大作业：豆瓣电影TOP250

1. 数据爬取



如代码所示，本次爬虫框架采用scrapy。在定义爬虫时，分别定义了start\_requests，parse，parse\_detail三个函数，分别起了定义爬取页面，默认数据处理方法和详情页数据处理办法的作用。

对于start\_requests，通过观察豆瓣电影对于TOP250的链接设计，发现其十页，每页25部电影的链接只有在start=部分呈现从0到225的递增，故采用手动拼接URL的方式并配合yield关键字实现生成器从而逐层对是个页面提取。

对于parse，这是scrapy默认生成数据处理方法。当没有指定爬取页面的处理方式时，默认采用此方法完成。定义此方法时，着重注意start\_page页面的处理，由于其中的电影名，电影评分，电影评价或金句在HTML源代码中有固定的标签，故直接用css方法读取出数据，而制作国家或地区在初始页面无法有效爬取，故用yeild关键字提交对详情页的提取，并指定用parse\_detail方法读取数据。

对于parse\_detail，阅读详情页源代码时，发现国家和地区和其他部分如导演部分所用标签一致，故将整个详细数据部分读出，而后采用正则化表达式指定读入国家和地区部分。最后，依次对字符串分割，去除头尾无效占位符，存入数据。



上述代码描述了如何将前述读取出的数据存入SQlite数据库。

首先是该类的构造函数，他首先创建了数据库文件douban.db，而后创建表格DOUBAN\_TOP250并定义表头，为后续数据填入创造条件。

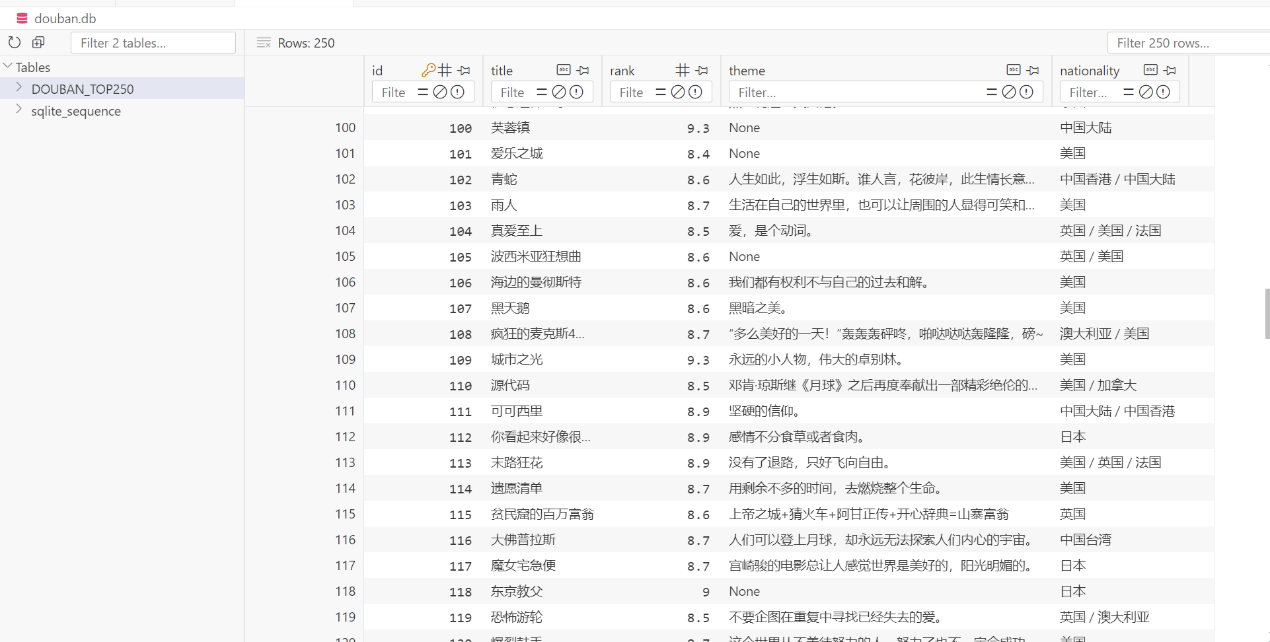
然后来到close\_spider函数，该函数被说明在scrapy官方文档中，是在爬虫停止运行时调用。它主要实现了数据库信息提交和数据库关闭两个功能。

