**2021年西北大学数学建模协会赛论文格式规范**

* 参赛队从A、B、C题中任选一题。
* 论文上下左右各留出至少2.5厘米的页边距**。**
* 论文第一页为竞赛论文封面，请认真准确填写封面信息(具体见下面)。
* 论文题目和摘要写在论文第二页上，从第三页开始是论文正文。
* 论文从第二页开始编写页码，页码必须位于每页页脚中部，用阿拉伯数字从“1”开始连续编号。
* 论文不能有页眉，论文中不能有任何可能显示答题人身份的标志。
* 论文题目用三号黑体字、一级标题用四号黑体字，并居中；二级、三级标题用小四号黑体字，左端对齐（不居中）。论文中其他汉字一律采用小四号宋体字，行距用单倍行距。
* 提请大家注意：摘要应该是一份简明扼要的详细摘要（包括关键词），在整篇论文评阅中占有重要权重，请认真书写（注意篇幅不能超过一页，且无需译成英文）。评阅时将首先根据摘要和论文整体结构及概貌对论文优劣进行初步筛选。
* 论文应该思路清晰，表达简洁（正文尽量控制在20页以内，附录页数不限）。
* 由于部分竞赛题目需要收集数据，因此要在正文部分注明数据的出处；并且电子版论文中必须提供源数据，以供程序验证。
* 引用别人的成果或其他公开的资料(包括网上查到的资料) 必须按照规定的参考文献的表述方式在正文引用处和参考文献中均明确列出。正文引用处用方括号标示参考文献的编号，如[1][3]等；引用书籍还必须指出页码。参考文献按正文中的引用次序列出，其中书籍的表述方式为：

[编号] 作者，书名，出版地：出版社，出版年。

参考文献中期刊杂志论文的表述方式为：

[编号] 作者，论文名，杂志名，卷期号：起止页码，出版年。

参考文献中网上资源的表述方式为：

[编号] 作者，资源标题，网址，访问时间（年月日）。

* 本规范的解释权属于西北大学数学建模协会。

**2021年西北大学**

**数学建模协会赛**

**题目**： C题 （ 填写A、B、C题 ）

|  |  |  |  |
| --- | --- | --- | --- |
|  | 参赛队员1 | 参赛队员2 | 参赛队员3 |
| **姓名** | 樊泽瑞 | 郭明皓 | 刘宇哲 |
| **学号** | 2020111089 | 2020111109 | 2021117188 |
| **学院** | 信息科学与技术学院 | 化学与材料科学学院 | 信息科学与技术学院 |
| **专业** | 软件工程 | 化学类 | 电子信息类 |
| **电话** | 15389422621 |  |  |
| **Email** | 477706421@qq.com |  |  |

**西北大学数学建模协会**

摘要

影视评价与定制问题基于大数据所提出，该问题受到水军刷票数据造假，夸大宣传等因素影响。为了解决这些问题，从互联网上搜索可靠的数据，过滤掉有瑕疵的值，确定权重并给出合理的预测是非常重要的。

针对问题一。在电视剧排名方面，我们选择电视剧得分、每部电视剧的评论人数和电视剧集数作为前三个重要指标来判断最终的排名。为了找到这三个指标中最合理的权重，我们使用AHP层次分析法来计算最佳权重，提供了一个新开发的指数R分数。在计算最终结果时，分别定义了三个加权值。

#

第二，我们认为，判断明星流行度的指标是各式各样的，并且没有非官方指标，所以Apriori算法可以用来过滤不重要的指标，只保留高权重指标。通过遍历从互联网收集的多个数据，我们得到最终的频繁n项集，其中最重要的指标。然后利用主成分分析法确定相关指标的权重。除此之外，还应考虑特殊情况，如花絮在短时间内引起的剧烈变化。最后，我们将基于我们的指数的排名与官方网站中存在的排名进行比较，发现近似相等。

#

第三，为了建立一个新的团队来创造新产品，我们可以使用爬虫从互联网上搜索到的数据，如点击率、评论、明星、制作团队等。为了过滤掉不重要的指标，可以采用逐步回归的方法，然后经过标准化得到回归方程。通过这个等式，每个指数将对应一个权重，该权重衡量对最终指数的贡献。然后，将导出的排名与官方排名进行比较，以获得可信度，并判断该指数是可接受的。根据最终指标，描述一个理想的生产团队。

最后，从观众的浏览历史和每个频道的评分中获得最合适的推荐。这里使用LDA算法，找出主要趋势和主要类型的历史。赋值主要依赖于历史数据具有不同类型的概率，然后通过余弦计算，找到拟合度最高的数据。使用标准化后的数据，我们证明了可信度高达93.2%。更重要的是，根据评级，这种模式也是可靠的。

每一个模型都经过了来自现实的数据测试，这些数据来自互联网，由crawler在Python中搜索。

1. 问题重述

1.1问题背景

目前中国电视市场规模大，竞争激烈，产品类型多。每年虽然可以产出大量电视剧，但过多的电视剧缺少电视台购买，造成了大量的投资浪费。其核心原因在于大量电视剧质量差。为了提升电视剧质量，得到更大的利润回报，需要对于如何评价和定制影视剧等问题进行预测很分析。

已知大数据作为分析工具可以非常精确地分析数据和预测。这可以应用于剧本写作，电视评级预测，电视广告的结果和电视剧购买。可以降低电视投资风险，提高脚本质量，并预测受众响应以确保最大的收益。

1.2问题重述

现在给出两个附件，其中附件1包含了429部电视剧的电视剧评分，电视剧评论数和电视剧所属类型的统计结果。数据为空说明缺少相关数据。其中附件2包含429部电视剧制作公司，发行时间等基本信息。

