INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Brasília
Campus Brasília

TECNOLOGIA EM SISTEMAS PARA INTERNET

Wendell Rodrigues Feliciano
Leonardo Campos Muniz
Daniel Evangelista Pereira

RELATÓRIO DE PRÁTICA INTEGRADA
DE
CIÊNCIA DE DADOS E APRENDIZADO DE MÁQUINA

Brasília - DF

07/03/2020

Sumário

1. Objetivos

Este é o primeiro relatório de atividades da PRÁTICA INTEGRADA DE CIÊNCIA DE DADOS E APRENDIZADO DE MÁQUINA. O desafio envolve um time misto de discentes das disciplinas optativas de Introdução à Ciência de Dados - ministrada no quinto semestre - e de Aprendizado de Máquina - ministrada no quarto semestre - no âmbito do curso de Tecnologia em Sistemas para Internet do campus Brasília do Instituto Federal. Os docentes são, respectivamente, o Professor Doutor Fábio Henrique de Oliveira e o Professor Doutor Diego Queiroz.

A equipe é composta pelos alunos Wendell Rodrigues Feliciano, Leonardo Campos Muniz e Daniel Evangelista Pereira, com a mentoria de Thiago Marinho e utilizará a metodologia de desenvolvimento ágil SCRUM, com um total de 4 sprints de uma semana.

Este documento inaugural, portanto, trata do desafio proposto para a primeira semana: o SCRIPT DE COLETA. Nele descreveremos o processo utilizado pela equipe para a raspagem dos dados de um sítio na web para fundamentar as próximas etapas. Ao final, os dados minerados estarão disponíveis em um arquivo com a extensão do tipo CSV.

2. Descrição do problema

Nesta fase reuniremos fatos interessantes relacionados a OVNIs, a partir de relatos realizados nos últimos vinte anos usando o site Nuforc (www.nuforc.org).

De forma específica, nosso problema é bem descrito em seu enunciado: desenvolver um script Python para executar a coleta de dados dos vinte anos, entre setembro de 1997 e agosto de 2017. Os dados serão colocados em um DataFrame de onde serão gravados em um arquivo denominado OVNIS.csv.

3. Desenvolvimento

Durante o desenvolvimento desta etapa do projeto praticaremos nossos conhecimentos em Python. Não à toa: trata-se da melhor linguagem para prospecção e análise de dados da atualidade, com uma infinidade de bibliotecas para raspagem, análise e aplicação de inteligência artificial, dentre as quais escolhemos, graças ao vasto material de aprendizagem:

- requests: Utilizamos esta biblioteca para poder realizar requisições HTTP com o sítio web Nuforc;
- BeautifulSoup: Esta biblioteca foi utilizada para efetuar a extração de dados em arquivos HTML e XML;
- Pandas: Biblioteca esta que utilizamos para desempenhar o papel de fazer a carga de dados e a transformação da mesma em forma de tabela;

3.1 Código implementado

Para cumprir o objetivo de pegar os dados foram criado duas funções:

1. get_meses_e_anos: função responsável por gerar uma lista de string mm/aaaa com o range proposto, pois os links na primeira página possuía uma string dessa forma.

2. main: responsável pela lógica de pegar os dados, utiliza a função get_meses_e_anos para filtrar os links da primeira página. Realiza o laço de repetição pelos links da tabela da primeira página. Verifica se o texto do link está dentro da lista de filtro, caso esteja acessa o link e pega os dados da tabela da próxima página, A tabela que contém os dados que vai ser extraído é uma tabela linear não paginada, conseguindo pegar seu conteúdo com uma única requisição.

