**广州商学院**

**课程论文**

**题目：电视节目推荐模型构建与评价**

|  |  |
| --- | --- |
| 课 程 名 称 | **商业智能应用案例** |
| 考 查 学 期 | **2018-2019学年第二学期** |
| 考 查 方 式 | **课程论文** |
| 姓 名 | **杨伟庭** |
| 学 号 | **201606050022** |
| 专 业 | **信息管理与信息系统** |
| 成 绩 |  |
| 指 导 教 师 | **吴晓玲** |

摘要

随着数字电视和通信技术的不断发展,电视节目资源越来越丰富。一方面用户为能够收看到如此之多的节目而感到兴奋不已，另一方面又为如何从成百上千个节目中找到他们真正喜爱的节目，而感到苦恼。

本论文内容为介绍了几种智能推荐算法并构建基于用户收视的电视节目推荐和基于电视节目产品的电视节目推荐,对数据进行预处理之后使用智能推荐算法进行建模为用户推荐电视节目。然后对模型进行评价和模型之间的对比。

**关键词：推荐算法;收视率;电视节目;协同过滤;Python;**

Abstract

With the continuous development of digital television and communication technology, TV program resources are becoming more and more abundant. On the one hand, users are excited to see so many programs, and on the other hand are distressed about how to find the programs they really love from hundreds of programs. The content of this thesis introduces several intelligent recommendation algorithms and constructs TV program recommendation based on user ratings and TV program recommendation based on TV program products. After pre-processing the data, the intelligent recommendation algorithm is used to model the recommended TV programs for users. The model is then evaluated and compared between the models.

**Key words**: Recommended algorithm; ratings; TV programs; collaborative filtering; Python;

目录

[摘要 II](#_Toc12431916)

[Abstract III](#_Toc12431917)

[目录 IV](#_Toc12431918)

[1 绪论 1](#_Toc12431919)

[1.1 背景 1](#_Toc12431920)

[1.2 挖掘目标 2](#_Toc12431921)

[1.3 模型构建流程 2](#_Toc12431922)

[2 数据预处理 4](#_Toc12431923)

[2.1 数据概况 4](#_Toc12431924)

[2.2 数据清洗 5](#_Toc12431925)

[3 智能推荐算法的构建 12](#_Toc12431926)

[3.1 推荐算法介绍 12](#_Toc12431927)

[3.2 划分数据集 12](#_Toc12431928)

[3.3 基于用户的协同过滤算法的模型构建与评价 12](#_Toc12431929)

[3.4 基于物品的协同过滤算法的模型构建与评价 15](#_Toc12431930)

[3.5 模型比较与总结评价 17](#_Toc12431931)

[总结与展望 19](#_Toc12431932)

[参考文献 19](#_Toc12431933)

1. 绪论
   1. 背景

信息数字化已经是不可逆转的趋势，电视广播也不例外,随着电视广播的信息化，家庭能收看到的电视频道和电视节目的数量也随之增加。电视节目一直以来是家庭娱乐中重要的一环.随着电视节目的数量和电视台频道在互联网时代下越来越多地工人们选择，给人们带来便利的同事，人们面对种类繁多的电视台和电视节目也或许会犯下选择困难.选择多了，人自然而然地不可能知道所有的选择是否适合自己是否值得观看。

以此为契机,智能推荐算法的应用显得尤为重要.运用智能推荐算法，基于用户的行为或者基于电视节目进行建模，分析用户的行为或者电视节目的相似性，对用户进行电视节目的推荐,既可以方便电视用户对节目的选择也可以达到电视节目的推广的目的,可谓是电视节目和用户的双赢。

* 1. 挖掘目标

本文主要利用提供的收视数据，针对电视节目推荐的特点：

1. 基于电视节目产品，利用基于物品的协同过滤为用户推荐具体的电视节目。
2. 基于用户点播观看行为，利用基于用户的协同过滤为用户推荐具体电视节目。
   1. 模型构建流程

首先对数据进行预处理，将数据转换成可量化计算的形式，再分析从用户和产品两个角度实现标签化。用户的收视行为能够侧面反映出用户的性别、年龄等特征，能通过用户的收视时段等特征判断用户的家庭构成，电视产品也有针对某类用户的特性。