1. 问题一：对于附件一429部电视剧的电视剧得分、每部电视剧的评论人数和电视剧集数进行分析评价，从电视剧自身质量出发，选出最优的十部电视剧。
2. 问题二：通过信息搜集一年内各平台明星的相关热度信息，对其进行分析评价，建立模型设定明星人气指数显示明星个人的真实人气，同时今年的实例证明该模型的可靠性
3. 问题三：根据目前已有的明星热度数据和观众画像，推测观众感兴趣的部分，由此描述一个理想的制作团队阵容。
4. 问题四：建立根据观众个人的历史观看记录，为其定制推广其最感兴趣，最适合该观众的相关的节目的模型方案

2.模型假设

1）数据可以正确反映受欢迎程度，没有互联网恶意灌水等非自然趋势提高受欢迎程度的行为。

2）所有以明星或电视剧名称命名的论坛都在谈论相关主题。

3）对知名网站的排名没有商业猜测，所有排名都依赖于真实数据，并且必须是客观的。

4）这些模型具有普适性，因为来自互联网的数据不能包含所有明星和电视剧，通过计算足够大的数据规模来考虑。派生的模型可以适用于所有明星和电视剧，并且产生的误差应该足够小。

3符号说明

|  |  |
| --- | --- |
| **符号** | **说明** |
| RScore | 排名指数 |
| S | 评价分数 |
| Chs | 归一化评论数 |
| Ehs | 归一化剧集数 |
|  |  |
|  |  |
|  |  |

表1 符号说明

## 4问题分析

## 4.1问题一的分析

### 4.1.1 问题概述

我们首先通过主观判断，可以明确在论坛中所给出的评分高并不代表着电视剧一定是一部好剧，例如说：A剧和B剧同样是4.8分，但是由于这两者的评论数不同导致这两部电视剧无法进行比较。大部分观众只是在电视剧播出之前提前通过观看评论判断出它是一部好剧，评论的走向也会很大程度的影响这部电视剧的观看意向。由于评论分数与评论数量之间存在偏差，我们需要一种综合其他从属因素的方法来完善电视剧的评价标准。经过认真的讨论，除前一个主要参数外，还考虑了剧集和著名明星的数量，以构成一个数学模型。

**层次分析法**是指将一个复杂的多目标决策问题作为一个系统，将目标分解为多个目标或准则，进而分解为多指标（或准则、约束）的若干层次，通过定性指标模糊量化方法算出层次单排序（权数）和总排序，以作为目标（多指标）、多方案优化决策的系统方法。经过我们的讨论，最终选择**层次分析法（AHP）**作为我们此次的分析模型。

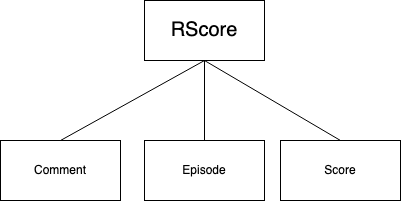


图1.1 层次分析法影响因素

### 4.1.2 模型提出（AHP层次分析法）

电视剧评论的数量实际上是观众数量的反映，因此我们认为**评论数量**代表观众评级，**评价分数**代表电视剧的质量，**剧集数量**代表特定电视剧制片厂的制作成本。

首先，我们排除了所有未评分的戏剧，因为如果没有最关键的评价因素，我们就无法评判电视剧。 所谓的“精彩且具有成本效益的电视剧”无疑是基于评论数量、高分和低预算的。

**·评价分数**

电视剧的评价分数是评论者所给的分数的均值，是给定R分数的主要基础。 我们将其权重值定义为1，这恰到好处，因为分数作为首要因素，直接反映了观众对电视剧的印象。

·**评论数量**

电视剧评论的数量决定了它的受欢迎程度。 在某种程度上，由于虚拟网络和现实世界的讨论指数飙升，而不是其质量，电视剧在一段时间内变得如此火爆。 这种现象往往导致这样一个事实，即如果我们只是将评论和分数的数量纳入AHP，我们将获得不准确的后果。 因此，我们将评论最多的电视剧的评论权重值定义为1，通过这个类比，我们分别将评论数量转换为百分比，并将其添加到R评分的计算中。

·**剧集数量**

众所周知，电视剧集越多，制作量就越多投资这部电视剧会花很多钱。 根据我们的第一个假设，不同系列中的每个剧集都有相同的成本，也就是说，电视剧集越多，电视剧的成本效益就越低。 我们通过设置适度的权重值0.2来重新定义这个因素，该值对结果的影响相对较小。

通过评价分数的计算公式我们可以知道，对于所有评论所给定的单一的评价分数取平均值即为最终的评价分数。因此我们对**评价指数RScore**指定一个合理的计算公式：

4.1.3 总结

### 4.1.3总结

### 我们通过计算得到前十名的电视剧：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 电视剧 | 评价分数 | 评论数量 | 剧集数 | 归一化评论数 | | 归一化剧集数 | 评价指数 |
| 战长沙 | 9.2 | 18419.0 | 32 | | 0.702399 | 0.256 | 6.410868 |
| 何以笙箫默 | 6.3 | 26223.0 | 32 | | 1.000000 | 0.256 | 6.248800 |
| 北平无战事 | 8.8 | 16084.0 | 53 | | 0.613355 | 0.424 | 5.312721 |
| 离婚律师 | 7.2 | 16872.0 | 46 | | 0.643405 | 0.368 | 4.558913 |
| 红高粱 | 7.5 | 10028.0 | 60 | | 0.382412 | 0.480 | 2.772093 |
| 武媚娘传奇 | 5.4 | 13625.0 | 82 | | 0.519582 | 0.656 | 2.674543 |
| 杉杉来了 | 6.9 | 9285.0 | 33 | | 0.354078 | 0.264 | 2.390342 |
| 一仆二主 | 7.2 | 8337.0 | 43 | | 0.317927 | 0.344 | 2.220274 |
| 我爱男闺蜜 | 7.4 | 7373.0 | 34 | | 0.281165 | 0.272 | 2.026224 |
| 约会专家 | 8.3 | 5664.0 | 32 | | 0.215994 | 0.256 | 1.741547 |

