DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO/ ICEx/ UFMG

Análise e Mineração de Mídias Sociais

Prof. Fabricio Benevenuto de Souza

Trabalho Prático 1: Coleta de Dados

João Pedro Braga Ennes

Rafael Gontijo Sabino Neves

01 de Outubro de 2023

Documentação

1. Introdução

No cenário atual, onde as mídias sociais desempenham um papel crucial na vida das pessoas e na maneira como elas interagem, a coleta e análise de dados dessas plataformas se tornaram uma fonte rica de informação. Este projeto busca portanto, explorar um pequeno escopo das mídias sociais coletando dados dos fóruns públicos da plataforma Clube Ceticismo, que se dedica a promover discussões sobre uma ampla gama de tópicos, incluindo religião, história, ciência e política, representando assim, um objeto de estudo fascinante no contexto de análise de mídias sociais. Todo o código utilizado para coleta e dados obtidos estão disponíveis em https://github.com/EnnesJp/python-crawler.

1.1 Clube Ceticismo

O Clube Ceticismo (https://clubeceticismo.com.br) é uma rede social baseada em fóruns de discussão, que foi criada a partir do antigo fórum do Clube Cético, que iniciou suas atividades em 2005 e foi encerrado em janeiro de 2020. Segundo seus administradores, o Clube tem como objetivo, ser um local para “intensos e prolíficos debates sobre os mais diversos assuntos”. Tendo isso então, como principal diretriz, na rede um usuário tem duas opções iniciais de interação: acessar um fórum já existente sobre um assunto que lhe interesse e participar das discussões, ou iniciar seu novo tópico com um questionamento que ele julgue como relevante.

Além dessa interação básica, no Clube Ceticismo o usuário tem algumas opções comuns a diversas redes sociais, como adicionar outras pessoas como amigos, criar grupos privados ou mandar mensagens privadas para outros users. Temos também algumas funções conhecidas, porém com nomes diferentes, como a possibilidade de adicionar uma pessoa a sua “lista de inimigos”, que seria uma espécie de bloqueio a um outro usuário, ocultando suas interações nos fóruns. Uma informação interessante a se ressaltar também, é que para se registrar nessa rede, não é tão simples como vemos nas mais conhecidas como Facebook, Instagram, Twitter, etc.., além de dar seus dados, para que seu cadastro seja efetuado é preciso também a permissão de um dos administradores da plataforma, logo existe um controle maior do perfil de pessoas que podem ou não ter acesso ao clube.

2. Coleta dos Dados

Como foi dito anteriormente, nossa coleta é limitada aos dados de fóruns públicos, que poderiam ser acessados por qualquer visitante, mesmo não tendo uma conta na rede, visto que não obtivemos permissão de um administrador para entrar mais a fundo na plataforma. Para isso foi utilizado a técnica de web scraping e elaborado um script em Python com o auxílio da biblioteca Selenium.

2.1 Código

O código não é muito complexo e utiliza das classes css utilizadas para estilização dos componentes, para navegar pelo site como se fosse um usuário comum, funcionando da seguinte forma:

Figura 1: Página Inicial

- A partir da página inicial (figura 1), na função “visit_foruns” buscamos todos os elementos da classe “forumtitle”, que representam os links para um fórum (figura 2), e armazenamos todos os endereços em um array (forumLinks, exibido na figura 3), para acessá-los em seguida.

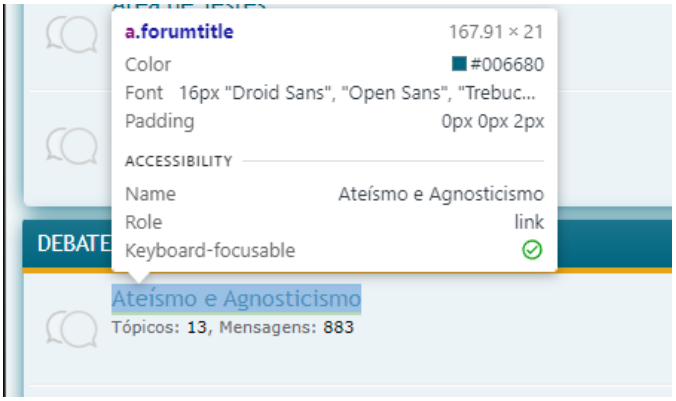

Figura 2

```python
forum_title_locator = (By.CLASS_NAME,'forumtitle')

def visit_forums():
    allForums = forumsContainer.find_elements(*forum_title_locator)
    forumTitles = [forum.text for forum in allForums]
    forumLinks = [forum.get_attribute("href") for forum in allForums]

    for index, link in enumerate(forumLinks):
        if(check_valid_link(link)):
            print('Visiting forum "', forumTitles[index], '"\n')
            driver.get(link)
            time.sleep(3)
            visit_topics()
```

