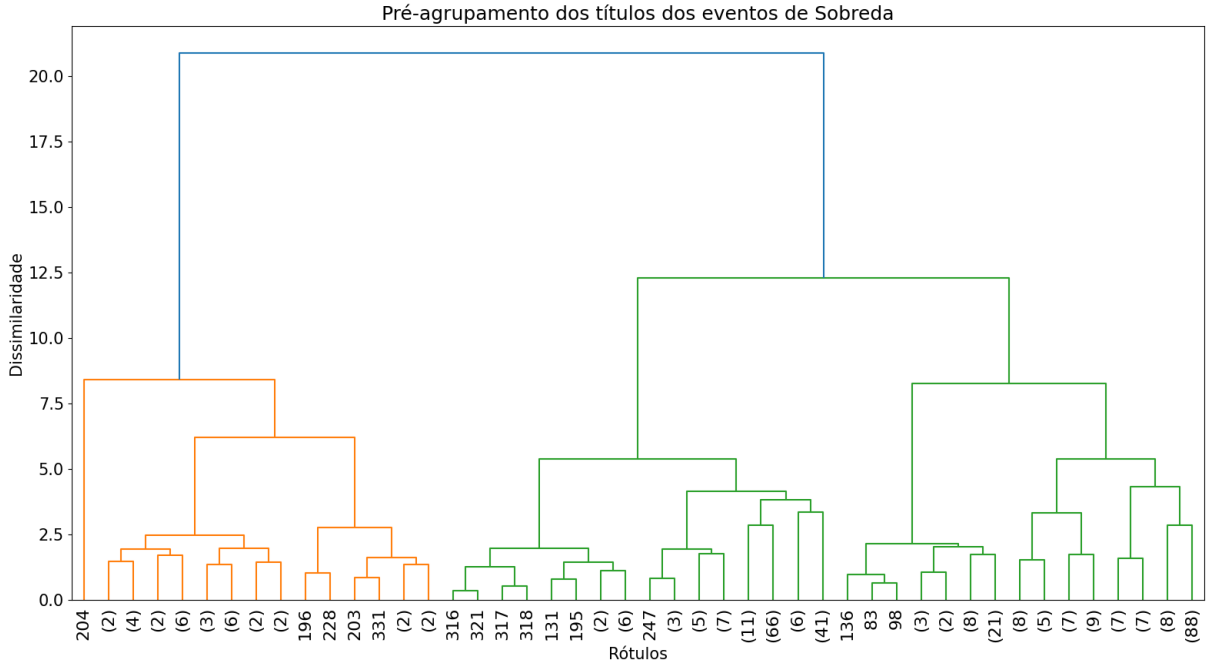
Quadro 4.3: Amostra de eventos públicos em Sobreda de 2016 a 2023

Título	**Data inicial**	**Data final**
Montagem das iluminações de Natal pela nossa Freguesia	09/11/2023	09/11/2023
19.º Aniversário das Cantadeiras de Essência Alentejan	07/11/2023	07/11/2023
Festa Jovem no hipódromo de Vale Figueira	04/06/2019	04/06/2019
Teatro de Rua incluído no Sementes	21/05/2019	21/05/2019
Ai que linda Brincadeira	11/01/2016	11/01/2016
Lançamento do livro "Na Sobreda"	05/01/2016	05/01/2016

Fonte: Elaboração própria

No total, foram coletados 379 eventos entre 2016 e 2022 para a freguesia de Charneca, de Caparica e Sobreda.

Figura 4.11: Pré-classificação obtida dos dados de eventos em Charneca de Carpenica e Sobreda



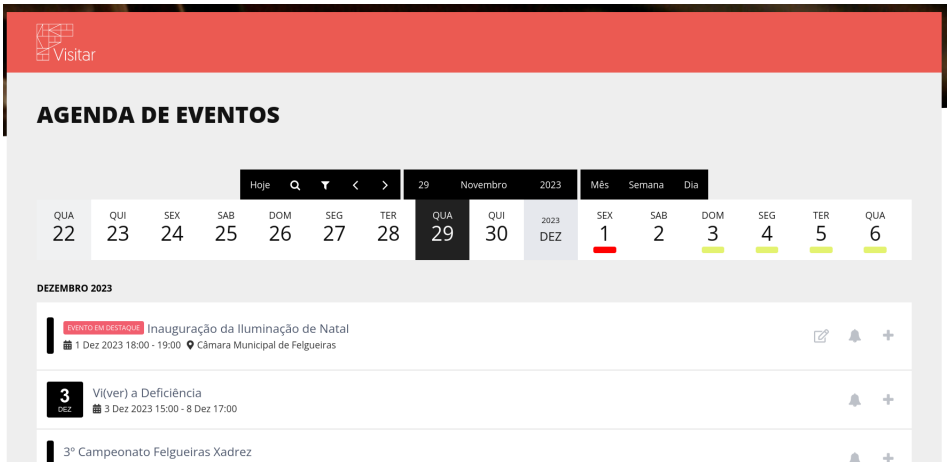
Fonte: Elaboração própria

Um conjunto de 366 títulos únicos foi transformado em 133 grupos de eventos semelhantes. Neste caso, destaca-se a baixa incidência de títulos duplicados, evidenciando a diversidade e especificidade dos eventos representados no conjunto de dados.

4.2.3 Eventos públicos em Felgueiras

A agenda de eventos da página da Câmara Municipal de Felgueiras (Figura 4.12) apresenta menus de seleção que indicam o período de datas dos eventos desejados.

Figura 4.12: Agenda de Eventos do Site da Camâra Municipal de Felgueiras



Fonte: Extraído de `https://cm-felgueiras.pt/venha-experimentar-saborear-e-sentir-felgueiras/agenda-de-eventos-municipal/`

Esse menu opera de modo que, ao escolher uma data, são exibidos os dados de eventos que ocorreram entre a data selecionada e o final do ano especificado. O carregamento da página é dinâmico, de forma que, dependendo da quantidade de eventos no período escolhido, pode ser necessário clicar em um botão "Ver Mais" para acessar os próximos eventos da página. A estrutura é apresentada na Figura 4.13.

Figura 4.13: Estrutura do site da Câmara Municipal de Felgueiras

```
...
└── TAG div
    ├── ATRIB class = stec-layout-event-preview-left-text
    ├── TAG span
    │   └── ATRIB class = stec-layout-event-preview-left-text-date
    └── TAG p
        └── ATRIB class = stec-layout-event-preview-left-text-title
```

Fonte: Elaboração própria

Esses elementos interagem com scripts em JavaScript, o que impossibilitou a aplicação da mesma estratégia utilizada para coletar dados de outras localidades. Como alternativa, foi necessário utilizar as capacidades da biblioteca `splinter` do Python, que automatiza esses processos ao interagir com navegadores em modo marionete e permitir injetar scripts codificados em JavaScript. Esta biblioteca opera com base nas funcionalidades do Selenium em seu núcleo.

