# 电影信息智能推荐平台的构建

**作者**：韦沛浪，郎书君，杨昊严，黄旺辉，杨磊，冯永

**单位**：重庆大学，计算机学院

**案例版权**：该案例归重庆大学计算机学院所有

**涉及的知识点**：网络爬虫、Hadoop 、Spark、推荐系统、SVD算法、ALS算法

**案例来源及案例真实性情况**：该案例来源于重庆大学计算机学院专业硕士（电子信息）课程《大数据架构与技术》中的学生精选汇编课程设计。

**摘要** 随着互联网和大数据技术的发展，以用户偏好为目标的推荐系统逐步应用至各大型网站中。目前，推荐系统在很多领域得到了成功应用，包括电子商务（如Amazon､eBay､Netflix､阿里巴巴等）､信息检索（如iGoogle､MyYahoo､百度等）､社交网络（Face-book､Twitter､腾讯等）､位置服务（如Foursquare､Yelp､大众点评等）､新闻推送（如GoogleNews､GroupLens､今日头条等）等各个领域。本案例选取电影信息智能推荐为主题，使用爬取于IMDb网站上的电影信息构建模拟数据集，结合数据挖掘、机器学习等相关方法进行电影个性化智能推荐。本案例核心内容有：（1）使用Hadoop 和Spark搭建服务平台；（2）使用网络爬虫获取电影数据并进行存储和分析；（3）使用热门推荐算法SVD算法以及ALS算法进行电影个性化智能推荐。

**关键词**：推荐系统，网络爬虫，SVD算法，ALS算法

## 1 引言

该教学案例来源于重庆大学计算机学院专业硕士（电子信息）课程《大数据架构与技术》中的学生精选汇编课程设计。该案例的关键问题为互联网推荐系统的构建，以搭建电影信息智能推荐平台为具体问题，需引导学生进行的主要内容有：（1）掌握并使用Hadoop 和Spark搭建服务平台；（2）掌握并使用网络爬虫获取电影数据并进行存储和分析；（3）掌握热门推荐算法SVD算法以及ALS算法并应用，进行电影个性化智能推荐。

## 2 背景介绍

近年来，随着云计算､大数据､物联网等技术的迅猛发展，互联网空间中各类应用的层出不穷引发了数据规模的爆炸式增长。大数据中蕴含着丰富的价值与巨大的潜力，将给人类社会带来变革性的发展，但同时也带来了严重的“信息过载”问题，如何快速有效地从纷繁复杂的数据中获取有价值的信息成为当前大数据发展的关键难题。推荐系统作为解决“信息过载”问题的有效方法，已经成为学术界和工业界的关注热点并得到了广泛应用，形成了众多相关研究成果。推荐系统根据用户需求､兴趣等，通过推荐算法从海量数据中挖掘出用户感兴趣的项目（如信息､服务､物品等），并将结果以个性化列表的形式推荐给用户。目前，推荐系统在很多领域得到了成功应用，包括电子商务（如Amazon､eBay､Netflix､阿里巴巴等）､信息检索（如iGoogle､MyYahoo､百度等）､社交网络（Face-book､Twitter､腾讯等）､位置服务（如Foursquare､Yelp､大众点评等）､新闻推送（如GoogleNews､GroupLens､今日头条等）等各个领域。

推荐系统发展迅速且应用十分广泛，无论是工业界还是学术界均仍需大量的相关人才，因此对专业硕士进行推荐系统设计的指导是有必要的。在本案例中，我们选取互联网推荐系统的构建作为主要研究问题，借助大数据分析与处理、数据挖掘、机器学习的相关方法，以满足培养具备创新能力与实践能力的人才的需求。

## 3 内容

该案例的主要内容主要分为三个小节，分别为服务平台的搭建、网络电影数据的爬取与存储以及电影个性化智能推荐的实现。

3.1 服务平台的搭建

（1）平台参数

本次课程实验使用两台腾讯云服务器完成Hadoop & Spark分布式环境搭建，最终搭建的相关详细环境如下：

* 操作系统：centos 7.6.64
* 图形界面：GNOME
* 语言环境：Python 3.6.8
* 相关软件：Hadoop 2.8.5、Spark 3.2.0等

