File1: githubpromax.py

import re

import scrapy

from scrapy import Request

from GithubSpiderProMax.settings import COOKIES\_DICT

class GithubpromaxSpider(scrapy.Spider):

    name = 'githubpromax'

    allowed\_domains = ['github.com']

    start\_urls = [f"https://github.com/topics/python?page={x}" for x in range(1, 51)]

    def parse(self, response):

        proj\_list = response.xpath('//article')

        for proj in proj\_list:

            basic\_info = {}

            user, project = proj.xpath('.//h3/a')

            basic\_info['username'] = user.xpath('./text()').get().strip()

            basic\_info['userurl'] = 'https://' + self.allowed\_domains[0] + user.xpath('./@href').get().strip()

            basic\_info['projname'] = project.xpath('./text()').get().strip()

            basic\_info['projurl'] = 'https://' + self.allowed\_domains[0] + project.xpath('./@href').get().strip()

            basic\_info['stars'] = int(proj.xpath('.//a/span[2]/@aria-label').get().split(' ')[0])

            about, topic, update = proj.xpath('./div[2]/div')

            emoji = about.xpath('./div/g-emoji/text()')

            if emoji:

                basic\_info['about'] = emoji.get().strip() + about.xpath('./div/text()').get().strip()

            else:

                basic\_info['about'] = about.xpath('./div/text()').get().strip()

            basic\_info['topics'] = []

            topic\_list = topic.xpath('./a')

            for topic\_basic\_info in topic\_list:

                basic\_info['topics'].append(topic\_basic\_info.xpath('./text()').get().strip())

            basic\_info['update'] = update.xpath('.//relative-time/text()').get().strip()

            yield Request(

                url=basic\_info['projurl'],

                callback=self.parse\_repo,

                cb\_kwargs=dict(basic\_info=basic\_info),

                cookies=COOKIES\_DICT

            )

    def parse\_repo(self, response, basic\_info):

        num\_str = response.xpath('.//span[@id="repo-network-counter"]/@title').get()

        forks = int(''.join(re.findall("\d+", num\_str)))

        readme = basic\_info['projurl']+'#readme'

        num\_str = response.xpath('.//a[@class="Link--primary no-underline"]/span/@title').get()

        release = int(''.join(re.findall('\d+', num\_str)))

        num\_str = response.xpath('.//a[@class="Link--primary no-underline"]/span/@title').getall()[-1]

        contributors = int(''.join(re.findall('\d+', num\_str)))

        num\_str = response.xpath('.//a[@data-pjax="#repo-content-pjax-container"]/strong/text()').get()

        branches = int(''.join(re.findall('\d+', num\_str)))

        langs = response.xpath('.//li[@class="d-inline"]/a/span[1]/text()').getall()

        percents = response.xpath('.//li[@class="d-inline"]/a/span[2]/text()').getall()

        languages = {}

        for lang, num\_str in zip(langs, percents):

            percent = round(float(''.join(re.findall('[\d\.]+', num\_str)))/100, 2)

            languages[lang] = percent

        basic\_info.update({

            'forks':forks,

            'readme':readme,

            'release':release,

            'contributors':contributors,

            'languages':languages,

            'branches':branches,

        })

        yield Request(

            url=basic\_info['projurl']+'/issues',

            callback=self.parse\_issues,

            cb\_kwargs=dict(basic\_info=basic\_info),

            cookies=COOKIES\_DICT

        )

    def parse\_issues(self, response, basic\_info):

        num\_str\_list = response.xpath('.//div[@aria-live="polite"]/a[1]/text()').getall()

        num\_str = '0'

        for num in num\_str\_list:

            if num.strip():

                num\_str = num.strip()

                break

        open\_issues = int(''.join(re.findall("\d+", num\_str)))

        num\_str\_list = response.xpath('.//div[@aria-live="polite"]/a[2]/text()').getall()

        num\_str = '0'

        for num in num\_str\_list:

            if num.strip():

                num\_str = num.strip()

                break

        closed\_issues = int(''.join(re.findall("\d+", num\_str)))

        basic\_info.update({

            'open\_issues': open\_issues,

            'closed\_issues':closed\_issues,

        })

        yield Request(

                url=basic\_info['projurl']+'/pulls',

                callback=self.parse\_pulls,

                cb\_kwargs=dict(basic\_info=basic\_info),

                cookies=COOKIES\_DICT

        )

    def parse\_pulls(self, response, basic\_info):

        num\_str\_list = response.xpath('.//div[@aria-live="polite"]/a[1]/text()').getall()

        num\_str = '0'

        for num in num\_str\_list:

            if num.strip():

                num\_str = num.strip()