O algoritmo desenvolvido inicia conectando o navegador ao site da Câmara Municipal. Ao acessar o site, uma janela de *pop-up* solicita a permissão para o uso de *cookies*. O script então simula um clique no botão. Após essa etapa, o algoritmo aguarda o carregamento da página e desloca a viewport para baixo, possibilitando a visualização dos campos de datas que realizam a busca dos eventos conforme as datas aplicadas.

O menu de seleção é preenchido para obter os dados a partir de janeiro, enquanto um laço de repetição é acionado para percorrer os anos de 2019 a 2023. A cada iteração, o script move o cursor sobre o campo correspondente no menu de seleção e, posteriormente, seleciona o ano desejado.

Após a seleção das datas, os dados de feriados são carregados na página até alcançarem um determinado limite. Neste ponto, a execução de cliques no botão "Ver Mais" se torna necessária, sendo repetida até que todos os dados estejam completamente visíveis na página. Este procedimento é implementado por meio de uma estrutura de repetição que persiste até que o botão não esteja mais presente.

Posteriormente, o código HTML resultante é processado pela biblioteca *BeautifulSoup*, mantendo a estratégia previamente delineada. No entanto, em vez de criar uma lista com os links de cada página, uma lista com os códigos HTML já interpretados é gerada. Cada elemento dessa lista corresponde ao código fonte contendo os dados de todos os eventos de um determinado ano.

Para cada elemento na lista, o código identifica os contêineres nos quais os eventos estão contidos. As *tags* associadas às datas e aos títulos dos eventos estão encapsuladas nesses contêineres. Os títulos dos eventos são encontrados em *tags* do tipo `<p>`.

A extração das datas, em contrapartida, é um pouco mais elaborada. Em situações em que um evento se estende por mais de um dia, a data de término é encontrada na mesma *tag* que a data de início, separadas apenas por um hífen. Nesse contexto, foi necessário verificar a presença desse hífen nos dados coletados e, quando confirmado, realizar a separação dos dados usando esse marcador.

Entretanto, a mera existência do hífen não é suficiente para garantir a presença das datas de término do evento. Isso se deve ao fato de que o hífen pode ser utilizado apenas para separar os horários de início e término do evento. Para abordar essa ambiguidade, observamos também a quantidade de dados existentes após o hífen.

O Quadro 4.4 exibe uma amostra dos dados coletados aplicando tal procedimento.

A totalidade dos dados contabilizou 145 eventos entre 2019 e 2023. Destaca-se que dados anteriores a 2019 não estão disponíveis no site. A Figura 4.14 exibe o resultado do agrupamento realizado.

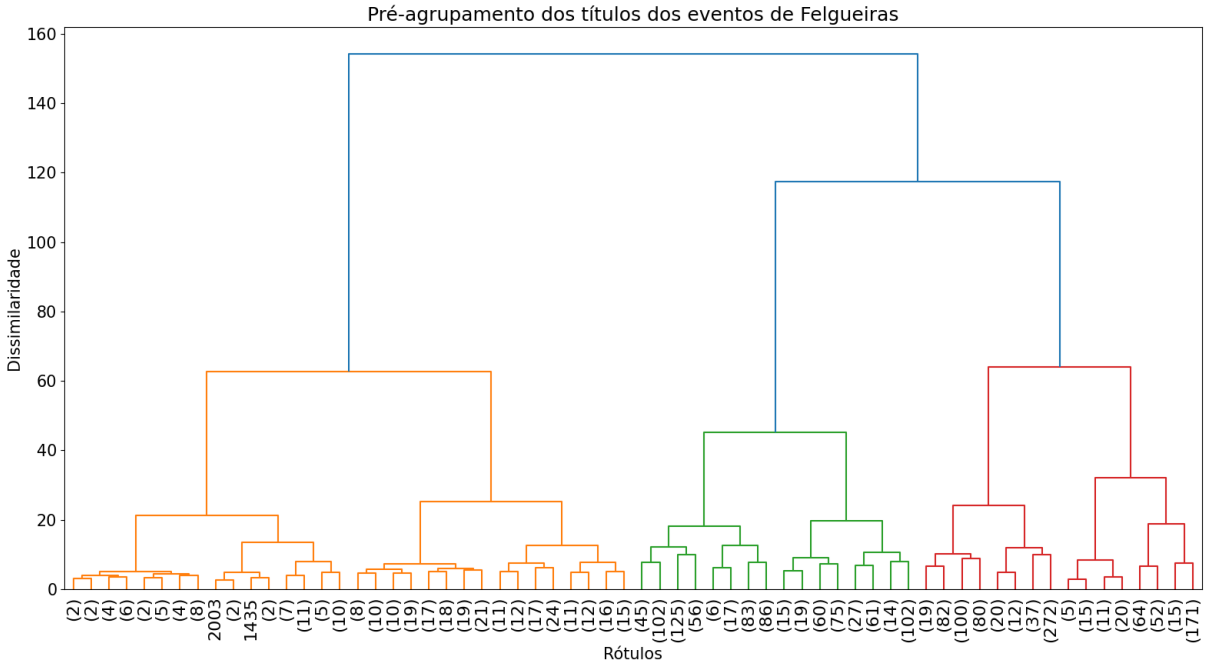
A partir do processo de agrupamento, emergiram 25 *clusters* em um conjunto de 105 títulos exclusivos. Novamente, o pequena proporção entre grupos e registros ocorre em detrimento da grande similaridade entre títulos de eventos recorrentes.

Quadro 4.4: Amostra de eventos públicos coletados em Felgueiras

Título	**Data inicial**	**Data final**
Oficinas de Inovação	26/01/2019	26/01/2019
Origo Ensemble na Igreja de Airães	20/02/2019	20/02/2019
Dia dos namorados	14/02/2020	14/02/2020
2ª Edição | Corrida para a VIDA	16/05/2020	30/05/2020
3º Campeonato Felgueiras Xadrez	08/12/2021	09/12/2023
Workshop de mini dioramas naturais	08/12/2023	9/12/2023

