

南京理工大学泰州科技学院

毕业设计说明书(论文)

作者: 韩睿枫 学号: 1909970215

学院(系): 计算机科学与工程学院

专业: 计算机科学与技术

题目: 交互式电影推荐算法的研究与实现

指导者: 谭立兴 讲师

评阅者: 成艳 讲师

2023 年 5 月

毕业设计说明书（论文）中文摘要

随着人工智能技术在各行各业的广泛赋能，学术界提出并研究了一系列相关推荐算法。现下基于深度学习的相关算法缺乏透明度和可解释性，本文在交互式推荐系统的背景下考虑了 Aspect-Item 算法，以此融合系统性能与可解释性研究。

本文从用户与偏好参数的角度研究了交互式电影推荐算法，方便用户选择电影并提供了推荐决策的论辩解释(argumentative explanations)。为了增强推荐决策的透明度，通过整合 Aspect-Item、KNN、Z-KNN、SVD、NMF、Slope One 及 CoClustering 算法，对数据进行预处理，设计并实现了 XAI 算法。通过相关算法的对比实验，根据准确率、精确率、召回率、调和均值、均方根误差、平均绝对误差和运行效率等指标考量了系统性能。相较于 Rago 的 Aspect-Item 算法，XAI 算法不仅运行时间减少 9%，推广适用于 Movielens 数据集，而且同时提高了精确率和召回率。实验结果表明，该算法不仅具有较高的系统性能，而且通过给予解释呈现了清晰的推荐机制，从而获得了友好的用户体验。

关键字 Aspect-Item算法 可解释性 用户满意度 交互式电影推荐算法

毕业设计说明书（论文）外文摘要

Title Research and Implementation of
 Interactive Movie Recommendation Algorithms

Abstract

With the widespread empowerment of artificial intelligence technology in various industries, a series of related recommendation algorithms have been proposed and studied by the academic community. Currently, the related algorithms based on deep learning lack transparency and explainability. Therefore, this paper considers Aspect-Item algorithm in the context of interactive recommendation system to integrate system performance and explainability research.

From the perspective of users and preference parameters, this paper studies the interactive movie recommendation algorithms, which facilitates movie selection for users and provides argumentative explanations for recommendation decisions. To enhance the transparency of recommendation decisions, XAI algorithm is designed and implemented by integrating Aspect-Item, KNN, Z-KNN, SVD, NMF, Slope One and CoClustering algorithms through data preprocessing. System performance is evaluated based on measures such as accuracy, precision, recall, F1, RMSE, MAE and operating efficiency through comparative experiments on related algorithms. Compared to Rago's Aspect-Item algorithm, the XAI algorithm not only witnesses a 9% reduction in operating time, but is also extended to be applicable to the Movielens dataset and leads to notable improvements in both precision and recall. Experimental results shows that not only does the algorithm exhibit high performance, but it also provides clear recommendation mechanism through explanation, thus achieving a friendly user experience.

Keywords Aspect-Item algorithm Explainability User satisfaction
 Interactive movie recommendation algorithms

目 录

1 绪论	1
1.1 研究背景	1
1.2 选题目的与意义	1
1.3 国内外发展现状	2
1.4 论文结构	3
2 相关技术与理论	4
2.1 推荐系统概述	4
2.2 协同过滤算法	5
2.3 机器学习算法	6
3 交互式电影推荐算法研究与实现	10
3.1 推荐系统的建模	10
3.2 数据预处理	22
3.3 算法实现	26
4 实验结果分析	46
4.1 准确率	47
4.2 精确率和召回率	49
4.3 均方根误差和平均绝对误差	56
4.4 运行效率	60
5 总结与展望	62
5.1 总结	62
5.2 展望	63
结束语	65
致 谢	66
参 考 文 献	67

1 绪论

1.1 研究背景

推荐系统(Recommendation System)是人工智能领域的重要方向之一,主要应用于互联网、电商、社交媒体等领域。通过分析用户的历史行为、兴趣爱好以及社交关系等多种数据,推荐系统能够为用户提供符合其需求和兴趣的商品、内容或服务。推荐系统最初起源于 1992 年 Goldberg 提出的 Tapestry 推荐系统,该系统首次引入了协同过滤的概念。之后,Resnick 在文献中正式提出了推荐系统这一概念,推荐系统逐渐受到商业界和学术界的关注,许多相关研究成果也应运而生。电商公司如亚马逊和 Netflix 也开发了基于协同过滤算法的推荐系统,这些系统主要利用用户的历史行为进行推荐,以推荐相似的商品和内容。这些推荐系统的出现得到了广大用户的欢迎,同时也促进了电商行业的快速发展。

1.2 选题目的与意义

随着互联网技术的发展,越来越多的人选择在家中观看电影。然而,用户往往会面临着选择困难症,因为市场上的电影种类繁多,每个人的口味和喜好也不同。为了解决这个问题,推荐系统逐渐成为一种重要的解决方案,通过对用户的历史行为和兴趣进行分析,为用户推荐符合其兴趣的电影。然而,传统的推荐系统往往只是一种静态的推荐,即系统根据用户的历史行为和兴趣推荐一些电影,然后将这些电影推荐给用户,用户没有机会对推荐结果进行反馈,也不能根据自己的实际需求进行筛选和修改。因此,交互式电影推荐系统应运而生,它允许用户通过与系统的交互,主动地指定自己的需求和偏好,并根据用户的反馈不断调整推荐结果,从而更好地满足用户的需求。尽管推荐系统常用的算法具有可扩展性和有效性,但它们很难被解释,因为它们表示因素的方式使它们无法解释。这些问题与最新的深度学习方法在推荐系统领域的应用类似^[1]。这些方法在有效性方面表现良好,但依赖于神经网络,尤其是对非专业用户而言,尽管已经做出一些努力来提供解释,例如使用注意力机制^[2],因此无法轻易解释如何生成推荐。如果可解释模型在任务中的表现令人满意,那么它们所拥有的额外可解释性可能被证明是比黑盒方法有价值的优势^[3]。本文采用一种新颖的推荐算法,将会与一些常见的推荐算法进行比较,以展现本文推荐系统的优秀性能。

交互式电影推荐系统是个性化推荐系统的典型代表，它可根据用户的兴趣和历史行为进行推荐，采用各种算法和模型对电影进行分析和推荐。同时，交互式电影推荐系统还充分考虑了用户的反馈和需求，通过与用户的交互来不断优化推荐结果。交互式电影推荐系统不仅可以给用户带来更好的观影体验，同时也为电影产业的发展提供了有益的参考。

随着信息技术的快速发展，人们对于信息的获取越来越依赖于推荐系统。交互式电影推荐系统的意义在于可以提高用户体验和满意度，根据用户的兴趣和喜好推荐更相关和优质的电影，提高用户观影的满意度和享受度；可以帮助用户发现更多好电影，让用户发掘更多符合其喜好的电影；可以提高电影产业的营收，通过推荐更符合用户需求的电影，提高用户观影率，从而增加电影产业的收益；可以推广电影，通过用户对电影的评分和推荐，使电影得到更多的关注和推广；可以实现个性化推荐，交互式电影推荐系统可以根据用户喜好和行为数据进行个性化推荐，满足用户多样化的需求。

1.3 国内外发展现状

随着互联网的普及和用户数量的增加，推荐系统的应用也越来越广泛。在电影领域，推荐系统也扮演着越来越重要的角色。本文将介绍国内外交互式电影推荐系统的发展现状。

在国外，交互式电影推荐系统已经有了一定的发展，如美国的 Netflix 和德国的 Maxdome，它们已经建立了基于用户电影观看历史和评分的个性化推荐系统。这些系统不断地改进和更新，加入越来越多的特征，提高了推荐精度。同时，一些研究还从用户行为、文本、情感等角度进行了推荐系统研究，并采用了深度学习、协同过滤等技术。Netflix 是一个全球领先的在线流媒体服务提供商，拥有超过 2 亿的用户。Netflix 的电影推荐系统从多个角度进行推荐，采用了协同过滤算法和深度学习技术，可以从大量的数据中学习到高层次特征，提高推荐的准确率。Maxdome 是德国领先的视频点播平台，也采用了基于用户历史行为和兴趣相似度的协同过滤算法和深度学习技术。Maxdome 的电影推荐系统还考虑了用户的年龄、性别、地域等因素，提高了推荐的准确率。Musto 等人提出了一种图形化的方法，用于语言解释推荐系统，推荐的基本原理独立于提出建议的算法生成，他们将解释分为不同类型^[4]，以满足不同用户的需求和偏好，这表明解释推

荐系统的方法不仅可以提高用户对推荐结果的理解和接受度,还可以为推荐系统的设计和优化提供有益的参考。

在国内,交互式电影推荐系统的研究也在逐渐发展。随着人工智能技术的不断进步,更多的电影推荐系统开始采用深度学习和自然语言处理等技术。然而,国内的研究主要局限于基于用户兴趣和观看历史进行推荐,还需要结合更多的特征和指标,建立更准确的推荐系统。目前,国内一些电影推荐系统主要采用协同过滤算法和基于内容的推荐算法^[5]。其中协同过滤算法通过用户历史行为及兴趣相似度进行推荐,但内容推荐算法根据电影的特征进行推荐。这些算法虽然可以提供一定的推荐效果,但是缺乏全面性和准确率。近年来,国内一些研究机构开始探索采用深度学习和自然语言处理等技术进行电影推荐^[6]。例如,华为在 2018 年提出了一种基于深度学习的电影推荐算法,该算法可以从海量的数据中学习 to 更高层次的特征,提高推荐的准确率。张永锋和陈旭的研究评估不同类型的解释如何向用户显示^[7],并考虑如何评估解释及其应用。此外,一些研究还从用户行为、文本、情感等角度进行了推荐系统研究,以提高推荐的精度和个性化程度。

1.4 论文结构

第一部分,从绪论、研究背景、研究目的与意义以及国内外发展现状入手,对选题研究进行大概的介绍;

第二部分,简要介绍了该研究的相关技术与理论,主要关于推荐系统、协同过滤以及机器学习算法等相关基础知识;

第三部分,简要介绍了交互式电影推荐算法的研究与实现,从推荐系统的建模、数据预处理以及算法实现三个方面进行阐述和分析;

第四部分,通过推荐算法的对比实验,对准确率、精确率和召回率、均方根误差和平均绝对误差以及运行效率四个方面对系统性能进行说明和分析;

第五部分,总结研究成果,展望前景方向。

2 相关技术与理论

2.1 推荐系统概述

推荐系统利用用户历史行为和偏好信息，通过数据挖掘技术进行分析，并向用户提供个性化推荐服务。它的应用领域包括电子商务、社交网络、音乐和视频等。推荐系统的核心任务是预测用户兴趣并向其个性化推荐物品，其目标是提高用户体验及促进销售及提升品牌价值。推荐系统需要解决数据稀疏性(sparsity)、冷启动(cold start)等问题，同时还需要考虑隐私保护、公平性等因素的影响。因此，在推荐系统的设计和实现中，需要综合考虑多种指标，并采用合适的算法和技术来解决各种问题。

推荐系统包含基于内容的推荐系统、基于协同过滤的推荐系统和混合推荐系统等类型。其中，基于内容的推荐系统分析产品或服务的内容，推荐与用户喜好相符的物品；基于协同过滤的推荐系统则根据用户历史行为和偏好进行分析，推荐与相似用户所偏好的物品；混合推荐系统将内容推荐算法和协同过滤算法结合起来，以提高推荐质量。基于内容的推荐推荐(content-based recommendation)利用物品的属性和用户的历史行为来发现物品之间的相似性，并向用户推荐与其先前选择的物品相似的物品。基于协同过滤的推荐(collaborative filtering recommendation)利用用户行为数据来发现用户之间的相似性，并向具有相似兴趣的用户推荐物品。基于协同过滤的推荐系统主要有以下三种方法：基于用户的协同过滤(user-based collaborative filtering)、基于物品的协同过滤(item-based collaborative filtering)和基于模型的协同过滤(model-based collaborative filtering)。其中，基于用户的协同过滤方法是根据用户之间的行为相似度进行推荐；基于物品的协同过滤方法是计算物品之间的相似度，并向用户推荐相似的物品；基于模型的协同过滤是通过构建用户和物品的向量表示以及隐含因子模型（例如矩阵分解），来预测用户对物品的评分，并向用户推荐物品。推荐系统主要包括离线训练和在线推荐两个阶段。其中，离线训练是在大规模数据集上计算和存储模型参数，以便在实时服务中利用这些模型参数提供个性化推荐服务。在线推荐是指在用户请求推荐服务时，推荐系统根据用户的历史数据及实时数据计算出推荐结果，并返回给用户。推荐系统的优化目标主要包括准确率、精确率、召回率、调和均

值、均方根误差、平均绝对误差和运行时间等系统指标。

2.2 协同过滤算法

协同过滤(Collaborative Filtering)是一种常见的推荐算法，其基本思想是根据用户的历史行为数据，结合协同相似性，预测用户对未知商品的兴趣，并将推荐结果呈现给用户。其主要有两种实现方式：基于用户的协同过滤和基于物品的协同过滤，它们在推荐物品时的计算方式不同，基于用户的协同过滤以用户为计算对象，通过计算用户之间的相似度来找到与目标用户相似度高的用户，并根据这些用户对目标物品的评分数据预测出目标用户对该物品的评分进行推荐。而基于物品的协同过滤则以物品为计算对象，通过计算物品之间的相似度找到与目标物品相似度高的物品，并结合目标用户已评价物品的评分数据预测目标用户对待预测物品的评分进行推荐。基于用户的协同过滤通常适用于物品数量较少、用户数量较多的情况下，因为其推荐结果更容易被解释，而且对于用户间的相似度能够更好地反映出用户的偏好。但是当物品数量很大时，在物品数量较大时，基于用户的协同过滤中计算用户相似度的复杂度会大大增加，效率和准确率可能会受到影响。相较之下，基于物品的协同过滤更适用于物品数量较多的情况，因为它避免了计算用户间相似度的复杂度，降低了计算复杂度，提高了推荐效率和准确率。此外，基于物品的协同过滤还有一个优点，即它对于长尾物品的推荐效果比较好。在实际应用中，基于物品的协同过滤通常比基于用户的协同过滤计算复杂度更低，因为前者只需要计算物品之间的相似度，而后者需要计算用户之间的相似度，这导致基于用户的协同过滤需要进行更多的计算。

2.2.1 基于用户的协同过滤算法

基于用户的协同过滤算法是通过计算用户之间的相似度来进行推荐的。具体来说，其先计算各个用户之间的相似度，找出与目标用户最相似的前 k 个用户，再根据这些用户的行为数据预测目标用户对未知商品的兴趣。常用的计算用户相似度的方法有 Pearson 相关系数、余弦相似度和欧几里得距离。找到最相似的前 k 个用户后，就可以利用这些用户的历史行为数据，预测目标用户对未知商品的评分。计算方法通常有加权平均和基于邻居的加权平均两种。其中，加权平均是根据各个邻居的相似度对其历史评分进行加权平均，而基于邻居的加权平均则是在加权平均的基础上，对邻居的数量进行加权。

2.2.2 基于物品的协同过滤算法

基于物品的协同过滤算法是根据商品之间的相似性进行推荐的。其基本思路是通过计算商品之间的相似度来进行推荐。具体而言，该算法会先计算各个商品之间的相似度，找出用户历史行为中与目标商品最相似的前 k 个商品，然后根据这些商品的评分预测目标用户对未知商品的兴趣。常用的计算商品相似度的方法是余弦相似度。在找到最相似的前 K 个商品后，可以利用这些商品的历史评分，采用加权平均或基于邻居的加权平均来计算目标用户对未知商品的评分。

协同过滤算法的优点是能够根据历史行为预测用户的潜在兴趣，推荐结果较为准确，并且能够扩展到任意类型的商品和用户，其缺点是易受稀疏数据和冷启动问题的影响，对于刚注册或者历史行为较少的用户缺乏个性化推荐。此外，协同过滤算法还容易受到独立偏好和流行度偏好的影响，导致推荐结果缺乏新颖性和多样性。为了克服协同过滤算法的缺陷，研究人员进行了很多改进尝试，包括融合多种算法、基于社交网络的推荐、针对冷启动问题的解决等。近年来，基于协同过滤算法的推荐系统也应用广泛，包括亚马逊、Netflix 等电商和娱乐领域的网站，以及 Google Play、App Store 等应用商店。还有许多新兴领域也在利用协同过滤算法进行推荐，例如智能家居、医疗健康等。

2.3 机器学习算法

机器算法是指在计算机领域中，通过运用数学、逻辑等方法来解决具体问题的一种技术，它是计算机科学中最基础的研究领域之一。机器算法可以应用于许多领域，包括数据处理、人工智能、图像和声音处理、网络和通信等等，其应用越来越广泛，也愈来愈重要。机器学习中的算法是一种用于从数据集中生成模型的数学公式或方法。本质上讲，这些算法都是利用现有的输入数据集作为基础，提供输出结果来预测未来的数据。机器学习算法通常被分类为监督学习 (supervised learning)、无监督学习 (unsupervised learning) 或半监督学习 (semi-supervised learning)。在监督学习中，对于给定的输入，通过有监督的学习来预测相应的输出，其中训练数据集包括已知的输出值；无监督学习则是指没有提供输出数据的一种学习方式；半监督学习则介于监督学习和无监督学习之间，其是指在数据集中只有一小部分数据被标记。

2.3.1 KNNBasic 算法

KNNBasic 是 KNN 算法的一种变体，它是基于用户之间的相似度来进行推荐的。在 KNNBasic 算法中，对于每一个用户，算法会找到与其最相似的 k 个用户，然后根据这 k 个用户的评分数据来预测该用户对未评分物品的评分。因此，KNNBasic 算法可以看作是 KNN 算法在推荐系统中的一种应用。但是，KNN 算法还可以应用于其他领域，比如分类和回归等。KNN(K Nearest Neighbors)是一种监督学习算法。它是基于实例的算法。KNN 模型不仅容易构造，而且非常适合处理高度结构化的数据，该算法最早由 Thomas Cover 和 Peter Hart 于 1967 年提出^[8]。

KNNBasic 算法是一种基于用户相似性的推荐算法，它通过查找与目标用户最相似的一组用户来预测目标用户对某个物品的评分。这种算法的主要思想是依靠用户之间的相似性来预测用户对物品的评分。作为协同过滤算法中的一种经典算法，KNNBasic 算法在推荐系统中得到了广泛的应用，属于一种基于距离度量的分类算法。它的亮点在于简单易懂，易于实现，适用于多分类问题和大规模数据集。它使用简单的平均值实现算法，其中，平均值指输出值的平均数。该算法使用欧几里德距离、曼哈顿距离或闵可夫斯基距离来计算相邻点之间的距离。当距离确定后，选择距离最近的标签来对该类型未知对象进行分类或预测对象的未知值。

2.3.2 Z-KNN 算法

Z-KNN(KNN with Z Score)算法是基于 KNN 和 Z-Score 技术的算法。Z-KNN 算法是在 KNN 算法的基础上发展而来的。该算法使用 Z 值进行标准化，通过消除评分者之间的偏见来提高预测准确率。Z-KNN 算法是 Surprise 库中的一种推荐算法，它是基于 K 最近邻算法的改进版本，在计算相似度时使用了 Z-Score 标准化方法。该算法是由 Simon Funk 在 2006 年 Netflix Prize 竞赛中提出的，并在 Surprise 库中得到了实现。Netflix Prize 竞赛是一个由 Netflix 公司发起的推荐算法竞赛，旨在提高 Netflix 电影推荐系统的准确率。