表2 前十名电视剧排名

## 4.2 问题2的分析

### 4.2.1 问题概述

对于明星在大众中的受欢迎程度，我们选择使用Apriori算法进行处理。我们使用32个可能相关的条件，通过对其**支持度的阙值**的计算来进行合适的项集选择。

对于支持度的阈值和置信度的的计算公式如下所示：

（支持度）

（置信度）

通过筛选，我们最终得到了14个相关的流行的评价指标。

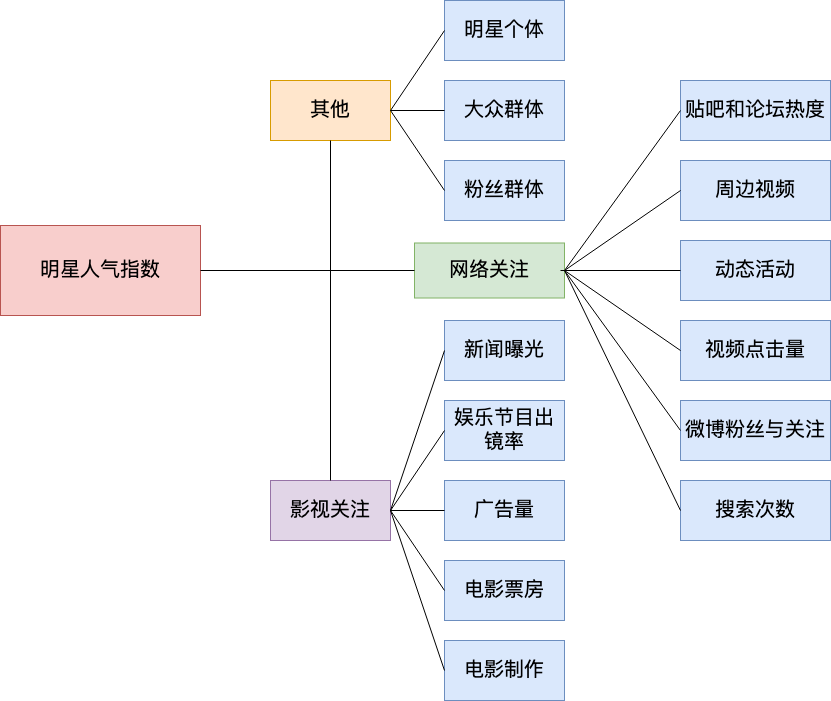


图3.1.1 可能流行的评价指标

上述评价指标的受欢迎程度，虽然部分受欢迎指数对受欢迎指数贡献显著，但会随着其他指标的重复，其重要性被消减。

因此，我们可以利用**主成分分析法**对指标进行分析，得到一些贡献较大的指标，而最终的指标没有或几乎没有一致性。

4.2.2指标选择

为了确保数据实时性，我们的数据来源是一年内互联网上的明星公开的搜索数据

为了确保数据来源的多样性，我们的数据来源由百度搜索量等社会影响热度和微博粉丝数、百度指数等明星日常粉丝社区活跃热度两部分组成，综合分析明星的真实影响力和热度

### 4.2.3 模型提出（PCA主成分析法）

#### 4.2.3.1 模型分析

对于图3.1.1 中所给出的数据模型，我们有以下分析：

虽然每个指标所针对的项目不是一致的，但不同指标之间的相关性往往会对一个明星的人气指数起到一个主导性的作用。为了找到相关性和分布权重，本文引入主成分析法进行分析：

1. 数据标准化处理：

我们使用z-score 标准化(zero-mean normalization)对数据进行标准化操作：

我们通过这种方式可以消除不同指标维度之间的影响，并且在归一化过程中，不会影响到数据之间的相关性。

1. 计算归一化数据的相关系数矩阵，找出特征值和特征向量：

最终的特征矩阵为 ,我们所给定的特征值以及所给定的向量。

1. 计算贡献率 以及累计贡献率 选定计算结果 的
2. 获取每个指数在受欢迎度指数上的权重：

计算贡献率并且得出所对应的作为新权重，这将是我们所需要得到的主成分的权重。

#### 4.2.3.2 模型求解

通过4.2.3.1章节所得出的，影响人气的几个重要指标有贴吧热度、微博热度、网络搜索量、粉丝数量和影视热度。以上的五个主要参数可以作为长期衡量指标。

最终计算所得出的明星人气指数的计算公式以及权值给出如下：

𝑃 = 0.2806𝑚1 + 0.2231𝑚2 + 0.1804𝑚3 + 0.0407𝑚4 + 0.0942𝑚5 +0.2010𝑒−0.041𝑑

## 4.2.4总结

最终根据此模型所给出模型，我们通过使用Python对于目前各大主流的贴吧平台、视频平台以及搜索引擎等进行数据收集，最终套用模型得出以下的人气排行榜：

|  |  |  |  |
| --- | --- | --- | --- |
| 排行 | 姓名 | 排行 | 姓名 |
| 1 | 杨紫 | 8 | 赵本山 |
| 2 | 刘涛 | 9 | 梅婷 |
| 3 | 赵丽颖 | 10 | 黄河 |
| 4 | 陈伟霆 | 11 | 金晨 |
| 5 | 韩雪 | 12 | 杨颖 |
| 6 | 靳东 | 13 | 霍建华 |
| 7 | 林心如 | 14 | … |