协同过滤

基于用户

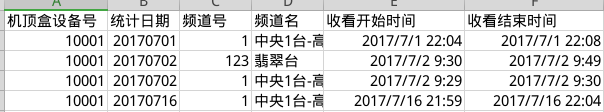
基于物品

数据

数据清洗

1. 数据预处理
   1. 数据概况

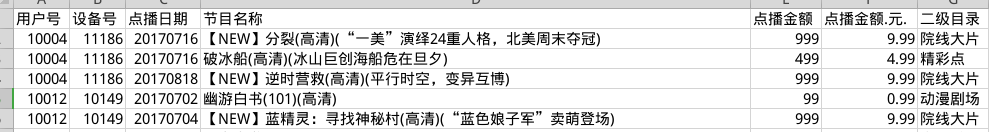
在附件1：用户收视信息.xlsx中有四个单元表,分别为用户收视信息、用户回看信息、用户点播信息、用户单片点播信息。



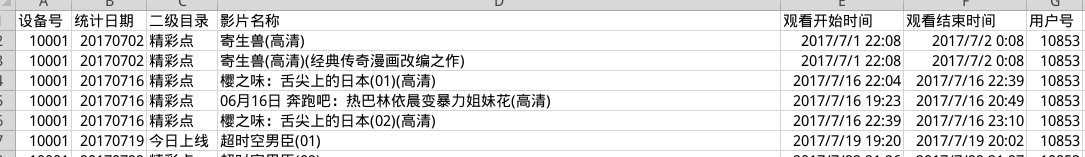
图表 2-1用户收视信息表字段



图表 2-2用户回看信息表字段



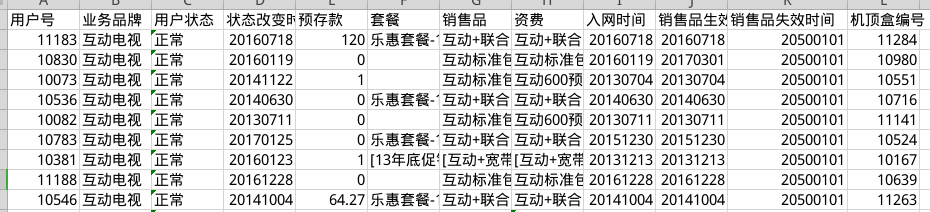
图表 2-3用户点播信息表字段



图表 2-4用户单片点播信息表字段

在附件2\_电视产品信息数据.csv 中是电视产品信息数据，记录了电视产品的敏子，创建日期，导演，演员,导演，出品年代，总集数，分类等等的信息。

附件3\_用户基本信息.csv是用户所使用的业务品牌，入网时间，套餐等信息。

.

图表 2-5用户基本信息表字段

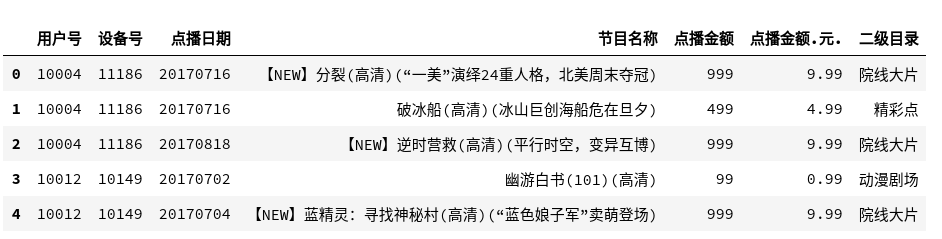
* 1. 数据清洗

2.2.1 脏数据汇总

图表 2-2-1用户收视信息表



图表 2-2-2用户回看信息表



图表 2-2-3用户点信息播表



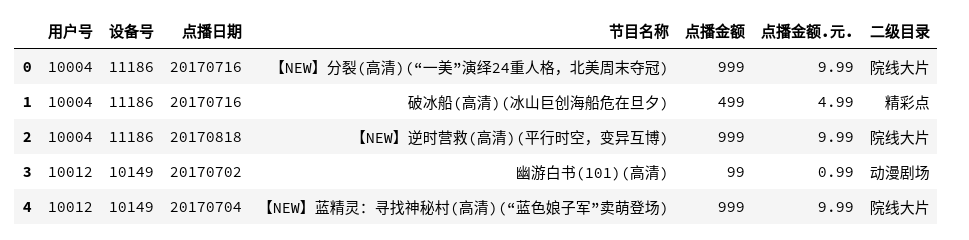
图表 2-2-4用户单片点信息播表

上面四图可看出各个数据中有很多重复的数据,例如频道号和频道名，并且频道名后面有“高清”和“[NEW]”字样,有的没有,因此要处理掉“高清”字样避免对后面的模型的准确度造成影响。



