# 项目技术说明书

豆瓣电影数据可视化

小组名称： 第五组

学员姓名： 肖子立

目录

[项目技术说明书 1](#_Toc17289747)

[1. 项目介绍 2](#_Toc17289748)

[2. 需求分析 2](#_Toc17289749)

[3. 技术选型 3](#_Toc17289750)

[4. 爬虫开发和优化 3](#_Toc17289751)

[5. 数据清洗和存储 4](#_Toc17289752)

[6. 数据查询和可视化展示 5](#_Toc17289753)

[7. 开发环境 5](#_Toc17289754)

[8. 产品原型 6](#_Toc17289755)

# 项目介绍

豆瓣电影数据可视化项目通过对大量电影关键数据进行收集与分析，利用可视化的方式为用户呈现了各种统计结果。

# 需求分析

进行数据可视化需要提前准备一个MongoDB数据库以及爬虫脚本，通过爬取大量数据，为最终的统计结果的呈现进行数据支撑。

# 技术选型

Node.js：前端应用开发方法

Cheerio：实现HTML数据解析

MongoDB：对需要的数据进行存储

eCharts：数据可视化展示

superagent：客户端请求代理模块（支持HTTPS）

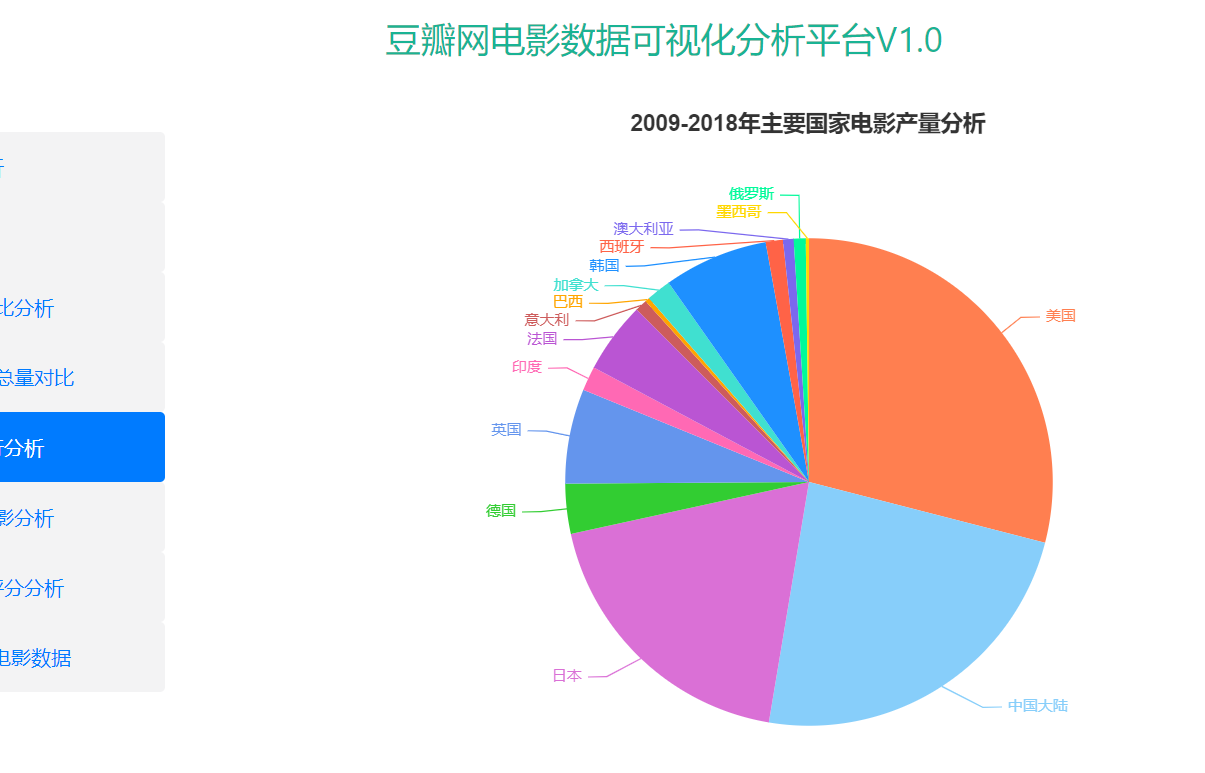
# 爬虫开发和优化

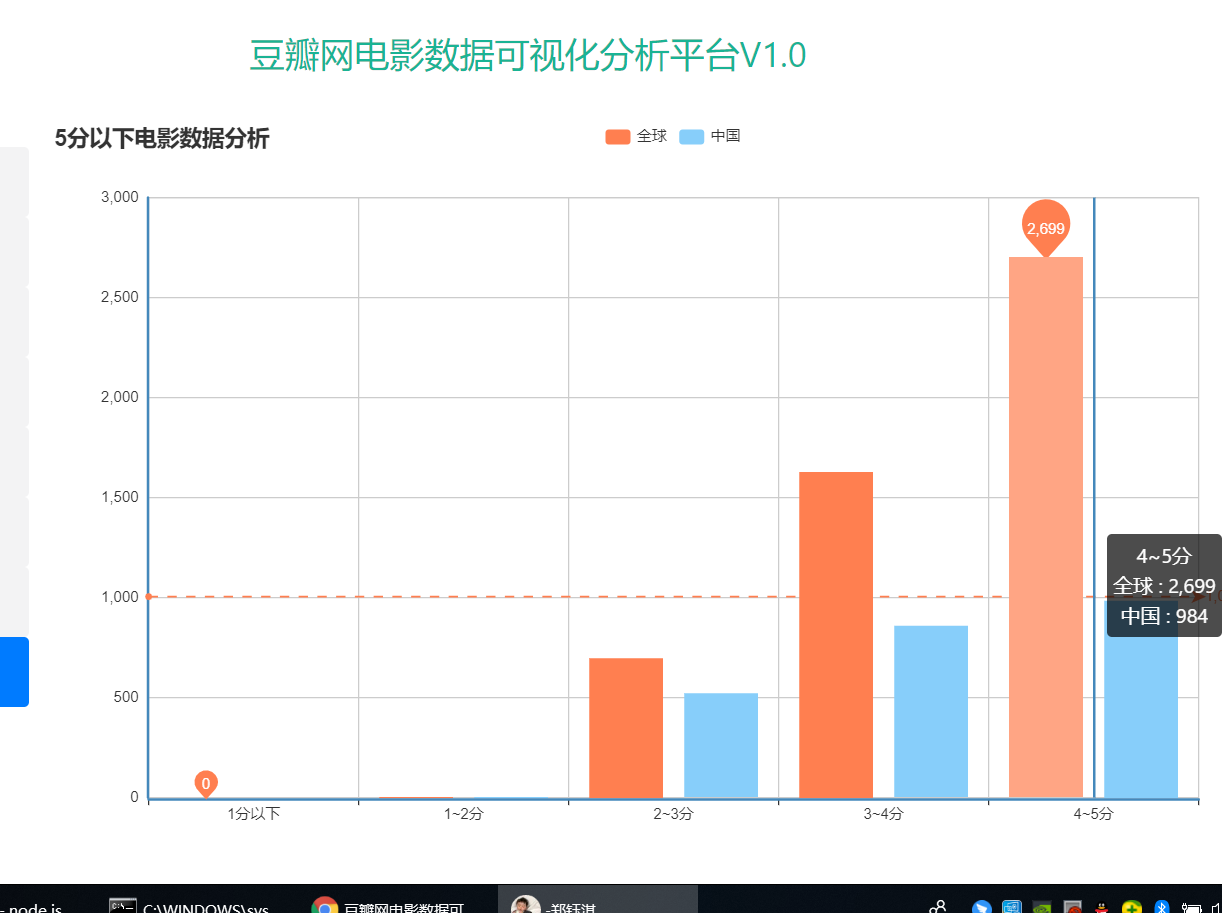
1. 连接数据库并建立model
2. 模拟豆瓣电影加载更多按钮发送请求获取一组（20个）电影对应的url
3. 遍历获取的一组url，根据url发送请求获取html
4. 使用cheerio解析HTML并配合正则获取所需数据
5. 判断所得数据在数据库里是否已存在，是则不进行存储，否则存储数据
6. 重复2-5直至所有电影数据均被爬取

# 数据清洗和存储

1. **const** mongoose = require('mongoose');
2. **const** dblink = 'mongodb://localhost/doubanmovies';
3. mongoose.connect(dblink, { useNewUrlParser: **true** });
4. **const** Movie = mongoose.model(`movie${year % 100}`, **new** mongoose.Schema({
5. name: String,
6. country: String,
7. rate: Number,
8. year: Number
9. }));
10. let item = {};
