Web Scraping: implementação de um modeloem R para extração de notícias jornalísticas.

Aluno: Carlos Alberto Alves de Meneses - 20180003202

**1 PROJETO**

**1.1 Introdução**

Na televisão, como em qualquer outro veículo de comunicação de massa, o jornalista busca a factualidade, ou seja, fatos que sejam importantes para serem passadas para seus telespectadores.

Nesse sentido, cabe ao produtor de notícias *“minerar”* esses fatos, localizando-os o mais rápido possível, de preferência de forma exclusiva, antes da concorrência, e enviá-los para os repórteres produzirem as matérias com o intuito de gerar índices altos de audiência.

A tarefa do produtor de notícias implica em:

* Verificar o que as empresas concorrentes estão noticiando
* Utilizar métodos de radioescuta, para obter noticias que estão sendo vinculadas nas rádios locais
* Pesquisar nas redes sociais (internet) sobre os assuntos mais comentados no momento
* Produzir as pautas do dia contendo um resumo das noticias que serão utilizadas como bases para que os repórteres preparem suas matérias

O produtor de notícias é, em geral, um profissional formado em jornalismo e faz parte de uma equipe de profissionais (chefe de redação, editores de texto, etc.) que trabalham em uma redação de jornalismo numa empresa de Televisão, por exemplo.

No entanto, nos dias atuais as notícias surgem com uma rapidez muito grande devido ao fato das pessoas possuírem equipamentos com câmeras e microfones de alta qualidade, como smartphones, que produzem dados em tempo real, gerando cotidianamente um volume de dados, principalmente nas redes sociais, gigantesco.

Com isso, faz-se necessário que a mineração de dados da WEB seja uma tarefa automatizada através da aplicação de algoritmos.

Diante disso, a proposta deste projeto é a utilização de técnicas de IA (Inteligência Artificial) e Aprendizado de Máquina a fim de facilitar a mineração desses dados e produzir relatórios para os produtores de televisão automaticamente.

**1.2 Fundamentação teórica**

Segundo Silva (2021) existem várias maneiras de extrair informações da web, sendo o uso de APIs, provavelmente, a melhor maneira de extrair dados de um website, os quais estão disponíveis no Twitter, Facebook, Google, StackOverflow, dentre outros.

Dentre as técnicas de programação de extração de dados da web, a *Web Scraping* permite obter informações de qualquer página da web por ser uma técnica que foca, principalmente, na transformação de dados não estruturados (formato HTML) da Web em dados estruturados (Banco de dados, JSON ou Planilha).

A web scraping é, portanto, uma prática de coletar dados por qualquer meio que não seja um programa interagindo com uma API (ou, obviamente, por um ser humano usando um navegador web). Isso é comumente feito escrevendo um programa automatizado que consulta um servidor web, requisita dados (em geral, na forma de HTML e de outros arquivos que compõem as páginas web) e então faz análises desses dados para extrair as informações necessárias. Na prática, a web scraping engloba uma grande variedade de técnicas de programação e de tecnologias, por exemplo, análise de dados, análise de idiomas naturais e segurança de informação.

Sendo assim, independentemente da área profissional, a web scraping quase sempre oferece uma forma de orientar práticas de negócios com mais eficiência, melhorar a produtividade ou até mesmo dar origem a uma área totalmente nova (MITCHELL, 2019).

Quanto à linguagem de programação de código aberto (open source), pode-se fazer uso do software R por ter muitas bibliotecas para executar funções de Web Scraping, tais como:

* rvest
* Rselenium

**1.3 Metodologia**

Em termos metodológicos, este projeto será desenvolvido para oferecer um guia rápido sobre técnicas e softwares de web scraping que podem ser usados para extrair dados de sites, tendo como objetivos:

Geral: implementar um modelo de web scraping para raspagem de notícias da internet.

Objetivos específicos:

* Analisar as bibliotecas da linguagem R utilizadas na web scraping;
* Elaborar e documentar o código de web scraping utilizado no modelo;
* Testar e implementar o modelo proposto.

Os dados são uma parte essencial de qualquer pesquisa, seja ela acadêmica, de marketing ou cientifica.

Para Lima (2022), uma das coisas mais importantes no Campo da Ciência de Dados é a habilidade de obter dados certos para o problema que você deseja resolver.

A **World Wide Web (WWW)** contém todos os tipos de informações de diferentes fontes. As pessoas podem querer coletar e analisar dados de vários sites.

