華中科技大学

课 程 实 验 报 告

课程名称： 大数据分析

专业班级： **CS2202**

学　　号： **U202215378**

姓　　名： 冯瑞琦

指导教师： 崔金华

报告日期： **2024 年 6 月 21 日**

计算机科学与技术学院

目录

实验五 推荐系统

3.1 实验内容

必做：

给定 Anime 数据集，包含用户对动漫评分、动漫标签等文件，其中动漫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-动漫效用矩阵，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动漫进行评分预测，选取评分最高的 n 个动漫进行推荐。预测评分按照以下方式计算：**

$$\text{predict\_rating} = \frac{\sum_{i=1}^{k} rating(i) * sim(i)}{\sum_{i=1}^{k} sim(i)}$$

在测试集中包含 100 条用户-动漫评分记录，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-动漫需要**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**。

基础版必做二：基于内容的推荐算法

将数据集 anime.csv 中的动漫类别作为特征值，**计算这些特征值的 tf-idf 值**，得到关于动漫与特征值的 n（动漫个数）*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**动漫**之间的相似度矩阵**。

对某个用户-动漫进行预测评分时，获取当前用户的已经完成的所有动漫的打分，通过动漫相似度矩阵获得已打分动漫与当前预测动漫的相似度，按照下列方式进行打分计算：

$$\text{score} = \frac{\sum_{i=1}^{n} score'(i) * sim(i)}{\sum_{i=1}^{n} sim(i)}$$

选取相似度大于零的值进行计算，如果已打分动漫与当前预测用户-动漫相似度大于零，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 个动漫，score 为我们预测的计算结果，score’(i)

为计算集合中第 i 个动漫的分数，sim(i)为第 i 个动漫与当前用户-动漫的相似度。如果 n 为零，则 score 为该用户所有已打分动漫的平均值。

要求能够对指定的 userID 用户进行动漫推荐，推荐动漫为预测评分排名前 k 的动漫。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-动漫进行预测评分，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（进阶）部分：

进阶部分的主要内容是使用**最小哈希（minhash）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**。同学可以把最小哈希的模块作为一种近似度的计算方式。

注意 minhash 采用 jarcard 方法计算相似度，需要对矩阵进行 01 处理。

对于**协同过滤推荐算法**，可以**将效用矩阵中动漫 0 - 5 的评分置为 0，5 - 10 的评分置为 1**。

对于**基于内容推荐算法，可以判断 tf-idf 矩阵中是否存在某特征值，如果存在则特征值为 1，反之则为 0。**

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**。

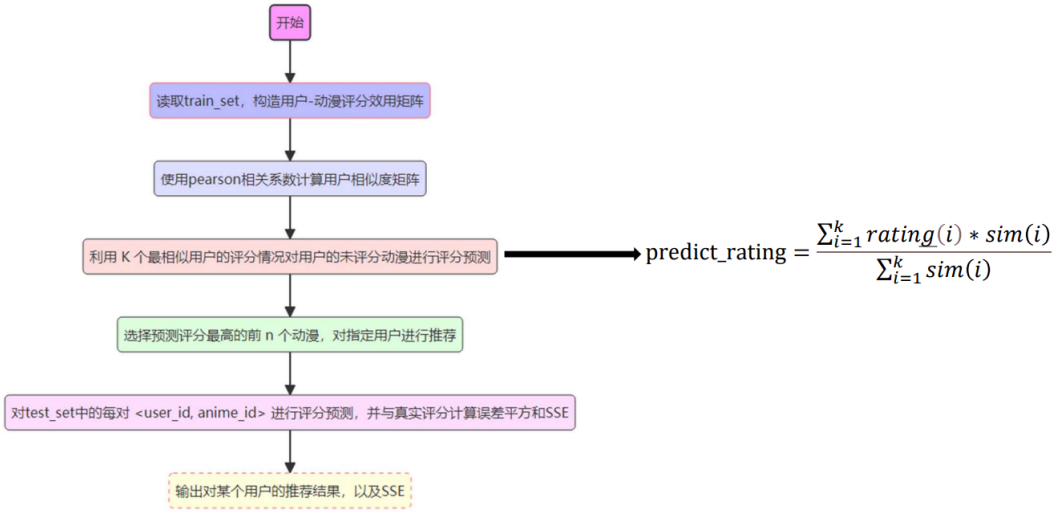
对基于用户的协同过滤推荐算法和基于内容的推荐算法进行推荐效果对比和分析，选做的完成后再进行一次对比分析。