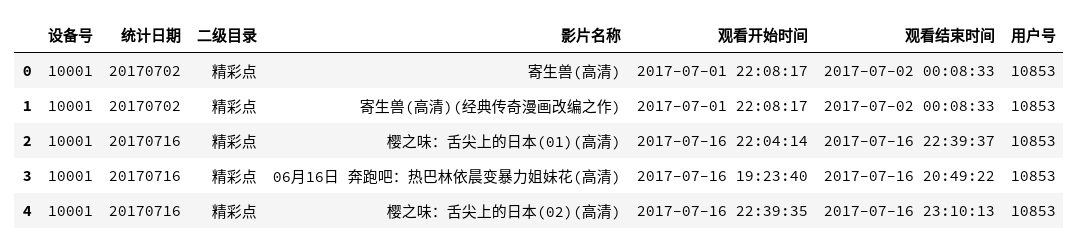
图表 2-2-5用户回看信息表

用户回看信息表也有重复值。



图表 2-2-6用户点播信息表

用户点播信息表也有重复的值,节目名称中有多余的没有意义的字符。



图表 2-2-7用户单片点播信息表

用户单片点播信息表中同样存在重复值，并且影片名称里面的数据有多余的字符.而且部分影片名有乱码的情况。

电视产品信息数据表中地区参数数据中“大陆 大陆”要改称“大陆”,“大陆、日本”要改成“日本”，“中国”要改成“大陆”，声道数据列中“英语语”改为“英语”,“普通话”改为“国语”，“粤语 粤语”改为“粤语”，“中文”改为“国语”,字幕语种中:“国语”为“中文”，“中文 中文”为“中文”，分类名称中“电视剧场 电视剧场\大陆剧场”为“电视剧场\大陆剧场”。

用户收视信息去重：

huikan = huikan.drop\_duplicates(['机顶盒设备号','统计日期','收看开始时间','收看结束时间'])

huikan.head()

对频道号做调整：

huikan['频道名'] = pd.Categorical(huikan['频道名'])

huikan['新频道号'] = huikan['频道名'].cat.codes



图表 2-2-8用户收视信息未清理



图表 2-2-9用户收视信息已清理

用户回看信息去重：

huikan = huikan.drop\_duplicates(['设备号','统计时间','用户号','频道','回看开始时间','回看结束时间'])

print(huikan.shape)

huikan.head()

重新计算回看时长：

huikan['新回看时长'] = (huikan['回看结束时间'] - huikan['回看开始时间']).values/np.timedelta64(1, 's')huikan.head()

去掉回看时间16小时以上的内容：

huikan = huikan[huikan['新回看时长'].apply(lambda x:True if x<=16\*3600 else False )]



图表 2-2-10用户回看信息未清理



图表 2-2-11用户回看信息已清理

用户点播信息处理：

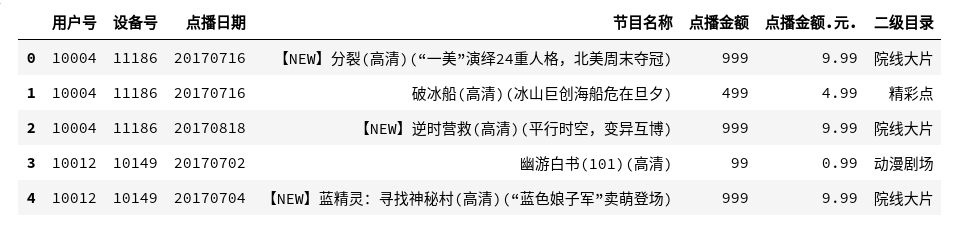
dianbo['节目名称'] = dianbo['节目名称'].apply(remove\_episode\_number)

dianbo['节目名称'] = dianbo['节目名称'].apply(remove\_high\_resolution)

dianbo['节目名称'] = dianbo['节目名称'].apply(remove\_inner\_bracket)

dianbo['节目名称'] = dianbo['节目名称'].apply(remove\_NEW)

dianbo.head()



图表 2-2-12用户点播信息未处理



图表 2-2-13用户点播信息已处理

用户单片点播信息去重,影片名称处理：

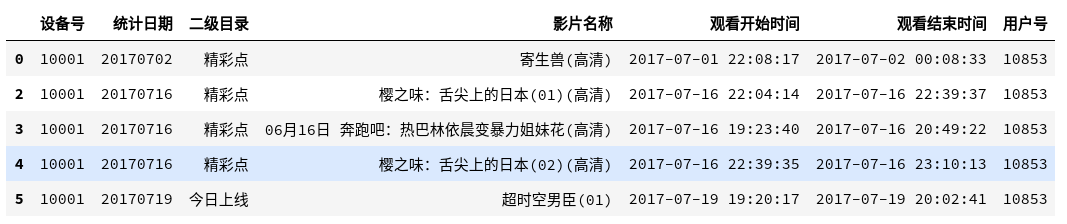
danpian = danpian.drop\_duplicates(['设备号','统计日期','二级目录','观看开始时间','观看结束时间','用户号'])

danpian['影片名称'] = danpian['影片名称'].apply(remove\_episode\_number)

danpian['影片名称'] = danpian['影片名称'].apply(remove\_high\_resolution)

danpian['影片名称'] = danpian['影片名称'].apply(remove\_inner\_bracket)

danpian['影片名称'] = danpian['影片名称'].apply(remove\_NEW)



