

**《实验报告》**

实验题目 设计Hugging Face数据集页面遍历爬取工具

学院系别 信息学院

专业名称 人工智能

学生姓名 郑凯航

学生学号 37220222203885

任课教师 曹冬林

20 24年4月11日

# 实验目的

列举实验要达到的几个目的。

1. 遍历所有数据页面

2. 存储的数据页面个数不少于总数目的2/3

3. 结合实验1，抽取数据页面的相关信息，并记录数据的类别信息

4. 数据存为csv格式文件

信息：数据简介、数据页面链接、下载次数、数据大小、数据数目、数据大类、数据小类

存为的格式为：

“数据简介”\t”http://XXXXXX”\t100\t”100”\t1000\t”Computer Vision”\t”Image Classification”

# 实验环境

列举实验机器环境，配置环境。

os: Windows

platform Windows-10-10.0.22621-SP0

version: 10.0.22621

python: 3.8.18

requests==2.27.1

bs4==0.0.2

lxml==4.9.3

# 实验步骤

详细说明所做实验的基本实验步骤及流程图。

基本流程

**# 本地准备工作**

**## 解析大小类标签**

**# 创建相关函数**

**## 数据页处理相关函数**

**## 抽取数据所需函数**

**# 爬取页面**

**## 下载所有小类网页文件**

**## 通过小类网页文件下载数据信息**

**# 数据写入**

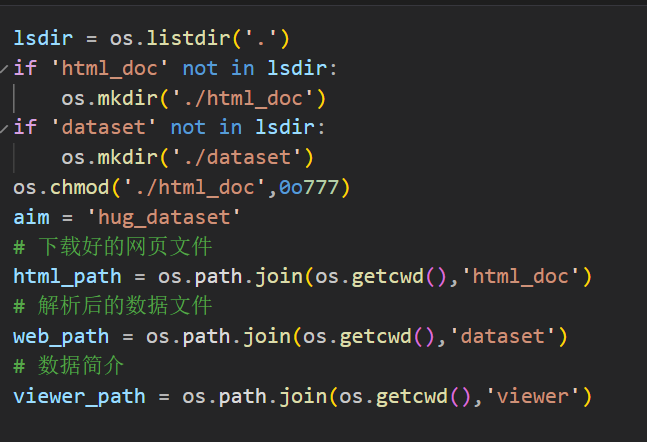
**# 本地准备工作**

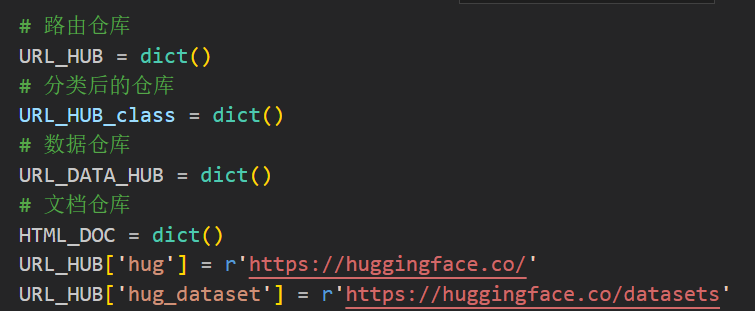
1. 创建存储相关路径

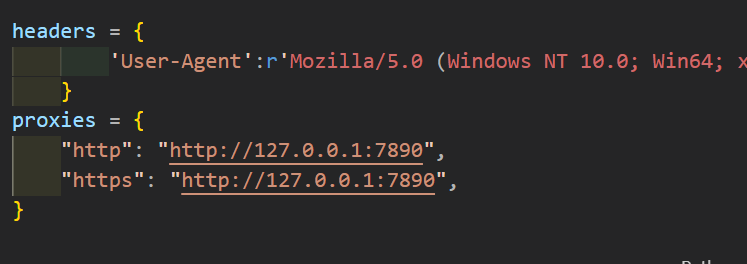
2. 记录大类，小类结构和常用url

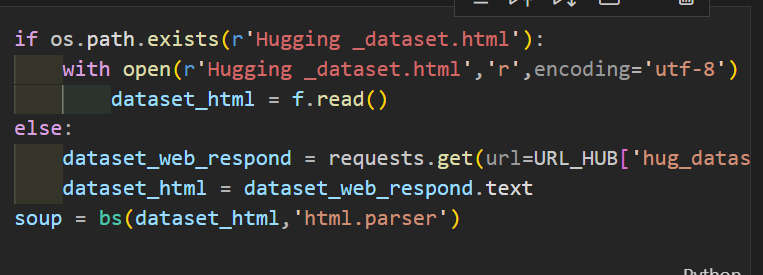
3. 配置爬虫头文件

4. 下载/读取本地数据首页



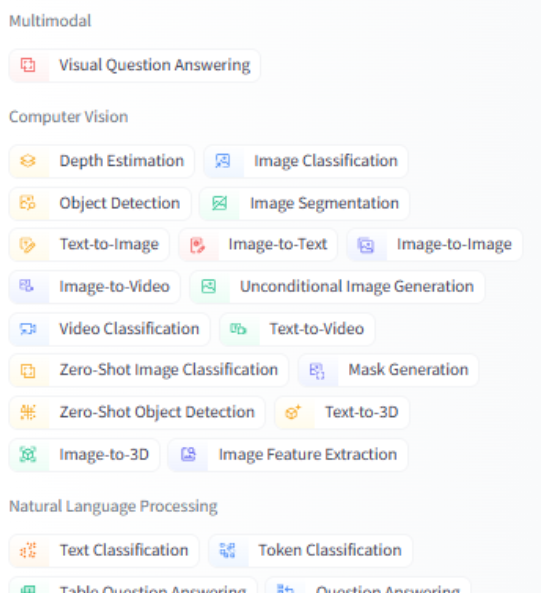






**## 解析大小类标签**

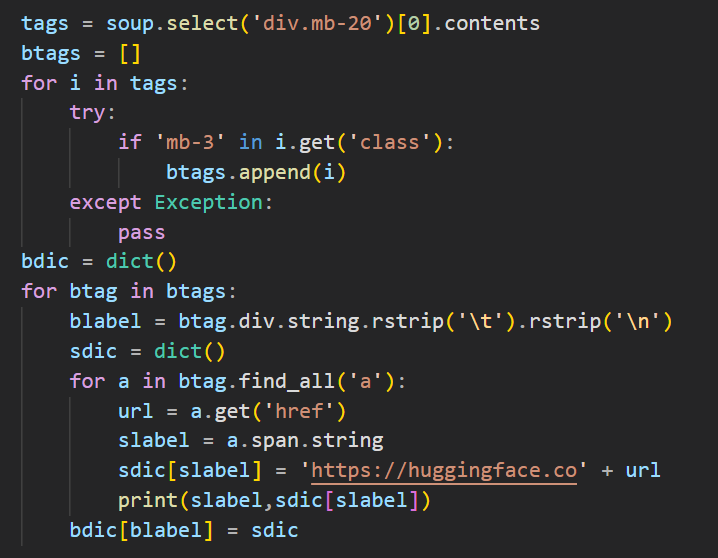
通过html文件解析找到有关大类小类标签，由于是超链接标签\<a\>所以可以读取href获得url