Z-KNN 算法在 KNN 算法的基础上，加入了 Z-Score 标准化处理，使得算法更加稳定和准确。它的特殊点在于对于数据集中存在异常值或者噪声的情况下，它的表现更加优秀。在该算法中，每个评分都会被标准化，使评分者的评分不会影响模型的输出结果。消除偏见之后，带 Z 值的 KNN 算法使用优先级排队来设

置阈值，并根据相邻数据点的距离来预测未知对象的值。

2.3.3 SVD 算法

SVD(Singular Value Decomposition)算法是一种基于线性代数的矩阵分解算法。该算法可以将一个大型矩阵分解成多个较小的矩阵，并使用这些矩阵对原始数据进行重建。奇异值分解理论可以追溯到 19 世纪末。1873 年，意大利数学家 Beltrami 和法国数学家 Jordan 独立发现了实正方矩阵的正交对角分解定理，即任何非奇异实正方矩阵都可以正交对角相似于对角矩阵。英国数学家 Sylvester 在 1889 年左右给出了该定理的第三个证明。1902 年，法国数学家 Autonne 将奇异值分解推广到了复正方矩阵。1939 年，Eckart 与 Young 将奇异值分解推广到了一般复长方形矩阵。后来人们发现 SVD 定理能够推广到任何形式的矩阵，尤其是宽矩阵和窄矩阵，这才有了后面的广泛应用。

SVD 算法是一种基于矩阵分解的推荐算法，它的亮点在于能够处理稀疏矩阵，具有较高的准确率和可解释性。该算法的特殊点在于能够处理缺失值和异常值，并且能够在不同的数据集上进行迁移学习。SVD 算法是一种被广泛使用的推荐算法，它使用梯度下降方法来确定最小的误差平方和，并将矩阵分解成三个部分：一个用户矩阵，一个物品矩阵和一个奇异价值矩阵。用户矩阵对应用户，商品矩阵对应商品，在矩阵分解时通过优化使得物品矩阵和用户矩阵相乘的结果与实际评分数据尽量接近。

2.3.4 NMF 算法

NMF(Non-negative Matrix Factorization)算法是一种非负矩阵分解算法。相对于标准矩阵分解算法，NMF 只能得出非负矩阵（由非负列和非负行组成），因此能够保持特征的可解释性。Lee 和 Seung 于 1999 年在《自然》杂志上提出了非负矩阵分解方法^[9]，Xin Luo 第一次将 NMF 算法应用于协同过滤的推荐算法^[10]，该方法将矩阵分解为非负分量，同时实现了非线性的维数约减。

NMF 算法是一种基于矩阵分解的推荐算法，它的亮点在于能够处理非负数据，具有较高的准确率和可解释性。它的特殊点在于能够处理长尾数据，能够对数据进行更好的拟合和预测。在推荐系统中，NMF 算法主要用于将评分矩阵分解成两个“小平面”，能够比较容易地理解数据集，该算法被证明对于大规模数据集具有较好的性能。

2.3.5 Slope One 算法

Slope One 算法是一种预测用户对物品评分的个性化算法，其原理是在评分矩阵中，每个用户评价商品的评价值之间存在一定的偏差，该算法就是通过对这些偏差进行计算生成推荐。Slope One 算法是由 Daniel Lemire 和 Anna Maclachlan 在 2005 年提出的一个基于物品的协同过滤推荐算法^[11]。

Slope One 算法是一种基于差值计算的推荐算法，它的亮点在于简单易懂，计算速度快，适用于处理大规模数据集，该算法的特殊点在于能够处理缺失值和异常值，并且能够对数据进行更好的拟合和预测。Slope One 算法首先求出每个物品与其它物品之间的评分差，然后对该物品出现的次数进行归一化，将其加权求和表示该物品的评分 A ，最后再加上该用户平均的值得到在未评分项上的预测值。

2.3.6 CoClustering 算法

CoClustering 算法是一种基于聚类的推荐算法，主要是将用户和物品打包在一起进行聚类。与常规聚类算法不同，CoClustering 函数是通过在行和列上迭代的形式来实现多维数组聚类。CoClustering 算法是由 Dhillon 等人在 2001 年提出的^[12]，这种算法在协同过滤中非常有用，因为该算法可以对评分矩阵进行划分，在扰动意义下，与多维模型相对应的二元矩阵进行匹配。因此，CoClustering 算法可以使用聚类和矩阵分解算法来处理大量的数据点。

CoClustering 算法的亮点在于能够同时聚类用户和物品，具有较高的准确率和可解释性。它的特殊点在于能够处理长尾数据和稀疏数据，并且能够在不同的数据集上进行迁移学习。

2.3.7 Aspect-Item 算法

Aspect-Item 算法是一种与传统的协同过滤算法不同的推荐算法，其结合了推荐系统中商品的属性信息，即“方面”(aspect)信息，通过对物品评分进行建模，得到了物品的方面信息，然后将其与用户历史行为数据结合起来，使用协同过滤算法对用户进行物品推荐。

Aspect-Item 算法的亮点在于能够缓解数据稀疏性问题，支持多样性推荐，使用商品方面信息来描述商品的特征，并利用这些特征来评估商品之间的相似度和相关性，可以更准确地预测用户的偏好和需求，从而提高推荐的精度和效果。

3 交互式电影推荐算法研究与实现

3.1 推荐系统的建模

本文的交互式电影推荐算法是在一种基于用户兴趣和历史行为数据的推荐系统上研究并实现的。本节将介绍该系统的构建过程，包括数据收集、数据预处理、模型选择和评估等方面。

如图 3.1 所示，数据收集是构建交互式电影推荐系统的第一步，需要用户的历史行为数据和电影元数据。历史行为数据包含用户的电影评分、浏览记录、收藏记录等，可以通过电影网站或移动应用的 API 接口获取。电影元数据包括电影的类型、演员、导演等信息，可以通过电影数据库获取。数据预处理是将原始数据转换为可用于建模的数据格式的过程。

在交互式电影推荐系统中，数据预处理包括数据清洗、数据集成、数据转换、数据规约等步骤。具体来说，需要处理缺失值、重复值、异常值等问题，将不同来源的数据进行合并，将数据转换为数值型或二进制型数据，规约数据的范围和精度等。

模型选择是选择合适的机器学习模型或深度学习模型来建立推荐模型的过程。在交互式电影推荐系统中，常用的模型有协同过滤模型、内容推荐模型等。本文的交互式推荐系统的建模如图 3.2 所示。

模型评估是评估推荐模型性能的过程，可以通过离线评估和在线评估来进行。本文选择的模型评估方式是离线评估，离线评估是在历史数据上进行模型评估，包括准确率(accuracy)、精确率(precision)、召回率(recall)、调和均值(F1)、均方根误差(RMSE)、平均绝对误差(MAE)和运行效率(operating efficiency)等指标。评估结果可以用于调整模型参数和优化推荐算法。

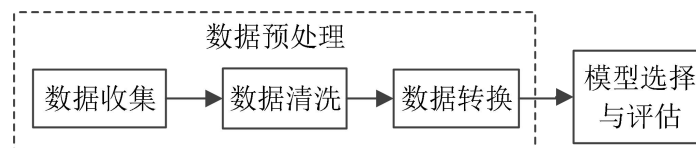


图 3.1 系统构建示意图

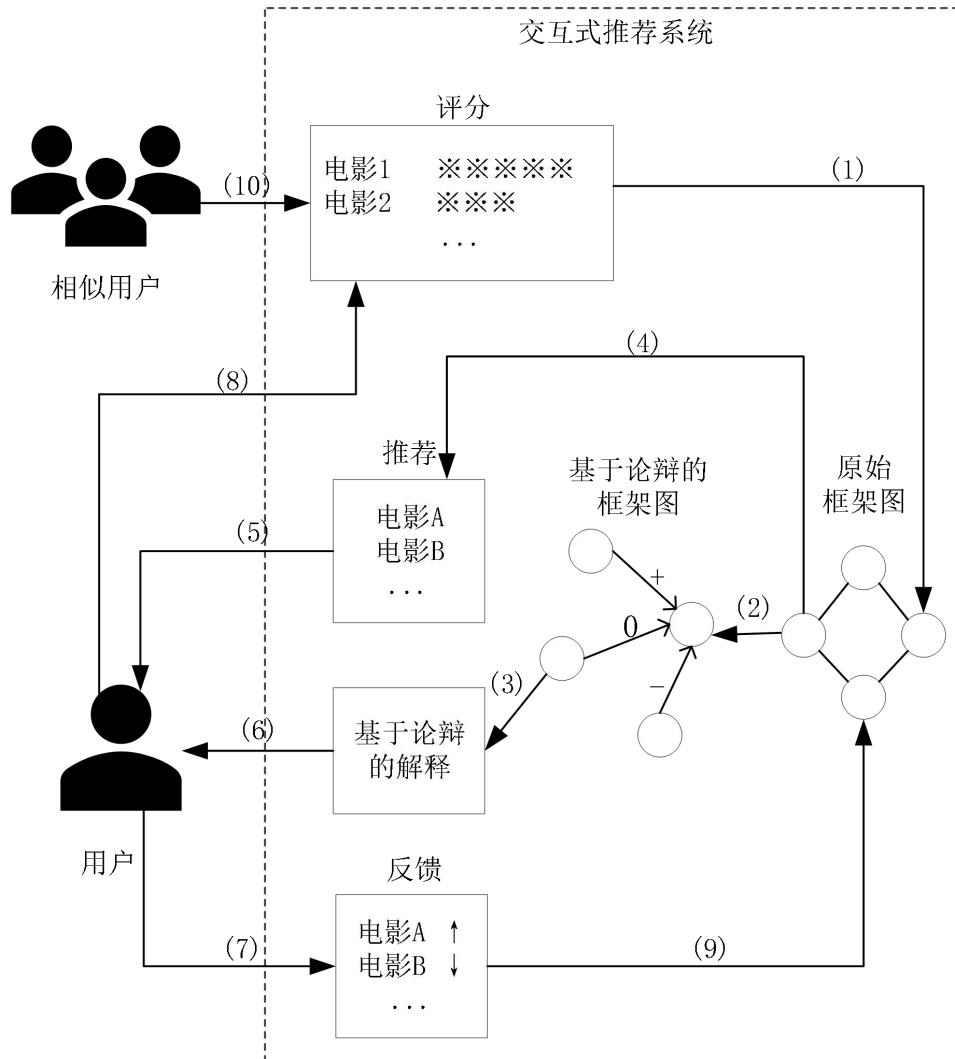


图 3.2 交互式推荐系统的建模

3.1.1 基于论辩的解释

本文引入了一种基于论辩的方法，以生成各种类型的解释，满足电影应用中的各种推荐评估措施。由于特定推荐系统的解释方式取决于计算推荐的方法，因此本文将重点关注如何根据不同的推荐方法生成相应的解释。例如，基于内容的推荐系统适合使用类似性质的解释，而协同过滤推荐系统更适合使用基于相似用户或相似项目的解释。本文为推荐的可解释性和透明性引入了 Aspect-Item 算法，方面-物品算法是一种基于图形化人工智能框架的推荐系统，其中物品（如电影）与其方面（如喜剧）相关联。这些联系形成一个基础图形架构，支撑着推荐系统。推荐是从用户和相似用户提供的评级计算出来的预测评级的混合方法的结果。为解决推荐系统中缺乏透明性和可解释性的问题，该算法采用论证技术提供可定制的推荐解释，同时有助于提高推荐的效果。算法使用三极论论证框架生成基于论

辩的解释。基于论辩的解释包括交谈和视觉解释等不同类型，以便为用户提供各种解释选择。研究表明，用户偏好解释的格式是多样化的。该算法可以支持反馈过程，以改进其行为。

本文采用的 **Aspect-Item** 算法依赖于连接电影项目和电影的各种信息方面。如图 3.2 所示，用户评分通过步骤(1)构成了底层的原始框架图，该框架包含了推荐信息，并组成了基于论辩的框架图的基础，经步骤(2)构成基于论辩的框架图，步骤(3)自动生成各种类型和格式的论辩解释，以支持用户的交互式推荐，包括提供对推荐项目及其方面的反馈机会。这些推荐来自于一种混合方法，即从用户和类似用户的评分中计算出预测的评分。该底层框架相当于三极论证框架，体现了基于论辩的解释，但通过在标准的“攻击”（标记为-）和“支持”（标记为+）关系之外加入一个“中和”关系（标记为0）来扩展经典的抽象^[13]和双极论证框架。如果将一个方面降低或增加其他方面的预测评级的影响程度，即攻击或支持强度，对这些攻击或支持强度的调整可被描述为削弱或加强攻击者或支持者。如图 3.3 所示，用户对电影 f_2 的评分是消极的，这种操作会使系统“攻击”类型 g_1 的预测评分，进而“攻击”用户对电影 f_1 的预测评分，最终综合多种方面对项目的影响，可以大致得出用户对项目的预测评分。这些关系被提取出来，以满足基于项目和方面的预测评级如何相互影响的逻辑要求。推荐系统和它的解释之间的这种同步性是本文方法的一个重要优势，因为用户可以相信解释描述了建议是如何产生的。论证性的解释包括对话式和视觉式的解释等多种形式。这些解释构成了与用户互动的基础，以解释建议和接收反馈，这些反馈可以被容纳到推荐系统中以改善其行为。因此，本文推荐的解释不仅是多种多样的，而且能够说明随着时间的推移，在有限的意义上提供可适应的建议。

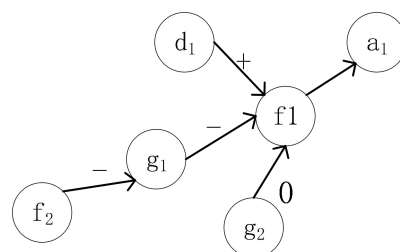


图 3.3 影响示意图

3.1.2 交互式实验

最近，学术界、工业界和政府都在推动人工智能的可解释性。尽管做出了这

些努力,增强可解释性并非易事。第一个问题是准确率和可解释性之间的权衡^[14],例如 Balog 等人认为在某种程度上牺牲前者来代替后者是值得的^[15]。另一个问题是,对于什么是好的解释没有通用的解决方案,考虑因素包括但不限于它应该包含的信息以及它如何传递给用户。事实上,不同的用户可能需要不同形式的解释并从中受益,因此解释需要适应他们所针对的人类用户。另一个问题是解释是否可以授权用户与系统交互,特别是提供对系统输出的反馈。社会科学清楚地表明,解释是人类的一个社会过程,人类对建议的解释仍然被认为比机器生成的解释质量更好,因此,激发了对用户的更多信任,尽管使生成的解释更丰富可能减轻了一些这种不足。此外,最近在推荐系统中的工作提倡将反馈集成为用户信任的关键。本文的目标是通过定义一个适应性强的混合推荐系统来解决其中的一些问题,它配备忠实于计算建议的方法,可根据不同的解释要求进行定制,并能够在旨在获得的交互中从人类那里引出反馈。这样既能提高用户的信任,也可以改进推荐系统,同时也是一种合理有效的方法。

如图 3.2,底层的原始框架通过混合推荐算法通过步骤(4)预测符合用户偏好的电影推荐,本文的交互式推荐系统经步骤(5)和步骤(6)将预测偏好电影和上文中提到的基于论辩的解释传递给用户,用户根据推荐的电影和各种形式的决策解释,即推荐这些电影的理由,并通过步骤(7)和步骤(8)反馈电影偏好和评分结果,由此可以得出用户实际偏好,步骤(9)将用户评分结果传递并影响原始框架图,以便于下次推荐可以更符合用户的喜好,通过反馈继续完善底层的原始框架,体现出该系统友好的交互式机制。

3.1.3 混合推荐系统

混合推荐系统(Hybrid Recommender System)是一种利用多种推荐算法协同工作的方法,旨在避免单个推荐算法所存在的问题,从而获得比单一算法更为优秀的推荐效果。混合推荐系统通常可以分为基于内容的推荐和基于协同过滤的推荐两类。基于内容的推荐系统会通过分析物品的属性等信息来进行推荐;而基于协同过滤的推荐系统则是根据相似性来寻找推荐目标。混合推荐系统能够充分利用多种算法的优势,提高推荐结果的准确率和覆盖率,并且能够根据实际应用情况灵活选择不同类型的算法以达到最佳的推荐效果。

基于内容的推荐系统是一种利用用户历史行为和物品属性等信息,通过计算

相似度来推荐与用户兴趣相关的物品的算法。相比协同过滤等推荐算法，基于内容的推荐系统更注重物品自身的属性和特征，能够更好地解决数据稀疏和冷启动等问题。基于内容的推荐系统的核心思想是根据物品的属性和特征来计算它们之间的相似度，从而推荐与用户兴趣相关的物品。具体来说，基于内容的推荐系统通常包括以下几个步骤：首先提取特征，对于每个物品，从其属性和特征中提取出一组关键词或特征向量，用于描述该物品的特征。然后计算相似度，通过计算不同物品之间的相似度来度量它们之间的相关性。最后生成推荐结果，根据用户历史行为和物品的相似度，计算推荐物品的得分，并按照得分排序生成推荐列表。

内容推荐系统不容易受到数据稀疏和流行度偏差等问题的影响，拥有较好的数据稳定性。基于内容的推荐系统不仅推荐准确率高：基于内容的推荐系统能够更好地理解物品的属性和特征，可以更准确地推荐与用户兴趣相关的物品，而且其具有较强可解释性，能够根据物品的属性和特征来解释推荐结果，更容易被用户理解和接受。但是基于内容的推荐系统也存在一些缺点，对于一些特征提取难度较大的物品，很难从它们的属性和特征中提取出有意义的特征。而且计算物品相似度复杂，计算物品之间的相似度需要消耗大量的计算资源和时间，尤其是在处理大规模数据时更为明显。对于新用户或新物品，由于缺乏历史数据，基于内容的推荐系统可能无法给出准确的推荐结果，难以解决冷启动问题，因此在本文中采用了基于用户的协同过滤算法应对冷启动问题，如图 3.2 所示，在用户没有评分时，通过步骤(10)收集相似用户评分，构建底层原始框架，以便于在用户冷启动时可以生成推荐。

协同过滤的推荐系统更适合基于相似用户或相似项目的解释，虽然一般人工智能方法在推荐系统领域也有所应用，但由于其黑盒特性，解释性方面存在一定难度。因此，本文认为推荐系统的可解释性趋势领先于一般的人工智能技术。基于协同过滤的推荐系统是一种常见的推荐算法，其基本思想是利用用户之间的相似性或物品之间的相似性来进行推荐。该算法不需要对物品或用户进行任何先验知识的假设，也不需要进行特征提取或建模，因此具有较好的灵活性和可扩展性。具体来说，协同过滤的推荐系统可以分为基于用户的协同过滤(User-based CF)和基于物品的协同过滤(Item-based CF)。

基于用户的协同过滤算法先计算用户之间的相似度，然后找到与目标用户相

似的一组用户，再将这些用户喜欢的物品作为推荐物品推荐给目标用户，最后去重并按照喜好程度排序。计算用户相似度使用的 Pearson 相关系数公式如式 3.1 所示。

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_{uv}} (r_{v,i} - \bar{r}_v)^2}} \quad (3.1)$$

