# MLB Search Engine

计 86　周恩贤　2018011438

2019 年 9 月 8 日

## 1　程序介绍

本程序是基于 `django` 的美国职棒大联盟（MLB）搜索引擎，从自由时报中爬取了 **20000** 篇新闻作为数据库，允许使用者搜索相关新闻并查阅相关内容。同时，本程序为每个球队做了一个介绍页，并加入了球队热搜榜，统计出了每个球队的相关新闻篇数。在实现方面，前端使用了 `CSS`、`jquery` 等进行美化，而后端使用了 `jieba` 分词、`TF-IDF` 算法等统计新闻匹配程度。欲使用此程序，请确保安装了 $django(version \geq 2.2)$ 以及 $python(version \geq 3.5.6)$，并于 $SportsNews\backslash SportsNewsSearchEngine$ 目录下输入 $python\ manage.py\ runserver$（端口号），即可使用浏览器进行相关操作。

## 2　界面介绍

### 2.1　首页

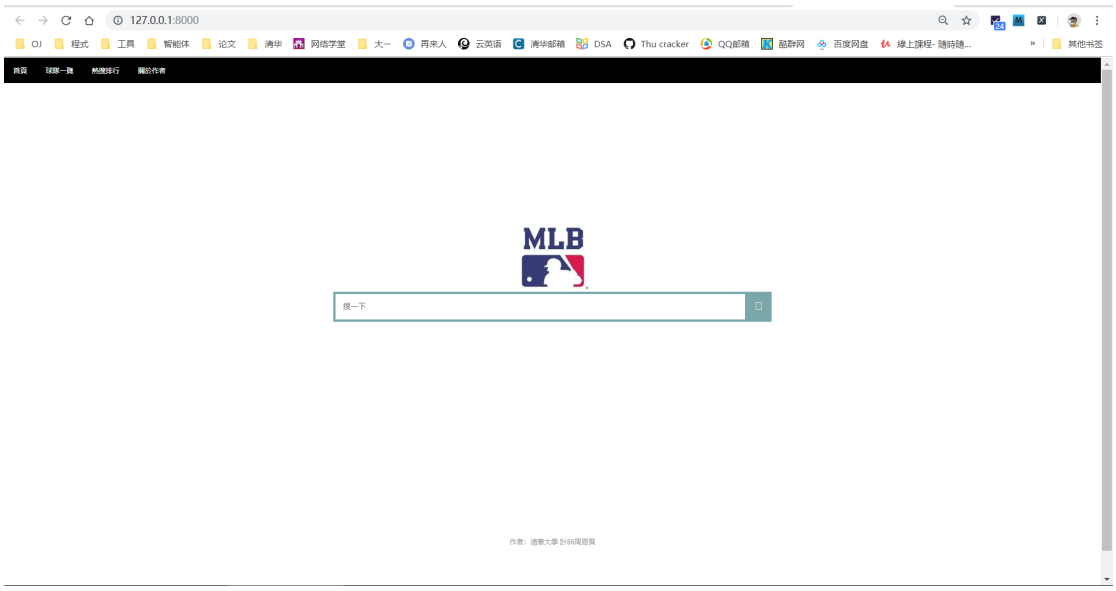


图 1: 搜索引擎的首页

图 1 是本搜索引擎的首页，利用正则表达式匹配 '$'(除 IP 以及端口号外没有任何后缀的网址)。网页上方有工具栏，中间有个搜索框，最下方显示了作者。

## 2.2 工具栏


图 2: 工具栏

图 2 是浮动在页面上方的工具栏，点击" 首页" 即可跳转到节 2.1 的页面。

### 2.2.1 球队一览


图 3: 球队一览表，网址为 */teamlist*

图 3 是点击工具栏" 球队一览" 后所跳转到的页面，显示出 MLB 三十个球队。用户可点击超连接跳转至图 10 的球队介绍页面。

### 2.2.2 热搜排行表


图 4: 热搜排行表，网址为 */rank*

图 5 是点击工具栏" 热搜排行" 后所跳转的页面，显示每个球队在数据库中的相关新闻数量以及排行。用户可点击超连接跳转至图 10 的球队介绍页面。