数据标签



小类数据页中的数据总数



**# 相关函数**

**## 数据页处理相关函数**

- find\_url

   可以找到页面中的相关数据链

- find\_cnt

   可以找到页面中的数据集总量，一页数据页中有三十个小数据集，则总量//30

   为需要查询页数

   可通过在get请求时传参来即可访问对应的所有小类数据页，其中url中加入的参数p=xxx为查找页数

    https://huggingface.co/datasets?task\_categories=task\_categories:image-classification&p=2&sort=trending

   p=2为参数

- download\_web

   下载网页，使用try高效避免下载错误

   若最终任然无法下载，则记录

   根据本地是否有相关文件来判断是否重复下载，解决记录问题

- download\_web\_page

   下载小类数据页，随后使用find\_cnt计算出当前小类的总页数，通过在请求中设置p=x访问每一页

**## 抽取数据所需函数**

- str2num

    文本形式字符串转数字，将其中的,等特殊符号去掉

- 抽取数据

    find\_page\_link

    find\_download\_time

    find\_dataset\_numb

    find\_dataset\_size

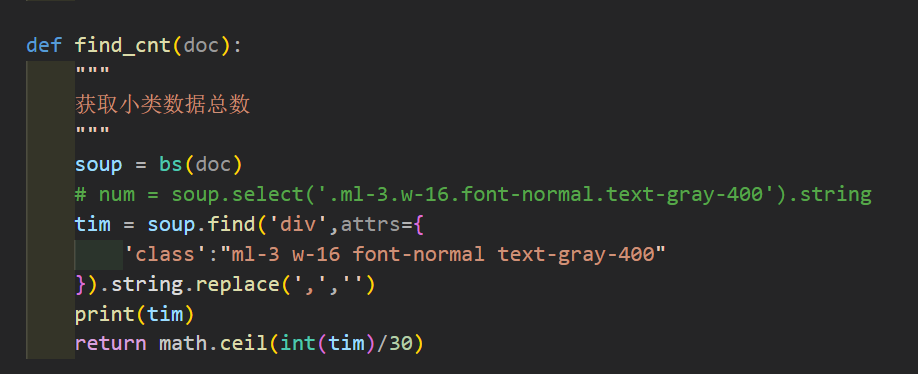