其中 u 和 v 是要计算相似度的两个用户， I_{uv} 表示两个用户共同评价过的物品集合， $r_{u,i}$ 表示用户 u 对物品 i 的评分， $\bar{r}_u$ 表示用户 u 的所有评分的平均值。Pearson 相关系数公式的计算过程中，首先计算出两个用户对于共同评价的物品的评分的差值，随后将这些差值乘积的和除以两个用户评分差值的标准差乘积，即可得到两个用户之间的 Pearson 相关系数。Pearson 相关系数的取值范围为 $[-1,1]$ ，数值越接近于 1，表示两个用户的评分具有较高的相关性，即两个用户的偏好比较相似。在协同过滤算法中，可以根据 Pearson 相关系数的大小对用户进行相似度排序，并选取与目标用户最相似的一组用户进行推荐。使用余弦相似度计算用户相似度的公式如式 3.2 所示。

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I_u} r_{u,i}^2} \sqrt{\sum_{i \in I_v} r_{v,i}^2}} \quad (3.2)$$

其中 u 和 v 是要计算相似度的两个用户， I_{uv} 表示两个用户共同评价过的物品集合， $r_{u,i}$ 表示用户 u 对物品 i 的评分。余弦相似度公式的计算过程中，首先计算出两个用户对于共同评价的物品的评分的乘积和，随后将这些乘积和除以两个用户评分向量的模长乘积，即可得到两个用户之间的余弦相似度。余弦相似度的取值范围是 $[-1,1]$ ，与 Pearson 相关系数类似，数值越接近于 1，表示两个用户的评分具有较高的相关性，即两个用户的偏好比较相似。在实际应用中，可以根据数据特点和实验结果选择合适的相似度计算方法。皮尔逊相关系数的优点是能够消除数据的偏移影响，但在数据稀疏的情况下会出现问题；余弦相似度计算简单，能够有效处理数据稀疏的问题，但可能忽略了一些有用的信息。因此，在选择相似度计算方法时需要结合具体应用场景进行权衡。计算用户间相似度时可以采用欧几里

得距离方法，这种方法在推荐系统中也有应用。具体来说，计算两个用户 u 和 v 之间的欧几里得距离可以使用式 3.3。

$$sim(u, v) = \frac{1}{1 + \sqrt{\sum_{i \in I_{uv}} (r_{u,i} - r_{v,i})^2}} \quad (3.3)$$

其中 I_{uv} 表示用户 u 和 v 共同评价过的物品集合， $r_{u,i}$ 表示用户 u 对物品 i 的评分， $r_{v,i}$ 表示用户 v 对物品 i 的评分。这个公式将两个用户在共同评价的物品上评分差值的平方求和后开根号倒数，加上常数 1，转换为相似度值。

基于物品的协同过滤算法是一种经典的推荐算法，它可以通过分析用户对物品的评分或行为，计算物品之间的相似度，从而向用户推荐可能感兴趣的物品。具体实现过程包括特征提取、相似度计算和推荐生成等步骤。在特征提取阶段，需要将每个物品的属性、标签、文本描述等特征转化为向量形式以方便后续的相似度计算。在相似度计算阶段，对于每个用户已评分的物品，可以采用 Pearson 相关系数、余弦相似度等方法计算物品之间的相似度。最后，在推荐生成阶段，选择用户未评价过的相似物品进行推荐，并去除重复物品，并按照用户喜好程度排序。计算出物品之间的相似度后可以将它们存储在一个相似度矩阵中，其中每个元素代表两个物品之间的相似度。最后进行推荐生成。可以根据每个用户已经评分过的物品，找到与之相似的物品进行推荐。具体来说，根据用户评分过的物品可以在相似度矩阵中找到与之相似度最高的物品，然后将这些物品推荐给用户。计算物品相似度可以使用 Pearson 相关系数公式，该公式能够衡量两个变量之间的线性相关程度，其计算公式如式 3.4 所示。

$$sim(x_i, y_i) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (3.4)$$

其中， x_i 和 y_i 分别表示两个物品在用户评价中的评分， $\bar{x}$ 和 $\bar{y}$ 分别表示两个物品在用户评价中的平均分， n 表示用户数量。根据 Pearson 相关系数计算公式，首先要求出两个物品的共同评价过的用户数，并且这些用户对两个物品都有评分。然后，可以通过该公式计算出两个物品之间的相似度得分。如果得分越高，说明

两个物品之间越相似，推荐的可靠程度也越高。需要注意的是，Pearson 相关系数公式在计算相似度时容易受到极端值的影响，因此在实际应用中，需要进行数据归一化或者使用其他相似度计算方法来避免这一问题。计算物品相似度也可以使用余弦相似度，计算公式如式 3.5 所示。

$$sim(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} \quad (3.5)$$

其中， $\mathbf{a}$ 和 $\mathbf{b}$ 分别表示两个物品的向量表示， $\cdot$ 表示向量的点积， $\|\cdot\|$ 表示向量的范数，可以是 L_1 范数或 L_2 范数。在基于物品的协同过滤算法中，一般采用 L_2 范数。对于每个物品，可以将其向量表示保存在一个物品向量矩阵中。当需要计算某个物品与其他物品之间的相似度时，可以将该物品向量与其他物品向量逐一进行点积和范数计算，得到它们之间的余弦相似度得分。如果得分越高，说明两个物品之间越相似，推荐的可靠程度也越高。余弦相似度计算方法在处理稀疏数据时比较适用，因为它可以忽略原始向量中的零值。但是，如果在计算过程中存在负值，可能会导致余弦相似度取值范围为 $[-1, 1]$ ，而不是 $[0, 1]$ 。此外，当两个物品之间的特征空间维度非常高时，计算复杂度也会比较高。计算物品相似度还可以使用欧几里得距离，计算公式如式 3.6 所示。

$$sim(x_i, y_i) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (3.6)$$

其中， x_i 和 y_i 分别表示两个物品在用户评价中的评分， n 表示共同评价过这两个物品的用户数量。在基于物品的协同过滤算法中，只有那些同时被多数用户评价的物品对应的特征向量会包含大量的非零值，因此在计算欧几里得距离时，可以忽略那些未评价的部分，可以将每个物品的评分向量保存在一个物品矩阵中。欧几里得距离计算方法在计算物品相似度时可能会受到数据尺度差异的影响，因此在实际应用中需要对数据进行归一化处理。此外当物品评分向量的维度非常高时，计算复杂度也会比较高。基于物品的协同过滤算法简单直观、具有较好推荐效果和较广的适用范围。由于该算法可以利用物品之间的相似度来进行推荐，因此可以避免一些用户之间的偏好差异，从而提高推荐效果。然而，基于物品的协同过滤算法也存在一些缺点。首先，该算法需要先对物品进行特征提取，因此在新物品上推荐效果较差，具有冷启动问题。其次，在用户评分数据较少的情况下，该

算法容易出现稀疏性问题，从而影响推荐效果。此外，物品之间的相似度需要计算，因此在物品数量较多的情况下，计算复杂度较高也是该算法的缺点之一。

基于用户的协同过滤适用于用户数量较多，物品数量较少的情况，因为用户对物品评分的次数相对较多，即评分稠密度高；推荐结果需要有一定的解释性，更容易被用户接受等场景。例如，在电影推荐系统中，用户的花费时间和精力通常更多，而每部电影评价的人数往往较少；推荐结果需要快速准确的响应时间，如在线推荐等领域。例如，在图书推荐系统中，有大量的图书，而且没有太多的用户评分数据，因此利用基于物品的协同过滤来寻找相似的图书，并为用户进行预测和推荐。此外，基于物品的协同过滤还适用于在线广告推荐系统等需要快速响应的场景。

本文在 Rago 的工作基础上进行优化和完善^[16]，Rago 虽然使用了混合推荐系统，但是只有在冷启动时使用了基于用户的协同过滤算法。进行推荐结果展示是将推荐结果呈现给用户的过程，输入合法的用户名可以输出自定义个数的相似用户偏好的方面和电影。本文中提出的方法提供了一种可解释性强的推荐系统，这对于用户提供反馈以帮助系统适应用户喜好非常重要。

3.1.4 系统性能指标

准确率(Accuracy)是指推荐系统所推荐的物品中，与用户真实兴趣相符的占比。其计算公式如式 3.7 所示。

$$Accuracy = \frac{TP}{TP + FP} \quad (3.7)$$

其中，TP 表示 True Positive，意为系统正确地将用户感兴趣的物品推荐给了用户；FP 表示 False Positive，意为系统错误地将用户不感兴趣的物品推荐给用户。因此，准确率反映了推荐系统被评估的样本中，预测正例占有所有预测结果的比例。举个例子来说，假设某个用户有 5 个喜欢的电影，而推荐系统在推荐列表中成功地找到了其中 3 部电影，那么其准确率即为 60%。准确率越高，表示推荐系统越能够为用户提供符合其兴趣爱好和需求的推荐结果，反之则表示推荐系统存在一定的不准确率和误导性。然而在实际应用中，准确率并不能完全代表推荐系统的质量，因为其仅仅关注了正确分类的情况而忽略了未被预测的情况。对于存在物品冷启动、用户兴趣漂移等现象的推荐系统而言，准确率可能低估了算法的效果，需要考虑其他评价指标的综合影响。

精确率(Precision)是指推荐系统在所有推荐列表中，成功推荐给用户的物品占比。其计算公式如式 3.8 所示。

$$Precision = \frac{TP}{TP + FN} \quad (3.8)$$

其中，FN 表示 False Negative，意为系统未能将用户感兴趣的物品推荐给用户。因此，精确率反映了推荐系统被评估的样本中，真正正例（即用户感兴趣的物品）占有所有预测结果的比例。同样以电影推荐为例，若某个推荐系统在列表中共推荐了 10 部电影，其中有 4 部电影是符合用户喜好的，而系统也只推荐了这 4 部电影，那么其精确率就是 100%。从这个例子可以看出，精确率强调的是推荐系统所推荐的物品中，对用户感兴趣的物品所占的比例。与准确率相比，精确率更侧重于推荐系统对目标用户所感兴趣的类别（如电影、音乐、图书等）的推荐效果。总的来说，精确率是推荐系统中较为重要的评价指标之一，因为其可以有效衡量推荐系统推荐结果的准确率和可行性。同时，它还能避免由于冷门商品、物品稀疏等因素引起的准确率不准确的问题，提高推荐系统的实际应用价值。

召回率(Recall)是指在所有实际正例中，被算法正确识别为正例的比例。在推荐系统中，召回率可以用来衡量推荐系统的覆盖度和召回率。具体而言，召回率可以反映出推荐系统对用户真实兴趣的完整识别情况，即推荐系统能否将用户感兴趣的物品全部推荐给用户，其计算公式如式 3.9 所示。

$$Recall = \frac{TP}{TP + FN} \quad (3.9)$$

其中，TP 表示 True Positive，即算法将用户真实感兴趣的物品识别为正例的数量；FN 表示 False Negative，即算法将用户真实感兴趣的物品识别为负例的数量。召回率反映了数据集中实际正例被算法正确检测出来的比例，它的值域为[0,1]。召回率的常见使用场景包括物品召回和异常检测，其中物品召回是指在用户偏好挖掘、物品推荐等场景中，通过计算召回率指标来评估推荐系统对用户兴趣喜好的全面识别程度；异常检测是指在大规模物品过滤、垃圾邮件识别等场景中，通过计算 recall 指标来评估算法对正常样本识别的完整性。虽然召回率能够全面反映推荐系统对所有实际正例的覆盖程度，但是由于召回率只考虑了被识别为正例的数量，而未考虑负例被错误地识别为正例的数量，因此无法全面评估推荐系统的质量，而且召回率在数据集不平衡的情况下容易导致评价结果失真，所以要综合多个指标对系统性能进行评测。

调和均值(F1)是 precision 和 recall 的综合度量，通常被用来综合评价分类模型的性能。F1 值越高，说明模型的准确率和泛化性能越好。在推荐系统中，F1 值可用于综合衡量推荐系统对用户兴趣和需求的准确识别能力。F1 值的计算公式如式 3.10 所示。

$$F1 = 2 * \frac{precision * recall}{precision + recall} \quad (3.10)$$

其中, precision 和 recall 的含义与上述讲解相同, F1 可以理解为 precision 和 recall 的加权平均值, 所以当 precision 和 recall 的值相等时, F1 值最大。F1 值能够综合考虑精确率和召回率两个指标, 使得评价更加全面和客观, 能够更加准确地评价模型的表现。虽然 F1 具有一定的可解释性, 易于理解和解释, 但是由于 F1 使用了加权平均值, 因此对于数据集存在类别不平衡时, 可能导致评价结果失准, 而且 F1 值难以直观地衡量每一个样本的特征, 可能会掩盖掉不同类别样本间的差异性。

均方根误差(Root Mean Square Error, RMSE)是衡量模型预测值与真实值之间差距的一种评价指标, 通常用于回归任务中。在推荐系统中, RMSE 可以用来衡量推荐算法对用户兴趣和需求的准确识别能力。RMSE 值越小, 说明模型预测的结果越接近真实值, 即模型的准确率越高, 其计算公式如式 3.11 所示。

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}} \quad (3.11)$$

其中, y_i 是真实评分, $\hat{y}_i$ 是预测评分, n 是评分数量。在推荐系统中, RMSE 可以用于评估推荐算法对用户偏好和需求的准确度; 在回归模型中, RMSE 可以作为评价指标, 用于衡量模型的预测精度; 在计算机视觉、自然语言处理等领域, RMSE 也被广泛应用于模型评价和优化。均方根误差易于计算和理解, 能够客观地反映出模型预测值与真实值之间的差距, 但是它对异常值比较敏感, 会受到极端值的影响, 无法直接解释每一个样本的特征差异。

平均绝对误差(Mean Absolute Error, MAE)是另一种评价回归模型性能指标, 衡量的是预测值与真实值之间的平均绝对误差。在推荐系统中, MAE 可用于评价推荐算法对用户个性化需求的准确率, 其计算公式如式 3.12 所示。

$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n} \quad (3.12)$$

其中， y_i 是真实评分， $\hat{y}_i$ 是预测评分， n 是评分数量。在推荐系统中，平均绝对误差可以用于评估推荐算法对用户偏好和需求的准确度；在回归模型中，平均绝对误差可以作为评价指标，用于衡量模型的预测精度；在其他领域中，平均绝对误差也被广泛应用于模型评价和优化。平均绝对误差对异常值不敏感，可以避免极端值对评价结果的影响，而且它易于计算和解释，能够直观地反映出模型预测值与真实值之间的差距，但是平均绝对误差无法区分预测值偏大和偏小的情况，因此可能会掩盖掉模型误差的一些特征，而且平均绝对误差不能直接用来比较不同模型的性能，因为不同模型的平均绝对误差是难以比较的。

运行效率(operating efficiency)是系统对用户进行个性化推荐时所需的时间和计算资源。具体来说，它由以下几个方面组成：数据处理效率是指系统对原始数据进行处理和分析的速度和效果，包括数据清洗、去重、归一化、特征提取等；模型训练效率指系统在对用户行为数据进行建模和训练时所需的时间和计算资源，包括模型选择、参数调整、训练算法等；推荐效率指系统在实际推荐过程中所需的时间和计算资源，包括向量相似度计算、推荐列表生成、排序等；并发性能指系统在面对大量并发请求时所能承受的压力和处理能力，包括负载均衡、缓存机制、分布式部署等。通过优化这些方面，可以提高推荐系统的运行效率和性能，提高用户对系统的满意度和使用体验。在实际应用中，需要根据系统的业务特点和用户需求来综合考虑这些指标，才能确定评估和优化的方向，提高系统的运行效率。根据运行时间得出运行效率变化，是通过对某个系统或过程的运行时间的测量，来推断它的运行效率的变化情况。这种方法可以对系统或过程的运行效率进行评估和提高，以达到优化系统或过程的目的。以运行时间为基准的运行效率可以通过如式 3.13 表示效率提升的程度。

$$x = \frac{t_1 - t_2}{t_1} \quad (3.13)$$

其中 t_1 为原有的运行时间， t_2 为优化后的运行时间。从式中可以看出，效率提升了 x 越大，代表着优化效果越明显。以运行时间为基准对比运行效率的方法简单

易操作，使用这种方法仅需要测量系统或过程的运行时间，具有较高的可靠性，通过多次测试，能够得到比较准确的结果，从而有助于进一步提高系统或过程的运行效率。而且具有较强的适用性，该方法适用于各类系统和过程的评估，能够帮助运维人员进行数据中心等的管理和维护，提高资源利用效率。但是使用这种方法会导致时间依赖性强，因为该方法所得出的结果仅基于运行时间，对于运行时间变化不大的系统或过程，效果可能不明显，如果在测试过程中发生其他因素（如网络连接不稳定、硬件故障等）对测试结果的干扰，可能会影响评估的准确率。该方法只关注系统或过程的运行时间变化，并没有考虑其他指标（如资源消耗量、运行稳定性等）的变化情况，可能导致评估的偏差。

3.2 数据预处理

本文采用 Movielens-100k, Movielens-latest-small 和 Netflix 三个数据集来测试实验的有效性，其中 Movielens-100k 是历史最悠久的数据集，Movielens-latest-small 是最新的小型数据集，而 Netflix 数据集则是 Netflix 公司在 2006 年推出的一个数据集，目的是为了吸引更多的数据科学家和工程师来帮助公司提升电影推荐算法的精度和效果。Movielens-100k 数据集和 Movielens-latest-small 数据集都是 Movielens 数据集的一部分，而 Netflix 数据集是为了精简样本容量，更快捷计算系统性能，选取了 Netflix Prize 数据集其中的一部分，Netflix Prize 数据集是由 Netflix 公司发布的独立数据集。

3.2.1 Movielens-100k 数据集

Movielens-100k 数据集是一个包含 10 万条评分数据的数据集，包含 943 名用户对 1682 部电影的评分数据。数据集中的用户评分分为 1-5 分，其中 1 分表示最差，5 分表示最好。除了评分数据之外，Movielens-100k 数据集还包含了电影的元数据，包括电影的名称、类型、导演、演员等信息。数据集的存储格式为 CSV 格式，其中包含三个文件：u.data、u.item 和 u.user，u.data 文件包含了所有的评分数据，每一行包含了用户 ID、电影 ID、评分和时间戳信息，其中时间戳信息可以用来进行时间序列分析，u.item 文件包含了所有电影的元数据信息，每一行包含了电影 ID、电影名称、发行时间、IMDB 链接、电影类型等信息，u.user 文件包含了所有用户的元数据信息，每一行包含了用户 ID、性别、年龄、职业等信息。

对于 Movielens-100k 数据集的处理方式，一般可以采用 Python 进行处理。常用的处理方式包括数据清洗、数据转换、数据可视化等。在数据清洗方面可以对数据进行去重、缺失值处理、异常值处理等。在数据转换方面可以将数据进行格式转换、数据合并等操作。在数据可视化方面可以采用 Matplotlib、Seaborn 等库来绘制各种图表，以便更好地理解数据。

3.2.2 Movielens-latest-small 数据集

Movielens-latest-small 数据集是一个包含 100000 条评分数据的数据集，包含了 610 名用户对 9724 部电影的评分数据。和 Movielens-100k 数据集相比，Movielens-latest-small 数据集更加新，也更加小型化，但是包含的评分数据和电影元数据信息都比较丰富。数据集的存储格式为 CSV 格式，包含三个文件：ratings.csv、movies.csv 和 tags.csv。其中，ratings.csv 文件包含了所有的评分数据，每一行包含了用户 ID、电影 ID、评分和时间戳信息，movies.csv 文件包含了所有电影的元数据信息，每一行包含了电影 ID、电影名称、电影类型等信息，tags.csv 文件包含了所有用户对电影的标签信息，每一行包含了用户 ID、电影 ID、标签和时间戳信息。

对于 Movielens-latest-small 数据集的处理方式，和 Movielens-100k 数据集类似，也可以采用 Python 进行处理。常用的处理方式包括数据清洗、数据转换、数据可视化等。