图表 2-2-14用户单片点播信息未处理



图表 2-2-15用户垫片点播信息已处理

电视产品信息表处理:

def area\_stand(strr):

if strr == '大陆 大陆':

strr = '大陆'

elif strr == '大陆、日本':

strr = '日本'

elif strr == '中国':

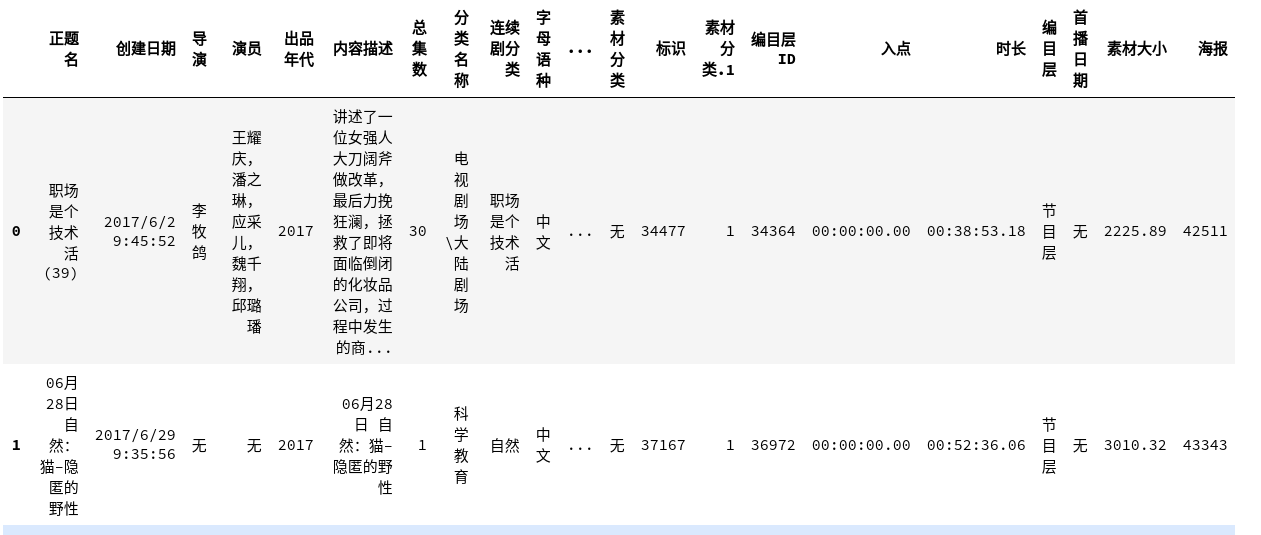
strr = '大陆'

elif strr == '其他国家':

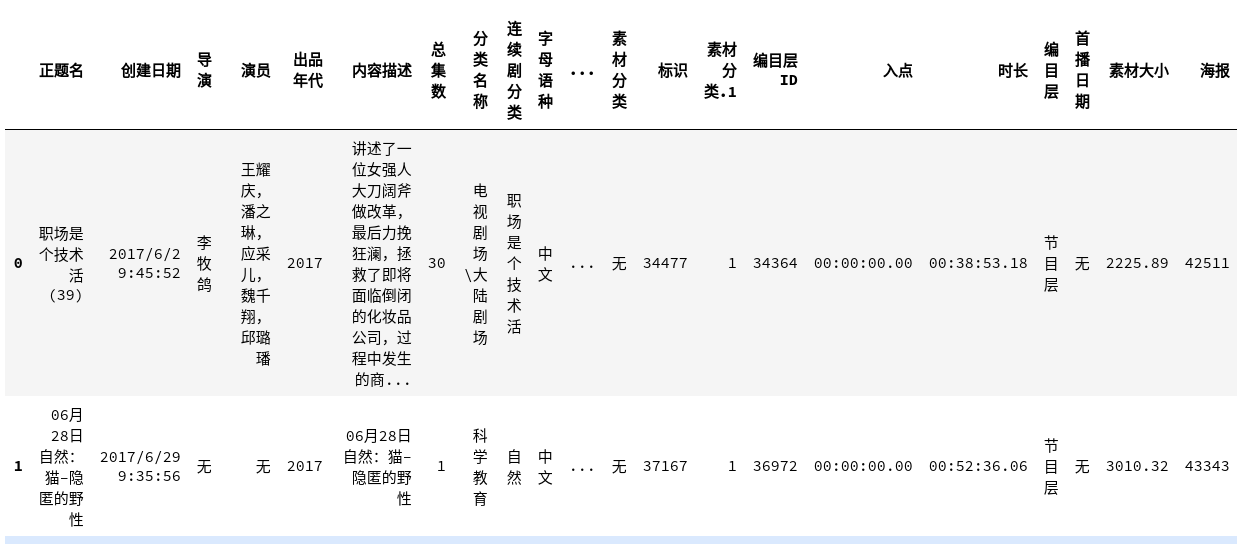
strr = '其他'