    find\_dataset\_viewer

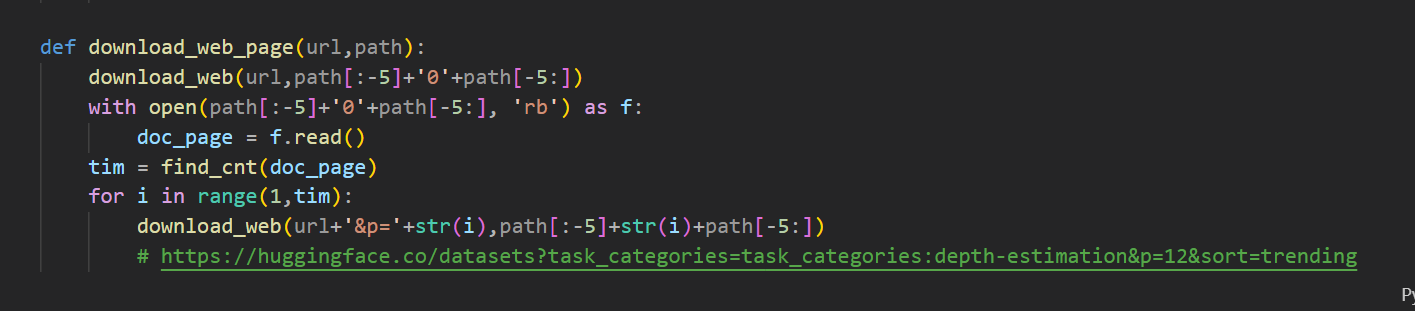
    传入文本，通过正则，解析，字符串操作等手段获取信息

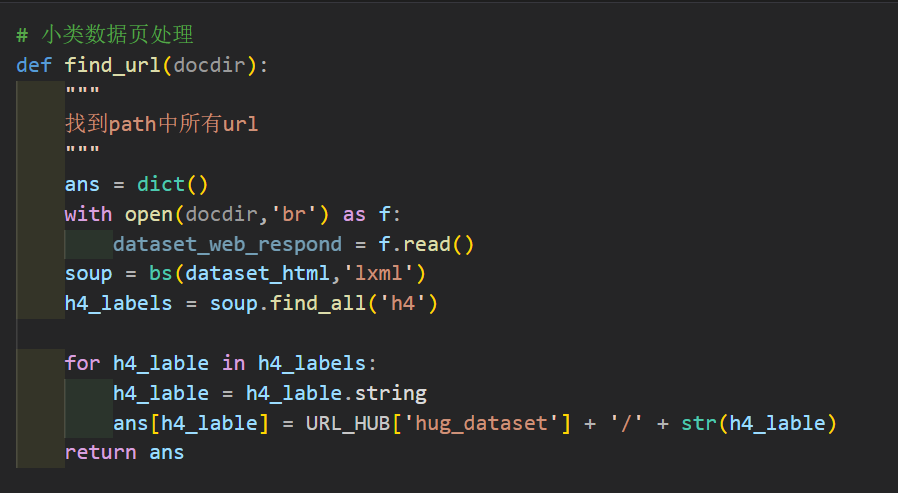
- aio

    整合前面函数，返回包含相关信息的字典

与上次实验使用函数相似











**# 爬取页面**

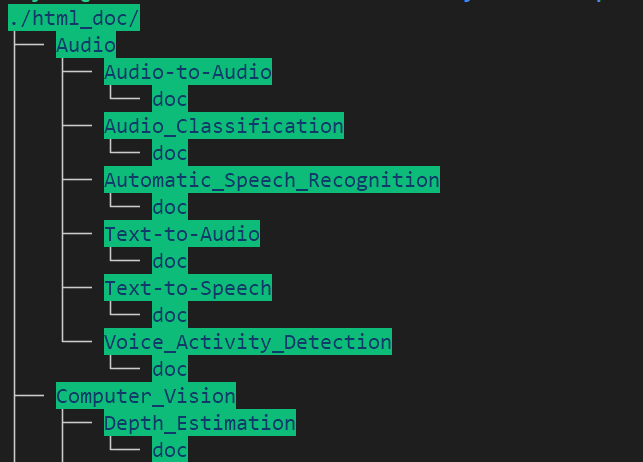
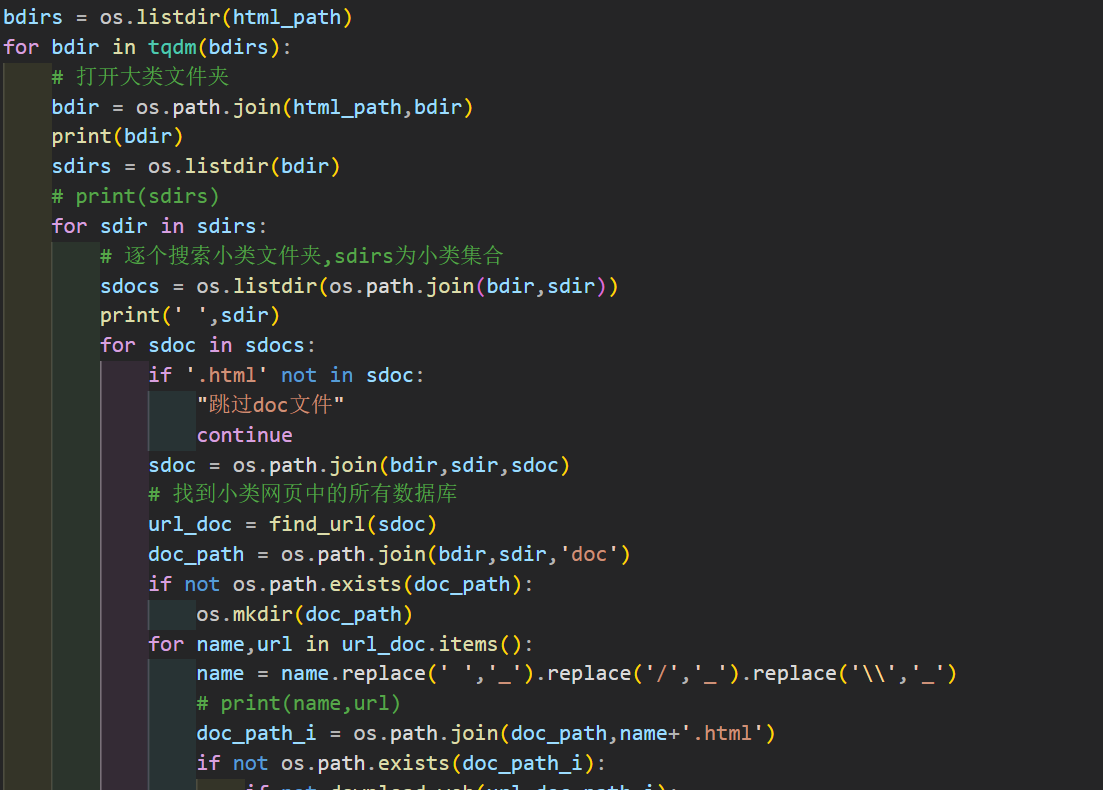
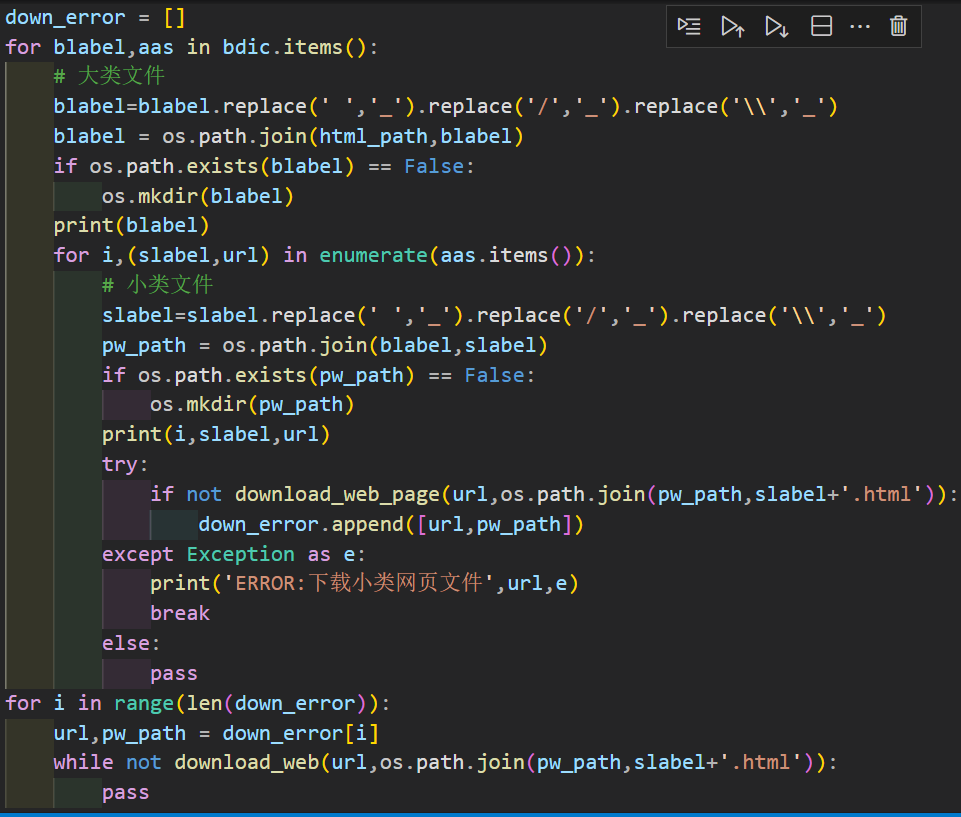
**## 下载所有小类网页文件**