3.2.3 Netflix 数据集

Netflix 数据集是一个包含 10 万条评分数据的数据集，包含了 480189 名用户对 17770 部电影的评分数据。和 Movielens 数据集相比，Netflix 数据集更加大型化，也更加复杂，包含了更多的评分数据和电影元数据信息。数据集的存储格式为文本格式，包含两个文件：training_set 和 probe_set。其中，training_set 文件包含了所有的评分数据，每一行包含了用户 ID、电影 ID、评分和时间戳信息，probe_set 文件包含了用于测试的评分数据，每一行包含了用户 ID、电影 ID 和时间戳信息。

对于 Netflix 数据集的处理方式，由于数据集比较大，因此需要使用分布式计算框架进行处理，例如 Hadoop、Spark 等。常用的处理方式包括数据清洗、特征工程、模型训练等。这些步骤需要仔细处理，以确保数据集的准确率和可靠性，

从而提高推荐算法的性能和效果。

总体来说，上述三个数据集都是电影推荐领域比较有名的数据集，每个数据集的原始格式都是 CSV 格式，但本研究为了方便处理数据，将数据集全部转换为 PKL 格式，每个数据集整合分为评分(ratings)数据集和电影(films)数据集。处理后的数据集分类信息见表 3.1 所示。

表 3.1 数据集分类

数据集	评分个数	电影个数
Movielens-100k	1682	1239
Movielens-latest-small	9724	9551
Netflix	528	528

Movielens-100k 数据集中的评分集有 1682 个记录、电影集有 1239 个记录，Movielens-latest-small 数据集中的评分集有 9724 个记录、电影集有 9551 个记录，Netflix 数据集中的评分集有 528 个记录、电影集有 528 个记录。

在本文中，为了数据的一致性和准确率，在使用这些数据集时删除了那些只有少量评分记录的电影和那些没有足够信息的用户。此外，本文还使用了 TMDb API 来获取电影的特定方面，如类型、演员和导演等。最终，本文使用余弦距离计算相似度，并针对每个用户的特定评分记录对推荐进行个性化。因此，本文去除了某些电影集和评分集中的数据，以确保数据集中的信息是足够且客观关联的，并且基于这些数据构建的混合推荐系统能够提供准确的推荐并解释推荐的原因。本文去除数据的方式是通过差集找出在评分集但不在电影集中的电影号存放在 `ids_to_del_rf` 集合中和在电影集但不在评分集中的电影号放在 `ids_to_del_fr` 集合中，通过这两个集合的交集找到在评分集和电影集中都不存在的电影号，并将这些电影号添加到 `ids_to_del` 集合中。通过该集合中的电影号可以删除无效的评分或电影，以确保数据集中的信息是足够的且客观关联的，这就是基于这些数据构建的混合推荐系统能够提供准确的推荐并解释推荐的原因。

评分集包含的方面有用户评分、用户评分日期和用户号，一部电影可以包含多个用户的评分信息。用户评分在集合 {1,2,3,4,5} 中选取，用户评分日期的表示方式在不同的数据集中不同。其中，在 Movielens 数据集中，用户评分的日期是以整数形式存储，表示形式为 Unix 时间戳，指从 1970 年 1 月 1 日 00:00:00 到该评分时间的秒数；在 Netflix 数据集中，用户评分日期是一个时间戳，以年-月-日的格式表示。在去除无效的评分和电影后数据集中，用户号存储格式为 'x'+初

始的用户号。因为每个电影都有多个用户对其进行评分，所以评分的数量会远远超过电影的数量。具体来说，评分集中的每一项都对应一个电影，其中包含了多个用户对该电影的评分记录，每条记录都包含了一组用户评分、评分日期、用户号的信息。因此，所有电影的评分记录数量之和会远远超过电影的数量。更新后的评分集的存储格式为：{'film_id': [{'user_rating': ' ', 'user_rating_data': ' ', 'user_id': ' '}], ...}。评分集具体分类信息见表 2 所示。

表 3.2 评分集包含方面

数据集	用户 评分 {1, 2, 3, 4, 5}	用户评 分日期 x>100	用户号				用户 总数
			评价 x 部 x>50	电影的用 x>30	户数 10<=x <=30		
Movielens-100k	259716	86572	312	510	689	254	943
Movielens-latest -small	301134	100378	244	376	498	112	610
Netflix	779454	259818	365	1077	1923	2849	26968

电影集包含的方面有导演、演员、标题和类型，一部电影可以包含多个演员信息和类型信息，但一部电影只能有一个导演信息和一个类型信息。在本文涉及的三个数据集中，演员方面和类型方面的存储格式是列表，标题方面的存储格式是字符串，但是导演方面的存储格式在不同的数据集中不相同。其中，在 Movielens 数据集中，导演方面的存储格式为列表，在 Netflix 数据集中，导演方面的存储格式为字符串。更新后的 Movielens 电影集的存储格式为：

{'film_id': {'director': [' '], 'actors': [' ', ' ', ' '], 'title': ' ', 'genre': [' ', ' '] }, ...}, Netflix 电影集的存储格式为：{'film_id': {'director': ' ', 'genre': [' ', ' '], 'actors': [' ', ' ', ' '], 'title': ' ' }, ...}，两者的不同在于在 movielens 中 director 以列表方式存储，在 netflix 中 director 以字符串方式存储，且两者存储方面的顺序不同。电影集具体分类信息见表 3.3 所示。

表 3.3 电影集包含方面

数据集	导演数	演员数	标题数	类型数	总方面数
Movielens-100k	867	3442	1239	19	4328
Movielens-latest-small	4347	18027	9551	20	22394
Netflix	389	745	528	20	1154

总而言之，上述三个数据集都有其独特的特点和优势。Movielens-100k 是一个包含 259716 个电影评分的数据集，其中包含 943 个用户对 1239 部电影的评分。

MovieLens-latest-small 是一个包含最新电影评分的较小数据集，其中包含 301134 个评分，由 610 个用户对 9551 部电影进行评分。Netflix 数据集是一个包含 779454 个评分的数据集，由 26968 个用户对 528 部电影进行的评分所构成。

MovieLens-100k 和 MovieLens-latest-small 包含更多的电影数据，而 Netflix 数据集包含更多的用户和评分数据。

3.3 算法实现

机器算法的基本思想是从计算机领域中解决的问题来推导出相应的数学结构，然后运用计算机算法来解决该问题。这些算法通常是数学公式和逻辑符号的集合，计算机可以根据这些算法指令进行编程，并在运算过程中实现具体的功能。因此，机器算法的实现需要对数学和计算机科学领域有深入的研究。

机器算法分为有监督学习、无监督学习和半监督学习三种类型。其中，有监督学习是指在学习过程中每个样本都有一个标签或目标值，计算机通过学习这些数据来理解问题，构建一个从输入到输出的映射关系，寻找最优解决方案。本文中进行对比实验的 KNN 算法、Z-KNN 算法属于有监督算法，因为这两个算法基于已有的评分数据集进行训练和预测。无监督学习则是在训练数据中没有标签或目标值，计算机需要根据已有的数据进行分类、聚类等操作，学习数据的内在结构，发现数据中的规律和模式，以便实现更加优秀的方案。本文中进行对比实验的 SVD 算法、NMF 算法、Slope One 算法、CoClustering 算法属于无监督算法，因为 SVD 算法和 NMF 算法不需要已知的标签或分类信息，只需要根据数据本身的结构进行模型训练，Slope One 算法主要用于协同过滤推荐系统中的评分预测，同样不需要训练数据中的标签或类别信息，CoClustering 算法是一种聚类算法，旨在将数据集中的数据点分组为具有相似特征的子集，以便于进一步的分析和处理，也不需要使用已知的标签或输出来训练模型。半监督学习是指在训练数据中，只有一部分样本有标签或目标值，利用这些有标签或目标值的样本来指导学习过程，同时利用没有标签或目标值的样本来发现数据中的结构和规律。为了与本文的方面-物品算法更明显地对比算法性能和优缺点，需要详细介绍本文涉及与该算法进行对比实验的基线算法。

3.3.1 基于 KNN 算法的实现

K 邻近(KNN, K-Nearest Neighbor)是数据挖掘分类技术中最基础的方法之一。该算法通过寻找样本空间中与当前样本最近的 K 个邻居来进行分类。每个样本可以用其 K 个最接近邻居的值来代表, 以实现数据集中每个记录的分类和回归。K 近邻法是一种监督学习方法, 它假设给定一个包含已定类别的训练数据集, 并利用这些样本数据来预测新实例的类别。在分类时, K 近邻法通过多数表决等方式, 根据新实例的 K 个最近邻居的训练实例类别来进行预测。这种算法简单易懂, 且适用于各种数据类型, 因此被广泛应用于数据挖掘、模式识别、图像处理等领域。KNN 方法是一种基于样本相似度的分类方法, 具有简单易懂、易于实现、效果较好等优点。但是, KNN 方法的计算复杂度较高, 对数据量大的数据集不太适用。

KNN 算法的步骤如下:

Step1: 准备一个有标签的数据集, 包括训练样本和测试样本。输入 k 值, 其中 k 表示选择距离待分类样本最近的 k 个已知类别的样本;

Step2: 计算每个待分类样本与训练集中所有样本之间的欧氏距离;

Step3: 根据计算出的距离排序, 选择与待分类样本距离最近的 k 个训练样本;

Step4: 选择 k 个最近邻的类别, 通过选择类别出现频率的方式确定待分类样本的类别。如果 k 为 1, 则将待分类样本分配给与其最近的训练样本的类别; 如果 k 大于 1, 则统计 k 个最近邻的类别;

Step5: 根据出现次数最多的类别确定待分类样本的类别。

KNN 算法具体流程步骤见图 3.3 所示。

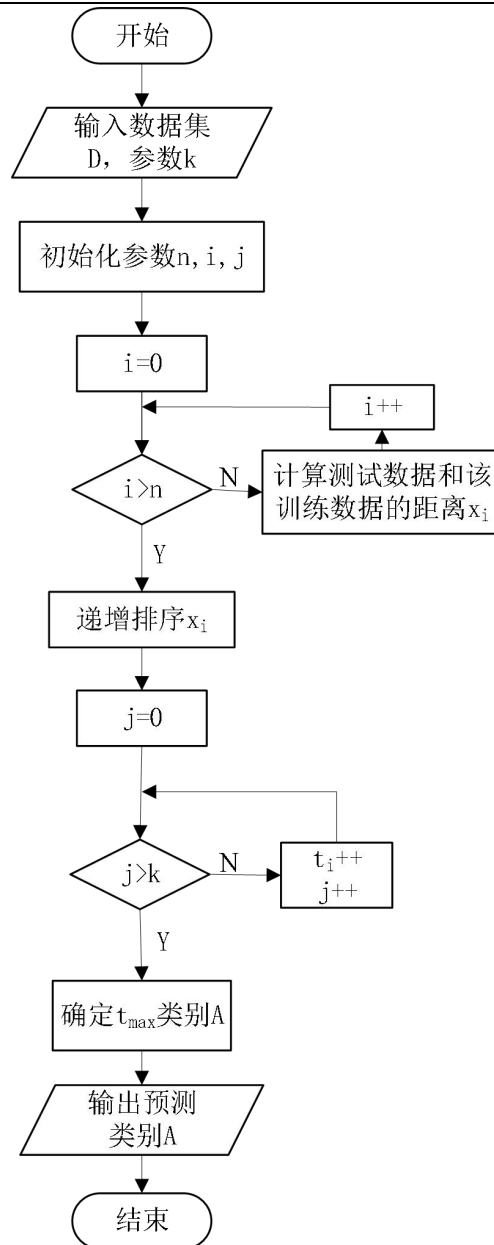


图 3.3 KNN 算法流程图

KNN 算法的优点在于简单易懂、适合处理稀有事件、适合多分类问题，并且可以通过施加距离权重来避免样本不平衡的问题。但是，该算法存在样本不平衡时分类效果差、计算量大的缺点，因为需要对每个待分类文本计算其到所有已知样本的距离，可以通过剪辑样本点或分层分群来优化。

3.3.2 基于 Z-KNN 算法的实现

Z-KNN 算法通过同时考虑均值和方差来对标准的 KNN 算法进行改进。Z-KNN 是一种基于邻居的协同过滤算法，可以用于推荐系统中的用户-物品推荐问题。它是基于 KNN 算法和 Z-Score 标准化技术的改进。KNN 算法是一种基于

相似度的算法，它通过计算物品之间的相似度来预测用户对未评价物品的评分。Z-KNN 算法在 KNN 算法的基础上，引入了 Z-Score 标准化技术。这种技术可以将物品的评分转换为标准分数，使得不同用户的评分具有可比性。具体来说，对于每个用户，Z-KNN 算法先计算其评分的平均值和标准差，然后将每个评分转换为标准分数。这样，不同用户之间的评分就可以进行比较，从而更准确地计算相似度。KNN 算法中基于用户的评分预估方法为式 3.14 所示。

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot r_{vi}}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)} \quad (3.14)$$

其中 $N_i^k(u)$ 为评价过项目 i 的用户中与当前用户 u 最相似的前 k 个用户的集合；基于物品的评分预估方法为式 3.15 所示。

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} \text{sim}(i, j) \cdot r_{uj}}{\sum_{j \in N_u^k(i)} \text{sim}(i, j)} \quad (3.15)$$

其中 $N_u^k(i)$ 为用户 u 评价过的项目集合中与当前项目 i 最相似的前 k 个项目的集合。但对于 Z 值标准化的 KNN 算法，基于用户的评分预估方法为式 3.16 所示。

$$\hat{r}_{ui} = \mu_u + \sigma_u \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot (r_{vi} - \mu_v) / \sigma_v}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)} \quad (3.16)$$

基于物品的得分预估方法为式 3.17 所示。

$$\hat{r}_{ui} = \mu_i + \sigma_i \frac{\sum_{j \in N_u^k(i)} \text{sim}(i, j) \cdot (r_{uj} - \mu_j) / \sigma_j}{\sum_{j \in N_u^k(i)} \text{sim}(i, j)} \quad (3.17)$$

其中和 KNN 算法评分计算公式相同参数的含义相同， μ_u 为用户 u 所有评分的均值， μ_v 为用户 v 所有评分的均值，且用户 v 是评价过项目 i 的用户中与用户 u 最相似的前 k 个用户中的其中之一， σ 为对应的用户评分的方差。Z 值标准化的 KNN 算法与普通的 KNN 算法的主要区别在于，前者考虑了每个用户的评分与其个人习惯之间的关系，即每个用户的评分会在一个特定的值上下波动，而且还会详细考虑用户的评分习惯规律。

Z-KNN 算法的优点在于它可以处理稀疏矩阵，因为它只考虑了用户和物品之间的关系，而没有考虑其他因素。此外，Z-Score 标准化技术可以使得算法对于不同用户的评分具有可比性，从而提高了推荐的准确率。Z 值标准化的 KNN 算法和 KNN 算法在算法流程上的区别是，Z 值标准化的 KNN 算法在遍历完所有训练数据，选取 k 个与测试数据最相似的邻居后，需要计算用户所有评分的均值 μ 和用户评分的方差 σ 。因为用户评分和用户习惯有关，所以要基于某个特定值确定 k 个邻居训练集，最后输出的预测类别即为该邻居训练集中出现频率最高的类别。Z-KNN 算法的具体算法流程图见图 3.4 所示。

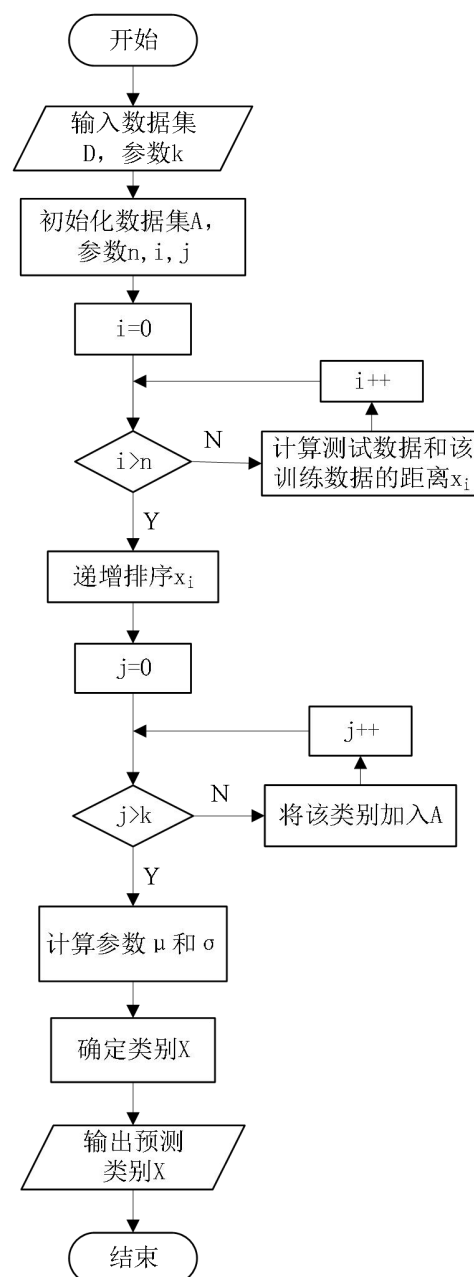


图 3.4 Z-KNN 算法流程图

总而言之，Z-KNN 算法是一种基于邻居的协同过滤算法，它通过计算物品之间的相似度和 Z-Score 标准化技术来预测用户对未评价物品的评分，具有较高的推荐准确率和稀疏矩阵处理能力。

3.3.3 基于 SVD 算法的实现

奇异值分解(SVD)是一种极为强大的算法，可用于解决最小二乘问题。它不仅适用于拟合直线和平面，还可获取点云的最小边界框。作为一种线性代数算法，奇异值分解算法能够对矩阵进行分解，并从中提取关键信息以降低原始数据规模。广泛应用领域包括信号处理、金融、统计学以及机器学习。实际上，在推荐系统、搜索引擎、数据压缩等诸多机器学习领域中，奇异值分解算法的应用也十分普遍。

奇异值分解是一种适用于任何矩阵的分解方法，核心思想是假设原始数据集矩阵 A 是一个 $m \times n$ 的矩阵，利用奇异值分解算法，可以将它分解成三个部分：

$A = U \Sigma V^T$ ，对任意 $m \times n$ 的矩阵，都可以找到一组正交基使得经过它变换后还是正交基。通过奇异值分解，可以得到一些类似于特征分解所获取信息的信息。虽然几乎所有实数矩阵都有一个奇异值分解，但并不是所有矩阵都有特征分解。例如，非方阵的矩阵就没有特征分解，这时只能使用奇异值分解来处理。

为了证明上述奇异值分解的核心思想成立，可以假设存在 $m \times n$ 矩阵 A ，事实上， A 矩阵将 n 维空间中的向量映射到 $k(k \leq m)$ 维空间中， $k = \text{Rank}(A)$ 。现在的目标就是：在 n 维空间中找一组正交基，使得经过 A 变换后还是正交的。假设已经找到这样一组正交基： $\{v_1, v_2, \dots, v_n\}$ ，则 A 矩阵将这组基映射为：

$\{Av_1, Av_2, \dots, Av_n\}$ 。如果要使他们两两正交，即 $Av_i \cdot Av_j = (Av_i)^T Av_j = v_i^T A^T Av_j = 0$ 。

根据假设，存在 $v_i^T v_j = v_i \cdot v_j = 0$ ，所以如果正交基 v 选择为 $A^T A$ 的特征向量的话，

由于 $A^T A$ 是对称阵， v 之间两两正交，那么 $v_i^T A^T Av_j = v_i^T \lambda_j v_j = \lambda_j v_i^T v_j = \lambda_j v_i \cdot v_j = 0$ 。