3.2 实验过程

3.2.1 编程思路

一、 基于用户的协同过滤推荐算法

基于用户的协同过滤推荐算法的核心思想是通过找出与目标用户兴趣相似的其他用户，利用这些用户的偏好来为目标用户推荐物品。具体而言，该算法首先计算用户之间的相似度（例如使用皮尔逊相关系数或余弦相似度），然后选取与目标用户最相似的 k 个用户，称为邻居。接着，根据这些邻居的评分信息，对目标用户未评分的物品进行评分预测，最终推荐评分最高的物品。这个过程依赖于用户的行为数据，认为具有相似兴趣的用户会对相同的物品产生相似的评分。算法的流程如下图。



具体编程思路如下：

1) 数据预处理

读取训练集和测试集的数据。使用 Pandas 库的 read_csv 函数来读取 CSV 文件，将数据分别存储在 train_set 和 test_set 两个 DataFrame 中。

接下来，要根据训练集数据构建一个用户-动漫效用矩阵。在这个矩阵中，每一行代表一个用户，每一列代表一个动漫，矩阵中的值表示用户对该动漫的评分。为了实现这一点，使用 Pandas 的 pivot 函数，将数据转换为所需的矩阵形式。然而，在实际数据中，并不是每个用户都会对每个动漫进行评分，因此会有一些缺失值。为了方便后续的计算，我们将这些缺失值填充为 0，表示用户未对该动漫评分。

2) 计算用户相似度矩阵

计算用户之间的相似度。采用皮尔逊相关系数来衡量用户之间的相似度。具体来说，对用户-动漫效用矩阵进行转置，使得每一列代表一个用户，然后使用 corr 函数计算每两个用户之间的皮尔逊相关系数。这样，我们就得到了一个用户相似度矩阵，其中的每一个元素表示两个用户之间的相似度。

3) 预测评分

为了预测某个用户对某个未评分动漫的评分，首先需要找到与该用户最相似的 k 个用户。这些用户被认为在口味上与该用户最为接近。

使用这些相似用户的评分信息来预测当前用户对该动漫的评分。具体来说，对每个相似用户的评分乘以其与当前用户的相似度，并将这些值累加起来。同时也累加这些相似度的值。最终，预测评分等于累加的

评分与累加的相似度的比值，确保了越是相似的用户对预测评分的影响越大，具体编程实现如下图。

```
# print(similar_users)
if(len(similar_users)<k) :
    return 0  # 如果相似用户数量少于k，返回0

numerator = 0  # 分子初始化
denominator = 0   # 分母初始化
for similar_user_id in similar_users:
    # 计算分子：相似度和评分的乘积的累加
    numerator += similarity_matrix.loc[user_id, similar_user_id] * utility_matrix.loc[similar_user_id, anime_id]
    # 计算分母:相似度累加
    denominator += similarity_matrix.loc[user_id, similar_user_id]
if denominator == 0:
    return 0

res = numerator / denominator
return res
```

4) 评估推荐算法

为了评估推荐算法的性能，需要在测试集中计算每一条用户-动漫记录的预测评分，并与真实评分进行对比。具体来说，遍历测试集中的每一条记录，计算用户对相应动漫的预测评分，然后将预测评分与真实评分之间的差值的平方累加起来，这就是误差平方和（SSE）。SSE 值越小，说明推荐算法的预测越准确。