表3 检验明星人气排名

我们将所得到的排位与当天的官方所发布的数据进行对比，其均方差均不超过50，对于前20名的均方差小余20，所以该算法所提出的明星人气模型是能够得出可接受结果的。

## 4.3问题3:问题三的分析

### 4.3.1 问题概述

对于市面上所能够见到的各式各样种类的电视剧，都有着不同的观众受众，为了找到不同观众对于何种电视剧更加感兴趣，我们首先收集了2021年年度人气电视剧排行榜：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 排行 | 电视剧名称 | 参演明星 | 电视剧类型 | 电视剧集数 |
| 1 | 谁是凶手 | 赵丽颖,肖央,董子健,姚安濂 | 悬疑, 犯罪 | 16 |
| 2 | 风起洛阳 | 黄轩,王一博,宋茜,宋轶,咏梅 | 剧情,悬疑,古装 | 39 |
| 3 | 就这样… | 莎拉·杰茜卡·帕克,克里斯汀·戴维斯,辛西娅·尼克松 | 剧情,喜剧,爱情 | 10 |
| 4 | 爱很美味 | 李纯,张含韵,王菊,周澄奥,刘冬沁 | 剧情,喜剧,爱情 | 20 |
| 5 | 女心理师 | 杨紫,井柏然,王嘉,菅纫姿,黄觉 | 剧情 | 40 |
| 6 | 那年，我们的夏天 | 崔宇植,金多美,金圣喆,卢正义 | 爱情 | 16 |
| 7 | 华灯初上 | 林心如,杨谨华,杨祐宁,凤小岳,张轩睿 | 剧情,爱情,悬疑,犯罪 | 8 |

表4 2021年年度人气电视剧排行榜

### 4.3.2 模型提出（逐步回归法）

#### 4.3.2.1 模型分析

观众的兴趣往往是导演们所需要考察的对象。即使电视剧的导演如何去宣传自己所拍摄的电视剧，但是如果观众对于这部电视剧表示不买账。因此，以下想法旨在调查观众的兴趣。根据表4我们可以看到，于2021年喜剧和爱情剧的观众受众十分的广泛。观众的评论往往反映了一部电视剧的实际情况，根据我们的常识而言，意见往往会在某种程度上影响某些人的观影体验，特别是我们面对一些资深观影人的评论时。因此我们提出使用逐步分析法对此问题进行求解。

通过以上的分析，我们可以看到影响电视剧受欢迎程度的指标很多

|  |  |
| --- | --- |
| 明星 | 影视剧类型 |
| 制作团队 | **拍摄成本** |
| 播出时间 | **播放频道** |
| 地域因素 | **受众人群** |
| 特效质量 | **…** |

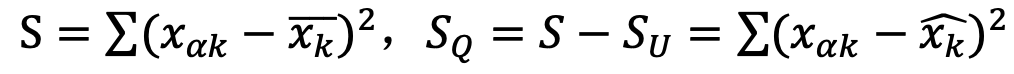
表5 影响电视剧受欢迎程度的因素

由于影响电视剧受欢迎程度的指标很多，可以采用逐步回归的方法选择最显著的指标。通过筛选自变量，自变量个数越大，回归平方和越大，残差平方和越小，回归分析质量越高，可以有效提高回归模型分析的精度。

#### 4.3.3.2 模型求解

1）对索引重新编号并标准化不同维度：

我们首先将，数字的下标为，所以数学模型为：

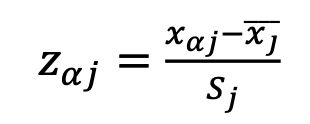


此外，的偏平方回归和是：

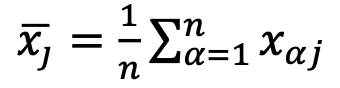
为是的矩阵对角线值。

2) 以新指数为参数的回归模型为：

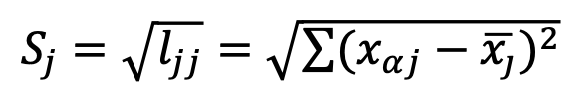
将初始数据转化为标准化回归数学模型，求解得到:



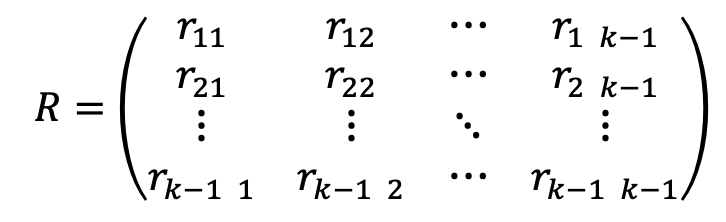
并且：



3)我们可以以此得到：



4)我们最终可以得到初始回归曲线模型和相关性系数。建立相关系数矩阵：



通过上述公式求解，我们可以得到电视剧的一些重要指标以及相应的权重。将这些指标作为参数，标准化后放入公式中，避免了单元的大小，使系统误差最小化。指数的权重与公式中的系数正相关，这意味着假设总权重为1，权重呈多重关系；所有权重之和为1，在最终的流行性判断公式中，可以轻松计算出相应的权重。求解得到电视剧受欢迎程度的最终公式是：