A segui, é exibido um exemplo inicial do script para a realização da raspagem dos dados do “portalt5”. O site: <https://www.portalt5.com.br/> é um portal de noticias geral e, principalmente do estado da Paraíba.

library(rvest)  
library(shiny)  
  
# Define a função de scraping  
scrape\_news <- function() {  
 # URL do site de notícias  
 url <- "https://www.portalt5.com.br/"  
   
 # Faz o scraping dos títulos das notícias  
 webpage <- read\_html(url)  
 titles <- webpage %>% html\_nodes("a") %>% html\_text()  
   
 # Retorna os títulos  
 return(titles)  
}  
  
# Define o UI do dashboard  
ui <- fluidPage(  
 # Título  
 titlePanel("Notícias do Portalt5"),  
   
 # Texto explicativo  
 sidebarLayout(  
 sidebarPanel(  
 p("Aqui estão os títulos das últimas notícias do portalt5.")  
 ),  
   
 # Exibe os títulos das notícias em um lista  
 mainPanel(  
 verbatimTextOutput("titles")  
 )  
 )  
)  
  
# Define o server do dashboard  
server <- function(input, output) {  
 # Chama a função de scraping e exibe os títulos  
 output$titles <- renderText({  
 scrape\_news()  
 })  
}  
  
# Roda o dashboard  
shinyApp(ui = ui, server = server)

##   
## Listening on http://127.0.0.1:5876

Nessa primeira fase do projeto iremos nos concentrar na obtenção dos dados de apenas um site, neste caso do *portalt5*, deixando a implementações de outros sites de noticias para trabalhos futuros.

**1.4 Executando a raspagem da web passo a passo, usando o pacote rvest R escrito por Hadley Wickham**.

O *rvest* é uma biblioteca R muito útil que ajuda a coletar informações da páginas da web.

O primeiro passo é baixar e carregar os pacotes necessários.

# Baixando e carregando o pacote  
library( rvest )  
library( xml2 )  
library(shiny)

O pacote rvest define o link da página da web como o primeiro passo. Depois disso, os rótulos apropriados devem ser definidos. A linguagem HTML edita o conteúdo usando várias tags e seletores. Esses seletores devem ser identificados e marcados para armazenamento de seu conteúdo. Em seguida, todos os dados gravados podem ser transformados em um conjunto de dados apropriado e a análise pode ser realizada.

Coletaremos um conjunto de dados de um portal de notícias (www.portalt5.com). Este site fornece informações gerais e principalmente do estado da Paraíba.

Vamos começar a coletar informações para descobrir quais as principais noticias que estão sendo vinculadas neste portal.

* Para coletar as informações sobre as principais manchetes do portal, usaremos a URL da página de destino do site.

# Link para:  
url <- " http://portalt5.com.br "  
url

## [1] " http://portalt5.com.br "

* Agora precisamos criar variáveis NULL, porque vamos salvar a contagem de artigos para cada categoria e o nome das categorias.
* Para isso, estamos criando variáveis de categoria e contagem:

# criando variáveis NULL  
 categoria <- NULL   
 contagem <- NULL

Como mencionamos, estamos interessado em coletar dados da primeira página do site.

Agora, a parte que mais nos importa: a coleta dos dados!

O script a seguir, fornece os seguintes procedimentos: visite o URL da página da Web, coletando nós HTML usando a função *read\_html* . Para analisar nós HTML, estamos usando as regras XPath.

*Faz o scraping dos títulos das notícias* *webpage <- read\_html(url)* *titles <- webpage %>%*  \*html\_nodes(“.portalt5-destaque\_\_lista\_\_item–last”) %>% html\_text()\*

XPath lida principalmente com os nós das árvores XML 1.0 ou XML 1.1. Ë usado para representar a estrutura hierárquica de um documento XML.

XPath usa sintaxe não XML e funciona na estrutura lógica de documentos XML. XPath é projetado para ser usado embutido em uma linguagem de programação. XPaath tem sete tipos diferentes de nós: elemento, atributo, texto, namespace, instrução de processamento, comentário e nós de documento.