在html\_doc路径下创建大类-小类目录，并在小类目录下下载小类界面文件，如https://huggingface.co/datasets?task\_categories=task\_categories:text-to-image&sort=trending

download\_web\_page可以计算小类的数据量，除以30后即为当前小类网页的张数，由p=x逐一访问

下载时记录失败的文件url,path,下载完成后重新下载失败的文件

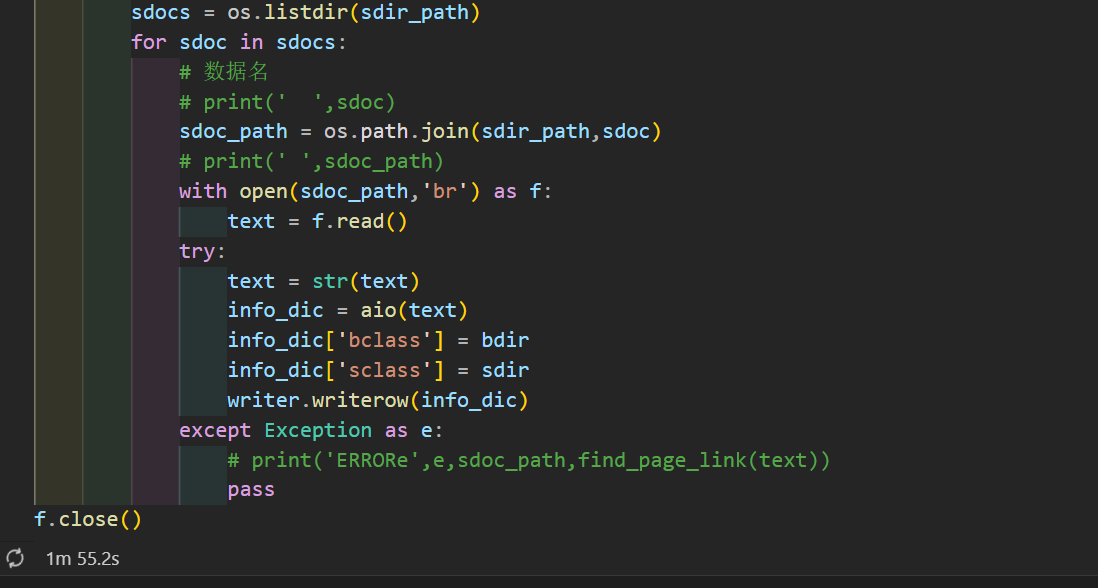
1. 按照大小类分类将网页文件下载到对应目录下



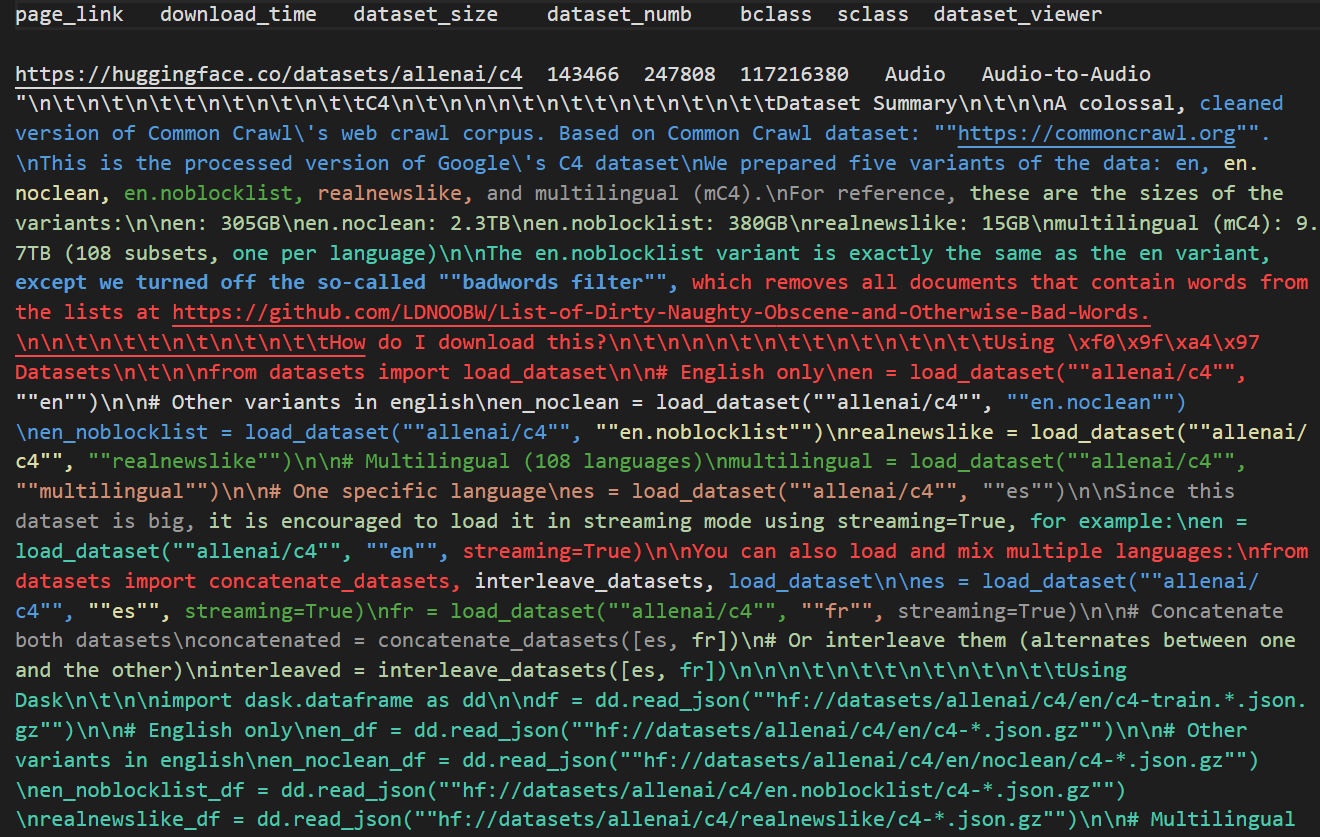
下载后的目录结构

1. 遍历所有目录，找到相关信息并写入csv文件