𝑃 = 0.3634𝑥1 + 0.2184𝑥2 + 0.1723𝑥3 + 0.1018𝑥4 + 0.0102𝑥5 + 0.0778𝑥6

通过这些数据，我们可以计算出不同类型的流行程度，并以数字的形式在公式中表示出来。广播频道和时间、特效水平和宣传力度可以用同样的方法计算。另外，明星人气指数见第4.2节。在R中，总权重不是1，因为某些指标对整个指标的贡献很小。这里我们可以忽略这些索引。

## 4.3.4总结

为了获得理想的生产团队的名单，我们必须考虑各种指标之间的搭配。因为理想的团队不是由最受欢迎的明星、最受欢迎的类型和其他最受欢迎的指数组成的简单团队。例如，一个明星不适合所有类型的戏剧，而一个团队通常在特定类型中表现出色。应该建立一个新的矩阵来描述搭配关系。

为第i个明星的特征矩阵，其中包含了与该明星所参与的各种戏剧的搭配程度。n是该明星出演戏剧的数量。当与戏剧类型匹配时，匹配系数，是戏剧类型的特征矩阵。匹配系数越大，匹配效果越好。同时，制作团队要像明星一样打字。每个索引都可以用一个特定的矩阵表示。结合公式，我们可以得到如下理想团队：

|  |  |
| --- | --- |
| 导演 | 孔笙 |
| 明星 | 赵丽颖、胡歌、白敬亭、杨紫 |
| 类型 | 当代城市剧 |
| 策划 | 王丽萍、海岩 |
| 组织 | 山东电影电视剧制作中心 |

表6 模型预测最佳团队

最受欢迎的戏剧类型是当代都市剧，这可以作为一个基本指标。结合公式，我们可以找到最匹配的明星、导演和编剧。

## 4.4问题4:问题四的分析

### 4.4.1 问题概述

在大数据信息爆发的时代，智能推荐作为一种能够针对于使用用户的个性化特征推荐相关内容的方式，面对数量如此庞大的影片资源和海量的用户群，如何对不同的用户进行精准、个性化的推荐，成为各大平台网站或手机 APP 提升自身竞争力的关键。而正因为影片资源和用户的数目都如此庞大，单纯依靠人力来实现个性化推荐已不具可行性，智能推荐系统以其高效、精准和即时的大数据处理能力而被各大平台网站或手机 APP 广泛采用。借助于注册信息、位置识别、个人通信录等来获得用户的性别、年龄、区域位置、文化程度、经济收入、社交人脉等个人信息，在此基础上进行“用户画像”，预测用户的兴趣或需求，建立用户和电影资源之间的匹配关系，并使每一个用户获得与其他用户不同的“精准”、个性化推荐，做到不同用户界面的“千人千面”，从而提升用户对平台的满意度和黏合度。

目前常见的智能推荐算法有以下几种：

（一）基于内容的推荐算法

这种推荐算法的基本思想是 ：用户会对跟之前的选择相类似的内容感兴趣。在电影推荐领域，该算法会抽取用户之前所选择的电影资源的特征，并以此作为判断其兴趣的参照，然后从候选资源中推荐与之相似的作品。要以这种算法来实现电影推荐，系统需要一些“元数据”，如编剧、导演、演员、制片人、所属类型以及电影评分等。当新用户登录电影网站或手机APP 时，通常，系统会首先随机向他推荐一些热门影片，随着该用户后续的点击、收藏、评论或评论，其行为记录不断增加，智能推荐算法以此为依据判断用户的兴趣，进而推荐与这种偏好相类似的短视频资源。“在基于内容的推荐系统中，我们得到了关于内容的信息（如领域和元数据）越多，算法就会变得越复杂（因为要涉及更多的变量），不过推荐也会更准确、更合理。”

（二）基于协同过滤的推荐算法

目前的智能推荐系统还不能真正“读懂”电影本身，而基于协同过滤的推荐算法并不尝试去识别影像或者声音，它通过分析目标用户的行为，找到与之相似的其他个体，再参照他们的选择来提供推荐内容，它是一种基于用户行为分析而运作的推荐算法。在电影推荐领域，基于协同过滤的推荐算法并不需要理解影片

（三）基于图模型的推荐算法

这种推荐算法以图计算理论为基础，将用户和影片资源视为图结构中没有直接相连的不同节点，从而把用户选择影片的过程转化为图模型，进一步构造出关系网络图，利用特定的算法（如基于随机游走的PersonalRank 算法等），计算出用户和项目（影片资源）在图模型中的相关性，相关性越高则该影片被推荐的权重越大。基于图模型的推荐算法更接近于真实的人际社交网络，尤其是在 web2.0 背景下的新媒体社交网络，每一个用户都可以是一个提供与发布内容的节点，图模型更适于用描述在这种社交网络中用户与电影资源之间的关系。但是基于图模型的推荐算法需要对二部图进行迭代，具有较高的复杂度和时间成本，在目前的技术条件下，它的应用受到了一定的限制。此外，还有综合上述各种推荐算法的优势，根据实际需要将各种算法混合使用的所谓“混合推荐算法”。目前，最广泛的混合方式是将 CB（基于内容）和 CF（基于协同过滤）相结合的方法，从而进一步提升了推荐的个性化及精准性。