5) 推荐未评分动漫

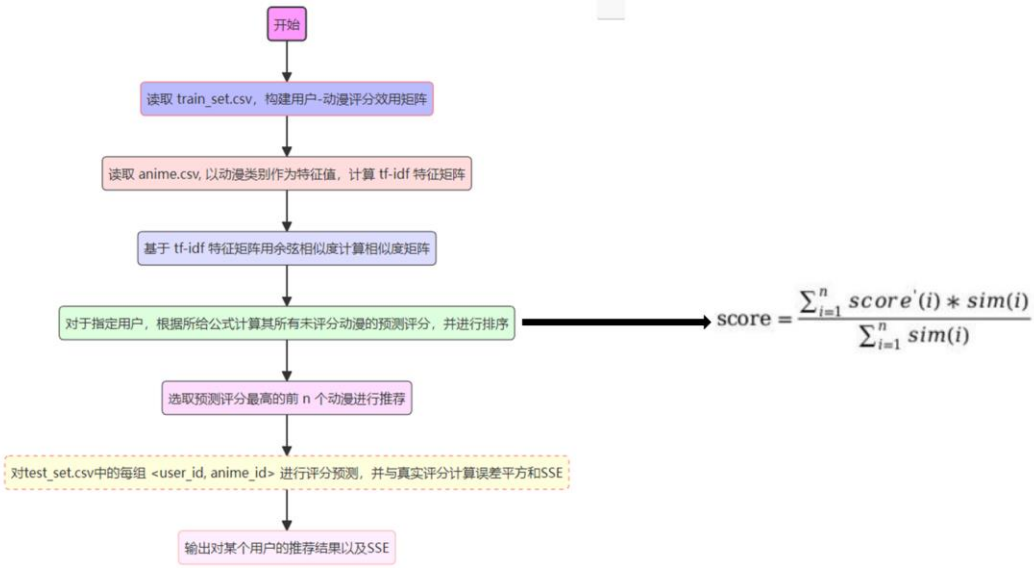
对于一个指定用户，要找到他未评分的所有动漫，然后对这些未评分动漫进行评分预测。将这些预测评分按从高到低排序，选择评分最高的前 n 个动漫进行推荐。这一过程确保推荐给用户的动漫是他最有可能喜欢的。

6) 输出结果

最后，将推荐的动漫列表及其预测评分，以及算法的 SSE 值写入一个文本文件中。这可以查看推荐结果和算法的性能。

二、 基于内容的推荐算法

基于内容的推荐算法的核心思想是通过分析物品本身的属性来为用户推荐物品。每个物品被描述为一组特征（如电影的导演、演员、类型，文章的主题词等），该算法通过学习用户对物品特征的偏好来生成推荐。例如，通过分析用户过去喜欢的电影，算法可以发现用户偏好某些类型的电影或特定导演的作品，然后推荐具有相似特征的其他电影。基于内容的推荐算法关注的是物品本身的属性，而不是用户之间的相似性，因此对于新物品的推荐更加有效，但可能无法捕捉用户之间的复杂兴趣关联。算法流程如下图所示。



具体编程思路如下：

1) 数据预处理

首先，读取训练集、测试集以及动漫数据集。使用 Pandas 库的 read_csv 函数来读取 CSV 文件，将数据分别存储在 train_set、test_set 和 anime 三个 DataFrame 中。接着，根据训练集数据构建一个用户-动漫效用矩阵。在这个矩阵中，每一行代表一个用户，每一列代表一个动漫，矩阵中的值表示用户对该动漫的评分。使用 Pandas 的 pivot 函数将数据转换为所需的矩阵形式。由于实际数据中并不是每个用户都会对每个动漫进行评分，因此会有一些缺失值。为了方便后续的计算，将这些缺失值填充为 0，表示用户未对该动漫评分。