表1：服务器的内网IP&公网IP

|  |  |  |
| --- | --- | --- |
| **主机名** | **内网IP** | **外网IP** |
| **master** | 10.0.12.15 | 101.35.3.201 |
| **slave01** | 10.0.12.7 | 101.34.233.213 |

（2）集群配置

首先使用Xshell 7连接云服务器，为每台机器创建Hadoop用户，同时给予root相同的权限，为master生成密钥，并修改权限。然后分别配置yum源、Java环境，并且安装python环境以及Hadoop和spark。

将一台云服务器作为主节点，另外一台云服务器作为子节点。此处将master作为主节点，slave01作为子节点，在master中修改hosts文件，将内网IP和子节点的外网IP填写其中。同样在子节点中修改hosts文件，编辑内网IP和主节点的外网IP并保存。

随后将master主机的id\_rsa.pub（公钥）传输到slave01子节点，并在子节点中加入验证，完成主节点到子节点的免密工作，实现单方向的免密登录。为了实验的便利，进行了逆向操作，实现子节点到主节点的免密登录。

在主节点中，将主节点作为NameNode，子节点作为DataNode，修改节点的slaves文件、core-site.xml文件、hdfs-site.xml文件、mapred-site.xml。template文件、yarn-site.xml文件，将配置完成的Hadoop文件通过scp命令传递给子节点，最后进行启动测试。

Spark集群配置相比Hadoop集群配置而言较为简单，仅需在主节点配置slaves文件、spark-env.sh文件，将Spark配置文件复制到子节点中。子节点中需要删除原来的Spark文件，然后将主节点传输过来的文件解压，并将spark文件权限拥有者设为hadoop用户。