                break

        open\_pulls = int(''.join(re.findall("\d+", num\_str)))

        num\_str\_list = response.xpath('.//div[@aria-live="polite"]/a[2]/text()').getall()

        num\_str = '0'

        for num in num\_str\_list:

            if num.strip():

                num\_str = num.strip()

                break

        closed\_pulls = int(''.join(re.findall("\d+", num\_str)))

        basic\_info.update({

            'open\_pulls': open\_pulls,

            'closed\_pulls':closed\_pulls,

        })

        yield Request(

            url=basic\_info['projurl']+'/network/dependencies',

            callback=self.parse\_dependencies,

            cb\_kwargs=dict(basic\_info=basic\_info),

            cookies=COOKIES\_DICT

        )

    def parse\_dependencies(self, response, basic\_info):

        dependencies = response.xpath('.//a[@data-octo-click="dep\_graph\_package"]/@href').getall()

        basic\_info.update({

            'dependencies': dependencies,

        })

        yield basic\_info

File2: pipelines.py

import pandas as pd

class GithubspiderpromaxPipeline:

    def open\_spider(self, spider):

            self.df = pd.DataFrame()

    def close\_spider(self, spider):

        self.df.sort\_values(by="stars", ascending=False, inplace=True)

        self.df.to\_csv("info7.csv")

    def process\_item(self, item, spider):

        item = pd.Series(item)

        self.df = pd.concat([self.df, item.to\_frame().T], ignore\_index=True)

        return item

File3: settings.py

BOT\_NAME = 'GithubSpiderProMax'

SPIDER\_MODULES = ['GithubSpiderProMax.spiders']

NEWSPIDER\_MODULE = 'GithubSpiderProMax.spiders'

FEED\_EXPORT\_ENCODING = 'utf-8'

def get\_cookies\_dict():

    cookies\_str = '\_octo=GH1.1.1449172623.1654507027; \_device\_id=0ef38b0fe6eed4c8d440332ab8093ebb; user\_session=F9wf3aUjGPrEEaoP8EiGoVBDCUuKcSspprcAnNsaL7hjev-P; \_\_Host-user\_session\_same\_site=F9wf3aUjGPrEEaoP8EiGoVBDCUuKcSspprcAnNsaL7hjev-P; logged\_in=yes; dotcom\_user=Awlgcu; has\_recent\_activity=1; color\_mode={"color\_mode":"auto","light\_theme":{"name":"light","color\_mode":"light"},"dark\_theme":{"name":"dark","color\_mode":"dark"}}; preferred\_color\_mode=light; tz=Asia/Shanghai; \_gh\_sess=cmp9akNSq8HNFj4rX4+BYsWXmKuzB+mbTLxXFoK/aGTSKBDRwEO7X+PL3MVU/CVXJ32fMTmwyJ+sixTR6L3dnYbZm1uE6MACo1U5hpxeSpPZFgXEKjhWztL+v6jhi33H9smdeCTOZYaOghklfL+9oV5GCrn7h5W51stLhw1Wz7SGLVRzrnBe3Mq8Z4XkOE5hQh4li+BXJ+ZJ4ZQF4vTvX1Y+5bu64LVC50SfsjMsLQxI05orIq327xfTVrVoLNZJKK+DGAs8sOL3M2SHVV1XndKrhAbKR4QOUyeFbVzYm5Q=--3eO5/45M4zl8VWkQ--q7i0awZkj5BCV5muEZ/4RA=='

    cookies\_dict = {}

    for item in cookies\_str.split(';'):

        key, value = item.split('=', maxsplit=1)

        cookies\_dict[key] = value

    return cookies\_dict

COOKIES\_DICT = get\_cookies\_dict()

USER\_AGENT = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36 Edg/106.0.1370.52'

ROBOTSTXT\_OBEY = False

FEED\_EXPORT\_ENCODING = 'utf-8'