2) 计算动漫的 TF-IDF 特征矩阵

为了表示动漫的内容特征，采用 TF-IDF（Term Frequency-Inverse Document Frequency）方法对动漫的类别信息进行处理。首先，初始化一个 TF-IDF 向量器，并去除英文停用词。（如下图所示）

对动漫数据集中的类别信息进行 TF-IDF 转换，生成一个 TF-IDF 特征矩阵。这个矩阵的每一行代表一个动漫，每一列代表一个特征，值表示该特征的重要性。

```
# 计算动漫的 TF-IDF 特征矩阵
# 初始化TF-IDF向量器，并去除英文停用词
tfidf_vectorizer = TfidfVectorizer(stop_words='english')
tfidf_matrix = tfidf_vectorizer.fit_transform(anime['Genres'].fillna(''))
```

3) 计算动漫相似度矩阵

通过计算 TF-IDF 特征矩阵的余弦相似度，可以得到动漫之间的相似度矩阵。余弦相似度是一种常用的衡量两个向量之间相似度的方法，值在 0 到 1 之间，值越大表示相似度越高。使用 Scikit-learn 库中的 cosine_similarity 函数来计算相似度矩阵，如下图所示。

```
# 计算余弦相似度矩阵，输出相似度矩阵的维度
cosine_sim = cosine_similarity(tfidf_matrix)# 计算动漫之间的余弦相似度
print(cosine_sim.shape[0])
```

4) 预测评分并推荐动漫

为了预测某个用户对某个未评分动漫的评分，首先获取该用户已经评分的动漫列表，并根据相似度矩阵计算这些已评分动漫与未评分动漫之间的相似度。对于每一个未评分动漫，根据相似度加权求和计算其预测评分。具体来说，遍历用户未评分的动漫，对于每个动漫，计算所有已评分动漫与其的相似度的加权和，然后除以相似度的总和，得到该动漫的预测评分。如果相似度总和为 0，表示没有相关的已评分动漫，此时预测评分设置为 0。

预测评分函数 predict_ratings 具体实现为：

获取用户的评分情况，并找到用户未评分的动漫。

初始化预测评分字典。

对每个未评分动漫，计算其与已评分动漫的相似度的加权和，得到预测评分。

根据预测评分，可以为用户推荐未评分的动漫。对所有未评分动漫的预测评分进行排序，选择评分最高的前 n 个动漫进行推荐。这个过程确保推荐的动漫是用户最有可能喜欢的。

5) 评估推荐算法并输出结果

为了评估推荐算法的准确性，需要在测试集中计算每一条用户-动漫记录的预测评分，并与真实评分进行对比。遍历测试集中的每一条记录，计算用户对相应动漫的预测评分，然后将预测评分与真实评分之间的差值平方累加，得到误差平方和（SSE）。SSE 值越小，说明推荐算法的预测越准确。计算 SSE 的函数 calculate_sse ，遍历测试集中的每一条记录，计算预测评分，并与真实评分比较，累加平方误差。

最后，将推荐的动漫列表及其预测评分，以及算法的 SSE 值写入一个文本文件中。这使得可以方便地查看推荐结果和算法的性能。

3.2.2 遇到的问题及解决方式

1) NaN 值处理

问题描述：在最初的数据预处理中，动漫效用矩阵中存在大量的缺失值（NaN），这些缺失值表示用户未对某些动漫进行评分。这些缺失值在后续的相似度计算和评分预测中会导致错误，还会导致在计算 TF-IDF 特征矩阵时出现错误。

解决方式：通过将缺失值填充为 0 来处理 NaN 值，这样可以表示用户未对该动漫进行评分，并确保矩阵运算的正常进行。在计算 TF-IDF 特征矩阵时，先将这些缺失的类别信息填充为空字符串，以避免计算过程中出现问题。