Figura 3

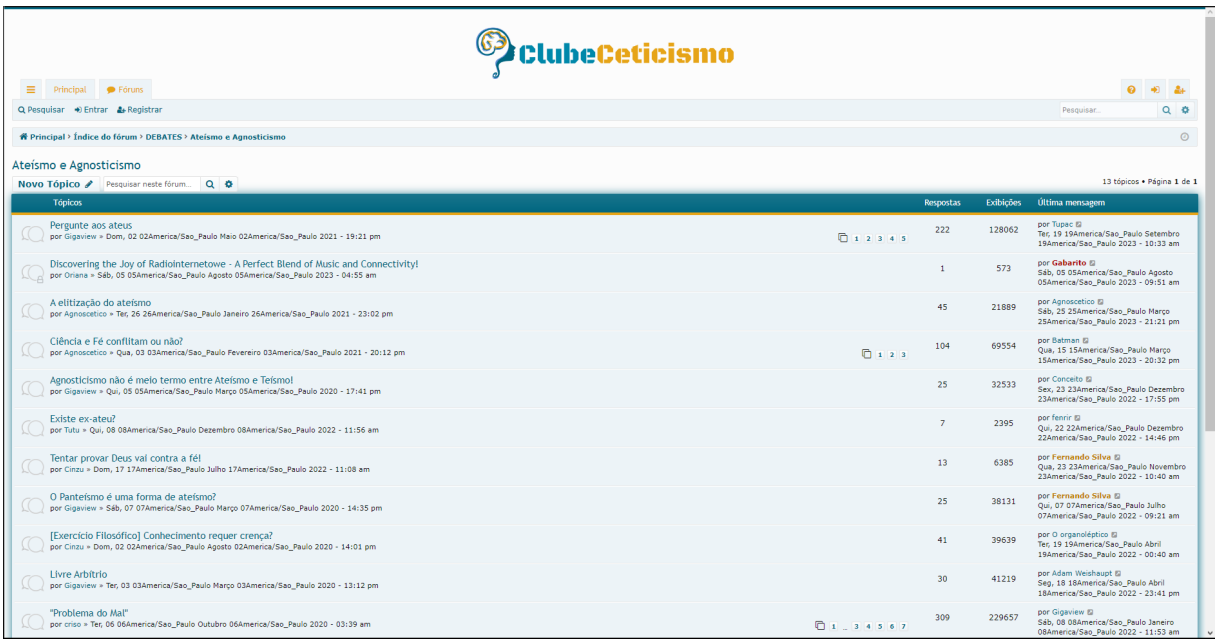

Figura 4: Fórum

- Já dentro de um fórum (figura 4), precisamos repetir o mesmo processo para armazenar os links para cada tópico de discussão na função “visit_topics”. Dessa vez, buscamos pela classe “topicTitle” (figura 5) e armazenamos os links em “topicLinks”. Além disso, buscamos o nome do fórum, a partir da classe “forum-title”, para que seja possível identificar futuramente a qual deles cada tópico pertence.