这样就找到了正交基使其映射后还是正交基了，现在可以将映射后的正交基单位化：

因为 $Av_i \cdot Av_i = \lambda_i v_i \cdot v_i = \lambda_i$ ，所以有 $|Av_i|^2 = \lambda_i \geq 0$ ，所以取单位向量

$u_i = \frac{Av_i}{|Av_i|} = \frac{1}{\sqrt{\lambda_i}} Av_i$ ，由此可得 $Av_i = \sigma_i u_i$ ， σ_i (奇异值) $= \sqrt{\lambda_i}$ $0 \leq i \leq k$ ， $k = \text{Rank}(A)$ 。

当 $k < i \leq m$ 时，对 $u_1, u_2, \dots, u_k$ 进行扩展 $u_{(k+1)}, \dots, u_m$ ，使得 $u_1, u_2, \dots, u_m$ 为 m 维空间中

的一组正交基，即将 $\{u_1, u_2, \dots, u_k\}$ 正交基扩展成 $\{u_1, u_2, \dots, u_m\}$ R^m 空间的单位正交基。

同样的，对 $v_1, v_2, \dots, v_k$ 进行扩展 $v_{(k+1)}, \dots, v_n$ （这 $n-k$ 个向量存在于 A 的零空间中，即 $Ax=O$ 的解空间的基），使得 $v_1, v_2, \dots, v_n$ 为 n 维空间中的一组正交基，即在 A

的零空间中选取 $\{v_{(k+1)}, \dots, v_n\}$ 使得 $Av_i=0, i>k$ ，并取 $\sigma_i=0$ ，则可得到

$$A[v_1 \ v_2 \ \cdots \ v_k \ | \ v_{k+1} \ \cdots \ v_n] = [u_1 \ u_2 \ \cdots \ u_k \ | \ u_{k+1} \ \cdots \ u_m] \begin{bmatrix} \sigma_1 & & & 0 \\ & \ddots & & \\ & & \sigma_k & \\ & & 0 & 0 \end{bmatrix}, \text{ 继而可以得到 } A$$

矩阵的奇异值分解： $A=U\Sigma V^T$ ，其中 V 是 $n \times n$ 的正交阵， U 是 $m \times m$ 的正交阵，

Σ 是 $m \times n$ 的对角阵，即可证明对任意 $m \times n$ 的矩阵，都可以找到一组正交基使得经过它变换后还是正交基。

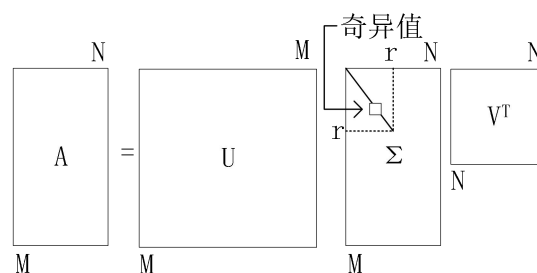


图 3.5 奇异值分解

将矩阵 A 分解成三个矩阵的乘积 $A=U\Sigma V^T$ ，矩阵 U 和 V 都定义为正交矩阵，而矩阵 Σ 定义为对角矩阵，矩阵 Σ 不一定是方阵。如图 3.5 所示，对角矩阵 Σ 对角线上的元素被称为矩阵 A 的奇异值。事实上，可以用与 A 相关的特征分解去解释 A 的奇异值分解。即： U 是 AA^T 的特征向量。 V 是 AA^T 的特征向量。 A 的非零奇异值即 Σ 中非零值是特征值的平方根，同时也是特征值的平方根。这是求 U 、 Σ 、 V 的方式。

奇异值分解算法的优点在于经过奇异值分解算法分解的奇异值降低很快，只要 10%甚至是 1%的奇异值就占据了全部奇异值之和的 99%以上的比例。所以只需要筛选出其中很少的 k 个奇异值和对应的左右奇异向量就可以近似描述原矩阵了，不需要完整的 SVD 分解结果。由图 3.6 可知，对角矩阵 Σ 对角线上的元素被称为矩阵 A 的奇异值，矩阵 U 的列向量被称为左奇异向量，矩阵 V 的列向量被称右奇异向量，相当于从分解出来的矩阵当中筛选一小部分来代替整体，并

且保留和整体近似的信息。

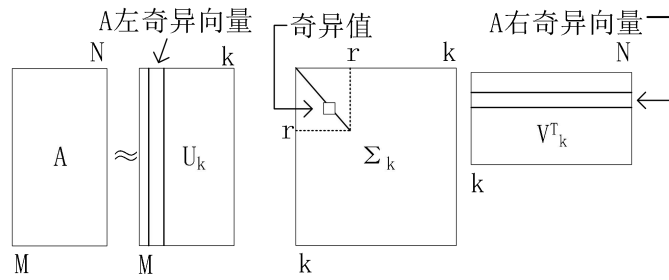


图 3.6 奇异值分解算法示意图

由 $A_{m \times n} = U_{m \times m} \Sigma_{m \times n} V_{n \times n}^T \approx U_{m \times k} \Sigma_{k \times k} V_{k \times n}^T$ 可知，这里的 k 远小于 n ，所以可以大大降低 SVD 分解之后得到的矩阵参数的数量。也就是说，通过奇异值分解，将一个 $m \times n$ 的大矩阵，分解成了三个小得多的小矩阵。并且通过这三个小矩阵，可以还原出原矩阵大部分的信息。

奇异值分解可以将一个矩阵分解成三个矩阵的乘积的形式，其中一个矩阵是左奇异矩阵，一个矩阵是右奇异矩阵，另一个矩阵则是由矩阵的奇异值组成的对角矩阵。该算法的具体步骤如下：

Step1: 将数据列表转换为矩阵 A ，对于该 $m \times n$ 的矩阵 A ，计算它的转置矩阵 A^T 与矩阵 A 的乘积 AA^T ，并求出 AA^T 的特征值和特征向量；

Step2: 将特征向量按照特征值大小从大到小排列，选择前 r 个特征向量构成一个 $n \times r$ 的矩阵 U 。其中 r 是一个较小的数，通常取 $r = \min(m, n)$

Step3: 对于矩阵 A ，计算其与矩阵 U 的乘积 $AV = U\Sigma$ ，其中 Σ 是一个 $r \times r$ 的对角矩阵，对角线上的元素是 AA^T 的特征值的平方根，对于矩阵 Σ ，除了对角线上的 r 个元素外，其余元素均为 0；

Step4: 将矩阵 Σ 扩展成一个 $m \times n$ 的矩阵 Σ' ，其中除了对角线上的 r 个元素外，其余元素均为 0；

Step5: 计算矩阵 V ，它是由 AA^T 的特征向量按照对应的特征值大小从大到小排列构成的 $n \times r$ 矩阵；

Step6: 将矩阵 U 、 Σ' 和矩阵 V 相乘得到原始矩阵 A 的近似矩阵： $A \approx U\Sigma'V^T$ ，由此得出预测矩阵。

奇异值分解算法流程图如 3.7 所示。

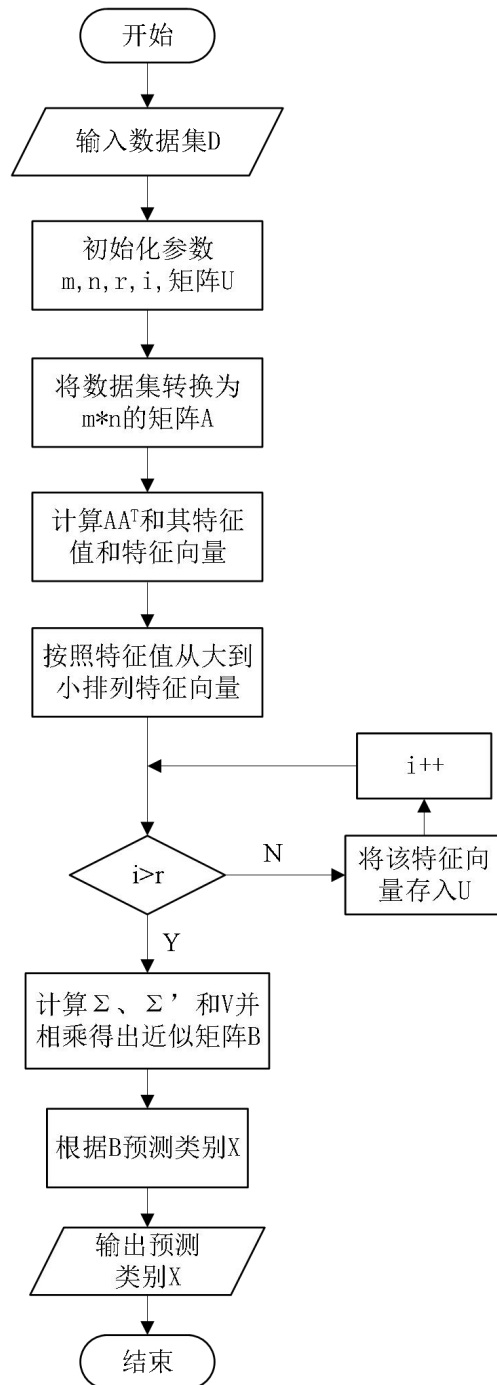


图 3.7 SVD 算法流程图

概括奇异值分解算法，对于一个 $m \times n$ 的实数矩阵 A ，它的秩为 r ，则有：

$A_{m \times n} = U_{m \times r} \Sigma_{r \times r} V_{n \times r}^T$ ，其中 $U_{m \times r}$ 和 $V_{n \times r}$ 是正交矩阵， $\Sigma_{r \times r}$ 里面的元素称为奇异值，

是一个非负对角矩阵。同时 U 和 V 两个矩阵都满足奇异值分解可以用于 $U^T U = I_r$ ，

$V^T V = I_r$ 。在实际应用中，常常采用截断 SVD 的方法，即选取比较小的 k 值进

行压缩，并且只保留前 k 个最大的奇异值和对应的左、右奇异矩阵。奇异值分解

算法可以用于数据压缩、数据降维、矩阵逆运算等多种场景，可以更好地理解和处理大量的数据，尤其是高维度的数据。

3.3.4 基于 NMF 算法的实现

非负矩阵分解(NMF)是一种用于数据降维和特征提取的技术。非负矩阵分解算法的目标是将一个非负矩阵分解成两个非负矩阵的乘积，即 $A_{m \times n} = W_{m \times k} H_{k \times n}$ ，其中 W 和 H 都是非负矩阵。NMF 算法的基本思想是将一个高维、稀疏的数据矩阵 M 分解为两个低维矩阵 W 和 H 的乘积，即 $M \approx WH$ 。其中， W 是一个大小为 $m \times r$ 的非负矩阵， H 是一个大小为 $r \times n$ 的非负矩阵， m 和 n 分别表示数据样本数和特征数， r 表示需要提取的潜在特征数量。可以将 W 看作是样本在新的潜在特征空间中的表示，将 H 看作是特征在样本空间中的表示。在推荐系统中，可以将用户评分或物品评分作为数据样本，从而得到用户-物品评分矩阵或物品-用户评分矩阵。这些矩阵中的项可以被看作是特征在样本空间中的表示，通过 NMF 算法可以对矩阵进行分解并得到用户和物品的潜在特征表示。

非负矩阵分解算法的步骤通常如下：

Step1: 需要进行数据预处理，准备用户-物品评分矩阵，并对缺失值进行处理，比如采用均值填充或零填充；

Step2: 初始化矩阵，根据需要提取的特征数量初始化矩阵 W 和矩阵 H ，其中 W 为一个 $m \times r$ 形状的矩阵， H 为一个 $r \times n$ 形状的矩阵， m 和 n 分别表示用户数和物品数， r 表示特征数量；

Step3: 进行迭代优化，NMF 算法采用交替最小二乘(Alternating Least Squares, ALS)法进行迭代优化，首先通过固定矩阵 H ，来优化矩阵 W ，然后再通过固定矩阵 W ，来优化矩阵 H 。具体来讲，优化矩阵可以通过公式完成，更新矩阵 W 的公式为式 3.18。

$$W_{a,k} = W_{a,k} \frac{(AH^T)_{a,k}}{(WHH^T)_{a,k}} \quad (3.18)$$

其中， A 为用户-物品评分矩阵。更新矩阵 H 的公式如式 3.19。

$$H_{a,b} = H_{a,b} \frac{(W^T A)_{a,b}}{(W^T WH)_{a,b}} \quad (3.19)$$

其中， $(W^T A)$ 是矩阵的内积， $(W^T WH)$ 是矩阵的外积。通过最小化误差函数来更

新矩阵,使得误差不断减小,通常使用均方根误差或平均绝对误差作为误差指标;

Step4: 在每一次迭代过程中,重复计算误差和更新矩阵操作,即步骤 3,当误差降到一定值或达到最大迭代次数时,算法停止迭代;

Step5: 输出矩阵 W 和矩阵 H 的乘积,即 WH 矩阵,可以利用这个矩阵来预测缺失值或进行推荐。推荐列表的生成可以基于用户-物品评分矩阵,使用余弦相似性或欧几里得距离等度量方法来计算用户与物品之间的相似度或距离,并将相似度或距离作为推荐结果的排序依据。需要说明的是,NMF 算法需要指定特征数量,在实践中需要通过交叉验证等手段来进行调参和评估算法性能。非负矩阵分解算法流程图如图 3.8 所示。

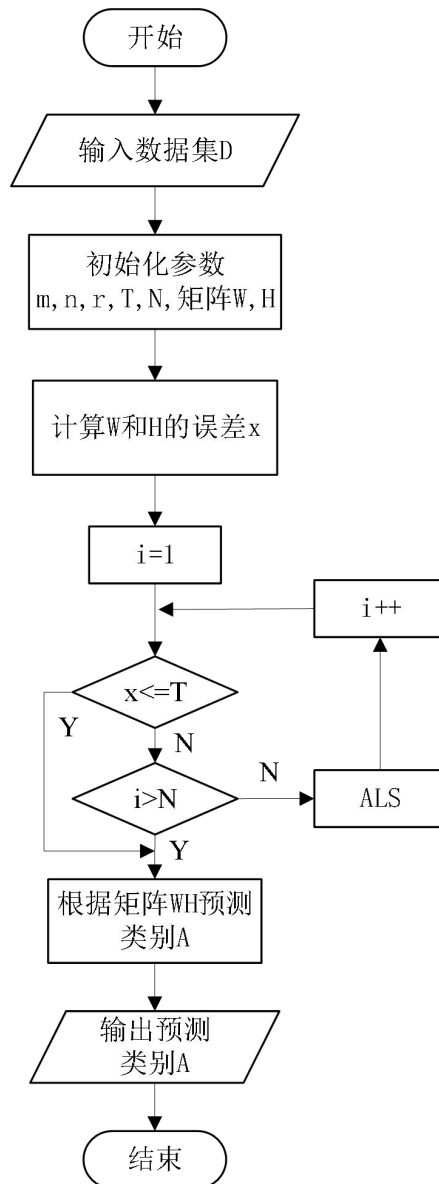


图 3.8 NMF 算法流程图

非负矩阵分解算法作为一种无监督学习算法，不依赖于用户的历史行为数据，不需要用户的显式反馈，因此该算法可以在没有或较少用户反馈信息的情况下进行推荐，而且该算法可以提取用户和物品的潜在特征表示，具有很好的可解释性，并且对于数据的噪声和异常值具有一定的鲁棒性。但是非负矩阵分解算法训练时间较长，需要经过多次迭代才能得到较好的模型效果。不仅非负矩阵分解算法的特征数量需要预先设定，不同的数据集需要不同的特征数量，通常需要进行多组参数调节才能得到最优参数，而且该算法容易陷入局部最优解，需要使用多组随机初值或增加正则化项等方法来减小这种风险。非负矩阵分解算法可以提取用户和物品的潜在特征表示，同时具有很好的可解释性和鲁棒性。推荐系统中的非负矩阵分解算法，需要通过数据预处理、初始化矩阵、迭代优化等步骤来实现。同时，在实际应用中，需要注意特征数量的设定、多组随机初值的采用以及局部最优解的问题。

3.3.5 基于 SlopeOne 算法的实现

推荐系统中的 Slope One 算法是一种基于膨胀型协同过滤算法(inflated collaborative filtering algorithm)的推荐算法，它可以通过计算物品之间的平均差异来预测用户对新物品的评分。Slope One 算法的原理很简单，它通过计算物品之间的平均差异来预测用户对未评分物品的评分。具体来说，它首先计算出每对物品之间的评分差异，然后对这些差异求平均值。这个平均值被称为“物品之间的偏差”，它表示了两个物品之间的相对评分差异。例如，如果 A 和 B 是两个物品，它们之间的偏差是 1.5，表示用户更喜欢 A 而不是 B。一旦计算出了物品之间的偏差，Slope One 算法就可以用它来预测用户对未评分物品的评分。具体来说，它首先找到用户评分过的所有物品，然后计算出这些物品与目标物品之间的偏差。最后，它将这些偏差加权平均，得到用户对目标物品的预测评分。

Slope One 算法的步骤通常如下：

Step1: 进行数据预处理，需要准备好用户-物品评分矩阵 R ，其中每个元素 $R_{u,i}$ 表示用户 u 对物品 i 的评分。在进行 Slope One 算法之前，需要对评分矩阵进行简单的预处理，计算出物品之间的评分差异信息。

Step2: 计算出物品之间的差异信息。用 $dev(i, j)$ 表示物品 i 和物品 j 之间的平均差异，可以定义如式 3.20 所示。

$$dev(i, j) = \frac{1}{|U_{i,j}|} \sum_{u \in U_{i,j}} (r_{u,i} - r_{u,j}) \quad (3.20)$$

其中， $U_{i,j}$ 表示同时对物品 i 和物品 j 进行评分的用户集合， $r_{u,i}$ 表示用户 u 对物品 i 的评分， $r_{u,j}$ 表示用户 u 对物品 j 的评分；

Step3: 通过计算用户已评价过的物品与待评价物品之间的评分差异，来预测用户对待评价物品的评分。假设用户 u 已经评价了若干个物品 $i_1, i_2, \dots, i_n$ ，对于一个待评价物品 j ，可以使用式 3.21 来预测用户 u 对它的评分。

$$p(u, j) = \frac{1}{|I_u^j|} \sum_{i \in I_u^j} (dev(i, j) + r_{u,i}) \quad (3.21)$$

其中， I_u^j 表示用户 u 已评价物品集合与待评价物品 j 的交集，即式 3.22 所示。

$$I_u^j = \{i \in R_u \mid r_{u,i} \neq ? \wedge (j, i) \in P\} \quad (3.22)$$

其中 P 表示所有用户-物品评分对的集合， $|I_u^j|$ 表示集合大小， $dev(i, j)$ 表示物品 i 和物品 j 的平均差异， $r_{u,i}$ 表示用户 u 对物品 i 的评分，"?"表示未知的评分值。