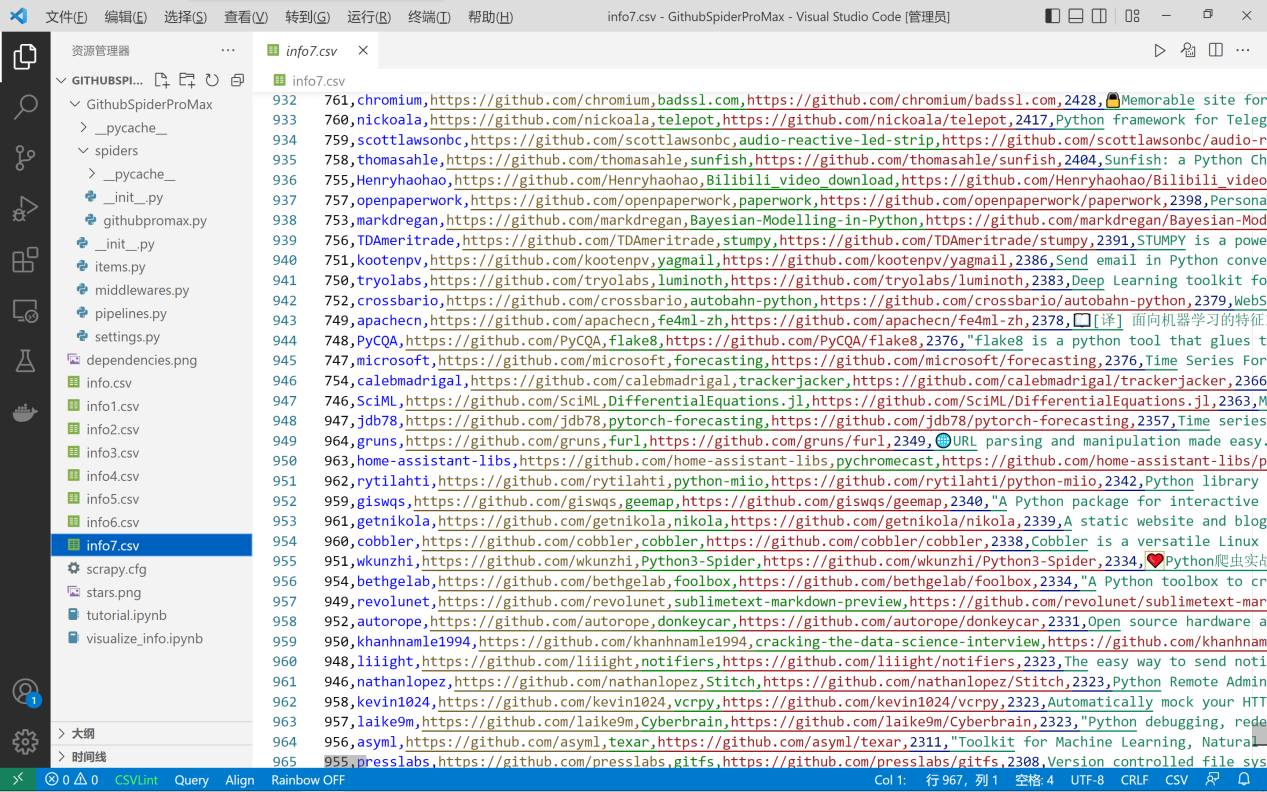
CONCURRENT\_REQUESTS = 32

DOWNLOAD\_DELAY = 0.2

ITEM\_PIPELINES = {

   'GithubSpiderProMax.pipelines.GithubspiderpromaxPipeline': 300,

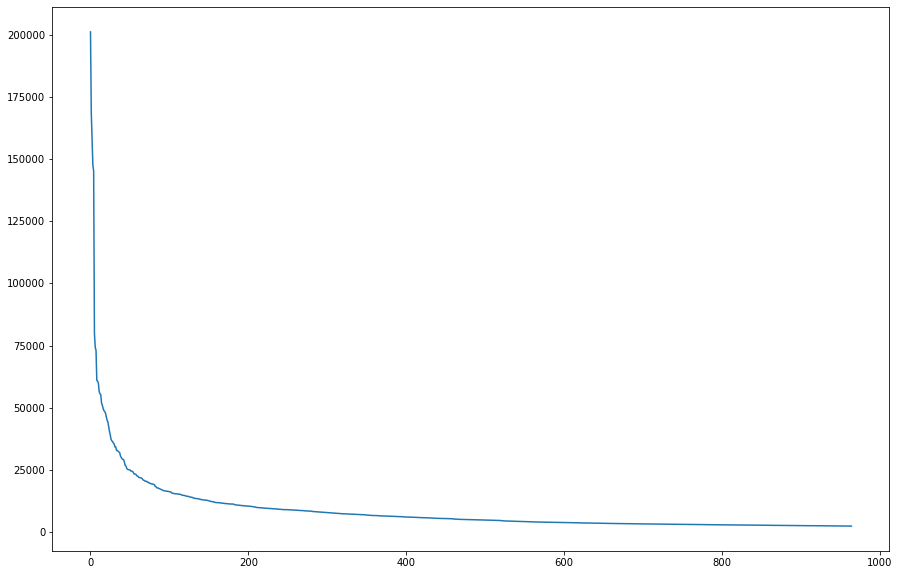
}

**数据保存成CSV文件，爬到了965条数据**

**可视化数据代码**



**星标排名可视化**



**项目依赖关系可视化**