最后是process\_item方法，该方法在每一次爬取数据完成时都会进行调用。该函数便只完成数据填入即可。

完成代码书写后，在终端中输入scrapy crawl the\_spider运行爬虫。

1. 数据绘图

在拿到数据前，需要思考这250部电影的什么数据用于作图是更有价值，最终我选择了制作国家和地区这一数据，希望能够分析出评分最高的250部电影的制作国家和地区的某些规律。

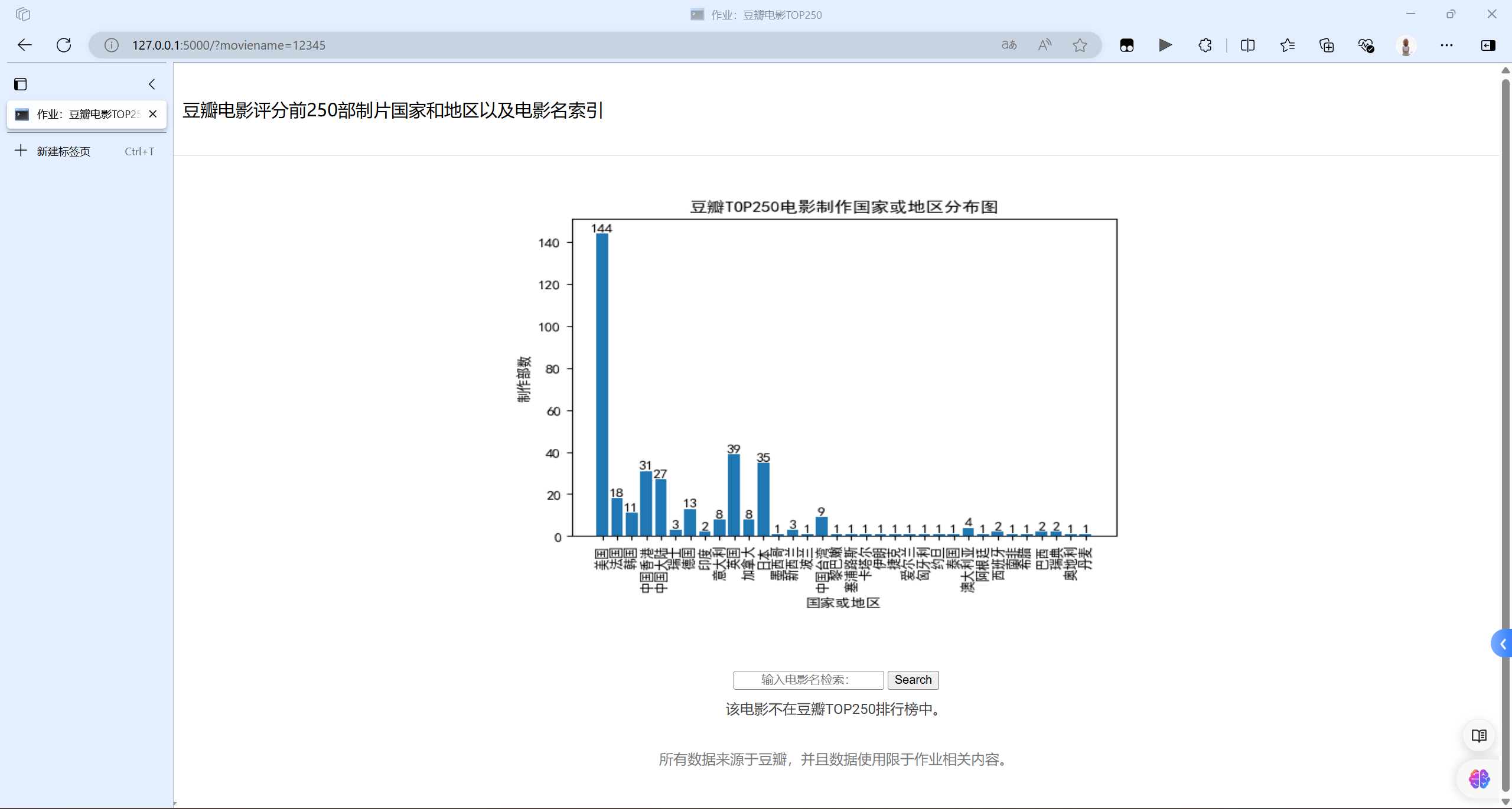


观察数据表，我们可以发现许多电影的而制作国家和地区可能不只有一个，所以在处理时决定将每一个出现的国家或地区均计入统计，从而观察优秀的电影大多的来源。