Fonte: Elaboração própria

Figura 4.14: Pré-classificação obtida dos dados de eventos em Felgueiras



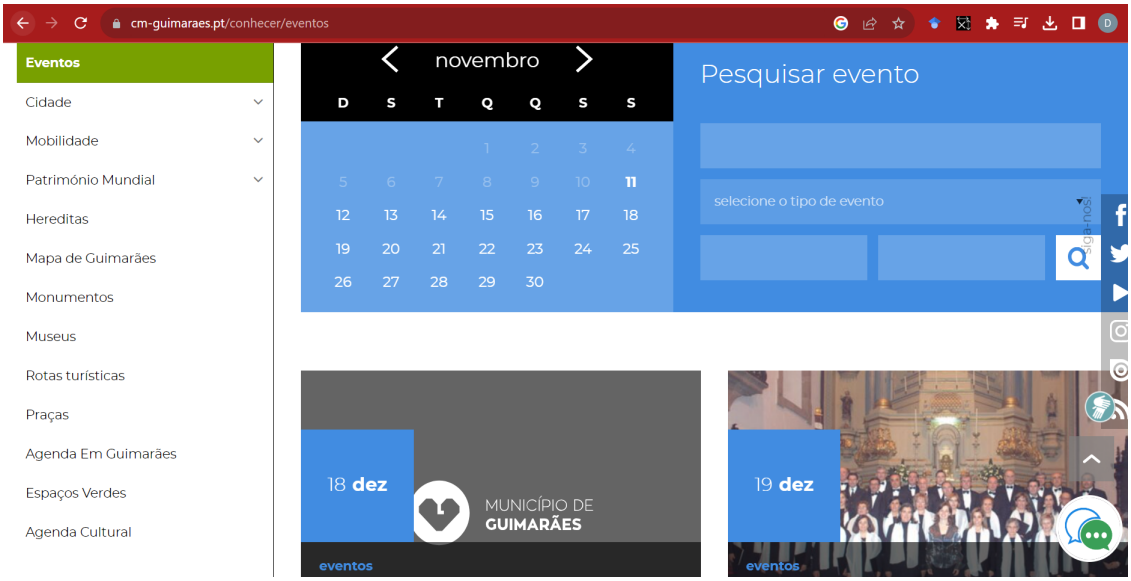
Fonte: Elaboração própria

4.2.4 Eventos públicos em Guimarães

Ao analisar as páginas da seção de eventos no site da Câmara Municipal de Guimarães (Figura 4.15), identificou-se a presença de filtros de data que possibilitam a especificação dos períodos desejados para a visualização dos eventos. Ao preencher esses filtros, observou-se que parâmetros de rotas relacionados às datas de início e fim dos eventos são modificados na URL da página, ainda que não sejam mostrados no navegador. Os anos dos eventos não são incluídos diretamente no HTML, tornando o filtro de data a única maneira de determinar em qual ano cada evento ocorreu.

Além disso, os dados apresentados em resposta ao filtro selecionado podem se distribuir em diversas páginas, dependendo da quantidade de informações no período. Dessa forma, a barra de paginação no site foi utilizada para identificar a quantidade de páginas a serem

Figura 4.15: Seção de eventos do site de Guimarães



Fonte: Extraído de `https://www.cm-guimaraes.pt/conhecer/eventos`

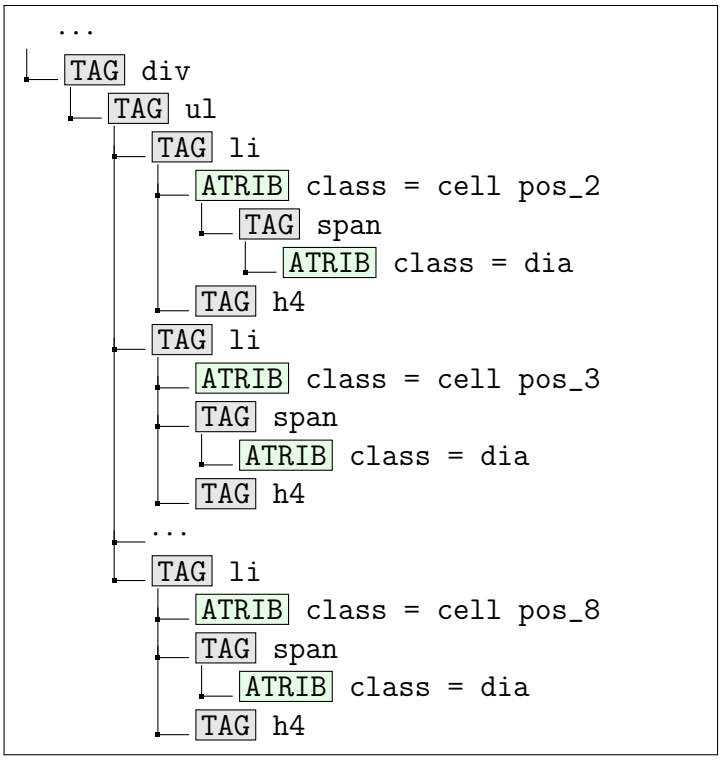
buscadas.

A solução final conta com um laço de repetição aninhado a outro. O laço mais externo lida com o filtro de datas, para poder percorrer cada ano de análise, enquanto o laço mais interno serve para popular a lista de links, alterando o parâmetro de rota relativo ao número das páginas.

Além disso, observou-se que as informações relativas aos eventos estão organizadas em uma lista HTML, contendo doze itens em cada página. Cada item contém uma imagem representativa do evento, seu título e sua data. Novamente, os meses são representados pela abreviação das três primeiras letras, e os anos são precedidos por um apóstrofo.

Inicialmente, foram identificados os itens da lista que contêm informações sobre os eventos. Embora esses elementos tenham classes diferentes, o nome de suas classes segue um padrão no qual apenas o número relativo à sua posição na página é modificado, com exceção do primeiro item da lista. Os elementos que exibem os títulos e as datas dos eventos são descendentes dos itens da lista HTML. As datas e os títulos são representados em *tags* `<span>` e `<h4>`, respectivamente. A Figura 4.16 apresenta a estrutura simplificada encontrada em relação aos dados alvejados, utilizada para extrair os dados. Uma amostra desses dados pode ser vista no Quadro 4.5.

Figura 4.16: Estrutura do site da Câmara Municipal de Guimarães



Fonte: Elaboração própria

Quadro 4.5: Amostra de eventos públicos em Guimarães de 2016 a 2020

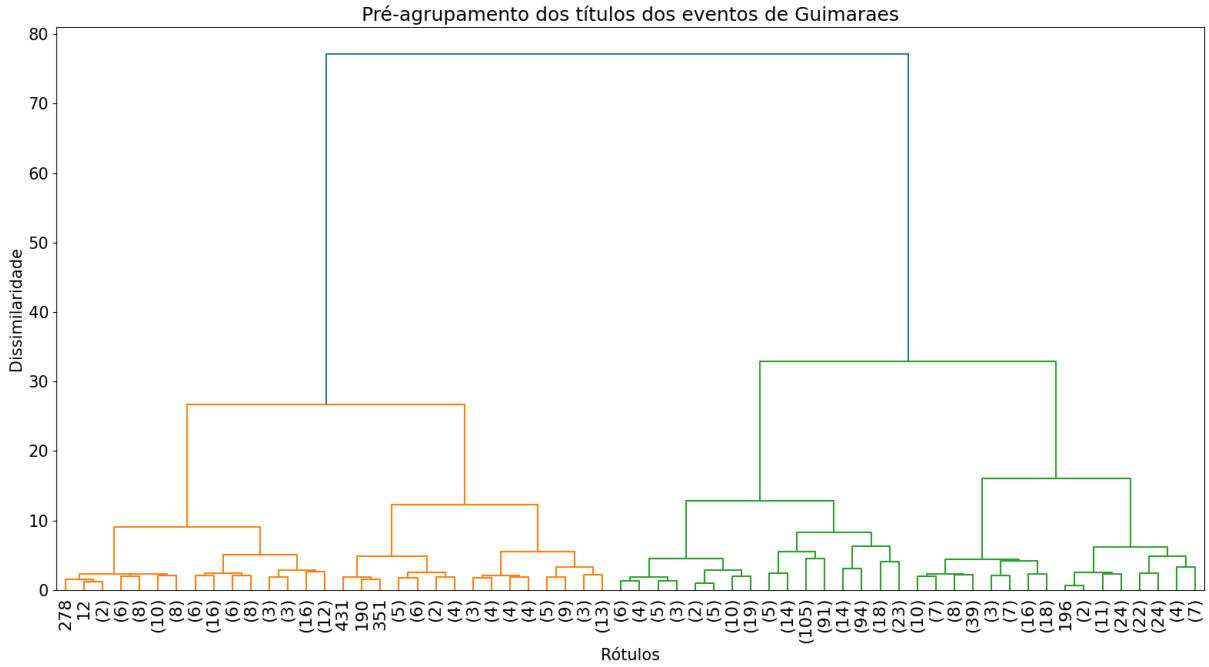
Título	**Data inicial**	**Data final**
GoGospel com Banda ao Vivo	23/11/2020	23/11/2020
Os Seres que nunca fomos	07/11/2020	07/11/2020
Mercadinho Local	22/12/2018	22/12/2018
GUIdance 2019 - Festival Internacional de Dança Contemporânea	07/02/2019	16/02/2019
Jacco Gardner atua no Centro Cultural Vila Flor	23/01/2016	23/01/2016
Iniciativa «Intramuros» decorre este sábado	05/01/2016	05/01/2016