### 4.4.2 模型提出（TF-IDF算法）

#### 4.4.2.1 模型分析

#### 4.4.3.2 模型求解

## 4.4.4总结

|  |
| --- |
| 数据预处理 |
| import pandas as pd  import numpy as np  from matplotlib import pyplot as plt  part = 2  if part == 1:  dt1 = pd.read\_excel("附件一-电视剧评估信息.xlsx",'Sheet1')  dt2 = pd.read\_excel("附件二-电视剧基本信息.xlsx",'Sheet1')  writer = pd.ExcelWriter("result.xlsx")  #print(dt1.merge(dt2,left\_on='TV Drama',right\_on='TV Drama'))  dtret = dt1.merge(dt2,*left\_on*='TV Drama',*right\_on*='TV Drama')  dtret.to\_excel(writer,'Sheet1')  writer.save()  writer.close()  classes = {}  for i in dt1.groupby('Theme'):  classes.update({i[0]:i[1]})  print(str([i for i in classes.keys()]))  cnt\_drama = [0 for i in range(12)]  cnt\_drama\_comments = [0 for i in range(12)]  for i,j in zip(dtret.sort\_values(*by*='Date of Issuance License')['Date of Issuance License'].reindex(),  dtret.sort\_values(*by*='Date of Issuance License')['Number of Comments'].reindex()):  cnt\_drama[int(i.split("/")[1])-1]+=1  cnt\_drama\_comments[int(i.split("/")[1])-1]+=j  #print(dtret[dtret['Score'] == 0])  month = [i for i in range(1,13)]  plt.bar(month,cnt\_drama)  plt.show()  dt = pd.read\_excel('result.xlsx','Sheet1')  comment\_max = dt['Number of Comments'].max()  Score\_max = dt['Score'].max()  Episode\_max = dt['Episode'].max()  Info\_total = []  for i in dt.index:  Info\_new = []  for j in dt.columns:  if j == 'TV Drama':  Info\_new.append(dt[j][i])  if j == 'Score':  try:  a = float(dt[j][i])  except:  a = 0  Info\_new.append(a)  elif j == 'Number of Comments':  try:  a = float(dt[j][i]/comment\_max)  except:  a = 0  Info\_new.append(dt[j][i])  Info\_new.append(a)  elif j == 'Episode':  try:  a = float(dt[j][i]/Episode\_max)  except:  a = 0  Info\_new.append(dt[j][i])  Info\_new.append(a)  Info\_total.append(Info\_new)  for i in Info\_total:  Rscore = i[1]\*i[2] - 0.2 \* i[3]  i.append(Rscore)  Info\_dt = pd.DataFrame(Info\_total)  Info\_dt.columns = ['TV Drama','Score','Number of Comments','Chs','Episode','Ehs','RScore']  print(Info\_dt.sort\_values(*by*='RScore',*ascending*=False).head(10)) |

|  |
| --- |
| 明星筛选 |
| import os  import shutil  import re  from bs4 import BeautifulSoup  listdir = os.listdir(os.getcwd())  if not os.path.exists("hm"):  if not os.path.isdir("hm"):  os.makedirs(os.path.join(os.getcwd(),"hm"),*mode*=0o777)  print("Success create the html folder")  else:  print("Warnning: Html dir is already existed!")  else:  print("Warnning: Html is already existed!")  try:  for name in listdir:  if re.match(r'.\*\_hm.txt',name) is not None:  shutil.move(name,"hm/")  print (name + "-->" + os.getcwd()+"/hm/"+name,end="\n")  print("Move files successfully")  except:  print("Already done!")  pass  stars\_bd\_file\_list = os.listdir("html")  for file in stars\_bd\_file\_list:  try:  with open(os.path.join(os.getcwd(),"html",file),"r") as f:  line = 0  try:  for i in f.readlines():  line+=1  except:  pass  print("length:"+str(line))  if line <= 10:  f.close()  os.remove(os.path.join(os.getcwd(),"html",file))  continue  bsdt = BeautifulSoup(f.read(),"html.parser",*from\_encoding*='utf-8')  try:  for i in bsdt.find(*attrs*={"class":re.compile("hint\_[a-zA-Z0-9]{0,9} c\_font\_[a-zA-Z0-9]{0,9}")}):  print(i)  except:  print(file)  pass  except:  print("utf-8"+file) |