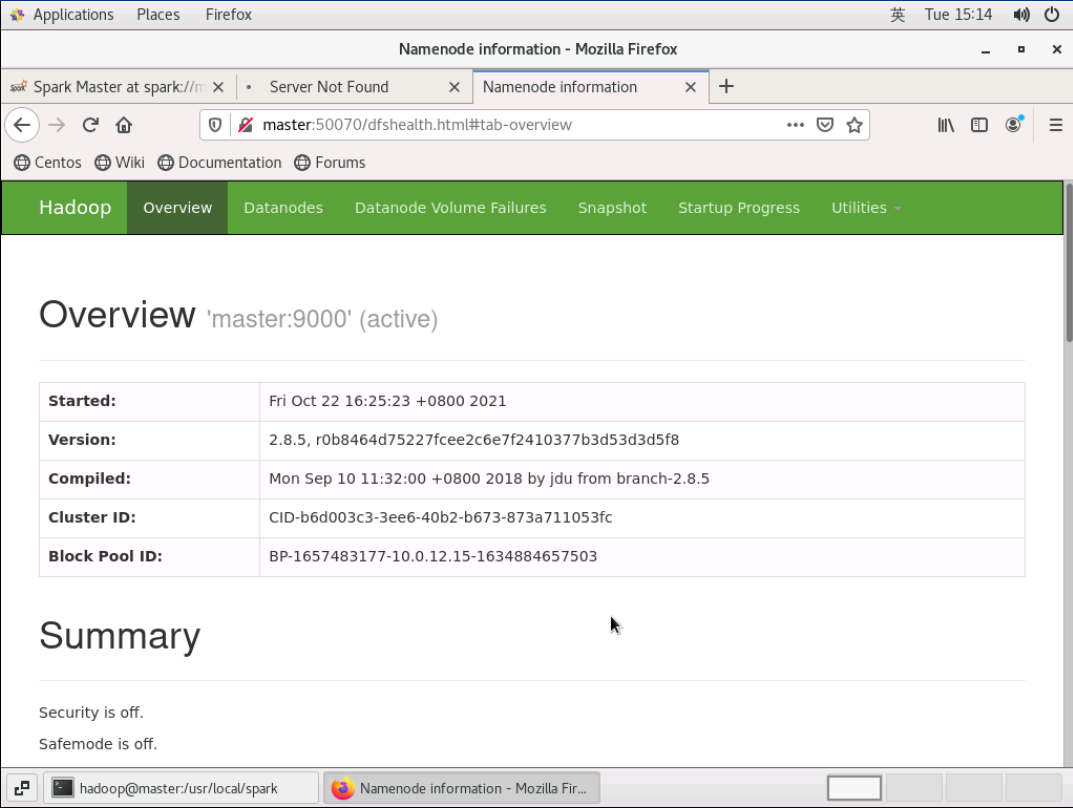


图1 Hadoop集群启动

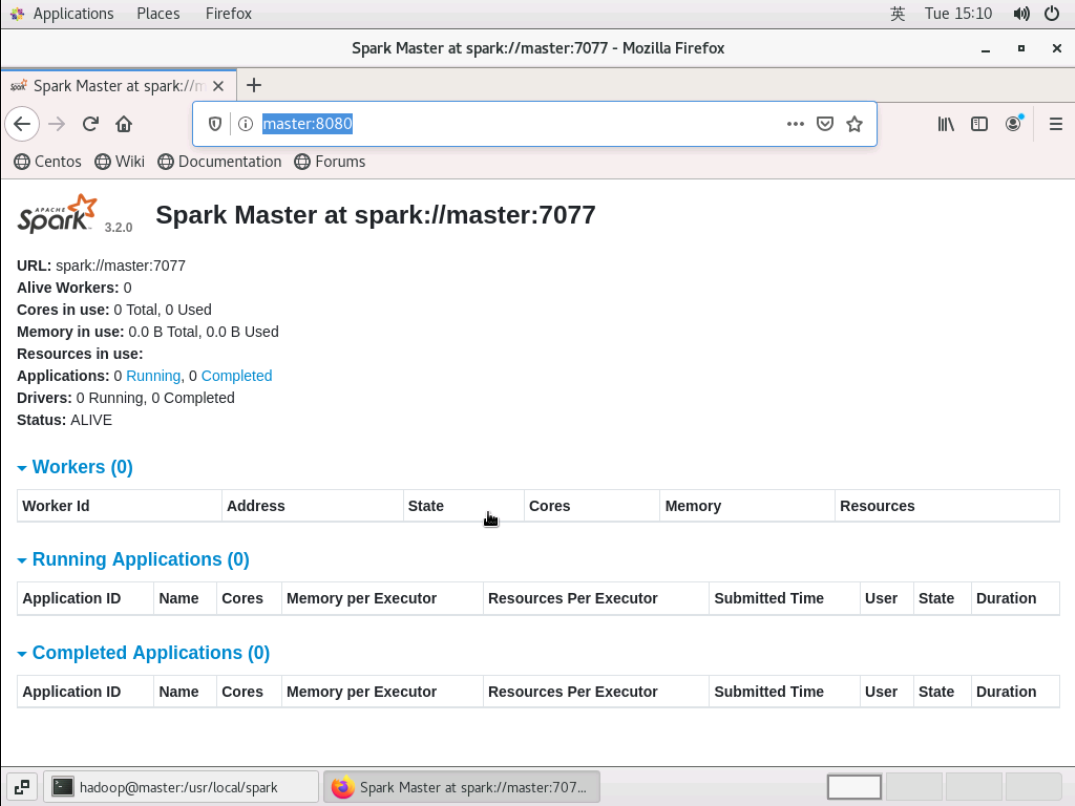


图2 Spark集群启动

3.2 网络电影数据的爬取与存储

（1）数据爬取

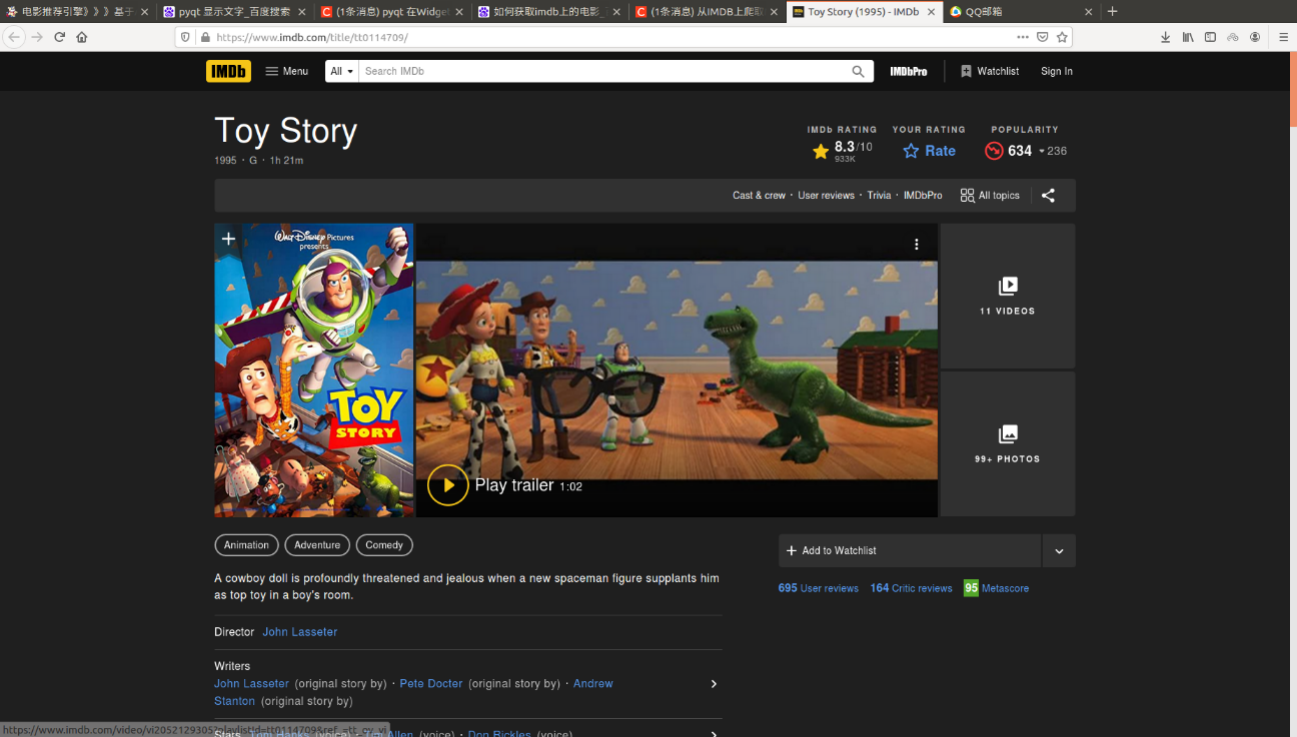


图3 爬取网页界面

本案例采用requests和BeautifulSoup库进行数据爬取。需要注意的是，所爬取的数据为海报的URL，因此在爬信息的过程中还需要再次爬取，发送请求下载海报。如图4，src存储的即为海报的链接.



图4 海报链接示意图

由上述分析过程可知，构造请求应该独立成一个函数，我们将爬虫包装为一个类，该类函数结构为：

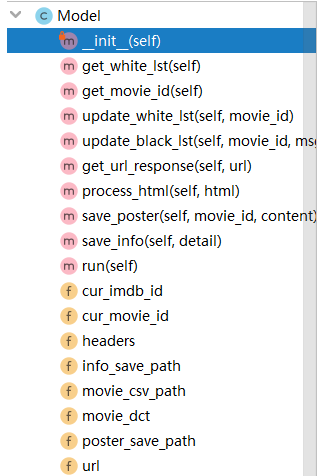


图5 类函数结构图

图5中的white\_lst和black\_lst分别表示已经爬取到的数据和爬取出错的电影，由于在爬取电影信息的过程中可能会出现报错，在修改完代码之后对剩余的信息重新进行爬取，white\_lst中保存的就是已经爬取好的电影信息不需要二次爬取，black\_lst中保存的为出错的电影，图6是运行日志中记录的爬取出错的电影。