Fonte: Elaboração própria

Os dados coletados abrangem um total de 894 eventos, compreendendo o período de 2016 a 2020. Registre-se que informações para anos subsequentes a 2020 não se encontram disponíveis no site.

O processo de agrupamento resultou na identificação de 548 grupos a partir de 792 títulos exclusivos. No entanto, a distribuição desses grupos sugere que cada um deles possui um número reduzido de elementos, o que indica a ausência de agrupamentos de eventos muito definidos. A Figura 4.17 apresenta o dendrograma gerado a partir dos resultados do agrupamento dos dados coletados.

Figura 4.17: Pré-classificação obtida dos dados de eventos em Guimarães

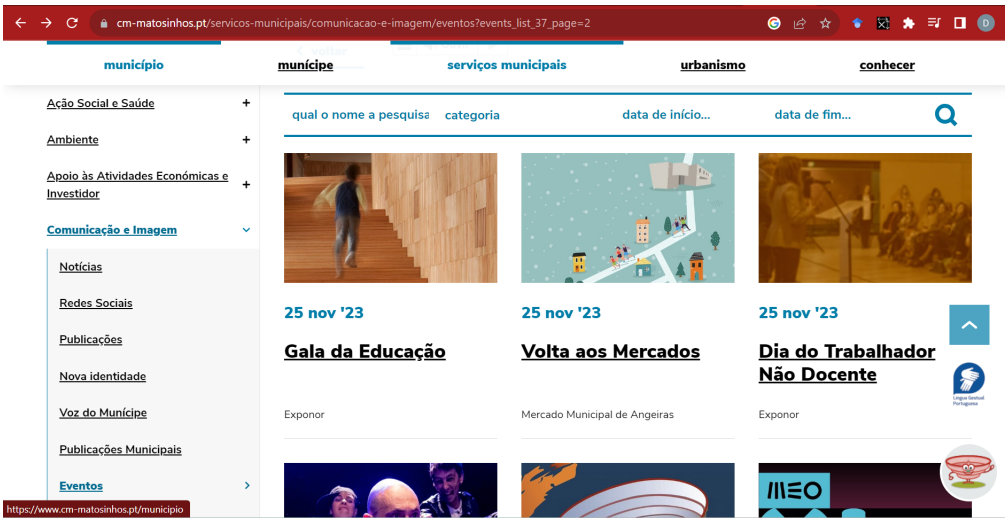


Fonte: Elaboração própria

4.2.5 Eventos públicos em Matosinhos

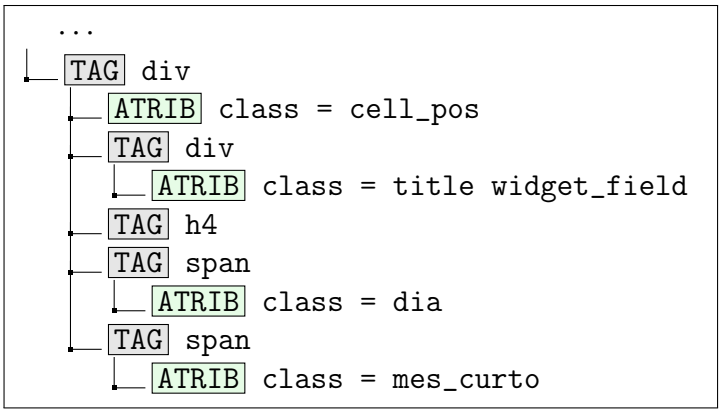
No site da Câmara Municipal de Matosinhos, observou-se, mais uma vez, um padrão nas URLs, onde apenas o valor do parâmetro de rota "events_list_37_page" é modificado, assumindo o número da página. A Figura 4.18 ilustra a visualização ao acessar a seção de eventos do site.

Figura 4.18: Seção de eventos do site da Câmara Municipal de Matosinhos



Fonte: Extraído de `https://www.cm-matosinhos.pt/servicos-municipais/comunicacao-e-imagem/eventos`

Figura 4.19: Estrutura do site da Câmara Municipal de Matosinhos



Fonte: Elaboração própria

Na Figura, também se observa a utilização de apóstrofos precedendo os anos em que os eventos ocorreram. Além disso, é possível notar que os meses são representados pelas suas três primeiras letras. Essas informações são importantes para o pré-processamento.

Com base no padrão identificado, a lista de links é gerada a partir da URL base. Cada elemento da lista é associado a uma URL com o parâmetro de rota alterado, começando com o link da primeira página e terminando com o da última.

Para executar a etapa de extração dos dados, analisou-se a hierarquia do código HTML. A Figura 4.19 apresenta o resultado dessa análise em relação aos dados alvejados.

Considerando a estrutura apresentada, cada evento é encapsulado em um contêiner que, por sua vez, inclui um outro com o título do evento. Além disso, as informações relativas às datas estão incluídas em elementos do tipo `<span>`. A partir dessas informações, realizou-se a extração dos dados.

O Quadro 4.6 apresenta os dados extraídos. Após a criação da tabela com todos os dados coletados, as colunas de anos de início e término são formatadas para remover os apóstrofos. Todas as colunas relacionadas às datas são transformadas para que contenham dados numéricos.