在制图时，采用matplotlib.pyplot库完成绘图和最后的web展示。首先，从数据库中读取出所有数据，然后将其中国家和地区部分作为字符串拆分并存入后续作图用字典。

最后，对图表，x标签，y标签命名，调整x标签显示方式，在将其转换为png格式存入graph变量中，为后续web展示做准备，图像如图。

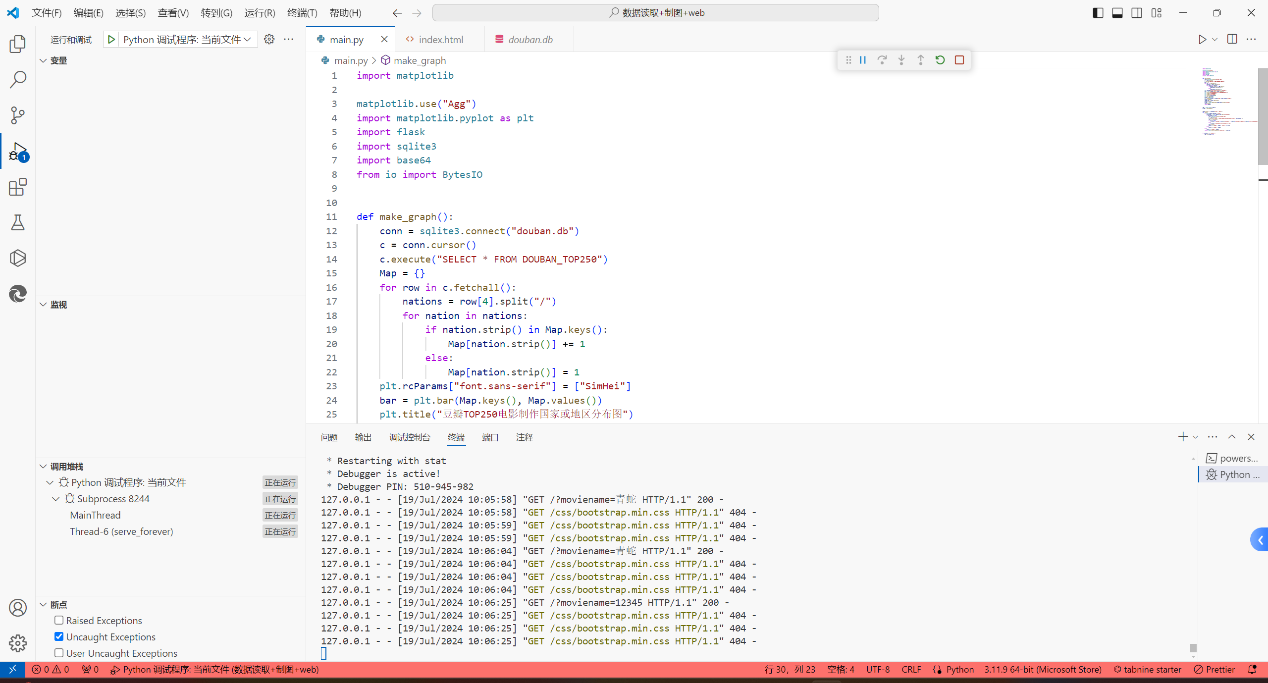


1. web设计

在设计web时遇到本次项目设计最为困难的部分，HTML源代码的复杂性很高，并且设计时也有css文件要处理，故选择了较为简单的方法，在线上设计模板，下载后根据实际情况更改部分变量和大小，使其可以作文flask的模板HTML使用。