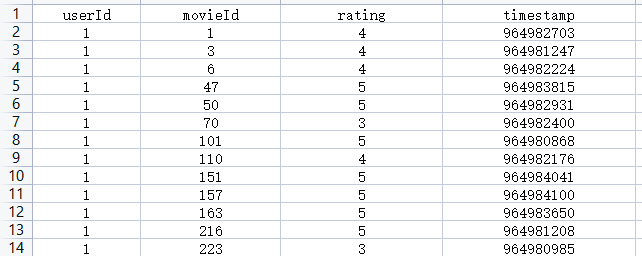


图6 爬取错误日志记录

在代码中分别使用 self.movie\_dct = {}存放每一步电影的id和imdb的id,self.white\_lst = []存放已经处理完的movie id，self.url = 'https://www.imdb.com/title/'为电影详情的初始url，self.poster\_save\_path='./poster'海报的保存路径， self.info\_save\_path='./info/info.csv'电影信息的保存文件，self.cur\_movie\_id = None，self.cur\_imdb\_id = None表示当前处理的电影，函数get\_white\_lst(self)表示获取处理的白名单，函数get\_movie\_id(self):获取电影的id和imdb的id，函数update\_white\_lst(self, movie\_id）用于更新白名单，函数update\_black\_lst更新错误名单，写入movie id 和imdb id，并且加上错误原因，msg=1是URL失效，msg=2是电影没有海报。通过函数get\_url\_response(self, url）访问网页请求，返回response，再通过函数def process\_html(self, html）解析html，获取海报，电影信息，最后通过save\_info(self, detail）将电影信息存储到CSV文件中。

（2）数据存储与处理

|  |  |  |  |
| --- | --- | --- | --- |
| **UserId** | **movieId** | **Rating** | **timeStamp** |
| 用户id | 电影id | 用户评分 | 评分时间 |



|  |  |  |
| --- | --- | --- |
| **movieId** | **title** | **genres** |
| 电影id | 电影名称 | 电影体裁 |



图7 爬取数据结构图

所爬取的数据如图7所示。随后对收集的数据集进行简单处理，其中缺失的数据进行填充。在rating数据集中，若用户未对某部电影评分，则将原始数据的nan值置为0.00000。处理后评分数据用相同形式进行表示，方便后续进行处理。



图8 处理后数据展示

3.3 电影个性化智能推荐的实现

（1）基于SVD的推荐算法

奇异值分解（Singular Value Decomposition，简称SVD）是在机器学习领域广泛应用的算法。它不仅可以用于降维算法中的特征分解，还可以用于推荐系统。以下我们使用SVD算法，对用户进行智能电影推荐。

算法细节：SVD是对矩阵进行分解，SVD并不要求分解的矩阵为方阵，假设矩阵A是一个m\*n的矩阵，我们定义矩阵A的SVD为：



其中U是一个m\*m的矩阵，∑是一个m\*n的矩阵，除了主对角线上的元素以外全为0，主对角线上的每个元素都称为奇异值，V是一个n\*n的矩阵。U、V都是酉矩阵，即满足。

求出SVD分解后的UV三矩阵的方法如下：

将A的转置和A做矩阵乘法，即可得到一个n\*n的方阵，对此方阵进行特征分解，得到的特征值和特征向量满足下式，



此时，得到的n个特征值和其对应的n个特征向量v，将的所有特征向量集合为一个n\*n的矩阵V，即可得到SVD公式里的V矩阵了。通常将V中的每个特征向量叫做A的右奇异值。

同理，将A和A 的转置做矩阵乘法，即可一个n\*n的方阵，对此方阵进行特征分解，得到的特征值和特征向量满足下式，



此时，得到的n个特征值和其对应的n个特征向量u，将的所有特征向量集合为一个n\*n的矩阵U，即可得到SVD公式里的U矩阵了。通常将U中的每个特征向量叫做A的左奇异值。

因为∑除了对角线上是奇异值，其它位置均为0，所以只需求出每个奇异值σ即可。

再者，

通过求出每个奇异值，进而求出奇异值矩阵∑。

由于，

我们可以看出特征值矩阵等于奇异值矩阵的平方，特征值和奇异值的关系式为，，即可通过的特征值取平方根来求奇异值。

利用SVD算法对评分矩阵进行分解，可以得到U、∑和V，把U当做用户空间，V当做电影空间。经过分解后，前面的值就已经可以表示大部分特征，所以这里选择了5个用户来表示原来的评分矩阵。