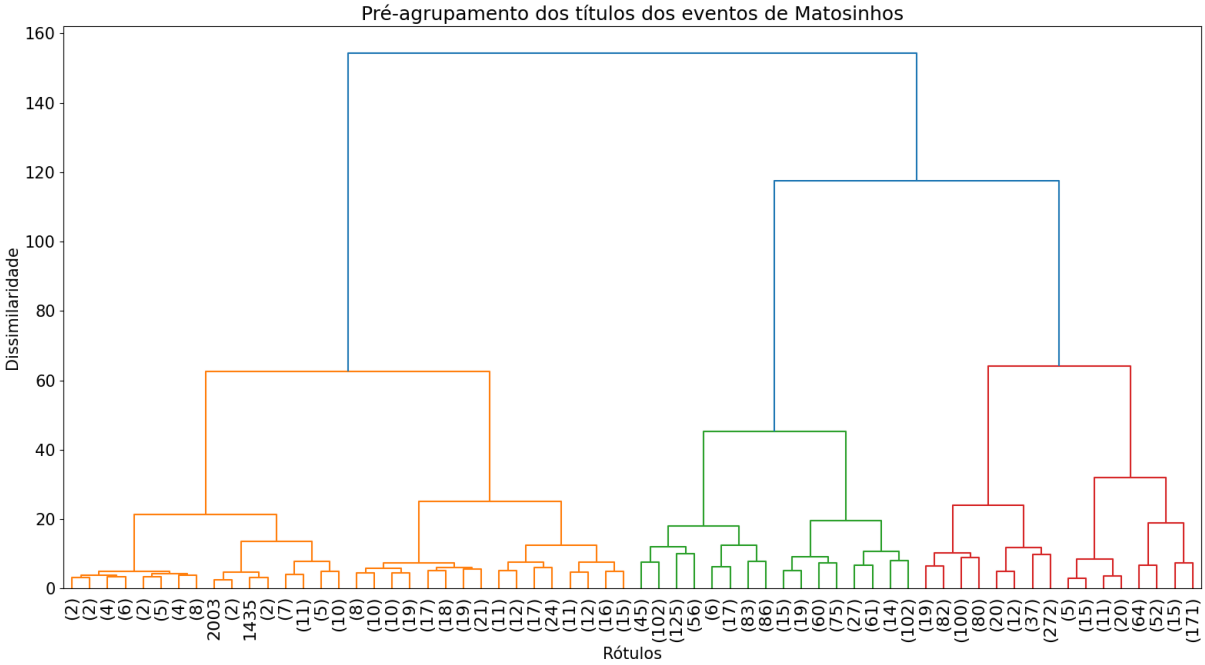
Quadro 4.6: Amostra de eventos públicos em Matosinho de 2016 a 2023

Título	**Data inicial**	**Data final**
GallaVoley 2023	26/12/2023	29/12/2023
Shout!	22/12/2023	22/12/2023
Dá-me Música	25/05/2018	25/05/2018
Concerto Coral Sinfónico	26/05/2018	26/05/2018
Vem descobrir a cascata gigante!	01/01/2016	31/12/2016
Feliz Natal Lobo Mau	19/12/2015	08/01/2016

Fonte: Elaboração própria

No total, foram coletados dados de 3229 eventos entre 2016 e 2023. A Figura 4.20 apresenta o dendrograma obtido após o agrupamento dos dados.

Figura 4.20: Pré-classificação obtida dos dados de eventos em Matosinhos



Fonte: Elaboração própria

Os títulos únicos contabilizaram 2180 dados e o número de grupos encontrados foi de 2129, o que sugere que a maioria dos eventos não estão grandemente relacionados entre si. Porém, nota-se que uma parte significativa dos dados coletados, 1100 eventos, possuem títulos exatamente iguais a pelo menos um outro evento, e desta forma não são incluídos na fase de agrupamento, considerando a etapa de remoção de duplicatas feita anteriormente.

4.3 Análise dos resultados obtidos

Comparando os resultados com a forma de descrição dos agrupamentos dos títulos dos eventos, observou-se que os conjuntos de dados que contém mais títulos descritos como notícias, contendo verbos descritivos, apresentaram resultados menos favoráveis em comparação aos compostos apenas pelos títulos dos eventos em si. Esse comportamento pode dificultar a identificação de títulos semelhantes, introduzindo ruído no processo de agrupamento.

Apesar disso, os conjuntos obtidos abarcam uma diversidade significativa de eventos ao longo de um extenso período, englobando diferentes tipos, embora algumas fontes não apresentem dados para o período desejado completo. No entanto, ao analisar os títulos dos eventos, percebe-se que alguns podem atrair audiências mais restritas, possivelmente

introduzindo ruído em análises futuras. A ausência de informações sobre a expectativa de público nas fontes selecionadas complica uma avaliação mais precisa.

Dentro desse contexto, a aplicação da técnica de pré-agrupamento aos eventos surge como uma solução para contornar essa limitação. A premissa central é que eventos com características semelhantes geralmente atraem públicos similares, embora possam existir nuances. Assim, ao realizar análises futuras, torna-se possível derivar conclusões distintas para conjuntos diversos de eventos, resultando em uma abordagem mais refinada e flexível.

Para a análise dos resultados do pré-agrupamento, é crucial levar em conta o período de 7 anos dos quais os dados foram extraídos, sendo que em 2 desses anos ocorreu a pandemia de COVID-19, o que potencialmente influenciou na redução da quantidade de edições de eventos recorrentes. É importante destacar que o pré-agrupamento realizado requer aprofundamento, funcionando inicialmente como um guia para análises mais detalhadas no futuro.

Por fim, vale ressaltar que a estratégia empregada não enfatizou a automação dos processos, o que significa que, caso haja alterações nos códigos-fontes dos sites selecionados, serão necessárias modificações nos códigos existentes. Apesar disso, essa abordagem foi escolhida para simplificar a operação, alcançando, de forma eficaz, os objetivos estabelecidos no trabalho, que consistiam em construir um conjunto de dados abrangente contendo informações sobre eventos e feriados.

Capítulo 5

Conclusão

Apesar dos dados de feriados e eventos já terem sido utilizados em sistemas de previsão, inclusive no setor varejista, o foco deste estudo, não há uma base quantitativa sólida sobre a influência destas ocasiões nos acidentes de trabalho. Assim, tais dados representam uma oportunidade para utilização em sistemas voltados para prever acidentes ocupacionais. Diante dessa lacuna, o propósito desta pesquisa foi coletar dados para respaldar futuras investigações sobre essa correlação. Para tanto, foram desenvolvidos scripts que realizam solicitações a *Web Services* e páginas Web, interpretam suas respostas, extraem os dados desejados, armazenam as informações coletadas e então aplicam processos de agrupamento.

O conjunto de dados obtidos apresenta potencial para servir como base em uma análise aprofundada do impacto que feriados e eventos públicos podem ter na ocorrência de acidentes. A intenção é correlacionar essas informações com os incidentes que ocorrem no setor varejista, buscando compreender de que maneira a presença de feriados ou eventos públicos influencia o comportamento tanto das lojas quanto dos funcionários envolvidos. No caso dos feriados, o processo aqui descrito já foi explorado na comunidade científica por Sena et al. (2022). Para os eventos, essa correlação ainda precisa ser comprovada. Então, esses dados poderão ser utilizados como parâmetros de entrada em sistemas de previsão de acidentes ocupacionais.