本次设计的web网页较为简单，仅有标题，图像，搜索框，搜索结果和页脚简单声明。其中的图象就是前述的国家和地区统计表，而搜索框可以通过电影名检索数据并展示在搜索框下方，若无法检索成功，则输出“该电影不在豆瓣TOP250排行榜中。”网页运行后台如图。

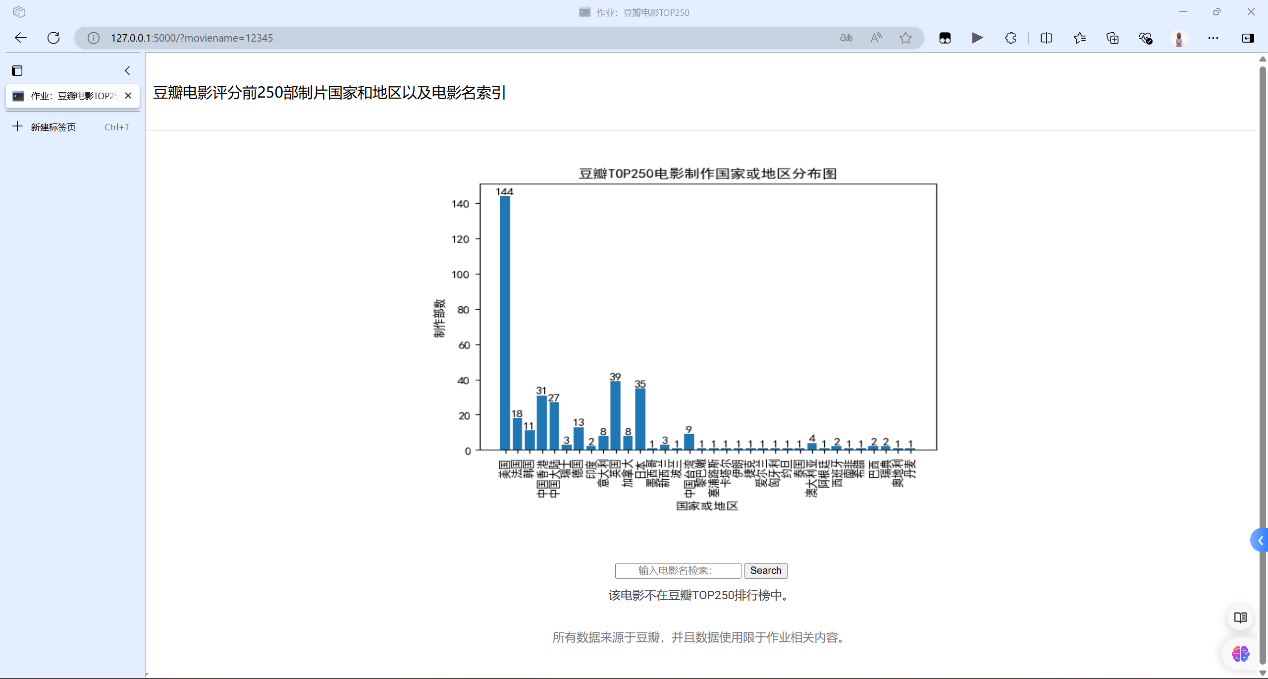


下面所示代码主要实现对模板的调用，变量填入和对请求的响应。提交请求为GET方法，读取到内容后，检索数据库，从而完成输出。

完成代码后，直接运行main.py后访问<http://127.0.0.1:5000/>即可成功。



1. 结果展示



上下两图分别展示了搜索电影是否存在的显示情况，具有一定的美观和实用性。