1. 保存后数据



# 实验分析

详细分析实验结果，包括程序中使用数据的基本情况、程序的正确性验证、实验的效果对比。

1. 使用数据的基本情况

Csv文件中的数据可以与网页上的匹配，部分网页上没有的数据以-1替代

1. 数据量

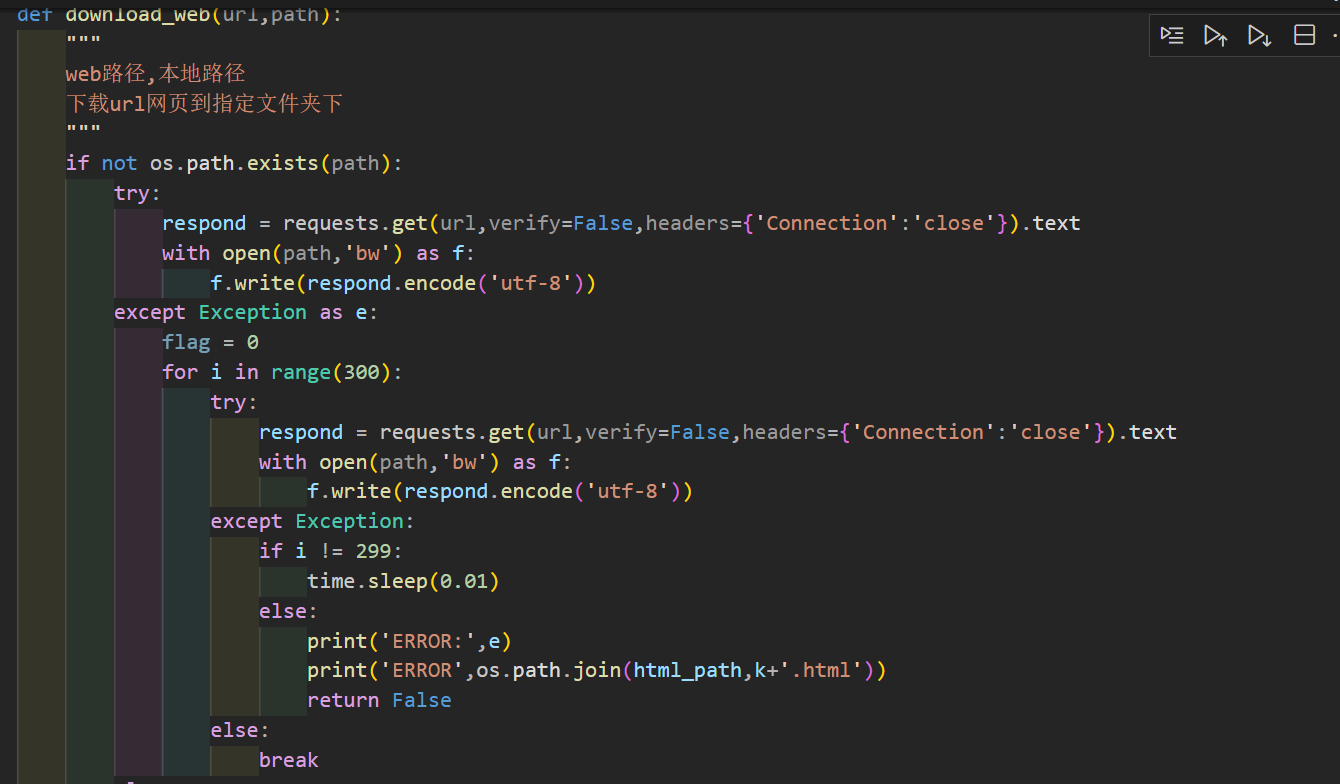
最终获取到了约两万条数据，由于是直接从大小类标签对应url获取，导致部分没有进行大小类标记的数据没有爬取，还有部分数据由于网络原因没有获得

# 问题和解决方案

详细列举实验中出现的问题以及自己所给出的解决方法。

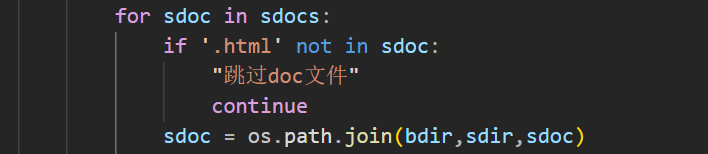
1. 数据未下载完成，但是由于一些原因导致下载中断

下载时进行保存，此后再次下载时如果此文件有记录，则直接跳过



1. 目录列出时发生错误

发现是由于添加了一些额外的目录，进行筛选



1. 部分数据抽取时发生错误

重写寻找相关信息的函数，使用一些更高效的html解析库

1. 数据简介较长，放在第一位会导致其他数据不宜观察，最后将其置后，并将其中的换行符去掉
2. 访问频率过高导致数据下载时出错，最后在访问之间设置了一些间隔
3. 部分数据没有对应信息，以-1作为标志

# 实验总结

总结实验效果，还有哪些地方可以继续改进。

1. 可以使用多进程，但是在jupyter中使用多进程时存在一些问题，后续可以将程序转移到py文件中
2. 一些信息抽取仍然存在问题，无法完全匹配所有的界面
3. 数据信息格式不统一，难以使用，可以考虑向量化等方法存储
4. 可以使用国内镜像从而提高下载速度
5. 可以考虑设置代理等方式增快访问速度