return strr

jiemu['地区参数'] = jiemu['地区参数'].apply(area\_stand)



图表 2-2-16电视节目信息未处理



图表 2-2-17电视节目信息已处理

from datetime import date

today = '20170930'

print(today)

user['当前套餐使用天数'] = (datetime.strptime(today, '%Y%m%d') - pd.to\_datetime(user['状态改变时间'],format="%Y%m%d")).astype('timedelta64[D]')

user['入网天数'] = (datetime.strptime(today, '%Y%m%d') - pd.to\_datetime(user['入网时间'],format="%Y%m%d")).astype('timedelta64[D]')



图表 2-2-18用户基本信息未处理



图表 2-2-19用户基本信息已处理

1. 智能推荐算法的构建
   1. 推荐算法介绍

协同过滤算法是与内容无关的，即不需要额外获取分析用户或物品的内容属性特征。是基于用户历史行为数据进行推荐的算法。其通过分析用户与物品间的联系来寻找新的用户与物品间的相关性。

该算法算法通常有两个过程，一个过程是预测，另一个过程是推荐。主流的协同过滤算法包括三种:基于用户的协同过滤、基于物品的协同过滤和基于模型的协同过滤。

1. 基于用户的协同过滤算法  
    基于用户的协同过滤推荐算法，先通过用户历史行为数据找到和用户u相似的用户，将这些用户感兴趣的且u没有点击过的物品推荐给用户。算法主要包括以下两个步骤:  
    (1)找到与目标用户喜好相似的邻居用户集合。  
    (2)在邻居用户集合中，为用户推荐其感兴趣的物品。

基于物品的协同过滤算法其主要思想是，为用户推荐那些与他们之前喜欢或点击过的物品相似的物品。不过基于物品的协同过滤算法并不是利用物品的内容属性特征来计算物品之间的相似度的。该类算法是利用用户的历史行为数据计算待推荐物品之间的相似度。在该类算法中，如果喜欢物品A的用户大都也喜欢物品B，那么就可以认为物品A和物品B之间的相似度很高。算法分为以下两个步骤:  
 (1)根据用户历史行为数据，计算物品间的相似度。  
 (2)利用用户行为和物品间的相似度为用户生成推荐列表。

本文将运用基于用户和基于物品的协同过滤算法进行建模.因为数据源有用户的点播和收视记录数据,可以方便地进行基于用户的协同过滤实现电视节目的推荐.而数据中也有丰富的电视节目内容,也比较方便地进行基于物品的协同过滤算法实现电视节目的推荐。

* 1. 划分数据集

十字交叉检验法。

* 1. 基于用户的协同过滤算法的模型构建与评价

计算单个点播某一节目的频次：

dianbo = dianbo.sort\_values(['用户号','节目名称'])

user\_only = np.sort(dianbo['用户号'].unique()) #用户号排序

jiemu\_only = np.sort(dianbo['节目名称'].unique()) #节目名称唯一

jiemu\_pinci = pd.DataFrame(index=list(user\_only), columns=list(jiemu\_only)) #矩阵中用户序列为行、节目序列为列

#收看对应节目的频次

for i in user\_only:#在用户中循环

user\_program = dianbo[dianbo['用户号']==i]

temp = pd.Series.to\_dict(user\_program['节目名称'].value\_counts()) #计算分类得分

for j in program\_unique\_num:#在节目中循环

if j in temp:

jiemu\_pinci[j][i] = temp[j]