|  |
| --- |
| 百度信息搜集 |
| import requests  import time  import pandas as pd  import numpy as np  import os  import json  name = "白雪"  url = r'http://index.baidu.com/api/FeedSearchApi/getFeedIndex?word=[[{"name":"'+name+'","wordType":1}]]&area=0&days=30'  print(url)  headers = {  "Host": 'index.baidu.com',  'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:94.0) Gecko/20100101 Firefox/94.0',  'Accept': 'application/json, text/plain, \*/\*',  'Accept-Language': 'zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2',  'Accept-Encoding': 'gzip, deflate, br',  'Referer': 'https://index.baidu.com/v2/main/index.html',  'Connection': 'keep-alive',  'Cookie': 'BAIDUID=6EE4B14CF0C5FFBD8BC51FE2BD5CC1A2:FG=1; BIDUPSID=6EE4B14CF0C5FFBDCDDAA3478440FB86; PSTM=1628165068; \_\_yjs\_duid=1\_6bfdb0ac0e3760480bbcac4da6e828871628176157265; BDUSS=X5NT1F4RjA2UmN4WnBOaVNkdX5QOC1iM2Nad3kyMmEzVUFmMmgxM1AzRUw4Mk5oRVFBQUFBJCQAAAAAAAAAAAEAAACPk8sQt67U88jwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAtmPGELZjxhV; BDORZ=FFFB88E999055A3F8A630C64834BD6D0; H\_PS\_PSSID=35295\_35104\_31254\_35435\_35456\_34584\_34504\_34606\_35329\_35315\_26350; BDSFRCVID=hFtOJexroG04sxcH4wwF-tse4ec\_23JTDYLEL5mveYl6oVFVJeC6EG0Ptq6Ex2P-EHtdogKKy2OTH9DF\_2uxOjjg8UtVJeC6EG0Ptf8g0M5; H\_BDCLCKID\_SF=tbAOVCD2JKL3fP36q4OHht\_tMq0X5-RLf273ah7F5l8-hCb6b-IKXUv3etoMeR3e5RnB3qrCBD3xOKQphUIhyjO-04FLBf0qbCJLLnbN3KJm8tP9bT3vLfuyjtcz2-biW5KH2MbdflbP\_IoG2Mn8M4bb3qOpBtQmJeTxoUJ25DnJhbLGe6L3-RJH-xQ0KnLXKKOLVKDyWl7ketn4hUt2jbtb-PAJ-bbjH6ne0CbHKMAKhpc2QhrdQf4WWb3ebTJr32Qr-qcH0KQpsIJM5bL-QJvX02c4Bn3baKvia-I-BMb1bqbDBT5h2M4qMxtOLR3pWDTm\_q5TtUJMeCnTDMFhe6oM-Uvy-40XKD600PK8Kb7VbnnD5MnkbJkXhPtjtnJOK2-JL-3IH4b5eDbN-pPKDUI7Qbrr0xAta6c-hR4KaMQPSlcNLTjpQT8r5hLq0RkHLPj4aPbeab3vOPI4XpO1ef0zBN5thURB2DkO-4bCWJ5TMl5jDh3Mb6ksD-FtqtJHKbDH\_KPMtMK; ab\_sr=1.0.1\_Mjk1MjAzY2E5NTZkODAxY2NiNDQ5YmY4N2U5NzExOWNmZmQ0OWE3YTM4ZGUxMmI1MzNjMTYxMDIxMWQxZWU3NmU5MTFlYzc2NTQ0MmQzYjZmZDVjNzQwMzE5NjQ4Yjk2ZjFmMmI3NGJhMGQ4NTlmN2FiZmMyOWIwYzgxZTI0NjAwMjkyYTBlODA3ODU2NjNkN2QyNTRhMDc1M2E0M2FhMA==; BA\_HECTOR=a021248020058524cd1gr6nok0r; delPer=0; PSINO=6; BDRCVFR[gltLrB7qNCt]=mk3SLVN4HKm; Hm\_lvt\_d101ea4d2a5c67dab98251f0b5de24dc=1639145239; Hm\_lpvt\_d101ea4d2a5c67dab98251f0b5de24dc=1639145471; RT="z=1&dm=baidu.com&si=6rc7piqj5be&ss=kx0gr626&sl=d&tt=6o8&bcn=https%3A%2F%2Ffclog.baidu.com%2Flog%2Fweirwood%3Ftype%3Dperf&ld=5109"; \_\_yjs\_st=2\_ZmE0MWQ4MjVkYmQwMzcyYWZlNGYzNDY0MGFlMGNlMTdiZTBmZDIyNzNhZTQyNWM0NWQ1OTRlZDE0MjFlNzhhMWU5OWUyZDdmZjE3MTgxNjBhOWY5YTNiMGQxYjIzYTU5ZDRmOWRhMTgyOWFkODFkOGMzOTQ0NTE5N2IxZGFmODY1YzUzM2NiZDUwOGE2YTVkMzQ5YzljMDk2Y2RkZmQwY2QwNDVhNTNmNmYyZWQ1Zjk4YWMzYTRiODM4NDAzMjNhNDdlZmUwMjE4N2M4YjY3Y2JkMzMzMjBlOWZhMDQ1Yzk0YmNmYzlkM2EwZWU3NjA0Zjg5YTgzNGMwZGQ2NWE3MmJkMzYxNGRiNjZlZDJiOGM2NGMwNzAwNzgxZWZlMzMzXzdfNDNkM2FjNmQ=; bdindexid=vo1c73ivop6an83hkaavraogl3',  'Sec-Fetch-Dest': 'empty',  'Sec-Fetch-Mode': 'no-cors',  'Sec-Fetch-Site': 'same-origin',  'Pragma': 'no-cache',  'Cache-Control': 'no-cache'  }  dt = pd.read\_excel('result.xlsx','Sheet1')  stars = set()  for names in dt['Starring']:  if names is not np.nan:  for name in str(names).replace('、',',').replace('，',',').split(','):  stars.add(name)  total\_file = open("summary\_hm.txt","a+")  total\_file.seek(0,0)  for name in stars:  time.sleep(1)  try:  if os.path.exists(name+'\_hm.txt'):  continue  with open(name+'\_hm.txt','w') as f:  time.sleep(1)  url = r'http://index.baidu.com/api/FeedSearchApi/getFeedIndex?word=[[{"name":"'+name+'","wordType":1}]]&area=0&startDate=2021-11-10&endDate=2021-12-09'  print(url)  r = requests.get(*url* = url,*headers* = headers)  r.raise\_for\_status()  r.encoding = r.apparent\_encoding  ret = str(r.text)  ret = json.loads(ret)  print(name+','+str(dict(ret['data'])['index'][0]['generalRatio'])+'Success')  f.write(name+','+str(dict(ret['data'])['index'][0]['generalRatio'])+'\n')  total\_file.write(name+','+str(dict(ret['data'])['index'][0]['generalRatio'])+'\n')  except:  print(name + "Failure")  continue  # for name in stars:  # time.sleep(1)  # try:  # if os.path.exists(name+'\_hm.html'):  # print(name,"Success")  # continue  # with open(name+'\_hm.html','w') as f:  # time.sleep(1)  # url = r'http://index.baidu.com/api/FeedSearchApi/getFeedIndex?word=[[{"name":"'+name+'","wordType":1}]]&area=0&days=30'  # r = requests.get(url = url,headers = headers)  # r.raise\_for\_status()  # r.encoding = r.apparent\_encoding  # ret = str(r.text)  # print(ret)  # f.write(ret)  # print(name,"Success")  # except:  # print(name,"Failure")  # continue |

