

**课 程 实 验 报 告**

**课程名称： 大数据分析**

**专业班级：**

**学 号：**

**姓 名：**

**指导教师：**

**报告日期：**

**计算机科学与技术学院**

**目录**

[实验五 推荐系统算法及其实现 1](#_Toc59543291)

[**1.1实验目的** 1](#_Toc59543292)

[**1.2 实验内容** 1](#_Toc59543293)

[**1.3 实验过程** 2](#_Toc59543294)

[1.3.1 编程思路 2](#_Toc59543295)

[1.3.2 遇到的问题及解决方式 3](#_Toc59543296)

[1.3.3 实验测试与结果分析 3](#_Toc59543297)

[**1.4 实验总结** 3](#_Toc59543298)

# 实验五 推荐系统算法及其实现

## **1.1实验目的**

1. 了解推荐系统的多种推荐算法并理解其原理。
2. 实现**User-User的协同过滤算法**并对用户进行推荐。
3. 实现**基于内容的推荐算法**并对用户进行推荐。
4. 对两个算法进行电影预测评分对比
5. 在学有余力的情况下，**加入minihash算法**对效用矩阵**进行降维处理**

## **1.2 实验内容**

给定MovieLens数据集，包含电影评分，电影标签等文件，其中电影评分文件分为训练集train\_set和测试集test\_set两部分

基础版必做一：**基于用户的协同过滤推荐算法**

对训练集中的评分数据构造用户-电影效用矩阵，使用**pearson相似度计算方法计算用户之间的相似度**，也即相似度矩阵。**对单个用户进行推荐时，找到与其最相似的k个用户，用这k个用户的评分情况对当前用户的所有未评分电影进行评分预测，选取评分最高的n个电影进行推荐。**

在测试集中包含100条用户-电影评分记录，用于计算推荐算法中预测评分的准确性，对测试集中的每个用户-电影需要**计算其预测评分**，**再和真实评分进行对比，误差计算使用SSE误差平方和**。

选做部分提示：此算法的进阶版采用minihash算法对效用矩阵进行降维处理，从而得到相似度矩阵，注意minihash采用jarcard方法计算相似度，需要对效用矩阵进行01处理，也即**将0.5-2.5的评分置为0，3.0-5.0的评分置为1**。

基础版必做二：**基于内容的推荐算法**

将数据集movies.csv中的电影类别作为特征值，**计算这些特征值的tf-idf值**，得到关于电影与特征值的n（电影个数）\*m（特征值个数）的**tf-idf特征矩阵**。根据得到的tf-idf特征矩阵，**用余弦相似度的计算方法，得到电影之间的相似度矩阵**。

对某个用户-电影进行预测评分时，获取当前用户的已经完成的所有电影的打分，通过电影相似度矩阵获得已打分电影与当前预测电影的相似度，按照下列方式进行打分计算：

**选取相似度大于零的值进行计算，如果已打分电影与当前预测用户-电影相似度大于零，加入计算集合，否则丢弃**。（相似度为负数的，强制设置为0，表示无相关）假设计算集合中一共有n个电影，score为我们预测的计算结果，score’(i)为计算集合中第i个电影的分数，sim(i)为第i个电影与当前用户-电影的相似度。如果n为零，则score为该用户所有已打分电影的平均值。

**要求能够对指定的userID用户进行电影推荐，推荐电影为预测评分排名前k的电影。userID与k值可以根据需求做更改。**

推荐算法准确值的判断：**对给出的测试集中对应的用户-电影进行预测评分，输出每一条预测评分，并与真实评分进行对比，误差计算使用SSE误差平方和**。

选做部分提示：进阶版采用minihash算法对特征矩阵进行降维处理，从而得到相似度矩阵，注意minihash采用jarcard方法计算相似度，特征矩阵应为01矩阵。因此进阶版的特征矩阵选取采用方式为，**如果该电影存在某特征值，则特征值为1，不存在则为0，从而得到01特征矩阵**。

**选做（进阶）部分：**

本次大作业的进阶部分是在基础版本完成的基础上大家可以尝试做的部分。进阶部分的主要内容是使用**迷你哈希（MiniHash）算法对协同过滤算法和基于内容推荐算法的相似度计算进行降维**。同学可以把迷你哈希的模块作为一种近似度的计算方式。

协同过滤算法和基于内容推荐算法都会涉及到相似度的计算，迷你哈希算法在牺牲一定准确度的情况下对相似度进行计算，其能够有效的降低维数，尤其是对大规模稀疏01矩阵。同学们可以**使用哈希函数或者随机数映射来计算哈希签名**。哈希签名可以计算物品之间的相似度。

最终降维后的维数等于我们定义映射函数的数量，我们设置的映射函数越少，整体计算量就越少，但是准确率就越低。**大家可以分析不同映射函数数量下，最终结果的准确率有什么差别**。