计算用户之间的相似度：

temp1 = jiemu\_pinci.T #频次

jiemu\_pincit\_cor1 = temp1.corr() # 皮尔逊相关系数

jiemu\_pinci\_cor2 = temp1.corr('kendall') # 肯德尔等级相关系数

jiemu\_pinci\_cor3 = temp1.corr('spearman') # 斯皮尔曼等级秩相关

计算节目评分：

indexx = temp1.index.values.tolist()

features = temp1.columns.values.tolist()

temp11 = pd.DataFrame()

for i in features:#特征评分

for j in indexx:#用户

if temp1[i][j] > 0:

temp11 = temp11.append(pd.DataFrame([j,i,temp1[i][j]]).T, ignore\_index=True)

temp11[0] = pd.Categorical(temp11[0])

temp11['节目号'] = temp11[0].cat.codes #分类

print(temp11['节目号'].unique().shape)

temp11.columns = ['节目名','用户号','得分','节目号']

temp11.head()

import usercf

from sklearn.model\_selection import KFold

kfold=KFold(n\_splits=10,shuffle=True,random\_state=1024)#10折交叉验证划分数据集

for train,test in kfold.split(full\_data):

user\_cf\_model = usercf.UserBasedCF() #利用usercf.py

user\_cf\_model.generate\_dataset(full\_data.iloc[train],full\_data.iloc[test])

user\_cf\_model.calc\_user\_sim()

acc\_list = []

for user in user\_cf\_model.trainset:

tmp\_rs = np.array([(user, a, b) for a,b in user\_cf\_model.recommend(user)])

if tmp\_rs.shape[0] > 0:

acc\_list.append(tmp\_rs)

final\_df = pd.DataFrame(np.concatenate(acc\_list), columns=['UserNo', 'ProgramNo', 'RecommendScore'])

final\_df['UserNo'] = final\_df.UserNo.astype('int32')

final\_df['ProgramNo'] = final\_df.ProgramNo.astype('int32')

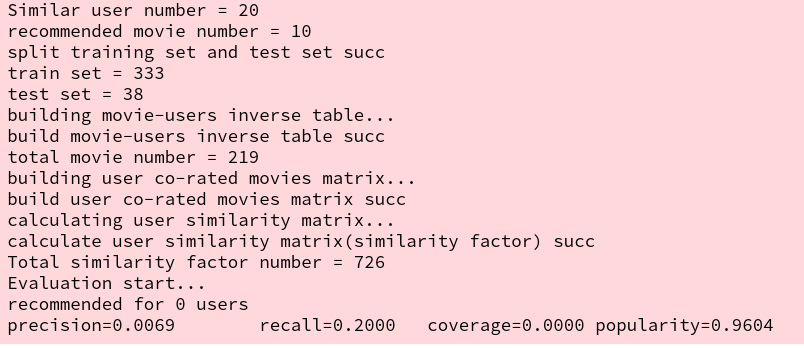
final\_df = final\_df.merge(movie\_df, left\_on='ProgramNo', right\_on='index')

del final\_df['index']

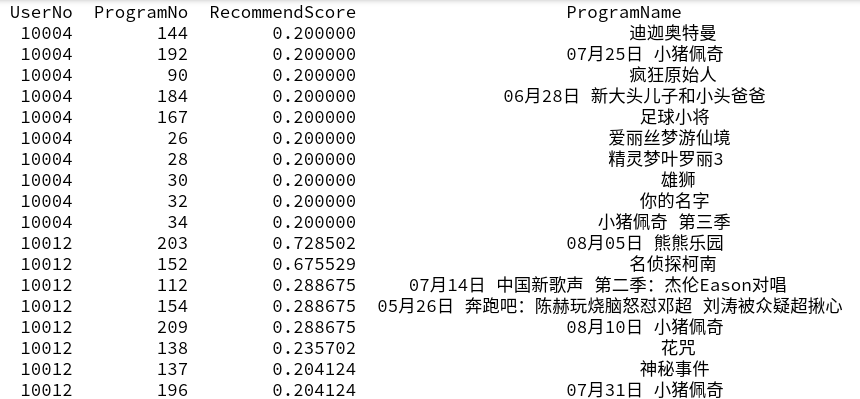
final\_df = final\_df.sort\_values(['UserNo', 'RecommendScore'], ascending=[True, False])

print(final\_df)