É importante destacar que outros elementos devem ser considerados na análise do impacto de eventos nos acidentes, como a localização exata do evento e a determinação da distância entre as lojas do setor que estão nas proximidades. O número médio de participantes no evento é igualmente relevante. No entanto, essas informações não estão disponíveis nas fontes de dados utilizadas ou exigem processos mais sofisticados, como a aplicação de inteligência artificial generativa para interpretar as descrições dos eventos e, assim, obter uma classificação dos eventos ainda mais precisa. Essas questões representam aspectos a serem explorados em estudos futuros.

Referências

ALMEIDA, Maurício Barcellos. Uma introdução ao XML, sua utilização na Internet e alguns conceitos complementares. **Ciência da informação**, SciELO Brasil, v. 31, p. 5–13, 2002.

ANTTONEN, Matti et al. Transforming the web into a real application platform: new technologies, emerging trends and missing pieces. In: PROCEEDINGS of the 2011 ACM Symposium on Applied Computing. [S.l.: s.n.], 2011. P. 800–807.

ATTWOOD, Daryl; KHAN, Faisal; VEITCH, Brian. Occupational accident models—Where have we been and where are we going? **Journal of Loss Prevention in the Process Industries**, v. 19, n. 6, p. 664–682, 2006. ISSN 0950-4230. DOI: https://doi.org/10.1016/j.jlp.2006.02.001.

AZIR, Mohd Amir Bin Mohd; AHMAD, Kamsuriah Binti. Wrapper approaches for web data extraction: A review. In: IEEE. 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI). [S.l.: s.n.], 2017. P. 1–6.

BARBAGLIA, Guido; MURZILLI, Simone; CUDINI, Stefano. Definition of REST web services with JSON schema. **Software: Practice and Experience**, Wiley Online Library, v. 47, n. 6, p. 907–920, 2017.

BARKHORDARI, Amir; MALMIR, Behnam; MALAKOUTIKHAH, Mahdi. An analysis of individual and social factors affecting occupational accidents. **Safety and health at work**, Elsevier, v. 10, n. 2, p. 205–212, 2019.

BOZKURT, Mustafa; HARMAN, Mark; HASSOUN, Youssef et al. Testing web services: A survey. **Department of Computer Science, King´s College London, Tech. Rep. TR-10-01**, 2010.

BRAGANÇA, Camara Municipal de. **Agenda de Eventos**. Acessado por último em 26/11/2023. Nov. 2023. Disponível em: <https://www.cm-braganca.pt/visitar/agenda-de-eventos>.

BROUCKE, Seppe vanden; BAESENS, Bart. The Web Speaks HTTP. In: PRACTICAL Web Scraping for Data Science: Best Practices and Examples with Python. Berkeley, CA: Apress, 2018. P. 25–48. ISBN 978-1-4842-3582-9. DOI: 10.1007/978-1-4842-3582-9_2. Disponível em: <https://doi.org/10.1007/978-1-4842-3582-9_2>.

CAMARGO-HENRÍQUEZ, Ismael; NÚÑEZ-BERNAL, Yarisel. A Web Scraping based approach for data research through social media: An Instagram case. In: IEEE. 2022 V Congreso Internacional en Inteligencia Ambiental, Ingeniería de Software y Salud Electrónica y Móvil (AmITIC). [S.l.: s.n.], 2022. P. 1–4.

CENIĆ, Aleksandar B; STOJKOVIĆ, Suzana. A Serbian Question Answering Dataset Created by Using the Web Scraping Technique. In: IEEE. 2023 58th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST). [S.l.: s.n.], 2023. P. 147–150.

CHANDRA, R. V.; VARANASI, B. S. **Python Requests Essentials**. 1. ed. [S.l.]: Packt Publishing Ltd, 2015.

CHAPMAN, Carl; STOLEE, Kathryn T. Exploring regular expression usage and context in Python. In: PROCEEDINGS of the 25th International Symposium on Software Testing and Analysis. [S.l.: s.n.], 2016. P. 282–293.

CRASSO, Marco et al. Revising WSDL Documents: Why and How. **IEEE Internet Computing**, v. 14, n. 5, p. 48–56, 2010. DOI: 10.1109/MIC.2010.81.

DAY, Andrea J; BRASHER, Kate; BRIDGER, Robert S. Accident proneness revisited: The role of psychological stress and cognitive failure. **Accident Analysis & Prevention**, Elsevier, v. 49, p. 532–535, 2012.

DIOUF, Rabiyatou et al. Web scraping: state-of-the-art and areas of application. In: IEEE. 2019 IEEE International Conference on Big Data (Big Data). [S.l.: s.n.], 2019. P. 6040–6042.

DIZDAREVIĆ, Jasenka et al. A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 51, n. 6, p. 1–29, 2019.

DYREBORG, Johnny et al. Safety interventions for the prevention of accidents at work: A systematic review. **Campbell Systematic Reviews**, v. 18, n. 2, e1234, 2022. DOI: https://doi.org/10.1002/cl2.1234.

FELGUEIRAS, Câmara Municipal de. **Agenda de eventos**. Acessado por último em 26/11/2023. Nov. 2023. Disponível em: <https://cm-felgueiras.pt/venha-experimentar-saborear-e-sentir-felgueiras/agenda-de-eventos-municipal/>.

FIELDING, Roy; RESCHKE, Julian. **Hypertext transfer protocol (HTTP/1.1): Message syntax and routing**. [S.l.], 2014.

_____. **Hypertext transfer protocol (HTTP/1.1): Semantics and content**. [S.l.], 2014.

FIELDING, Roy T; TAYLOR, Richard N. Principled design of the modern web architecture. **ACM Transactions on Internet Technology (TOIT)**, ACM New York, NY, USA, v. 2, n. 2, p. 115–150, 2002.

FREED, Ned; BORENSTEIN, Nathaniel. **Multipurpose internet mail extensions (MIME) part two: Media types**. [S.l.], 1996.

FREGUESIA DE CHARNECA DE CAPARICA E SOBREDA, Junta de. **Eventos**. Acessado por último em 26/11/2023. Nov. 2023. Disponível em: <https://www.jf-charnecacaparica-sobreda.pt/noticias/cultura>.

FRIEDL, Jeffrey EF. **Mastering regular expressions**. [S.l.]: "O'Reilly Media, Inc.", 2006.

GLEZ-PEÑA, Daniel et al. Web scraping technologies in an API world. **Briefings in Bioinformatics**, v. 15, n. 5, p. 788–797, abr. 2013. ISSN 1467-5463. DOI: 10.1093/bib/bbt026.

GOURLEY, David et al. **HTTP: the definitive guide**. [S.l.]: O'Reilly Media, Inc., 2002.

GUIMARÃES, Câmara Municipal de. **Eventos**. Acessado por último em 26/06/2023. Nov. 2023. Disponível em: <https://www.cm-guimaraes.pt/conhecer/eventos>.

GUNAWAN, Rohmat et al. Comparison of web scraping techniques: regular expression, HTML DOM and Xpath. In: ATLANTIS PRESS. 2018 International Conference on Industrial Enterprise and System Engineering (ICoIESE 2018). [S.l.: s.n.], 2019. P. 283–287.