2) 动漫 ID 与索引的映射

问题描述：在计算动漫相似度时，需要将动漫 ID 映射到 TF-IDF 特征矩阵的行索引。如果未正确处理这个映射关系，可能会导致索引错误。

解决方式：通过创建一个字典，将动漫 ID 映射到 TF-IDF 特征矩阵的行索引，以确保在相似度计算和评分预测时能够正确获取相应的索引，如图。

```
# 构建动漫与索引的映射关系
anime_id_to_index = {anime_id: idx for idx, anime_id in enumerate(anime['Anime_id'])}
```

3) 用户未评分动漫的预测评分

问题描述：在预测用户未评分的动漫时，如果所有已评分动漫与未评分动漫的相似度都为 0，会导致分母为 0，从而无法计算预测评分。

解决方式：在这种情况下，将预测评分设置为 0，表示无法根据已有评分信息进行预测，避免分母为 0 的错误。

4) 计算 SSE 时的预测评分

问题描述：在计算 SSE 时，需要对每条用户-动漫记录进行预测评分。如果预测评分函数返回的结果不正确，可能导致 SSE 计算错误。

解决方式：在 SSE 计算函数中，通过调用评分预测函数并确保返回正确的预测评分，避免误差累积，确保 SSE 计算的准确性。

3.2.3 实验测试与结果分析

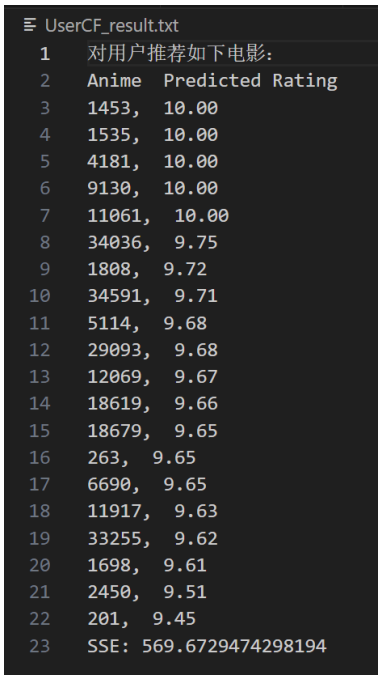
一、 基于用户的协同过滤推荐算法

基于用户的协同过滤推荐算法最终输出一个名为 UserCF_result.txt 的文本文件。

文件中列出了对特定用户推荐的动漫列表及其预测评分。每一行显示一个动漫的 ID 和预测的评分值。例如，动漫 ID 为 1453 的预测评分为 10.00，动漫 ID 为 1535 的预测评分也为 10.00，以此类推。预测评分值表示推荐系统根据用户的历史评分和其他用户的行为模式，预测用户可能对这些动漫的喜好程度。此处将预测评分值最高的 20 个动漫推荐给用户。

SSE（Sum of Squared Errors，误差平方和）是评估推荐算法预测准确性的指标之一。这里的数值是 569.67，表示在测试集中对每一条用户-动漫记录进行预测评分后，预测值与真实值之间的误差平方和。较低的 SSE 值通常意味着推荐系统能够较准确地预测用户的喜好，推荐的动漫更符合用户的实际兴趣。

输出文件的截图如下图所示。由图可知，给用户推荐的动漫预测评分都很高，在 9 分以上，最终的 SSE 也在合适范围，说明实验功能正确。



二、 基于内容的推荐算法

基于内容的推荐算法最终输出一个名为 ItemCF_result.txt 的文本文件。这个输出文件是基于用户的协同过滤推荐算法的结果和评估指标。

推荐的动漫列表每一行列出了对特定用户推荐的动漫及其预测评分。此处也同样将预测分数排名前 20 的动漫推荐给用户。例如，动漫 ID 为 4103 的预测评分为 10.0，动漫 ID 为 20355 的预测评分也为 10.0，以此类推。这些预测评分值表示推荐系统根据用户的历史评分和其他用户的行为模式，预测用户可能对这些动漫的喜好程度。