**对基于用户的协同过滤推荐算法和基于内容的推荐算法进行推荐效果对比和分析，选做的完成后再进行一次对比分析。**

## **1.3 实验过程**

### 1.3.1 编程思路

1.基于用户的协同过滤推荐算法

（1）处理训练集中的评分数据，忽略文件中的观看时间这一列，构造用户-电影效用矩阵；

（2）编写pearson相似度计算逻辑，主要包括评分向量的构建以及计算公式的实现，评分向量的构建是找出两个user电影列表里的相同电影来实现，计算公式为：；

（3）编写预测评分逻辑，对于特定user，找出user列表中与其相似度最高的k个user，然后在这k个user中找出目标电影的评分，计算平均值；

（4）编写测试函数，解析测试文件，将userId和movieId带入预测评分函数，得出的结果与正确结果进行误差分析，最后统计SSE即可。

### 1.3.2 遇到的问题及解决方式

1.基于用户的协同过滤推荐算法

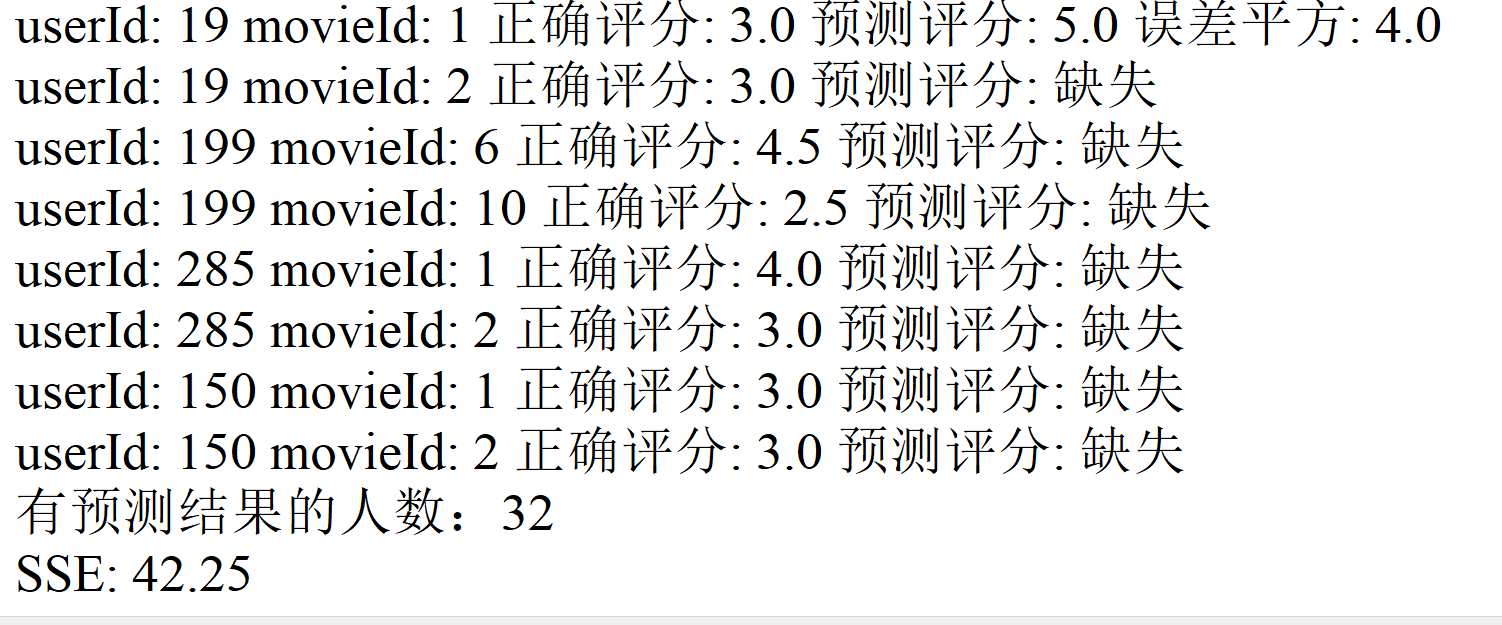
（1）文件解析问题：给定的训练集中包含timestamp这一项，最后认定为无关项，直接在解析文件时忽略掉即可；

（2）特殊测试案例问题：当k值取得不是很大时，存在某一个测试集里的用户-电影对，与该用户相似度最高的k个用户里都没有看过这个电影，所以无法预测其评分，我的处理方式是显式的告知该预测评分缺失，直接不纳入最后的SSE统计，当然最后肯定要统计一些可预测的数目；另一种方式就是找到满足所有测试集可预测的最小k值，经过不断测试，发现k值为66。

### 1.3.3 实验测试与结果分析

1.基于用户的协同过滤推荐算法

（1）k值取6时，测试结果文件如下图所示：



图：k=6测试结果

（2）k值取20时，测试结果文件如下图所示：

图：k=20测试结果

（3）k取40时，测试结果文件如下图所示：

图：k=40测试结果

（4）k取66时，测试结果文件如下图所示：

图：k=66测试结果

## **1.4 实验总结**

1.基于用户的协同过滤推荐算法