HALILI, Festim; RAMADANI, Erenis et al. Web services: a comparison of soap and rest services. **Modern Applied Science**, Canadian Center of Science e Education, v. 12, n. 3, p. 175, 2018.

HAN, Saram; ANDERSON, Christopher K. Web scraping for hospitality research: Overview, opportunities, and implications. **Cornell Hospitality Quarterly**, SAGE Publications Sage CA: Los Angeles, CA, v. 62, n. 1, p. 89–104, 2021.

HO, Hoang Phuong Thao. Leveraging web scraping for collecting competitive market data: Case: A case study of an Airbnb rental unit in Helsinki, 2020.

IANA. **Internet Assigned Numbers Authority**. Acessado por último em 30/06/2023. Jun. 2023. Disponível em: <https://www.iana.org/>.

ISO CENTRAL SECRETARY. **Date and time format**. en. Geneva, CH, 2016. Disponível em: <https://www.iso.org/standard/70907.html>.

JIANG, Haichen; RUAN, Jiatong; SUN, Jianmin. Application of Machine Learning Model and Hybrid Model in Retail Sales Forecast. In: 2021 IEEE 6th International Conference on Big Data Analytics (ICBDA). [S.l.: s.n.], 2021. P. 69–75. DOI: 10.1109/ICBDA51983.2021.9403224.

JØRGENSEN, Kirsten. Prevention of "simple accidents at work" with major consequences. **Safety Science**, v. 81, p. 46–58, 2016. Learning and Training in Safety and Health. ISSN 0925-7535. DOI: https://doi.org/10.1016/j.ssci.2015.01.017.

KHATTER, Harsh; SHARMA, Akshat; KUSHWAHA, Ajay Kumar et al. Web Scraping based Product Comparison Model for E-Commerce Websites. In: IEEE. 2022 IEEE International Conference on Data Science and Information System (ICDSIS). [S.l.: s.n.], 2022. P. 1–6.

KHDER, Moaiad Ahmad. Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application. **International Journal of Advances in Soft Computing & Its Applications**, v. 13, n. 3, 2021.

KIM, Kyung Woo et al. Safety Climate and Occupational Stress According to Occupational Accidents Experience and Employment Type in Shipbuilding Industry of Korea. **Safety and Health at Work**, v. 8, p. 290–295, 3 set. 2017. ISSN 20937911. DOI: 10.1016/j.shaw.2017.08.002.

KSENHUCK, Brayan C. et al. Desenvolvimento e Análise de Algoritmos Aplicados na Predição de Acidentes em Ambiente Fabril. In: ANAIS do Congresso Latino-Americano de Software Livre e Tecnologias Abertas (Latinoware 2019). [S.l.]: Sociedade Brasileira de Computação - SBC, nov. 2019. P. 22–31. DOI: 10.5753/latinoware.2019.10329.

LANTHALER, Markus; GRANITZER, Michael; GÜTL, Christian. Semantic Web services: state of the art. In: IADIS PRESS. PROCEEDINGS of the IADIS international conference-Internet technologies and society 2010. [S.l.: s.n.], 2010. P. 107–114.

LAWSON, Richard. **Web scraping with Python**. [S.l.]: Packt Publishing Ltd, 2015.

LEMOS, Angel Lagares; DANIEL, Florian; BENATALLAH, Boualem. Web service composition: a survey of techniques and tools. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 48, n. 3, p. 1–41, 2015.

LI, Fumin; ZHOU, Yisu; CAI, Tianji. Trails of data: Three cases for collecting web information for social science research. **Social Science Computer Review**, SAGE Publications Sage CA: Los Angeles, CA, v. 39, n. 5, p. 922–942, 2021.

LIE, Håkon Wium; SAARELA, Janne. Multipurpose Web publishing using HTML, XML, and CSS. **Communications of the ACM**, ACM New York, NY, USA, v. 42, n. 10, p. 95–101, 1999.

LUZ POLA, Charles da. **Aplicação de processo de classificação e técnica de bayes na base de dados de acidentes ocupacionais de uma empresa metalúrgica.** 2018. Tese (Doutorado) – Universidade de Caxias do Sul.

MANZOOR, Bilal; OTHMAN, Idris; WAHEED, Abdul. Accidental safety factors and prevention techniques for high-rise building projects – A review. **Ain Shams Engineering Journal**, v. 13, n. 5, p. 101723, 2022. ISSN 2090-4479. DOI: https://doi.org/10.1016/j.asej.2022.101723.

MARTINS, Danilo et al. Dynamic extraction of holiday data for use in a predictive model for workplace accidents. In: 2ND Symposium of Applied Science for Young Researchers. [S.l.: s.n.], 2022.

MATOSINHOS, Camara Municipal de. **Eventos**. Acessado por último em 26/11/2023. Nov. 2023. Disponível em: <https://www.cm-matosinhos.pt/servicos-municipais/comunicacao-e-imagem/eventos>.

MEULSTEE, Patrick; PECHENIZKIY, Mykola. Food Sales Prediction: "If Only It Knew What We Know". In: 2008 IEEE International Conference on Data Mining Workshops. [S.l.: s.n.], 2008. P. 134–143. DOI: 10.1109/ICDMW.2008.128.

MITCHELL, R. **Web Scraping with Python: Collecting More Data from the Modern Web**. 2. ed. [S.l.]: O'Reilly Media, Inc., 2018.

MORE, CoLAB. **iSafety**. [S.l.: s.n.], 2022. Acessado por último em 10 de Abril de 2023. Disponível em: <https://isafety.morecolab.pt>.

NIERMAN, Andrew; JAGADISH, HV. Evaluating Structural Similarity in XML Documents. In: CITESEER. WEBDB. [S.l.: s.n.], 2002. v. 2, p. 61–66.

NURSEITOV, Nurzhan et al. Comparison of JSON and XML data interchange formats: a case study. **Caine**, Citeseer, v. 9, p. 157–162, 2009.

PENG, Dongcheng et al. Research on Information Collection Method of Shipping Job Hunting Based on Web Crawler. In: 2018 Eighth International Conference on Information Science and Technology (ICIST). [S.l.: s.n.], 2018. P. 57–62. DOI: 10.1109/ICIST.2018.8426183.

PEREIRA, Francisco C; RODRIGUES, Filipe; BEN-AKIVA, Moshe. Using data from the web to predict public transport arrivals under special events scenarios. **Journal of Intelligent Transportation Systems**, Taylor & Francis, v. 19, n. 3, p. 273–288, 2015.

PERSSON, Emil. **Evaluating tools and techniques for web scraping**. 2019. Diss. (Mestrado) – KTH Royal Institute of Technology.

R, Ruchitaa Raj N; S, Nandhakumar Raj; M., Vijayalakshmi. Web Scrapping Tools and Techniques: A Brief Survey. In: 2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT). [S.l.: s.n.], 2023. P. 1–4. DOI: 10.1109/ICITIIT57246.2023.10068666.

RATHOD, Digvijaysinh. Performance evaluation of restful web services and soap/wsdl web services. **International Journal of Advanced Research in Computer Science**, v. 8, n. 7, p. 415–420, 2017.