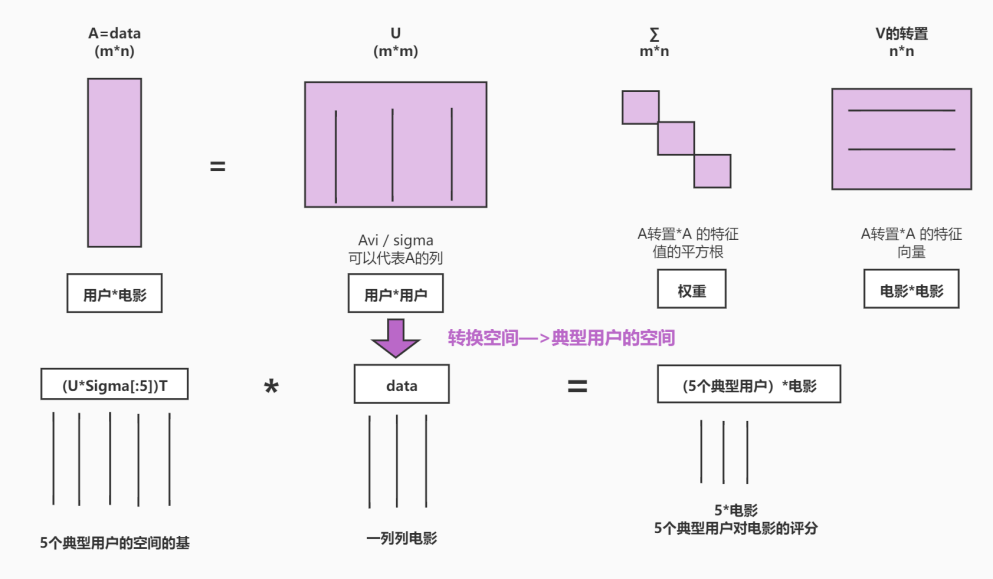


图9 SVD分解示意图

得到5个典型的用户后，我们计算电影之间的余弦相似度，这样就可以得到每一个电影和其他电影之间的相似度评分，如下所示。

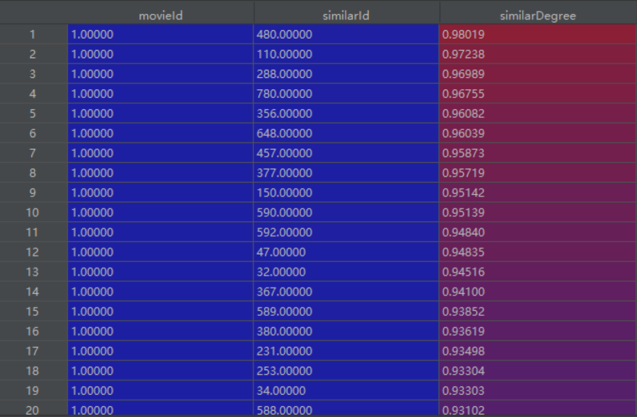


图10 相似度评分结果

在获取相似度评分之后，即可为用户进行推荐。其核心思想是：如果我们要推荐某一部电影，就观察他对其它电影的评分以及电影之间的相似度，将其相乘作为一个评判标准。如果某一个电影与这部电影相似，同时用户对另外一部电影的评分很高，我们就认为该用户可能更喜欢此类的电影。相反，如果评分很低，那么就说明用户不喜欢。将所有的看过的电影评分和相似度加权，预测该电影的评分。将所有没有看过的电影都预测完之后，选择前五部电影推荐给用户，即实现基于SVD的电影推荐算法。

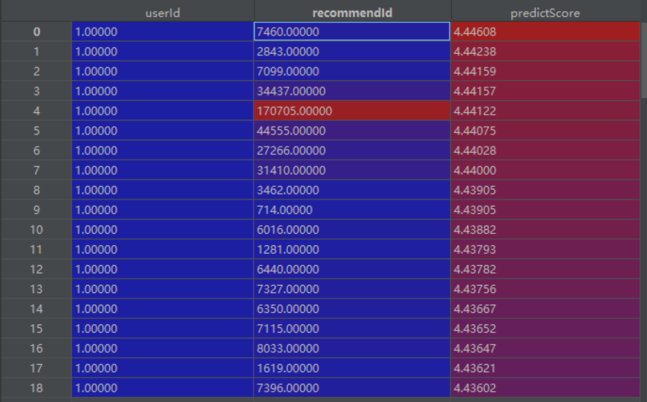


图 11 推荐结果

优点：SVD作为一个基础的算法，在很多机器学习算法中都能融合使用。SVD可以实现并行化，可以简化数据、去除噪声、提高算法的结果。

缺点：SVD的计算复杂度较高，且SVD分解出的矩阵解释性往往不强，有点类似于黑盒子。

（2）基于ALS的推荐算法

ALS是指交替最小二乘法（Alternating Least Squares），它是一个基于模型的协同过滤，通过降维的方法来补全用户-物品（电影）矩阵，对矩阵中没有出现的值进行估计，基于此要求，提出了ALS算法。