11. let html = await superagent.get(i.url).set('Cookie', myCookie).set('User-Agent', userag[Math.floor(Math.random() \* userag.length) % userag.length]).set('Host', 'movie.douban.com');
12. let $ = cheerio.load(html.text);
13. let reg1 = /制片国家\/地区:\s([A-z\u4e00-\u9fa5]+(\s\/\s[A-z\u4e00-\u9fa5]+)\*)/m;
14. let reg2 = /\((\d\*)\)/g;
15. item.name = $("#content h1 span[property]").text();
16. item.country = reg1.exec($("#info").text())[1];
17. item.rate = parseFloat($("#interest\_sectl .rating\_self strong.rating\_num").text());
18. item.year = parseInt(reg2.exec($("#content h1 span.year").text())[1]);
19. let needWrite = await check(Movie, item.name);
20. **if** (needWrite) {
21. Movie(item).save(**function** (err) {
22. **if** (err) {
23. console.log(err.message + "url:" + i.url);
24. } **else** {
25. saved++;
26. }
27. });
28. }
29. **function** check(Movie, name) {
30. **return** **new** Promise((res, rej) => {
31. Movie.find({ name: name }, **function** (err, docs) {
32. docs.length ? res(**false**) : res(**true**);
33. });
34. });
35. }

# 数据查询和可视化展示





# 开发环境

IDE：VS Code ：免费，插件丰富，调试方便

# 产品原型

|  |  |
| --- | --- |
| 豆瓣电影数据可视化分析平台V1.0 | |
| 2015-2016两年电影数据的爬取 |  |
|  |
|  |
|  |
|  |
|  |
|  |
| 自己自由发挥 |