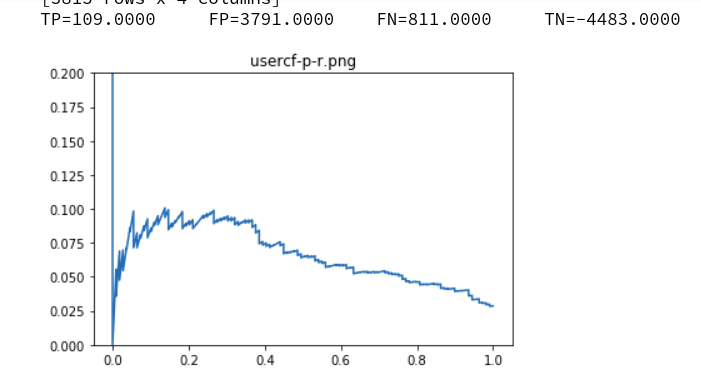
print(user\_cf\_model.evaluate()) #模型评估



图表 3-3-1基于用户的协同过滤评估



图表 3-3-2部分用户推荐结果



图表 3-3-3基于用户pr曲线

* 1. 基于物品的协同过滤算法的模型构建与评价

temp1 = jiemu\_pinci #频次

jiemu\_pinci\_cor1 = temp1.corr() # pearson相关系数

jiemu\_pinci\_cor2 = temp1.corr('kendall') # Kendall Tau相关系数

jiemu\_pinci\_cor3 = temp1.corr('spearman') # spearman秩相关

indexx = temp1.index.values.tolist()

features = temp1.columns.values.tolist()

temp11 = pd.DataFrame()

for i in features:

for j in indexx:

if temp1[i][j] > 0:

temp11 = temp11.append(pd.DataFrame([j,i,temp1[i][j]]).T, ignore\_index=True)

temp11[1] = pd.Categorical(temp11[1])

temp11['ProgramNo'] = temp11[1].cat.codes

temp11.columns = ['UserNo','ProgramName','Score','ProgramNo']

temp11.head()

import itemcf0422

from sklearn.model\_selection import KFold

kfold=KFold(n\_splits=10,shuffle=True,random\_state=1024)#10折交叉验证划分数据集

for train,test in kfold.split(full\_data2):

item\_cf\_model = itemcf0422.ItemBasedCF() #利用itemcf.py

item\_cf\_model.generate\_dataset(full\_data2.iloc[train],full\_data2.iloc[test])

item\_cf\_model.calc\_movie\_sim()

acc\_list = []

for user in item\_cf\_model.trainset:

tmp\_rs = np.array([(user, a, b) for a,b in item\_cf\_model.recommend(user)])

if tmp\_rs.shape[0] > 0:

acc\_list.append(tmp\_rs)

final\_df = pd.DataFrame(np.concatenate(acc\_list), columns=['UserNo', 'ProgramNo', 'RecommendScore'])

final\_df['UserNo'] = final\_df.UserNo.astype('int32')

final\_df['ProgramNo'] = final\_df.ProgramNo.astype('int32')

final\_df = final\_df.merge(movie\_df2, left\_on='ProgramNo', right\_on='index')

del final\_df['index']

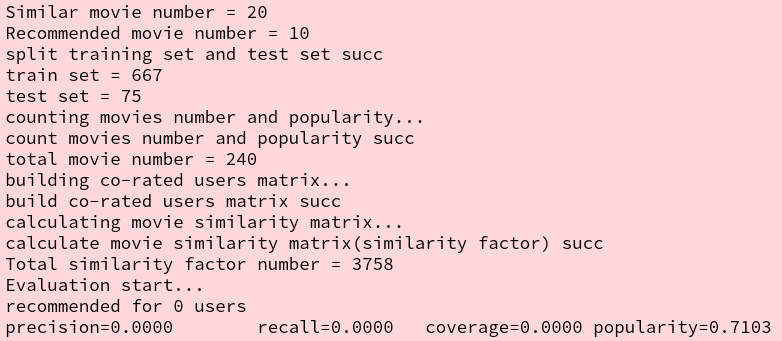
final\_df = final\_df.sort\_values(['UserNo', 'RecommendScore'], ascending=[True, False])

print(final\_df)