算法细节：最小二乘法（Least Squares：LS）是ALS的基础。人们常用最小二乘法寻找最优的未知数据，通过最小化误差平方和寻求数据的最佳匹配，保证求得数据与已知数据误差最小。我们用简单的一元线性回归模型说明最小二乘法。

假设有一组数据(x1,y1),(x2,y2),(x3,y3)...其符合线性回归，假设其符合的函数如下：



使用平方差函数，表达参数的好坏，Ln越小表示误差越小。N个样本的平均平方差为：。L越小参数w越精确，故寻找最合适的w0 w1很关键。概括即为：。

将线性回归函数代入到最小二乘损失函数中，得到的结果如下：



L函数取得最小值时，w0 w1的一阶偏导为0，因此对L函数分别求偏导，即可对w0 w1求解。

用户对物品的打分行为可以用矩阵R表示，R矩阵的行向量对应每个用户的U，列向量对应每个用户的V。ALS的核心思想是：将用户和物品（电影）都投射到K维空间，也就是说假设有K个隐向量特征，将每个用户和每个物品都用K维的向量表示，把它们的内积近似为打分值，这样便可以得到近似的评分：



R为打分矩阵（m\*n，m表示用户个数，n表示物品个数），U表示用户对隐含特征的偏好矩阵（m\*k），V表示物品对隐含特征的归属矩阵（n\*k），上述模型的参数就是U和V，求得U和V之后，就可以近似的得到用户对未评分物品的评分。

求上述公式中的U和V，通过显示反馈代价函数，来衡量参数的拟合程度。



通过先固定U求V，再固定V求U，交替使用最小二乘法去求解，如此迭代下去，问题即可解决。



因为spark提供了ALS算法，所以我们只需要将数据输入即可，下面是ALS算法推荐的结果，前面表示的是推荐的电影，后面表示的推荐电影的评分。

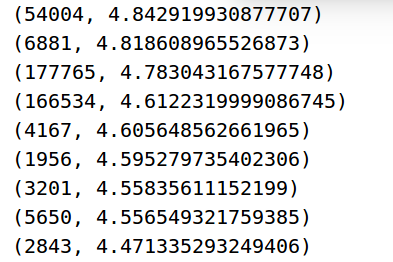


图12 ALS算法推荐结果

优点：由于待分解的矩阵常常是非常稀疏的，ALS能有效地解决过拟合的问题；基于ALS的矩阵分解的协同过滤算法的可扩展性是优于SVD的。

缺点：它是一个离线算法，不能体系实时性；无法准备评估新加入的用户或商品，即存在冷启动问题。

（3）平台展示

为了实现整个项目的完整性，随后进行了前端的设计。为了充分发挥小组成员的能力，我们分为了B\S端和C\S端两种不同的界面。在B\S端，我们主要实现两个web界面，分别是用户电影信息输入界面和智能电影推荐界面；在C\S端，我们主要实现了两个界面，分别是用户信息输入界面和智能电影推荐界面。