|  |
| --- |
| 新浪微博信息收集 |
| import pandas as pd  import numpy as np  import os  import time  import matplotlib.pyplot as plt  from bs4 import BeautifulSoup  import requests  from selenium.webdriver import Chrome  from selenium.webdriver.chrome.options import Options  from selenium.webdriver.support.wait import WebDriverWait  from selenium.webdriver import \*  from selenium import webdriver  from selenium.common.exceptions import TimeoutException  from selenium.webdriver.support.wait import WebDriverWait  from urllib.parse import quote  from pyquery import PyQuery as pq  dt = pd.read\_excel('result.xlsx','Sheet1')  stars = set()  for names in dt['Starring']:  if names is not np.nan:  for name in str(names).replace('、',',').replace('，',',').split(','):  stars.add(name)    url = "https://s.weibo.com/weibo?q="  headers = {  "Host": "s.weibo.com",  "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,\*/\*;q=0.8",  "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2",  "Accept-Encoding": "gzip, deflate, br",  "Connection": "keep-alive",  "Cookie": "www.52jingsai.com,widget.weibo.com,www.baidu.com; SUBP=0033WrSXqPxfM725Ws9jqgMF55529P9D9WWbjVd7EgcvSe75XM2kBhOX5JpX5KMhUgL.Foe4eh2pSo.7S0-2dJLoIp7LxKML1KBLBKnLxKqL1hnLBoM01K5peKq4ehMf; SINAGLOBAL=8505429497315.529.1632922843183; ULV=1639136383200:2:1:1:3763013819565.094.1639136383093:1632922843184; ALF=1670672351; SCF=AoMImLiEPADgRCx1ffu6aj6AK92GFULp5oaCrYYu3UyFQ1CaK7GE8lZnOmRTsnXxymUaky46S2SrG5WmaKwfqDc.; SUB=\_2A25Mt0wwDeRhGeVH61MQ9ifMzDmIHXVvxTr4rDV8PUNbmtAKLUGtkW9NT2jo5JhLkwNEFUwWQ\_iTWmiOS-eqHHhv; SSOLoginState=1639136352; \_s\_tentry=www.baidu.com; Apache=3763013819565.094.1639136383093",  "Upgrade-Insecure-Requests": "1",  "Sec-Fetch-Dest": "document",  "Sec-Fetch-Mode": "navigate",  "Sec-Fetch-Site": "cross-site",  "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:94.0) Gecko/20100101 Firefox/94.0",  }  # for name in stars:  # time.sleep(1)  # try:  # if os.path.exists(name+'\_sina.html'):  # continue  # # with open(name+'\_sina.html','w') as f:  # time.sleep(1)  # r = requests.get(url = url+name,headers = headers)  # r.raise\_for\_status()  # r.encoding = r.apparent\_encoding  # ret = str(r.text)  # print(ret)  # #f.write(ret)  # except:  # continue  for name in stars:  time.sleep(1)  try:  if os.path.exists(name+'\_sina.html'):  print(name,"Success")  continue  with open(name+'\_sina.html','w') as f:  time.sleep(1)  r = requests.get(*url* = url+name,*headers* = headers)  r.raise\_for\_status()  r.encoding = r.apparent\_encoding  ret = str(r.text)  # print(ret)  f.write(ret)  print(name,"Success")  except:  print(name,"Failure")  continue |

|  |
| --- |
| 新浪微博文件处理 |
| from bs4 import BeautifulSoup  import os  import re  list\_file = os.listdir(os.getcwd())  for file\_name in list\_file:  if re.match(r'.\*.\_sina.html',file\_name) is None:  list\_file.remove(file\_name)  total\_fans\_cnt = open("total\_fans\_cnt.txt","a+")  total\_fans\_cnt.seek(0,0)  for file in list\_file:  try:  with open(file,"r") as f:  name = file.split('\_')[0]  bsdt = BeautifulSoup(f.read(),"html.parser",*from\_encoding*="utf-8")  if bsdt.find("span",*attrs*={'class':'s-nobr'}) is not None:  print(name+" "+str(bsdt.find("span",*attrs*={'class':'s-nobr'}).contents[0]))  total\_fans\_cnt.write(name+" "+str(bsdt.find("span",*attrs*={'class':'s-nobr'}).contents[0])+'\n')  except:  continue  print(list\_file) |

|  |
| --- |
| 明星排位 |
| import os  import re  import json  import ast  filelist = os.listdir(os.getcwd())  for i in filelist:  if re.match(r'.\*\_hm.txt',i) is None:  filelist.remove(i)    total = []  for file in filelist:  with open(file,"r") as f:  for line in f.readlines():  splited = line.split(",")  str1 = "".join(','+splited[i] for i in range(1,len(splited)))  ret = ast.literal\_eval(str(str1[1:]).replace(" ","").replace('\n',''))  ret.update({'name':splited[0]})  print(ret)  total.append(ret)    for i in sorted(total,*key*=lambda *x*:x['avg'],*reverse*=True)[:34]:  print(i['name']) |