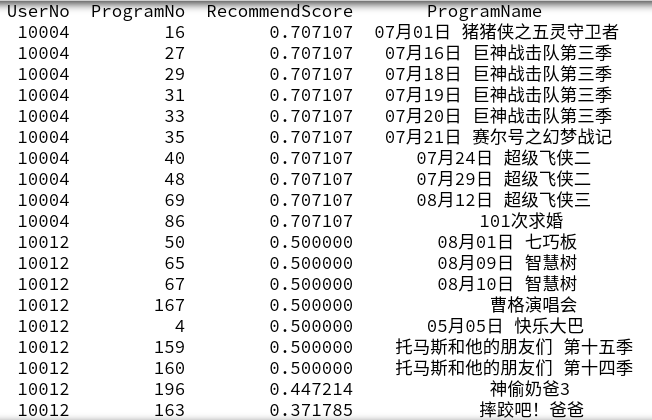
print(item\_cf\_model.evaluate()) #模型评估

%pylab inline

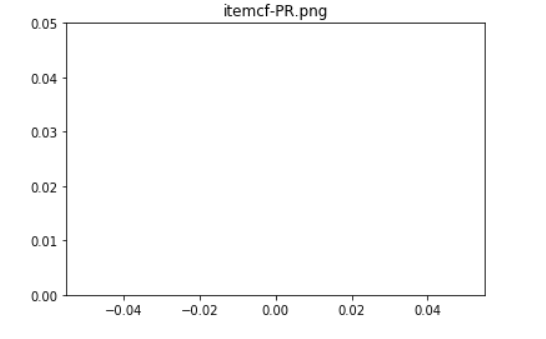
from IPython.display import Image



图表 3-4-1基于物品协同过滤评估



图表 3-4-2基于物品协同过滤推荐结果



user\_cf\_model = usercf.UserBasedCF() #利用usercf.py

图表 3-4-5基于电视节目的pr曲线



* 1. 模型比较与总结评价

通过两个模型的比较可以发现基于用户的推荐要比基于物品的推荐的效果和准确率要高的多，而基于物品的推荐中甚至存在推荐不出电视节目的情况。

对比对多个用户推荐的pc曲线可发现基于用户协同过滤的推荐模型相对来说还是比较高准确度的。

基于用户的协同过滤推荐算法先使用统计技术寻找与目标用户有相同喜好的邻居，然后根据目标用户的邻居的喜好产生向目标用户的推荐。基本原理就是利用用户访问行为的相似性来互相推荐用户可能感兴趣的资源。基于用户的协同过滤算法适用于用户较少的场合，如果用户很多，计算用户相似度矩阵代价很大,适合效性较强，用户个性化兴趣不太明显的领域。

优缺点：群体/个体：更依赖于当前用户相近的用户群体的社会化行为。计算代价：适用于用户数较少的场合。适用场景：时效性强，用户个性化兴趣不太显著的场合。冷启动：新加入的物品能很快进入推荐列表。实时性：用户新的行为不一定导致推荐结果的变化。

itemCF适用于物品数明显小于用户数的场合，否则物品相似度矩阵计算代价很大，适合长尾物品丰富，用户个性化需求强的领域，对新用户友好，对新物品不友好，因为物品相似度矩阵不需要很强的实时性利用，用户历史行为做推荐解释，比较令用户信服

因此，可以看出 UserCF 适用于物品增长很快，实时性较高的场合，比如新闻推荐。而在图书、电子商务和电影领域，比如京东、天猫、优酷中，ItemCF 则能极大地发挥优势。在这些网站中，用户的兴趣是比较固定和持久的，而且这些网站的物品更新速度不会特别快，一天一更新是在忍受范围内的。

## 总结与展望

基于用户和基于物品的协同过滤推荐算法各有各的优势，各有各的应用场景，应该根据具体的应用场景和数据来综合考虑使用哪一种或者结合两种算法来决绝推荐电视节目的问题。

总体来说,基于用户和基于物品(电视节目)的协同过滤推荐算法的构建还是比较顺利的,但是模型的准确率相对来说是比较低的,召回率也有点高.可能是数据量不够或者数据清理不够干净而造成的模型准确率总体来说比较低。

模型的运行起来比较耗资源,以后可能会在提高模型准确率的同时降低模型的资源消耗。

推荐算法随着信息的快速增长，信息重复和信息过多导致的被动获取的信息过载，通过搜索引擎主动获得高质量的信息也会花费更多的成本，推荐系统是解决这些问题最有潜力的方法，其帮助用户快速发现感兴趣和高质量的信息，提升用户体验。增加用户使用产品时间减少用户浏览到重复或者厌恶的信息带来的不利影响，提供个性化信息，信息的推荐更为精准。

参考文献

1. 刘玮:电子商务系统中的信息推荐方法研究.情报科学.2006,2   
   [2]Badrul Sarwar,George Karypis,Joseph Konstan,etc.Item-Based Collabrative Filtering Recommendation Algorithms.WWW10.2001  
   [3]张锋, 常会友. 使用BP神经网络缓解协同过滤推荐算法的稀疏性问题[J].   
   [4]陈曦, 成韵姿. 一种优化组合相似度的协同过滤推荐算法[J].

[5]郭彩云, 王会进. 改进的基于标签的协同过滤算法[J].

**课程论文（设计）评分表**

|  |  |  |
| --- | --- | --- |
| 评分项目 | 分值 | 得分 |
| 论文内容框架完整性与逻辑合理性 | 10 |  |
| 论文排版格式规范性 | 10 |  |
| 数据清洗设计合理正确 | 15 |  |
| 数据探索内容形式（图表、文字等）丰富合理性 | 15 |  |
| 模型构建算法选择合理性，模型对比内容完整性 | 30 |  |
| 回答问题的正确与流利程度 | 20 |  |
| **总得分**： |  | |