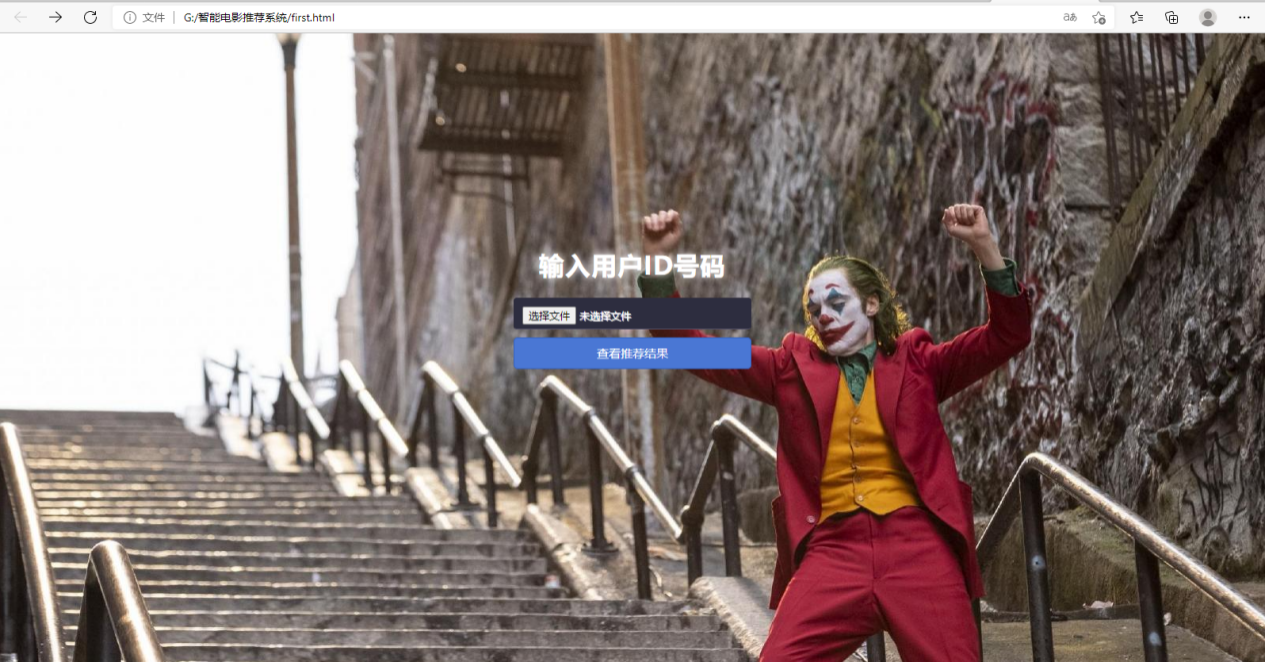


图13 B\S端用户电影信息输入界面

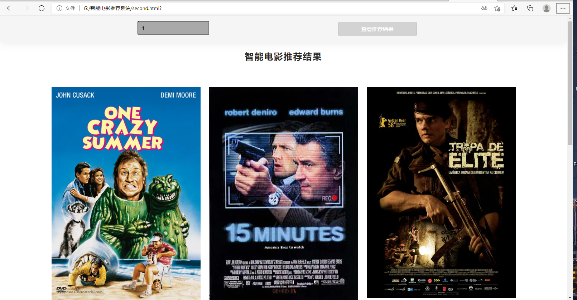
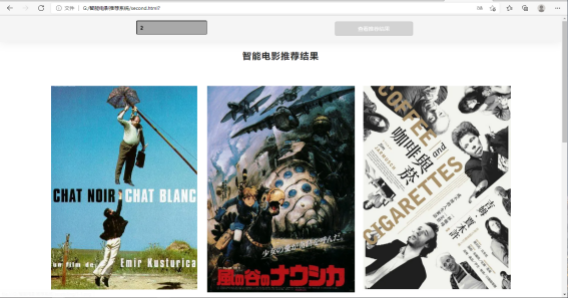


图14 B\S端智能电影推荐界面

## 4 小结

本案例选取互联网推荐系统的设计作为主题，使用爬取于热门电影网站的电影信息数据作为实验数据集，结合数据挖掘、机器学习等相关方法进行电影个性化智能推荐平台的构建。主要内容包括服务平台的搭建、网络电影数据的爬取与存储以及电影个性化智能推荐的实现。该案例主题新颖，结合了工业界现实需求与大数据分析与挖掘的多种理论与技术，可以充分增强学生的实践能力与理论基础。另外，本案例的内容仅为指导性的过程，在实际教学中，可保持基本研究内容不变，鼓励学生引入其它的数据预处理、数据挖掘、机器学习方法完成任务。

## 附录

1. 本案例提供配套的PPT、视频、数据集与代码等，发布于Github，链接为：https://github.com/Wanghui-Huang/CQU\_bigdata。

2. 本案例涉及数据预处理以及多种机器学习算法，建议使用python语言进行编写，推荐的工具包有requests（网络请求库），BeautifulSoup（HTML解析库），scikit-learn（机器学习库）。

3. 本案例参考文献如下：

[1] 周志华. 机器学习[M]. 北京: 清华大学出版社, 2016.

[2] 黄立威,江碧涛,吕守业,刘艳博,李德毅.基于深度学习的推荐系统研究综述[J]. 计算机学报,2018,41(07):1619-1647.