exp2-王玉麟

基本信息

时间：2025.3.26
课程：数据分析与实践
姓名：王玉麟
实验：exp2

具体操作

1. 使用python获取第一页html内容并解码为文本串，以utf8编码格式写入page1.txt

使用bs4，将nature高级搜索后的主页解析，得到html，写入page1.txt中。

```
import requests

from bs4 import BeautifulSoup



url ="https://www.nature.com/search?q=llm&order=relevance&date_range=2023-2024"

response = requests.get(url)



if response.status_code == 200:

    html_content=response.text

    file_path="page1.txt"

    with open(file_path,"w",encoding="utf-8") as file:

        file.write(html_content)
```

2. 打开page1.txt，提取论文相关信息（标题，地址，简介，作者列表，文章类型，期刊名称，卷宗），按照期刊名进行分类

在page1.txt中先查找所有`<article>`标签的段落并进行遍历；检查html格式，在段落中逐一查找title，url，authors等信息（使用`find()`函数）；最后写入字典

```
期刊 Nature Machine Intelligence 包含 6 篇论文
期刊 Nature Communications 包含 10 篇论文
期刊 Scientific Reports 包含 12 篇论文
期刊 Nature Methods 包含 1 篇论文
期刊 npj Digital Medicine 包含 3 篇论文
期刊 Nature Medicine 包含 2 篇论文
期刊 Nature 包含 4 篇论文
期刊 Nature Human Behaviour 包含 2 篇论文
期刊 npj Precision Oncology 包含 1 篇论文
期刊 Nature Computational Science 包含 2 篇论文
期刊 BDJ Open 包含 1 篇论文
期刊 Nature Reviews Urology 包含 1 篇论文
期刊 Communications Materials 包含 1 篇论文
期刊 Humanities and Social Sciences Communications 包含 1 篇论文
期刊 npj Biodiversity 包含 1 篇论文
期刊 Eye 包含 1 篇论文
期刊 npj Computational Materials 包含 1 篇论文
```

3. 从page1.txt中提取每篇文章的链接，从链接中获取全部作者，替换journal_dict中每篇文章的作者。

从字典中获取每篇文章的url，将其与`https://www.nature.com`拼接，再次使用bs4进行解析，考虑nature网站遵循**COinS** (ContextObjects in Spans) 和 **Highwire Press** 元数据标准，使用统一的 `citation_*` 系列 meta 标签存储论文信息，故直接查找带有`citation_author`属性的`meta`元素，将其替换字典中的`authors`即可。

```
base_url="https://www.nature.com"

for journal1 in journal_dict:

    for paper in journal_dict[journal1]['papers']:

        paper_url=paper['url']

        full_url=base_url+paper_url

        response = requests.get(full_url)

        if response.status_code == 200:

            html_content=response.text
```

```
            soup=BeautifulSoup(html_content,'html.parser')

            authors_meta = soup.find_all('meta', {'name': 'citation_author'})

            if authors_meta:

                authors = [f"{meta['content'].split(', ')[1]}
{meta['content'].split(', ')[0]}"

                      for meta in authors_meta]

                paper["authors"]=authors
```

4. 将page1.txt储存至json文件中

调用 `json` 库，使用 `open()` 以及 `json.dump` 将字典转换成 `json` 文件即可.

```
import json

with open("nature_llm.json","w",encoding="utf-8") as f:

    json.dump(journal_dict, f, indent=2, ensure_ascii=False)
```

下图为对应 `json` 文件


```
{
  "Nature Machine Intelligence": {
    "journal": "Nature Machine Intelligence",
    "papers": [
      {
        "title": "LLM-based agentic systems in medicine and healthcare",
        "authors": [
          "Jianing Qiu",
          "Kyle Lam",
          "Guohao Li",
          "Amish Acharya",
          "Tien Yin Wong",
          "Ara Darzi",
          "Wu Yuan",
          "Eric J. Topol"
        ],
        "url": "/articles/s42256-024-00944-1",
        "description": "Large language model-based agentic systems can pr
        "type": "Comments & Opinion",
        "volume_page_info": "Volume: 6, P: 1418-1420"
      },
```


心得与体会

1. `find()` & `find_all()`

查找时会涉及 `class` 标签，而 `class` 为python中的关键字，故需要将 `class` 替换成 `class_`，否则报错

2. `.text` 及判空处理

如果使用 `find()` 返回查找结果，需判断结果是否为空，否则用 `find().text` 处理时 `none` 可能被 `text` 处理，进而报错

3. 遍历字典中的url

1. 首先对 `journal_dict` 遍历，得到各个期刊名字的序号
2. 再对journal_dict `[journal1]['papers']` 遍历，其中 `journal1` 是第一步中的变量，`'papers'` 为字典中储存各个期刊中的论文的key

3. 此时 `paper` (第二部中的变量)['url']，即可根据paper在journal1及journal1在journal_dict这两次遍历跑遍各个论文