RICHARDSON, L.; RUBY, S. **RESTful web services**. 1. ed. [S.l.]: O'Reilly Media, Inc., 2007. P. 29–43.

RODRIGUES, Filipe; MARKOU, Ioulia; PEREIRA, Francisco C. Combining time-series and textual data for taxi demand prediction in event areas: A deep learning approach. **Information Fusion**, Elsevier, v. 49, p. 120–129, 2019.

RODRIGUEZ, Juan Manuel; ZUNINO SUAREZ, Alejandro Octavio; CRASSO, Marco Patricio. An approach for web service discoverability anti-pattern detection for journal of web engineering. Rinton Press, Inc, 2013.

SANTURTÚN, Ana; SHAMAN, Jeffrey. Work accidents, climate change and COVID-19. **Science of the total environment**, Elsevier, v. 871, p. 162129, 2023.

SAPO. **Serviço de Feriados Portugueses**. Acessado por último em 26/11/2023. Nov. 2023. Disponível em: <https://www.sapo.pt/>.

SARKAR, Sobhan; MAITI, J. Machine learning in occupational accident analysis: A review using science mapping approach with citation network analysis. **Safety Science**, v. 131, p. 104900, 2020. ISSN 0925-7535. DOI: https://doi.org/10.1016/j.ssci.2020.104900. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925753520302976>.

SARKAR, Sobhan; PATEL, Atul et al. Prediction of occupational accidents using decision tree approach. In: 2016 IEEE Annual India Conference (INDICON). [S.l.: s.n.], 2016. P. 1–6. DOI: 10.1109/INDICON.2016.7838969.

SENA, Inês et al. Integrated Feature Selection and Classification Algorithm in the Prediction of Work-Related Accidents in the Retail Sector: A Comparative Study. In: SPRINGER. INTERNATIONAL Conference on Optimization, Learning Algorithms and Applications. [S.l.: s.n.], 2022. P. 187–201.

SHAHZAD, Farrukh et al. A review of latest web tools and libraries for state-of-the-art visualization. **Procedia Computer Science**, Elsevier, v. 98, p. 100–106, 2016.

SHARMA, TN; BHARDWAJ, Priyanka; BHARDWAJ, Manish. Differences between HTML and HTML 5. **International Journal Of Computational Engineering Research**, Citeseer, v. 2, n. 5, p. 1430–1437, 2012.

SRINIVASAN, Latha; TREADWELL, Jem. An overview of service-oriented architecture, web services and grid computing. **HP Software Global Business Unit**, Citeseer, v. 2, p. 1–13, 2005.

SRIVASTAVA, Shobhit; HAROON, Mohd.; BAJAJ, Abhishek. Web document information extraction using class attribute approach. In: 2013 4th International Conference on Computer and Communication Technology (ICCCT). [S.l.: s.n.], 2013. P. 17–22. DOI: 10.1109/ICCCT.2013.6749596.

TABARÉS, Raúl. HTML5 and the evolution of HTML; tracing the origins of digital platforms. **Technology in Society**, v. 65, p. 101529, 2021. ISSN 0160-791X. DOI: https://doi.org/10.1016/j.techsoc.2021.101529. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0160791X2100004X>.

TEOTIA, Harsh et al. Instagram Analysis and Activity Automation: Using Python and Selenium Automation Tools. In: 2023 International Conference on Computational Intelligence, Communication Technology and Networking (CICTN). [S.l.: s.n.], 2023. P. 522–526. DOI: 10.1109/CICTN57981.2023.10140356.

THEISEN, Kevin J. Programming languages in chemistry: a review of HTML5/JavaScript. **Journal of Cheminformatics**, Springer, v. 11, n. 1, p. 11, 2019.

THIVAHARAN, S; SRIVATSUN, G; SARATHAMBEKAI, S. A survey on python libraries used for social media content scraping. In: IEEE. 2020 International Conference on Smart Electronics and Communication (ICOSEC). [S.l.: s.n.], 2020. P. 361–366.

THOMAS, D. M.; MATHUR, S. Data Analysis by Web Scraping using Python. **3rd International Conference Electronics, Communication and Aerospace Technology (ICECA)**, p. 450–454, 2019. DOI: 10.1109/ICECA.2019.8822022.

TIHOMIROVS, Juris; GRABIS, Jānis. Comparison of soap and rest based web services using software evaluation metrics. **Information technology and management science**, v. 19, n. 1, p. 92–97, 2016.

WANG, Feng et al. Using Regression Algorithms to Forecast Merchandise Sales in the Presence of Independent Variables. In: IEEE. 2022 7th International Conference on Cyber Security and Information Engineering (ICCSIE). [S.l.: s.n.], 2022. P. 113–118.

WANG, Hongbing et al. Web services: problems and future directions. **Journal of Web Semantics**, v. 1, n. 3, p. 309–320, 2004. ISSN 1570-8268. DOI: https://doi.org/10.1016/j.websem.2004.02.001.

WHATWG, Web Hypertext Application Technology Working Group. **HTML**. Acessado por último em 30/06/2023. Jun. 2023. Disponível em: <https://html.spec.whatwg.org/multipage/>.

WORLD WIDE WEB CONSORTIUM. **"Extensible Markup Language (XML) 1.0 (Fifth Edition)"**. en. [S.l.], 2008. Disponível em: <https://www.w3.org/TR/xml>.

_____. **"Web Services Architecture"**. en. [S.l.], 2004. Disponível em: <https://www.w3.org/TR/ws-arch>.

_____. **"Web Services Description Language (WSDL) Version 2.0"**. en. [S.l.], 2007. Disponível em: <https://www.w3.org/TR/wsdl/#markup>.

ZANETTI, A. R.; CAMARGO, V. V. Web Service no ambiente escolar: um estudo de caso. **Tecnologias, Infraestrutura e Software**, v. 1, p. 35–53, 1 2012.

ZEGERS, Roland. HTTP header analysis. **University of Amsterdam Master Thesis**, 2015.

ZHAO, B. Web scraping. **Encyclopedia of Big Data**, p. 1–3, mai. 2017. DOI: 10.1007/978-3-319-32001-4_483-1.

ZISMAN, ANDREA. An overview of XML. **Computing & Control Engineering Journal**, IET, v. 11, n. 4, p. 165–167, 2000.

Emitido em 18/12/2023

CÓPIA DO TRABALHO Nº 234/2023 - DELMAX (11.57.05)

(Nº do Protocolo: NÃO PROTOCOLADO)

(Assinado digitalmente em 18/12/2023 20:43)
LEANDRO RESENDE MATTIOLI
PROFESSOR ENS BASICO TECN TECNOLOGICO
DELMAX (11.57.05)
Matrícula: ###731#3

(Assinado digitalmente em 18/12/2023 17:23)
DANILO MAGONE DUARTE MARTINS
DISCENTE
Matrícula: 2018######0

Visualize o documento original em https://sig.cefetmg.br/documentos/ informando seu número: **234**, ano: **2023**, tipo: **CÓPIA DO TRABALHO**, data de emissão: **18/12/2023** e o código de verificação: **2f7b889cd3**