### 2.2.3 关于作者


图 5: 个人介绍页，网址为 $/zex$

图 5 是点击工具栏" 关于作者" 后所跳转的页面，这是我预习 WEB 编程所编写的个人介绍页，之后会再继续美化维护。

# 3 功能介绍

## 3.1 搜索功能

用户可在图 1 或图 6 的搜索框输入关键词句，并点击右侧绿色小按钮进行新闻搜索。

### 3.1.1 搜索结果页

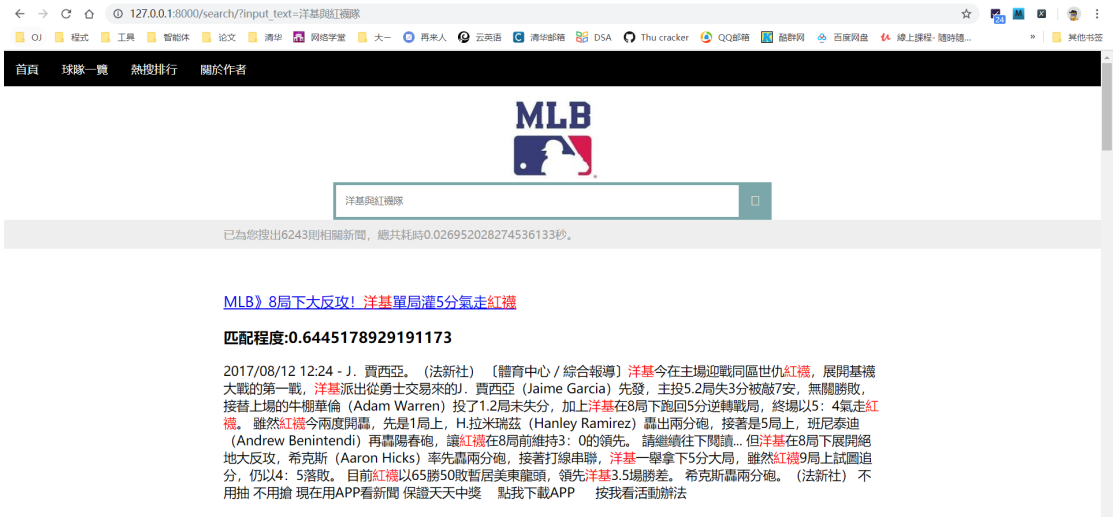


图 6: 搜索结果页，网址为 $/search/?input\_text = xxx$

图 6 为用户搜索" 洋基与红袜队" 后点击按钮后所跳转到的搜索结果页。前端利用表单设置跳转网址，后端**利用了 GET 方法获取了用户输入的搜索词句**并进行搜索。搜索出的相关新闻依序显示于页面中标题加入了超连接并高亮显示关键词汇。若没有相关新闻则会输出" 已无更多搜索结果" 的提示。

### 3.1.2 分页展示

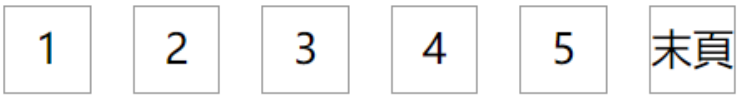


图 7: 页码框，点击以跳转至不同页

图 7 为每个搜索结果页面底下的页码框，当前页面仅显示 10 条新闻，用户点击数字 $i$ 后可跳转至 $/search/?input\_text = xxx\&page\_num = i$ 的搜索结果页，后端并利用 GET 方法获取目标页码以展示正确页面。

### 3.1.3 多词汇搜索

本搜索引擎支持多词汇的搜索。多词汇搜索是指当用户输入多关键词或长句时，会先利用节 4.1的技术进行分词，再按照节 4.3 所计算出的匹配程度排序新闻。

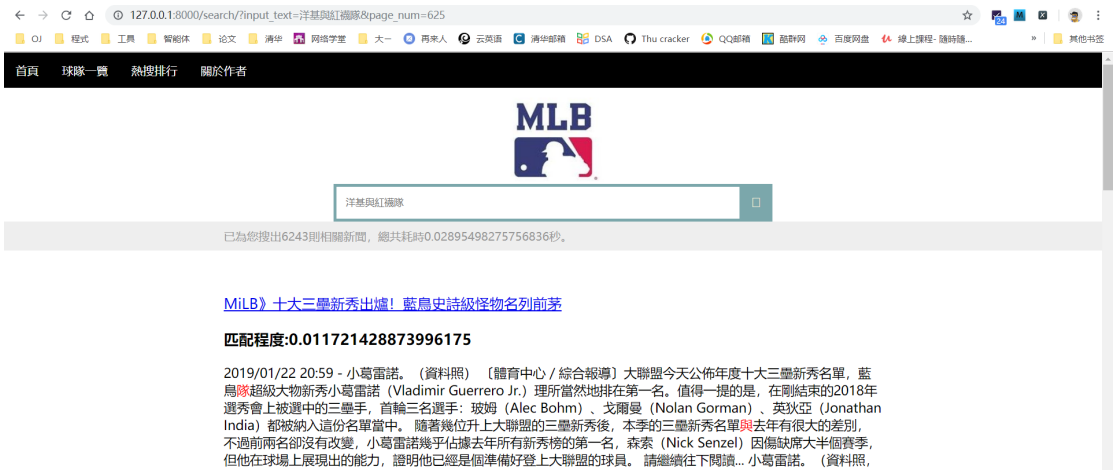


图 8: 匹配程度不高的例子

图 8 为同样搜索" 洋基与红袜队" 的最后一页搜索结果页。相比于图 **??** ，可看出不仅在关键词的数量上两篇新闻有所落差，且洋基、红袜等词汇相比于队这个词汇更重要，因而一个在首页，一个在末页。