Step4: 将预测评分进行排序，选择前 N 个评分最高的物品作为推荐结果输出，根据预测评分来推荐物品。

SlopeOne 算法具体流程如图 3.9 所示。

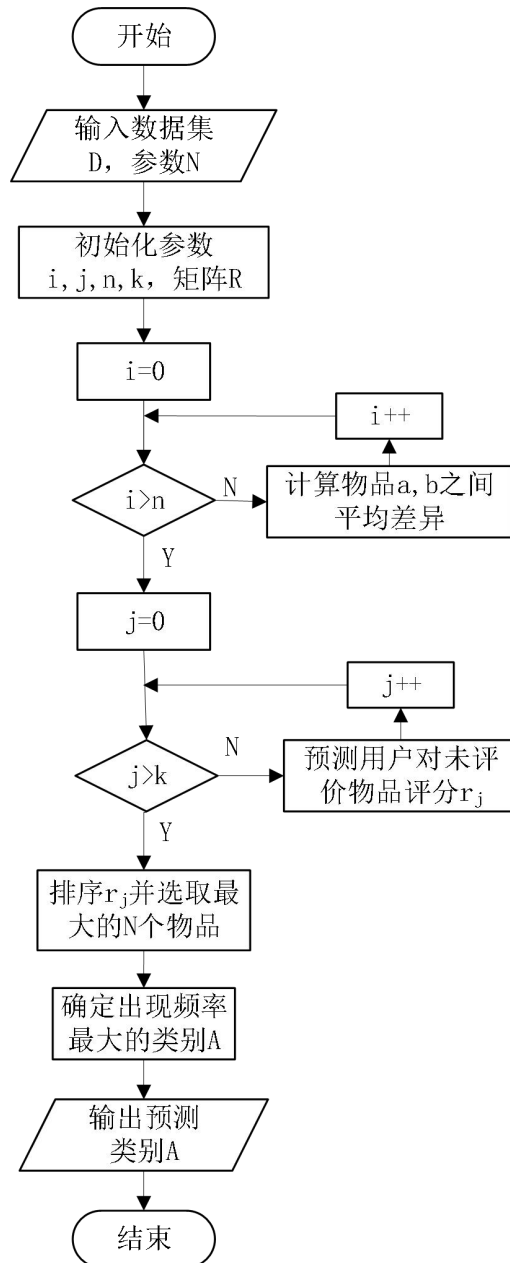


图 3.9 SlopeOne 算法流程图

Slope One 算法的优点是它原理简单，易于实现和理解。它不需要复杂的数学公式或高级的机器学习技术，只需要一些简单的数学运算和数据处理。此外，它的计算复杂度比其他协同过滤算法低，可以处理大规模的数据集。然而 Slope One 算法也有一些缺点。首先，它只考虑了物品之间的偏差，而没有考虑用户之间的偏好差异。这可能导致一些用户的预测评分不准确。其次，它对于稀疏数据集的处理效果不佳，因为它需要大量的评分数据才能计算出物品之间的偏差。最后，它不能处理非数值型的评分数据，例如文本评价或二元评价。Slope One 算法是一种简单但有效的协同过滤算法，适用于大规模的推荐系统。它的实现简单，

计算复杂度低，可以处理数值型的评分数据，但使用该算法需要注意其适用范围和局限性。

3.3.6 基于 CoClustering 算法的实现

CoClustering 算法是一种基于矩阵分解的聚类算法，它可以同时对行和列进行聚类，将矩阵分解成多个子矩阵，每个子矩阵都包含一组相似的行和列。CoClustering 算法可以应用于文本挖掘，用于将文本数据集分成多个主题。例如，可以将一组新闻文章分成多个主题，每个主题包含一组相关的文章。CoClustering 算法可以应用于图像处理，用于将图像分成多个子图像。例如可以将一张大图像分成多个小图像，每个小图像都包含一组相似的像素。CoClustering 算法可以应用于生物信息学，用于将基因表达数据分成多个基因簇。例如，可以将一组基因表达数据分成多个基因簇，每个基因簇都包含一组相似的基因。CoClustering 算法的基本原理是将原始矩阵分解成多个子矩阵，每个子矩阵都包含一组相似的行和列。这些子矩阵可以看作是矩阵的“子群”，它们在行和列方向上都有一定的相似性。通过分解矩阵，可以得到一组“子群”，这些“子群”可以用来描述原始数据集的结构和特征。CoClustering 算法的分解过程可以用矩阵分解的方式来实现。具体来说，可以将原始矩阵分解成两个矩阵 U 和 V ，其中 U 表示行的聚类结果， V 表示列的聚类结果。这两个矩阵的乘积可以得到原始矩阵。

CoClustering 算法的流程如下：

Step1: 初始化两个矩阵 U 和 V ，其中 U 的行数等于原始矩阵的行数， V 的列数等于原始矩阵的列数；

Step2: 计算原始矩阵与 $U \cdot V$ 的误差，其中误差可以用欧几里得距离来表示；

Step3: 通过最小化误差来更新 U 和 V ，具体来说，可以使用梯度下降法来更新 U 和 V ；

Step4: 重复步骤 2 和 3，直到误差收敛或达到最大迭代次数；

Step5: 根据 U 和 V 的聚类结果得到“子群”，即为预测结果。

CoClustering 算法具体流程如图 3.10 所示。

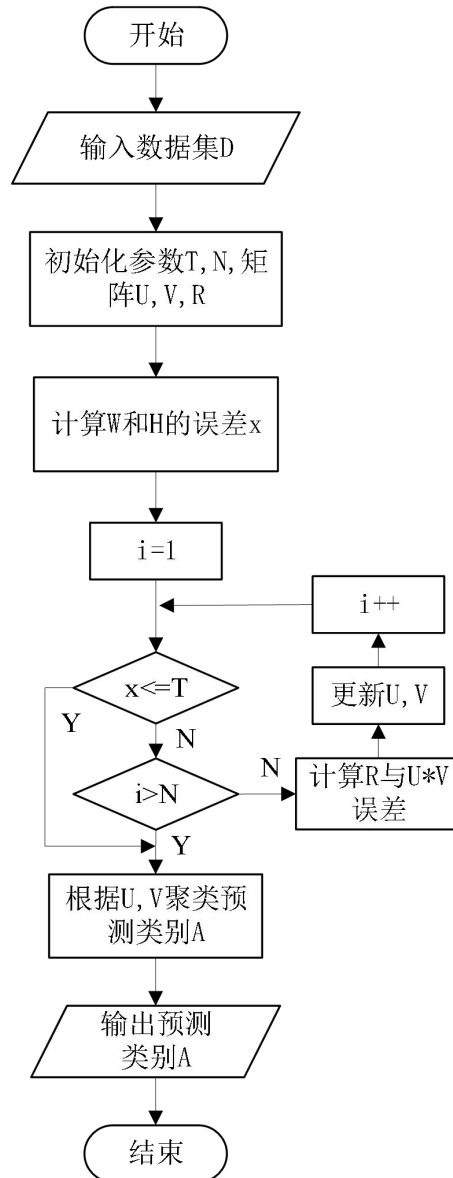


图 3.10 CoClustering 算法流程图

CoClustering 算法是一种基于矩阵分解的聚类算法，它可以同时对行和列进行聚类，将矩阵分解成多个子矩阵，每个子矩阵都包含一组相似的行和列。CoClustering 算法可以应用于各种领域，如文本挖掘、图像处理、生物信息学等。常见的 Coclustering 算法包括 Spectral CoClustering、Biclustering、Probabilistic CoClustering 等。

3.3.7 基于 Aspect-Item 算法的实现

Aspect-Item 算法是推荐系统中一种常见的解决方案，其主要思想是将用户评价中的不同方面和物品进行分离，并建立相应的模型进行推荐。Aspect-Item 算法相较于传统的基于物品或基于用户的协同过滤算法，需要额外考虑 aspect

之间的关系,并通过子模型对 aspect 进行建模。该算法需要更加高效的计算资源、更多的数据量以及更合适的评价指标才能取得较好的推荐效果。

Aspect-item 算法相较于其他推荐算法可以更好地考虑用户兴趣的多样性,根据用户的不同兴趣点,为其推荐不同的商品,从而更好地考虑用户兴趣的多样性;该算法不仅可以考虑商品的多个方面,比如商品的价格、品牌、风格等,从而更好地满足用户的需求;也可以推荐长尾商品,从而更好地解决长尾问题;还可以根据商品的属性信息,为新用户推荐适合其兴趣的商品,从而更好地解决冷启动问题;并且 Aspect-item 算法可以根据商品的属性信息,为用户提供更加明确的推荐解释,从而更好地解决推荐过程中的“黑盒子”问题。Aspect-Item 算法的目标是通过考虑用户兴趣和项目特征的细节,更加精确地预测用户对项目的喜好并提供更好的推荐。

Aspect-Item 算法的流程如下:

Step1: 收集用户行为数据,包括用户对不同电影项目的评分等;

Step2: 从用户评价中提取出重要的方面;

Step3: 将每个电影项目与其相关的方面进行匹配,计算电影项目在每个方面的评分;

Step4: 对于每个用户,根据历史行为和兴趣,计算出用户对每个方面的偏好程度;

Step5: 根据用户的偏好程度和电影项目在方面上的得分,计算出每个电影项目的预测评分;

Step6: 根据预测评分对电影项目进行降序排序,输出排名前列的电影项目作为推荐结果。

Aspect-Item 算法具体流程如图 3.11 所示。

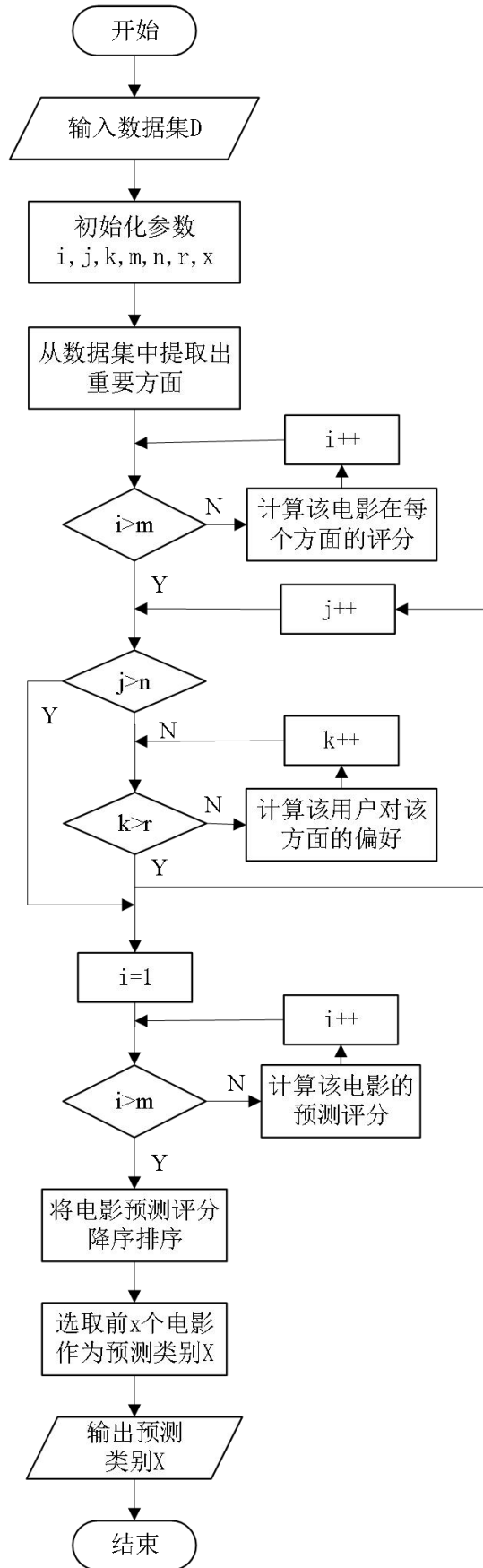


图 3.11 Aspect-Item 算法流程图

3.3.8 基于 XAI 算法的实现

XAI 算法通过考虑数据格式和处理方式来对 RAI 算法，即 Rago 的 Aspect-Item 算法进行改进。为了数据的一致性和准确率，在使用数据集实验时删除了那些只有少量评分记录的电影和那些没有足够信息的用户。XAI 算法将所有导演方面数据格式统一为字符串格式，并改进了处理数据的方式，使得 XAI 算法不仅保持 RAI 算法在 Netflix 数据集上实验，而且推广适用于 Movielens 数据集。

同时提高推荐精确率和召回率是 XAI 算法的亮点之一。本文设计与实现的 XAI 算法引入了论辩解释的概念，通过给予解释，向用户明确了推荐决策的依据和理由，使得用户可以更好地理解推荐结果背后的原因。这不仅有助于提高用户的信任度，还能够提高推荐系统的精确率和召回率，因为用户在理解推荐决策的过程中能够提供更加准确的反馈信息，以此提高推荐决策的透明度与精确率和召回率的调和均值，使得用户能够得到更加满意的推荐结果。相较于 RAI 算法，本文设计的 XAI 算法不仅运行时间减少 9%，而且在 Movielens 数据集上表现更加出色。通过减少运行时间，本文不仅提高了推荐系统的效率，还可以为用户提供更为快捷和便利的服务，提升用户体验。

XAI 算法的流程如下：

Step1: 收集用户行为数据，包括用户对不同电影项目的评分等；

Step2: 从用户评价中提取出重要的方面；

Step3: 对方面数据进行预处理；

Step4: 将每个电影项目与其相关的方面进行匹配，计算电影项目在每个方面的评分；

Step5: 对于每个用户，根据历史行为和兴趣，计算出用户对每个方面的偏好程度；

Step6: 根据用户的偏好程度和电影项目在方面上的得分，计算出每个电影项目的预测评分；

Step7: 根据预测评分对电影项目进行降序排序，输出排名前列的电影项目作为推荐结果。

XAI 算法具体流程如图 3.12 所示。

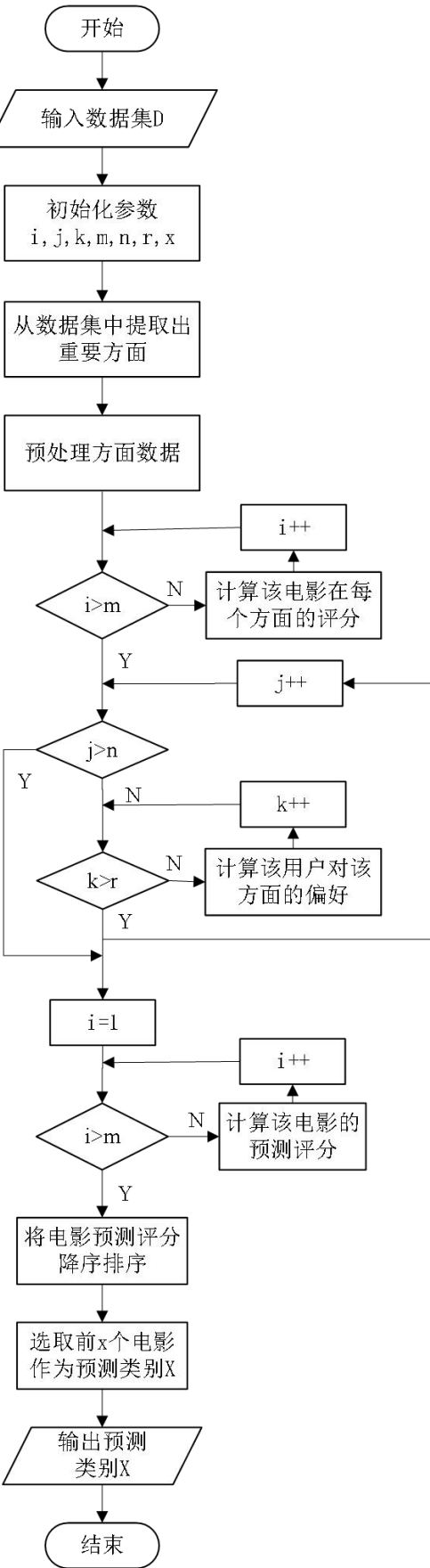


图 3.12 XAI 算法流程图

4 实验结果分析

交互式电影推荐系统是一种基于用户历史行为和偏好的推荐算法，能够为用户提供个性化的电影推荐服务。本文使用了多种不同的算法，包括基于协同过滤的推荐算法、基于内容的推荐算法以及混合推荐算法，对推荐系统的性能进行了评估和比较。

首先，本文对用户满意度和准确率进行了评估。在实验中采用了交叉验证的方法，将数据集分为训练集和测试集，通过对测试集中用户对电影的评分进行预测，来评估推荐系统的准确率和用户满意度。通过对比不同算法的表现，可以发现基于协同过滤的推荐算法和混合推荐算法在准确率和用户满意度方面表现较好，而基于内容过滤的推荐算法则表现较差。其次，本文对精确率和召回率进行了评估。精确率是指推荐系统中推荐的电影中，用户实际喜欢的电影所占的比例，而召回率则是指在所有用户喜欢的电影中，推荐系统成功推荐的电影所占的比例。通过对比不同算法的精确率和召回率，可以发现基于协同过滤的算法和混合算法在精确率和召回率方面表现较好，而基于内容过滤的算法则表现较差。接下来，本文对均方根误差和平均绝对误差进行了评估。均方根误差和平均绝对误差都是评估推荐系统预测准确率的指标。通过对比不同算法的均方根误差和平均绝对误差，可以发现基于协同过滤的算法和混合算法在预测准确率方面表现较好，而基于内容过滤的算法则表现较差。最后，本文对算法的性能进行了比较。在实验中使用了不同的评估指标来评估算法的性能，包括准确率、精确率和召回率、均方根误差和平均绝对误差。

在数据集方面，本文对比了 Netflix 数据集、Movielens-100k 数据集和 Movielens-latest-small 数据集；在算法方面，本文对比了 KNN、Z-KNN、SVD、NMF、Slope One 和 CoClustering 常用的推荐算法、Rago 研究的 Aspect-Item 算法（简称 RAI）和本实验的 Explainable Aspect Item 算法（简称 XAI）。相较于 RAI 算法只适用于 Netflix 数据集，XAI 算法可适用于 Netflix、Movielens-100k 和 Movielens-latest-small 数据集，即本文所有数据集。从实验结果角度，由于对数据进行了预处理，XAI 算法比相关算法的精确率有显著提高，并且运行时间上相较于 RAI 算法有 9% 的提升。各算法在各数据集上具体实验数据见表 4.1 所示。

表 4.1 实验数据表

数据集	系统指标	算法							
		KNN	KNNZ	SVD	NMF	Slope1	CoClust	RAI	XAI
Netflix	准确率	0.85	0.85	0.88	0.84	0.85	0.84	0.77	0.77
	精确率	0.82	0.84	0.83	0.85	0.84	0.85	0.82	0.82
	召回率	0.99	0.96	0.98	0.93	0.95	0.94	0.84	0.84
	调和均值	0.90	0.90	0.90	0.89	0.90	0.89	0.83	0.83
	均方根误差	1.05	1.02	0.99	1.05	1.02	1.04	1.32	1.32
	平均绝对误差	0.77	0.73	0.71	0.76	0.74	0.75	0.97	0.97
	运行时间/s	/	/	/	/	/	/	2100.37	1912.28
Small	准确率	0.79	0.80	0.82	0.79	0.80	0.79	/	0.73
	精确率	0.84	0.86	0.86	0.86	0.87	0.87	0.84	0.89
	召回率	0.94	0.92	0.94	0.90	0.90	0.87	0.85	0.95
	调和均值	0.89	0.89	0.90	0.88	0.88	0.87	0.85	0.92
	均方根误差	0.98	0.93	0.91	0.96	0.94	0.98	1.19	1.71
	平均绝对误差	0.74	0.69	0.68	0.73	0.71	0.74	0.90	1.16
	运行时间/s	/	/	/	/	/	/	/	46127.12
100k	准确率	0.83	0.83	0.87	0.83	0.84	0.82	/	0.77
	精确率	0.85	0.86	0.85	0.86	0.86	0.86	0.86	0.87
	召回率	0.97	0.97	0.99	0.95	0.96	0.95	0.93	0.94
	调和均值	0.91	0.91	0.92	0.90	0.91	0.90	0.89	0.90
	均方根误差	1.12	1.06	1.02	1.11	1.09	1.10	1.13	1.44
	平均绝对误差	0.81	0.76	0.74	0.80	0.79	0.80	0.85	1.00
	运行时间/s	/	/	/	/	/	/	/	1885.52