A função sleep da biblioteca time foi utilizada para não sobrecarregar o site com várias requisições seguidas e o site bloquear o link por achar que é um ataque.

```
import requests
from requests.api import head
from bs4 import BeautifulSoup
import pandas as pd
import time

def get_meses_e_anos():
    meses_anos = list()
    #laco para percorrer os anos
    for ano in range(1997, 2018):

        #laco para percorrer os meses
        for mes in range(1, 13):

            #exclui meses antes de setembro de 1997
            if ano == 1997 and mes < 9:
                continue

            #exclui meses apos agosto de 2017
            if ano == 2017 and mes > 8:
                continue

             #forma a string MM/ANO para buscar apenas pelos meses e anos solicitados
              #converte o mes para string e usa a funcao zfill da string para preencher com 0 a esquerda
            href_text = str(mes).zfill(2)+'/'+str(ano)

            meses_anos.append(href_text)
    return meses_anos


def main():
    # base_url principal
    base_url = 'http://www.nuforc.org/webreports/'

    #realiza request inicial
    req = requests.get(base_url + 'ndxevent.html')

    #pega a resposta html em form de string
```

```
        body = req.text

        #passa o corpo da pagina para o BeautifulSoup
        soup = BeautifulSoup(body, features="html.parser")

        #pega a tag tbody da tabela contida na pagina
        tbody = soup.find('tbody')
        header=True

        meses_anos = get_meses_e_anos()

        #laco que percorre os links da tabela inicial
        for link in tbody.find_all('a'):

            #filtra o link por mes/ano
            if(link.get_text().strip() not in meses_anos):
                continue

            #realiza uma nova request com o link encontrado
            data = requests.get(base_url + link.get('href'))

             #cria uma nova instancia do BeautifulSoup com o conteudo
da resposta
            table = BeautifulSoup(data.text, features="html.parser")

             #pega a tabela da pagina convertida em string e passa para
o pandas na funcao read_html
            #foi necessario instalar a biblioteca lxml
            df = pd.read_html(str(table))[0]

            print(df)

             #header verdadeirao cria o arquivo csv com o cabecalho da
tabela
            if header:
                df.to_csv('OVNIS.csv', mode='w', header=True)

                    #seta para falso para que as novas linhas sejam
inserias ao final do arquivo
                header = False

                   #sleep para nao sobrecarregar o servidor com varias
requests
```

```
                time.sleep(30)

                continue

            #demais linhas ele apenas da o apend no arquivo
            df.to_csv('OVNIS.csv', mode='a', header=False)

                #sleep para nao sobrecarregar o servidor com varias
requests
            time.sleep(30)


    if __name__ == "__main__":
        main()
```

4. Considerações Finais

A primeira dificuldade encontrada no processo de desenvolvimento foi elaborar uma lógica para pegar os dados, pois tinhas duas páginas para serem acessadas, a primeira com os links de dados separados por meses e anos, e a segunda os dados do respectivo mês e ano.

A parte de filtragem dos links acreditamos que pode ser melhorada pois ela resulta em uma lista que só vai verificar se os links estão dentro dela, não conseguimos pensar em outra forma de otimizar essa busca.

Na primeira parte percorremos a lista de links e verificamos se o texto da tag (“<a>”), está contida na lista, após essa filtragem acessamos o link e pegamos os dados que realmente queremos.

Observamos que não precisávamos manipular os dados da página seguinte, apenas obtê-los. Pesquisamos a melhor forma de fazer isso, e foi utilizando a própria função do pandas read_html, para isso pegamos a tabela inteira com o beutifulsoup e transformamos seu conteúdo em string e passamos para o pandas, como poderia ter mais de uma tabela na página e sabíamos que só existiria uma, utilizamos diretamente o índice 0 para pegarmos apenas a primeira

Uma verificação que poderia ser adicionada no futuro é, se há um conteúdo retornado pelo beautifulsoup, pois hoje procuramos uma table, pegamos seu conteúdo, transformamos em string e passamos para o pandas, porem se nao tiver nenhuma table, na página pode ocasionar um erro no futuro.

Uma boa prática que percebemos no início é que se não controlarmos os intervalos entre as requisições o site poderia bloquear as requisições por pensar que estava sofrendo um ataque, por isso utilizamos a função sleep da biblioteca time do python com um intervalos de 30 segundos entre as requisições.

Referências

THE NATIONAL UFO REPORTING CENTER: Dedicated to the Collection and Dissemination of Objective UFO Data. Nuforc, 2021. Disponível em: < http://www.nuforc.org/ >. Acesso em 05/03/2021.

PANDAS DOCUMENTATION. Pandas, 2021. Disponível em: < https://pandas.pydata.org/docs/ >. Acesso em 05/03/2021.

BEAUTIFUL SOUP DOCUMENTATION. Crummy, 2021. Disponível em: < https://www.crummy.com/software/BeautifulSoup/bs4/doc/ > Acesso em 05/03/2021.