## 3.2 新闻页面

当用户点击图 6 的新闻标题超连接后，会跳转到该新闻页面。


图 9: 新闻页面，网址为 $/news/xxx$

图 9 为新闻页面的展示。显示出了新闻标题、新闻时间、新闻内文。内文中关于球队名、球员名等关键字进行了超连结，点击后会跳转至节3.3 的球队介绍页

## 3.3 球队介绍页

当用户点击图 5 或图 9 的超连接后，会跳转相应的球队介绍页。


图 10: 球队介绍页上半部份，网址为 $/team/xxx$

图 10 为球队介绍页的上半部份的展示，显示队名字、创立时间、地点及简介。

图 11 为球队介绍页的下半部份，显示出了该球队的球员列表以及相关新闻。


图 11: 球队介绍页下半部

# 4 信息检索相关技术与实现

## 4.1 分词

利用 `jieba` 库进行分词，具体内容写于 *views.py* 中

## 4.2 倒排索引

将分词、`TF-IDF` 的结果加入全局变量 $InvertedIndexMap$ 中，一个从 $string \to set$ 的 `dict`，透过词可查询所有包含该词的新闻编号集合

## 4.3 TF-IDF 值

利用 `sklearn` 库进行 `TF-IDF` 的计算,具体内容写于 $tfidf.py$ 中,结果存于 $tfidf.txt$、$tfidf1.txt$ 以及 $tfidf2.txt$

# 5 爬虫相关技术与实现

利用 `scrapy` 进行新闻爬取以及球队爬取，具体内容写于 $getNews$ 目录中，结果储存于 $data.json$ 以及 $team.json$

# 6 性能统计

## 6.1 总新闻量

本次作业从自由时报 MLB 专栏中 (https://news.ltn.com.tw/topic/MLB) 提取,总共提取 23035 条。为方便后续匹配以及管理，我使用了**前两万条**作为新闻库。

## 6.2 查询时间

平均查询时间落在 **0.01 - 0.02 秒**之间。

## 6.3 精确率和召回率

依照用户输入的关键词，我们可以将每一条新闻分为两类：正类（与关键词相关）以及负类（与关键词无关），而搜索结果也有两种可能。因此对于每条新闻有四种情况：相关且有搜索到 (TP)、不相关且没被搜索到 (TN)、相关但没被搜索到 (FN)、不相关但被搜索到 (FP)。而在信息检索领域有两个关键指标：

- 精确率 (precision)$= \frac{\text{检索出的相关信息量}}{\text{检索出的信息总量}} = \frac{TP}{TP+FP}$
- 召回率 (recall) $= \frac{\text{检索出的相关信息量}}{\text{系统中的相关信息总量}} = \frac{TP}{TP+FN}$

为了加快搜索速度，我采取的搜索方式是：预处理将每篇新闻标题、内容分词并计算完 TF-IDF 值，**把所有值非 0 的关键词以及其子词加入倒排索引**，因此搜索出的新闻必定相关，在单个词语的查询情况下，$precision \approx 100\%$ 。

但鱼与熊掌不可兼得，此方法的缺点会遗漏掉部份因为 TF-IDF 值过小而被筛漏的新闻。以搜索" 红袜" 为例，直接在搜索框中搜出来的结果为 3002 条，但实际包含" 红袜" 的总新闻共 3460 条，可说明$recall \leq \frac{3002}{3460} = 86\%$ 。

高级搜索时，因为长句子分词时会存在歧义，指标会再低一些；且每个新闻没有一个确定的指标，比较难去判断与计算。但至少在我多次反复试验下，**单词搜索能准确且涵盖多数相关新闻，高级搜索能按匹配程度降序排序**，已符合简易搜索引擎的特性。

# 7 心得

第三周学习的内容为主要为 *python* 以及 *django* 编程，但要完成一个大作业要学习的内容就更多了：Web 前端部份，除了课上的基础内容 HTML 与 CSS 外，为了实现与交互功能，我还尝试着**学习 JS 及 jquery** 以完整完善用户交互功能；Web 后端部份，除了掌握 MVT 思想并学习正则表达式匹配 URL 外，更要完整理解 **GET 方法与 POST 方法**以完善搜索以及分页功能；python 的部份除了基本语法外，更要**查询并学习 scrapy、jieba、sklearn** 等第三方库的用法以完善整个爬虫与信息检索的功能。因此，这个大作业是我三周投入最多的一个，但也是收获最多的！

这周最震撼的部份就是了解到了 python 的强大~~（还有题目整整一页的斗地主考试题）~~。相对前两周使用 C++ 写 Qt，这周的 python 方便、简洁、第三方库多且实用（短短二

十行的 scrapy 就能完成爬虫任务!)，也难怪现在 python 如此火热且广泛应用于各领域中。

不过很可惜因为时间缘故，我还没来得及完善整个程序，包含但不限于动态爬虫功能、利用 bootstrap 美化页面、学习信息检索的机器学习算法..... 但这周的学习~~(爆肝)~~让我掌握了基本 Web Programming 以及信息检索的基础，相信到未来我能继续精进!

奋战一星期，画个 DMFB；奋战一星期，写个国际象棋；奋战一星期，搭个搜索引擎。在三周的奋斗下小学期终于结束了！小学期结束了，但我不会忘记这三周以来坚持刻苦的学习过程！~~(毕竟马上开学了)~~ 再次感谢助教与老师这几周来的指导 >O<