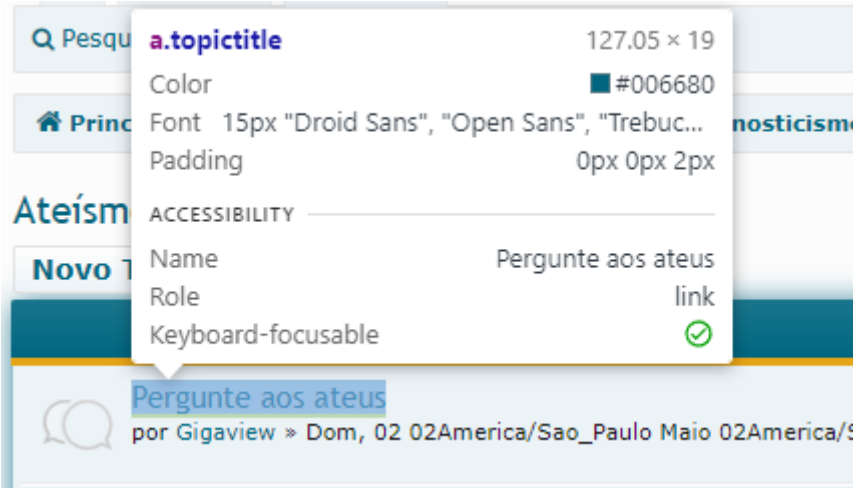

Figura 5

```
body_locator = (By.CLASS_NAME,'page-body')
next_page_locator = (By.CLASS_NAME, 'arrow.next')
topic_title_locator = (By.CLASS_NAME,'topictitle')

def visit_topics(page = 1):
  print("Getting forum page", page, "\n")

  global currentTopic
  forumsWrapper = get_wrapper()
  forumTitle = forumsWrapper.find_element(By.CLASS_NAME, 'forum-title').text
  topicsContainer = forumsWrapper.find_element(*body_locator)
  allTopics = topicsContainer.find_elements(*topic_title_locator)
  topicTitles = [topic.text for topic in allTopics]
  topicLinks = [topic.get_attribute("href") for topic in allTopics]

  nextPage = topicsContainer.find_elements(*next_page_locator)
  if nextPage:
    nextPageUrl = nextPage[0].find_element(By.TAG_NAME,'a').get_attribute("href")

  for index, link in enumerate(topicLinks):
    if(check_valid_link(link)):
      print('Visiting topic "', topicTitles[index], '"\n')
      driver.get(link)
      time.sleep(3)
      get_topics_data(forumTitle)
      currentTopic = currentTopic + 1

  if nextPage:
    driver.get(nextPageUrl)
    visit_topics(page + 1)
```

Figura 6

É importante notar também que um fórum pode ter infinitos tópicos e por isso é feita uma paginação dos resultados, logo sempre que terminamos de acessar todos os links de uma página, verificamos na tela se existe um elemento com as classes “arrow” e “next” que representam a seta para a próxima página de resultados (figura 7), e em seguida, caso exista, acessamos ela e chamamos a mesma função “visit_topics” para repetir o processo.

Figura 7

Figura 8: Tópico

- Por fim, já dentro do tópico (figura 8), buscamos as informações de cada interação que achamos relevante para o trabalho, dentre elas autor da postagem (classe “author”, tag “strong”), conteúdo (classe “content”), data (classe “time”, atributo “datetime”), link, etc. Assim como na página do fórum, temos também que navegar em toda a paginação das respostas de cada tópico, utilizando do mesmo elemento com classe “arrow” e “next” (figura 7). Ao final da função “get_topics_data”, salvamos todos os dados em um csv.

```python
def get_topics_data(forumTitle, previousIndex = 0, page = 1):
  print("Getting topic page", page, "\n")

  url = driver.current_url
  topicsWrapper = get_wrapper()

  wait = WebDriverWait(driver, timeout=60,ignored_exceptions=[NoSuchElementException])
  wait.until(lambda d : driver.find_element(By.CLASS_NAME, "page-body"))

  topicTitle = topicsWrapper.find_element(By.CLASS_NAME, 'topic-title')
  posts = topicsWrapper.find_elements(By.CLASS_NAME, 'post')

  for index, post in enumerate(posts):
    print('Getting post', index+previousIndex+1, 'data\n')

    authorElement = post.find_element(By.CLASS_NAME, 'author')
    authorNameElement = authorElement.find_element(By.TAG_NAME, 'strong')
    authorURL = authorNameElement.find_element(By.TAG_NAME, 'a').get_attribute('href')
    authorTotalMessages= post.find_element(By.CLASS_NAME,'profile-posts').find_element(By.TAG_NAME, 'a')
    postDateTime = authorElement.find_element(By.TAG_NAME, 'time').get_attribute('datetime')
    content = post.find_element(By.CLASS_NAME, 'content')

    csv_writer([
      currentTopic,
      index + previousIndex,
      topicTitle.text,
      forumTitle,
      url,
      authorNameElement.text,
      authorURL,
      authorTotalMessages.text,
      postDateTime,
      content.get_attribute('innerHTML')
    ])

  nextPage = topicsWrapper.find_elements(*next_page_locator)
  if nextPage:
    nextPage[0].click()
    get_topics_data(forumTitle, previousIndex + 50, page + 1)
```

Figura 9

3. Dados

Após rodar o script implementado, foi possível coletar dados de 30.529 postagens do Clube Ceticismo, cobrindo todos seus 15 fóruns públicos e todos os 702 tópicos que haviam sido incluídos nestes fóruns, até o dia da última coleta. No arquivo CSV gerado portanto, temos basicamente 10 colunas que representam os seguintes dados:

1. ID do tópico de discussão (Incluído por nós na coleta, para facilitar análise futura)
2. ID do post dentro de cada tópico (Incluído por nós na coleta, para facilitar análise futura)
3. Nome do tópico
4. Nome do fórum

5. Link para o tópico
6. Username do usuário que realizou o post
7. Link para o perfil do usuário que realizou o post
8. Número de interações do usuário na plataforma
9. Data e horário do post
10. Conteúdo do post

Apesar da coleta interessante que conseguimos realizar, esbarramos com algumas limitações durante o trabalho, devido a não termos conseguido uma autorização de um administrador da plataforma. Uma parte dos dados da plataforma não são disponibilizados para visitantes, tais como visualização de perfil de um usuário e grupos privados, o que acabou impossibilitando uma coleta de dados mais profunda na rede.