4.1 准确率

随着互联网和智能设备的普及，交互式电影推荐系统越来越受到人们的关注。其中，准确率是交互式电影推荐系统的一个重要指标，它能够直接影响到用户的认可度和使用体验。本文将从准确率对交互式电影推荐系统的影响和作用两个方面进行探讨。本文整合了 KNN、Z-KNN、SVD、NMF、Slope One、CoClustering、RAI 及 XAI 算法，准确率对比如图 4.1 所示，可以清晰看出数据均值比较，也可以在图 4.2 中更直观看出了数据中位数及分布情况，图 4.2 中上下线为各算法性能的上下四分位数，箱中横线即为各算法数据中位线，空心正方形为各算法该性能的均值。

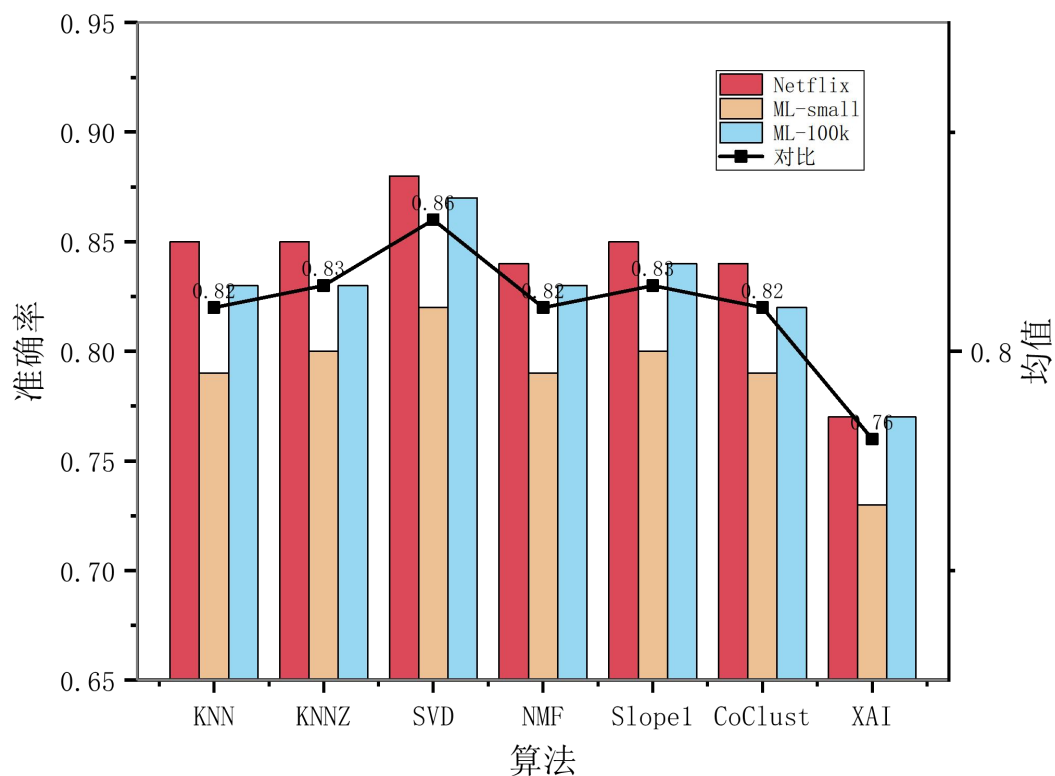


图 4.1 准确率柱状折线图

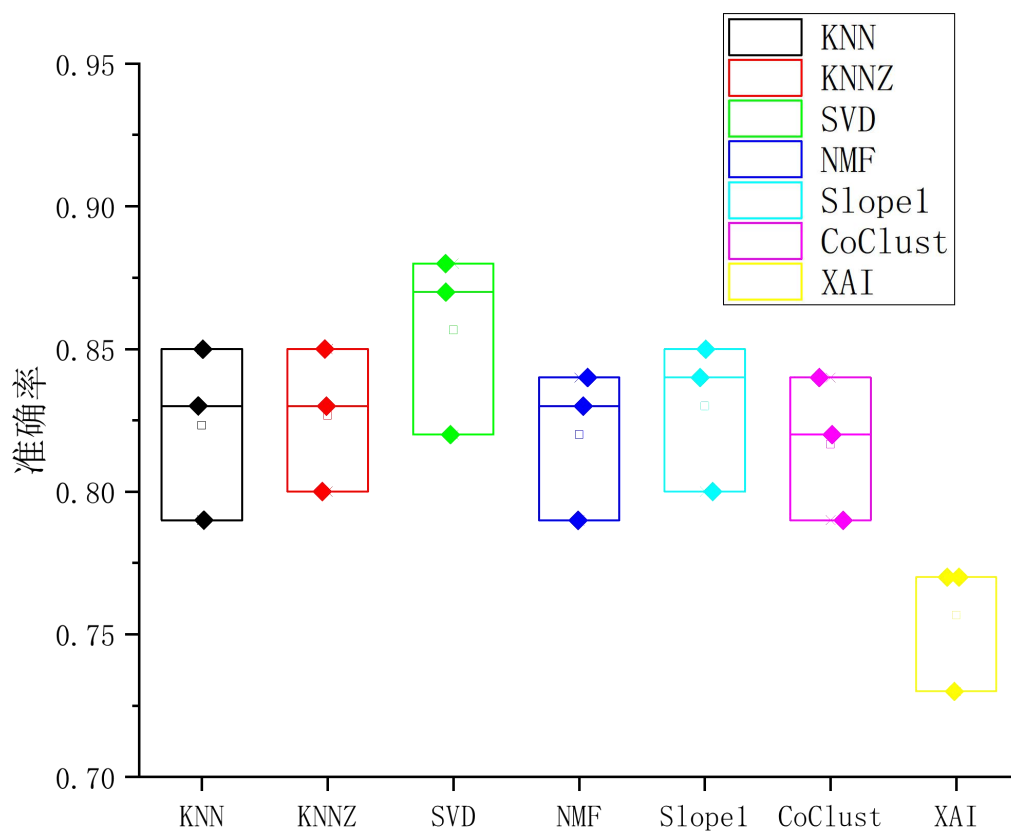


图 4.2 准确率箱线图

当涉及到交互式电影推荐系统时，准确率是影响其推荐效果的重要因素之一。在实际应用中，准确率往往是评价一个推荐系统的重要指标之一。因此，一个准确率高的交互式电影推荐系统对于用户的体验和满意度具有重要的影响和作用。准确率是交互式电影推荐系统的核心指标之一，它能够直接影响到推荐系统的推荐结果和用户的满意度。准确率高的推荐系统能够更好地满足用户的需求，提供更加精准的推荐结果，从而增强用户的体验和满意度。相反，准确率低的推荐系统可能会给用户带来不必要的困扰，降低用户的满意度。因此，提高交互式电影推荐系统的准确率是非常重要的。准确率对交互式电影推荐系统的推荐效果具有重要的影响和作用。为了提高准确率，可以采用以下几种方法：优化推荐算法。推荐算法是影响推荐准确率的关键因素之一，因此，优化推荐算法是提高推荐准确率的重要手段。可以采用协同过滤、内容过滤、混合过滤等多种算法进行优化。收集更加全面和准确的数据。数据是推荐系统的重要基础，收集更加全面和准确的数据能够提高推荐系统的准确率。可以通过多种方式收集数据，如用户行为数据、社交网络数据等。优化用户体验。用户体验是影响用户满意度的重要因素之一，因此，优化用户体验能够提高用户的满意度和推荐准确率。可以通过优化界面设计、提高响应速度等方式来优化用户体验。

简言之，准确率是交互式电影推荐系统的一个重要指标，它能够直接影响到用户的满意度和使用体验。因此，交互式电影推荐系统应该不断提高准确率，以提高推荐效果和用户满意度。

4.2 精确率和召回率

随着互联网技术的发展，人们对于个性化服务的需求越来越高，推荐系统作为一种个性化服务的实现方式，已经被广泛应用于各行各业。在电影推荐领域，交互式电影推荐系统已经成为了主流的推荐方式。精确率、召回率和调和均值是评价推荐系统性能的常用指标，本文将从这三个方面对交互式电影推荐系统的影响和作用进行介绍。

精确率是指推荐系统推荐的物品中，用户感兴趣的物品所占的比例。在电影推荐中，精确率可以理解为推荐列表中用户最终观看的电影数量与推荐列表总数的比例。提高精确率可以增加用户对推荐系统的信任度，从而提高用户的满意度。在交互式电影推荐系统中，精确率的提高可以通过以下几种方式实现：增加用户

反馈信息、考虑用户的历史行为及提高推荐多样性，不仅考虑用户的兴趣爱好，考虑电影之间的相似性和差异性，还可以根据电影的类型、导演、演员等不同因素进行推荐。本文将数据集分为 80%训练集、20%测试集和 20%测试集、80%训练集两种情况进行对比实验，可以发现即使 NMF 算法、SlopeOne 算法、CoClustering 算法有良好的精确率，但是本文的 XAI 算法在 movielens-100k 数据集中的精确率要优于对比实验的所有算法，如图 4.3 和图 4.4 所示，从图 4.5 中可以直观看出数据中位数及分布情况。因为精确率指的是算法预测结果为正例中实际结果为正例的比例，也就是算法预测正确的样本数占预测结果中所有正例的比例，即意味 XAI 算法相较于对比实验的推荐算法的误判率更低，预测的结果更可靠。由此可以看出本文的 XAI 算法不仅拥有较高的透明性和可解释性，在精确率方面也优于其他推荐算法。

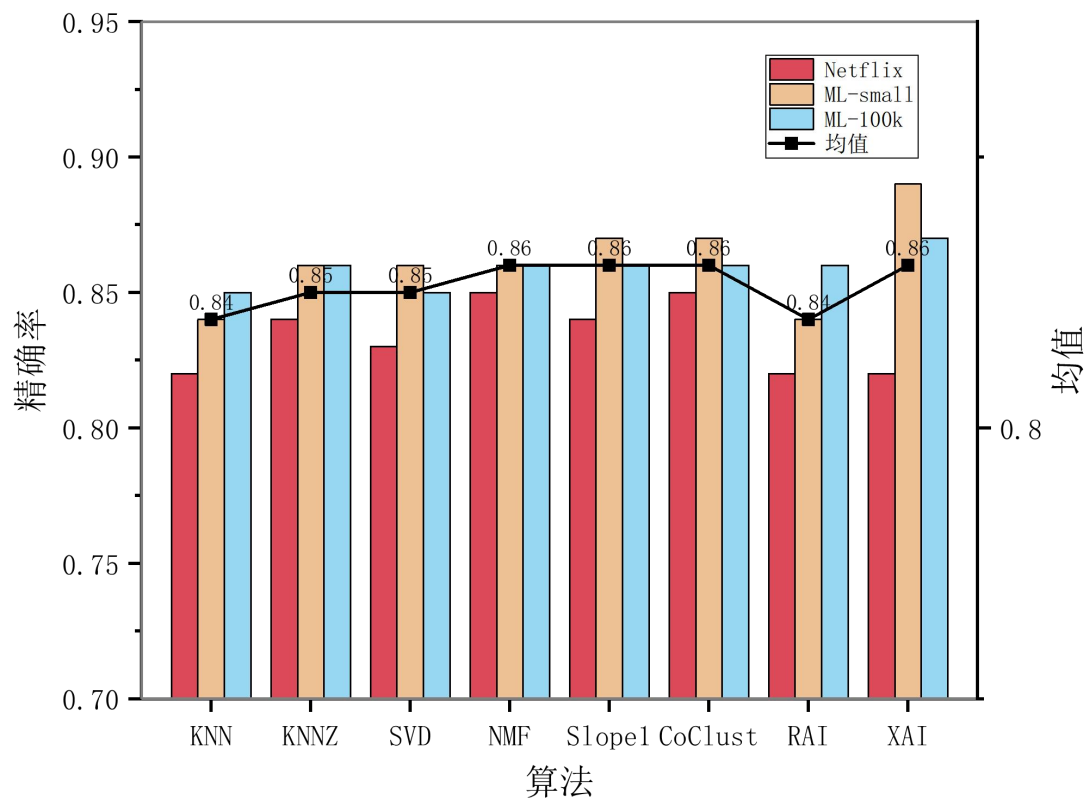


图 4.3 80%训练集 20%测试集精确率柱状折线图

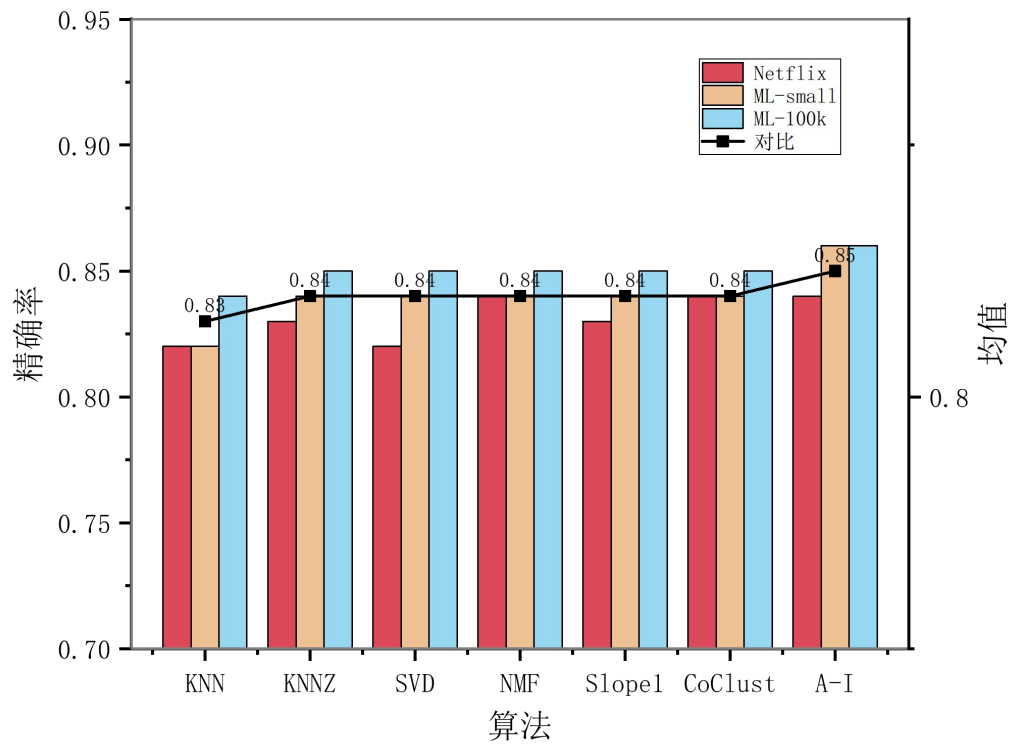


图 4.4 20%训练集 80%测试集精确率柱状折线图

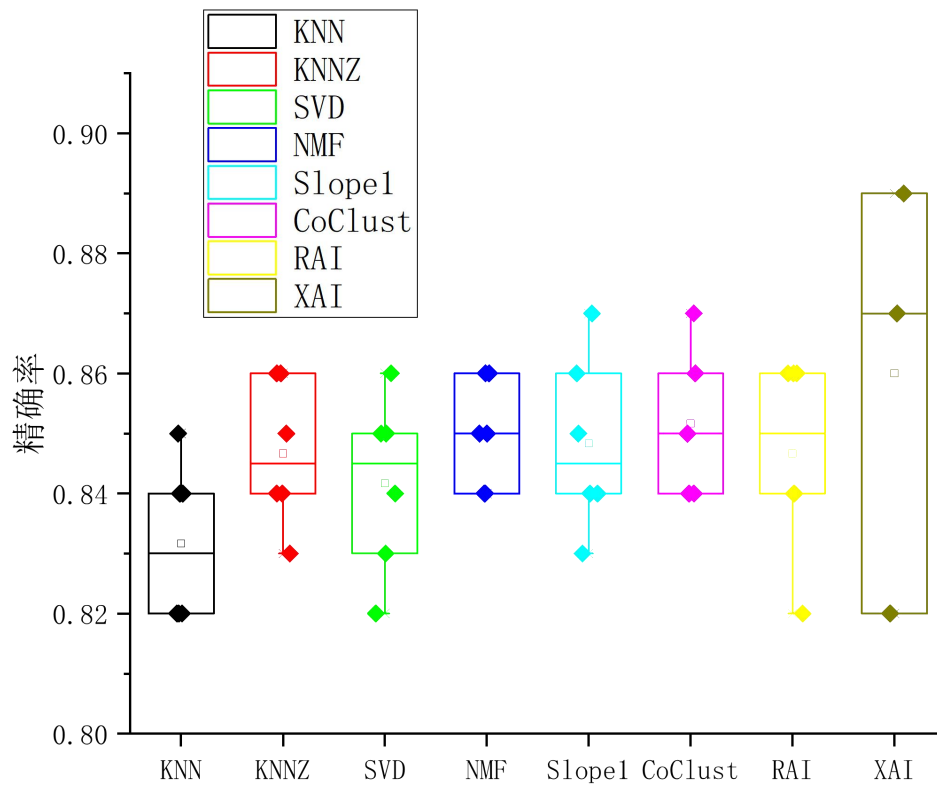


图 4.5 精确率箱线图

召回率是指推荐系统能够找到用户感兴趣的物品所占的比例。在电影推荐中，召回率可以理解为用户感兴趣的电影数量与用户实际感兴趣的

的电影数量的比例。提高召回率可以增加用户发现新电影的机会，从而提高用户的满意度。在交互式电影推荐系统中，召回率的提高可以通过以下几种方式实现：

考虑用户的历史行为：用户的历史行为可以反映出用户的兴趣爱好和偏好，因此在推荐系统中，可以考虑用户的历史行为，例如用户最近观看的电影、用户收藏的电影等，从而提高推荐的召回率。

推荐多样性：推荐多样性是指推荐系统在推荐电影时，不仅考虑用户的兴趣爱好，还考虑电影之间的相似性和差异性，从而提高推荐的多样性和召回率。例如，推荐系统可以根据电影的类型、导演、演员等因素进行推荐，从而提高推荐的多样性和召回率。

增加推荐列表长度：增加推荐列表长度可以增加推荐系统找到用户感兴趣电影的机会，从而提高召回率。但是需要注意的是，推荐列表长度过长也会影响用户体验。

各算法召回率柱状折线对比图和箱线图如图 4.6、图 4.7、图 4.8 所示。因为召回率指的是算法预测结果为正例中实际结果为正例的比例，也就是算法成功找到的正例数量占有所有正例数量的比例。由对比图可知，虽然 Aspect-Item 算法的召回率要逊色与其他推荐算法，但优化的 XAI 算法在 Movielens 数据集上不逊色其他推荐算法，甚至在 Movielens-latest-small 数据集上优于对比的所有推荐算法，意味着 XAI 算法找到的有用信息更多，能够覆盖更多的正例。