SSE（Sum of Squared Errors，误差平方和）是评估推荐算法预测准确性的指标之一。这里的数值是 409.66，表示在测试集中对每一条用户-动漫记录进行预测评分后，预测值与真实值之间的误差平方和。较低的 SSE 值通常意味着推荐系统能够较准确地预测用户的喜好，推荐的动漫更符合用户的实际兴趣。

输出文件的截图如下图所示。由图可知，给用户推荐的动漫预测评分都较高，在 8 分以上，最终的 SSE 也在合适范围，说明实验功能正确。

ItemCF_result.txt

```
1   Anime ID: 4103 Predicted Rating: 10.0
2   Anime ID: 20355 Predicted Rating: 10.0
3   Anime ID: 34625 Predicted Rating: 10.0
4   Anime ID: 34930 Predicted Rating: 10.0
5   Anime ID: 32384 Predicted Rating: 8.536065427451165
6   Anime ID: 40441 Predicted Rating: 8.536065427451165
7   Anime ID: 11483 Predicted Rating: 8.254625998164261
8   Anime ID: 12919 Predicted Rating: 8.132224607195198
9   Anime ID: 20889 Predicted Rating: 8.132224607195198
10  Anime ID: 28831 Predicted Rating: 8.132224607195198
11  Anime ID: 29904 Predicted Rating: 8.132224607195198
12  Anime ID: 4460 Predicted Rating: 8.126368585190331
13  Anime ID: 5622 Predicted Rating: 8.126368585190331
14  Anime ID: 8689 Predicted Rating: 8.126368585190331
15  Anime ID: 28657 Predicted Rating: 8.126368585190331
16  Anime ID: 2349 Predicted Rating: 8.113502900163228
17  Anime ID: 32139 Predicted Rating: 8.106585208201565
18  Anime ID: 6842 Predicted Rating: 8.089270607712894
19  Anime ID: 7836 Predicted Rating: 8.089270607712894
20  Anime ID: 9748 Predicted Rating: 8.089270607712894
21  SSE: 409.66199749017716
```

3.3 实验总结

在本次实验中，我深入学习和实践了基于用户的协同过滤推荐算法及基于内容的推荐算法。这两种算法是推荐系统中常用的方法，分别从用户行为数据和物品属性数据出发，实现个性化推荐。以下是我对实验的总结和心得体会：

我学习了基于用户的协同过滤算法，知晓了依赖于用户-物品评分矩阵和用户相似度矩阵的构建方法。在实现过程中，我遇到了如何处理数据缺失（NaN 值）、计算皮尔逊相似度、选择相似用户和预测评分等问题。通过详细分析和处理，我确保了推荐系统能够准确地预测用户对未评分物品的喜好程度。在评估算法准确性时，使用 SSE 作为指标，帮助我量化推荐结果与实际评分之间的差距，进一步优化算法。

我还体会到了基于内容的推荐算法则侧重于分析物品的属性（如动漫的类别信息），学习怎样构建 TF-IDF 特征矩阵，并计算物品之间的相似度。在实现过

程中，我处理了动漫类别信息缺失、构建 TF-IDF 特征矩阵、计算余弦相似度矩阵等关键步骤。通过这些步骤，我能够基于动漫的内容特征，推荐与用户已评分动漫相似的未评分动漫，提升了推荐的个性化水平。

在实验中，我深刻体会到推荐系统的核心挑战之一是如何在准确性和效率之间取得平衡。特别是在处理大规模数据集时，算法的优化和数据结构的选择显得尤为重要。同时，推荐系统的性能不仅取决于算法本身，还与数据质量、特征选择和评估方法密切相关。通过不断调整和优化，我逐步提升了推荐系统的预测精度和用户体验。

本次实验使我对推荐系统的工作原理和实现细节有了更深入的理解，也让我很有成就感。在未来的研究和工作中，我将继续探索和学习各种推荐算法，并关注推荐系统在实际应用中的优化和创新，以更好地满足用户个性化需求。