Para este projeto, estamos usando a função html\_nodes e definindo nossas regras XPath , que já temos, dentro da função:

html\_nodes(“.portalt5-destaque\_\_lista\_\_item–last”)

library(rvest)  
library(shiny)  
library(xml2)  
  
# Define a função de scraping  
scrape\_news <- function() {  
 # URL do site de notícias  
 url <- "https://www.portalt5.com.br"  
   
 # Faz o scraping dos títulos das notícias  
 webpage <- read\_html(url)  
 titles <- webpage %>%   
 html\_nodes(".portalt5-destaque\_\_lista\_\_item--last") %>%   
 html\_text()  
   
 # Retorna os títulos  
 return(titles)  
 #return(headline\_text)  
}

Iremos utilizar um dashboard produzido utilizando o pacote *Shiny* do R, para a apresentação do resultado da raspagem da página do site, implementando novas linhas de comando ao nosso código.

# Define o UI do dashboard  
ui <- fluidPage(  
 # Título  
 titlePanel("Notícias do PortalT5"),  
   
 # Texto explicativo  
 sidebarLayout(  
 sidebarPanel(  
 p("Aqui estão os títulos das últimas notícias do PortalT5.")  
 ),  
   
 # Exibe os títulos das notícias em um lista  
 mainPanel(  
 verbatimTextOutput("titles")  
   
 )  
 )  
)  
  
# Define o server do dashboard  
server <- function(input, output) {  
 # Chama a função de scraping e exibe os títulos  
 output$titles <- renderText({  
 scrape\_news()  
 })  
}  
  
# Roda o dashboard  
shinyApp(ui = ui, server = server)

##   
## Listening on http://127.0.0.1:3693

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Shiny é um pacote R que permite criar facilmente aplicativos da Web ricos e interativos. Shiny permite que você pegue seu trabalho em R e o exponha por meio de um navegador da Web para que qualquer pessoa possa usá-lo.

Nesse ponto, concluímos a primeira fase do nosso projeto, realizando a raspagem da manchete principal do *portalT5*.

Começaremos agora a coleta de contagens de categorias e informações sobre quantos artigos estão sendo vinculado pelo portal.

Para isso, iremos criar variáveis NULL, porque iremos salvar a contagem de artigos e o nome das categorias a qual pertencem:

**2 CRONOGRAMA DE ATIVIDADE E VIABILIDADE DE EXECUÇÃO**

**2.1.1 Cronograma de atividade**

| Atividade | Trimestre | Trimestre | Trimestre | Trimestre |
| --- | --- | --- | --- | --- |
| Pesquisa bibliográfica | X |  |  |  |
| Modelo | X | X |  |  |
| Implementação |  |  | X | X |
| Elaboração do Relatório Final | X | X | X | X |

**2.1.2 Viabilidade de execução**

Os experimentos computacionais serão realizados em um computador MacBook Pro (13-inch, Mid 2012), 2,5 GHz Intel Core i5 Dual-Corecom memória *RAM de* 16 GB 1600 MHz DDR3. Os códigos foram gerados utilizando o software open source R, version 4.2.2 (2022-10-31). R é uma linguagem de programação de código aberto e é uma das linguagens de programação preferida entre cientistas e pesquisadores de dados. R não apenas fornece algoritmos para modelos estatísticos e métodos de aprendizado de máquina, mas também fornece um ambiente de web scraping para pesquisadores. Como os programas utilizados são todos de códigos aberto, não haverá custo financeiro nesse projeto.

Uma atenção especial deve ser dado a lei de proteção de dados, realizando apenas a raspagem de sites que disponibilizam seus dados publicamente.

Os sites utilizados nesse estudo serão:

<http://www.clickpb.com.br/>,

<https://portalcorreio.com.br>/

<https://www.portalt5.com.br/>

<https://www.pbagora.com.br/>.

Para a realização desse projeto, serão necessários pesquisarmos e aprendermos sobre:

• Dados na internet;

• Utilização do XPath (Caminhos XML);

• Técnicas de raspagem da web.

**Referências**

LIMA, Acervo. **Web Scraping usando a linguagem R**. 2022. Disponível em: https://acervolima.com/web-scraping-usando-a-lingugem-r/. Acesso em: 01 abr. 2023.

MITCHELL, Ryan. Web Scraping com Python: coletando mais dados na web moderna. 2. ed. São Paulo: Oreille Novatec, 2019.

SILVA, Réulison. Web Scraping com Python: uma maneira de extrair dados da web. Uma maneira de extrair dados da web. 2021. Disponível em: <https://reulison.com.br/web-scraping-python/>. Acesso em: 28 jan. 2023.

**Anexo**

[**https://github.com/carlostvcb-ux/Web\_Scraping\_Inicio**](https://github.com/carlostvcb-ux/Web_Scraping_Inicio)