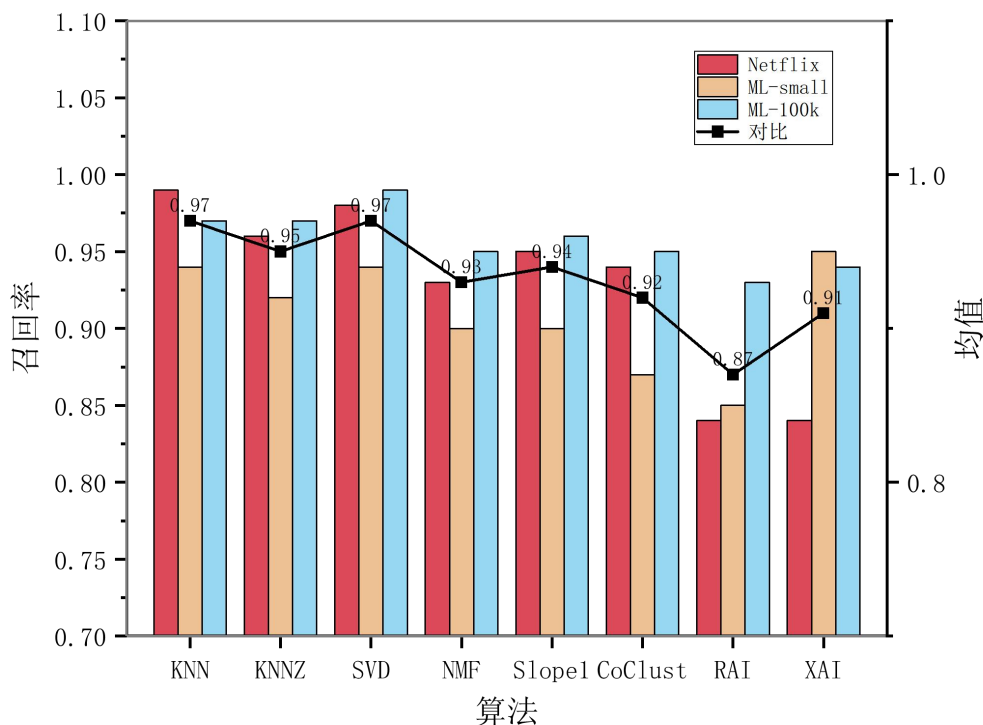


图 4.6 80%训练集 20%测试集召回率柱状折线图

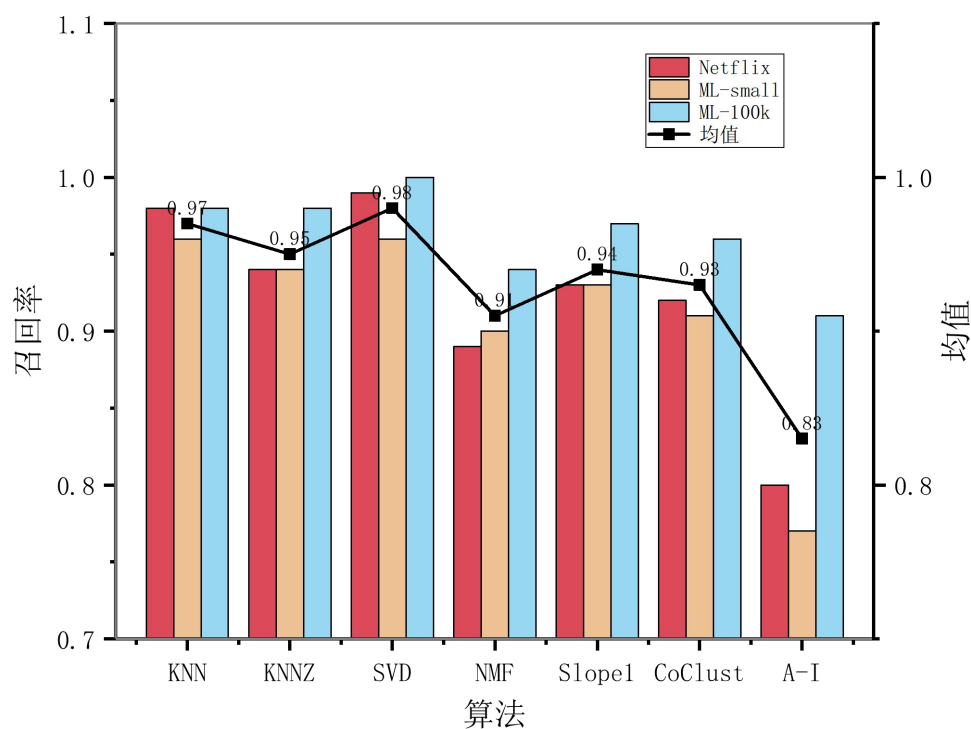


图 4.7 20%训练集 80%测试集召回率柱状折线图

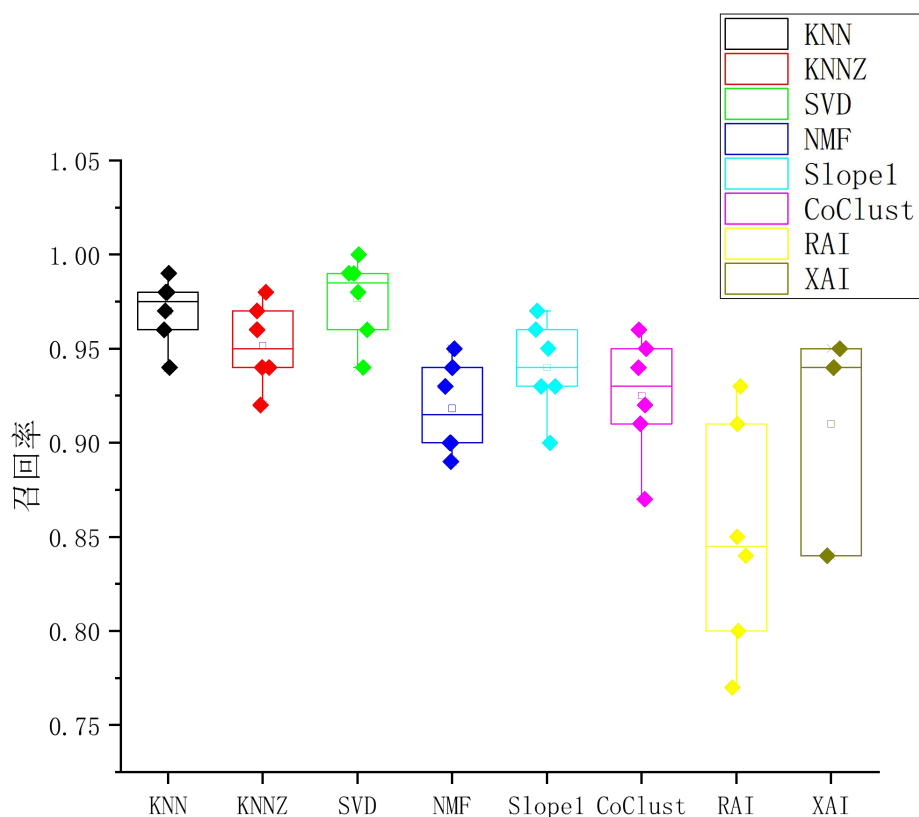


图 4.8 召回率箱线图

调和均值是精确率和召回率的综合指标，可以用来评价推荐系统的整体性能。调和均值越高，说明精确率和召回率都比较高，推荐系统的性能也就越好。在交

交互式电影推荐系统中，调和均值的提高可以通过以下几种方式实现：综合考虑精确率和召回率两者的影响。例如，可以根据用户的历史行为和反馈信息，综合考虑精确率和召回率，从而提高调和均值；优化推荐算法，提高推荐系统的性能。例如，可以采用内容推荐算法、协同过滤推荐算法等，从而提高推荐的准确率和召回率；优化用户交互体验，提高用户对推荐系统的满意度，从而提高调和均值。各算法调和均值柱状折线对比图和箱线图如图 4.9、图 4.10、图 4.11 所示。因为综合考虑精确率和召回率可以得到更全面、准确的模型性能评估结果，精确率和召回率的调和均值可以有效权衡预测结果正确和发现有用信息，从而提高模型的实际应用效果。由对比图可知，虽然 Aspect-Item 算法的调和均值要逊色与其他推荐算法，但优化的 XAI 算法在 Movielens 数据集上不逊色其他推荐算法，甚至在 Movielens-latest-small 数据集上显著优于对比的所有推荐算法，意味着 XAI 算法具有更好的系统性能，可以更有效融合预测结果正确和发现有用信息。

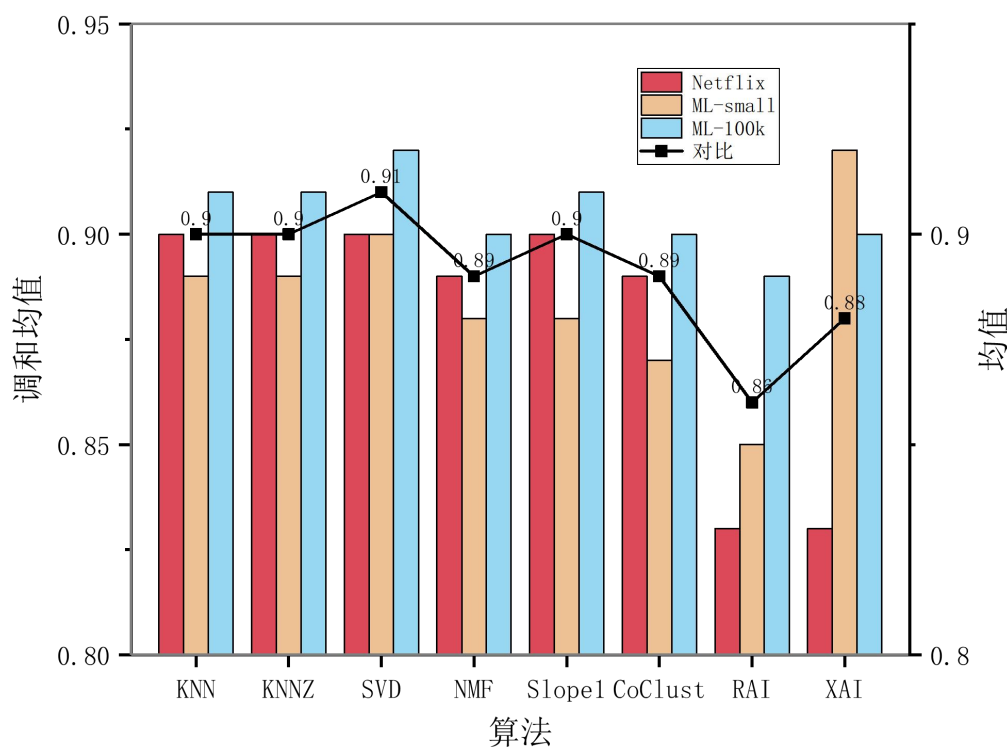


图 4.9 80%训练集 20%测试集调和均值柱状折线图

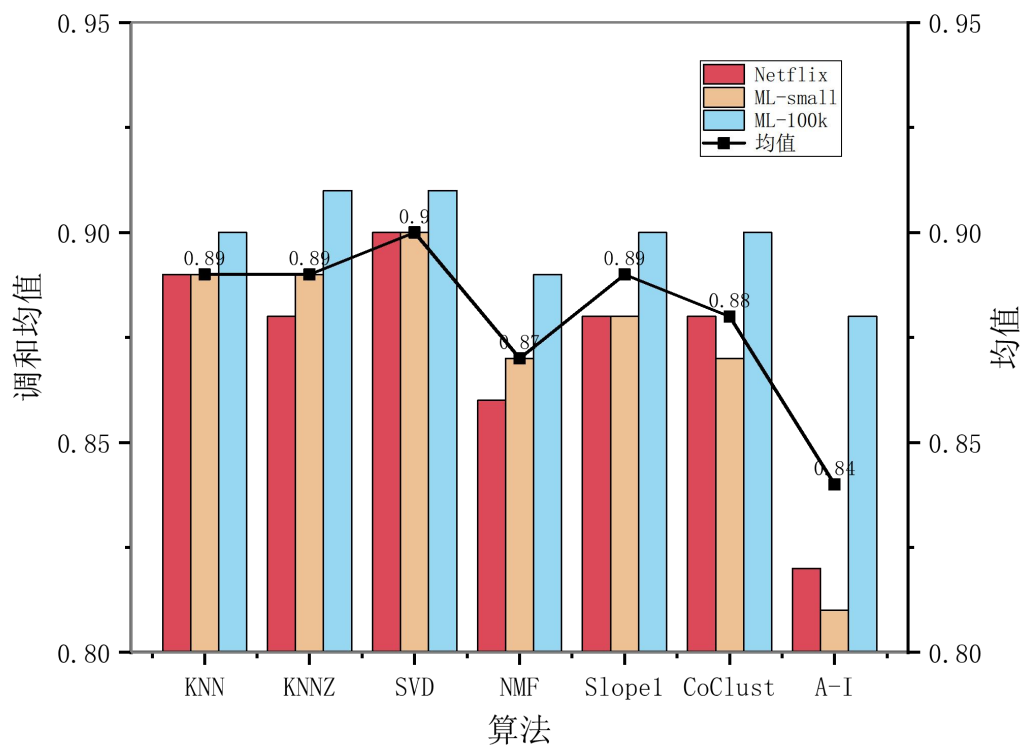


图 4.10 20%训练集 80%测试集调和均值柱状折线图

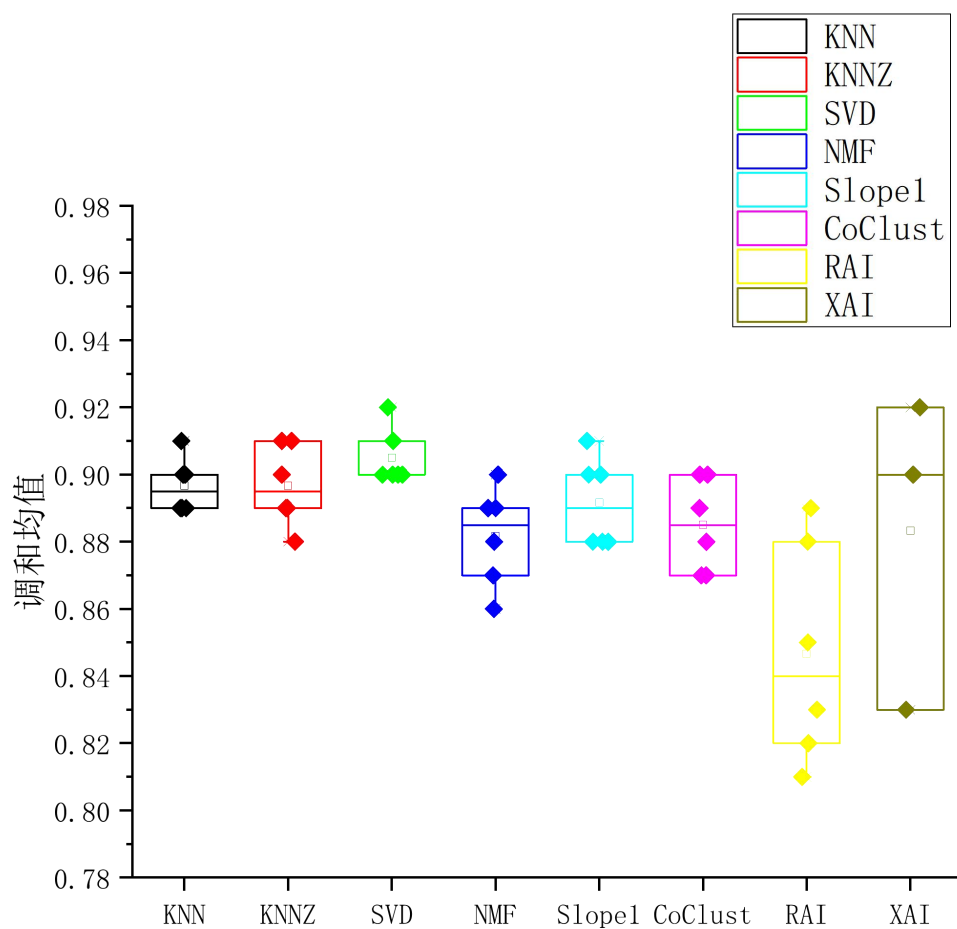


图 4.11 调和均值箱线图

4.3 均方根误差和平均绝对误差

交互式电影推荐系统是一种基于用户兴趣和行为的个性化推荐系统，以帮助用户在海量电影中找到自己感兴趣的电影。在电影推荐系统中，评估推荐算法的性能是至关重要的。均方根误差(RMSE)和平均绝对误差(MAE)是评估推荐算法的常用指标。使用均方根误差和平均绝对误差这两个指标对同一个模型得出的结果进行比较，可以很好地评估模型的性能和稳定性。

均方根误差是衡量预测值与真实值之间差异的一种方法。在电影推荐系统中，均方根误差是指预测评分与真实评分之间的平方误差取平均值再开根号，相较于平均绝对误差更为敏感。均方根误差越小说明推荐系统预测的精度越高。各算法均方根误差柱状折线对比图和箱线图如图 4.12、图 4.13、图 4.14 所示。

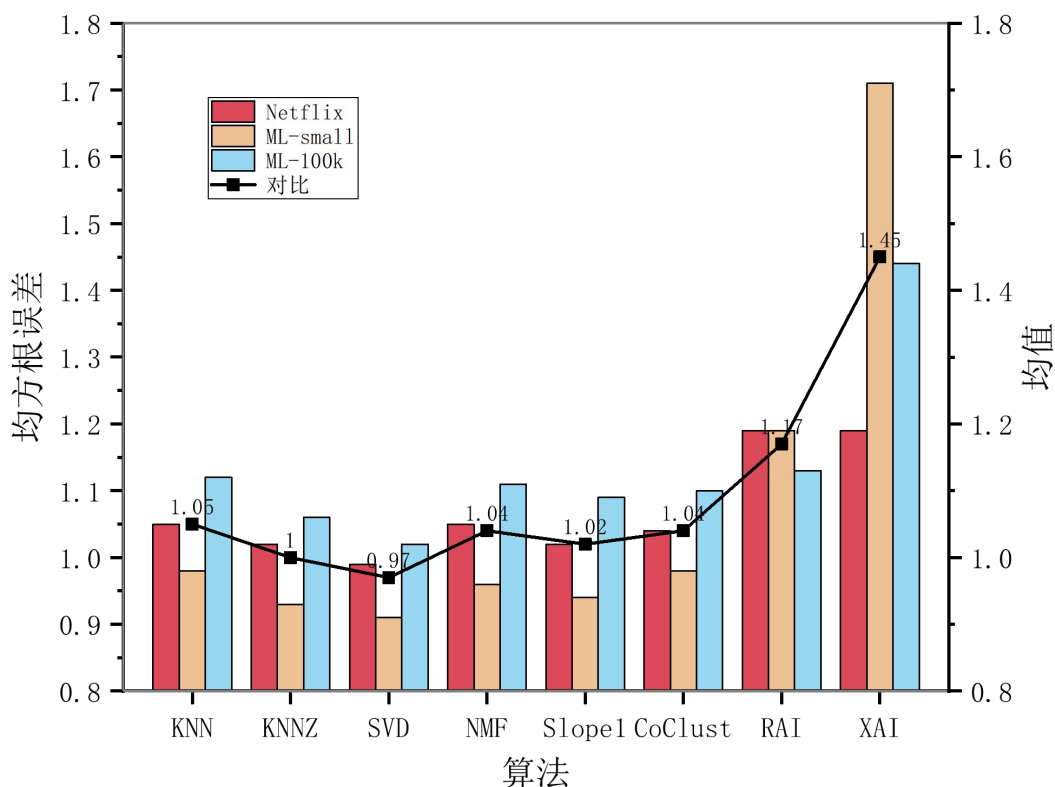


图 4.12 80%训练集 20%测试集均方根误差柱状折线图

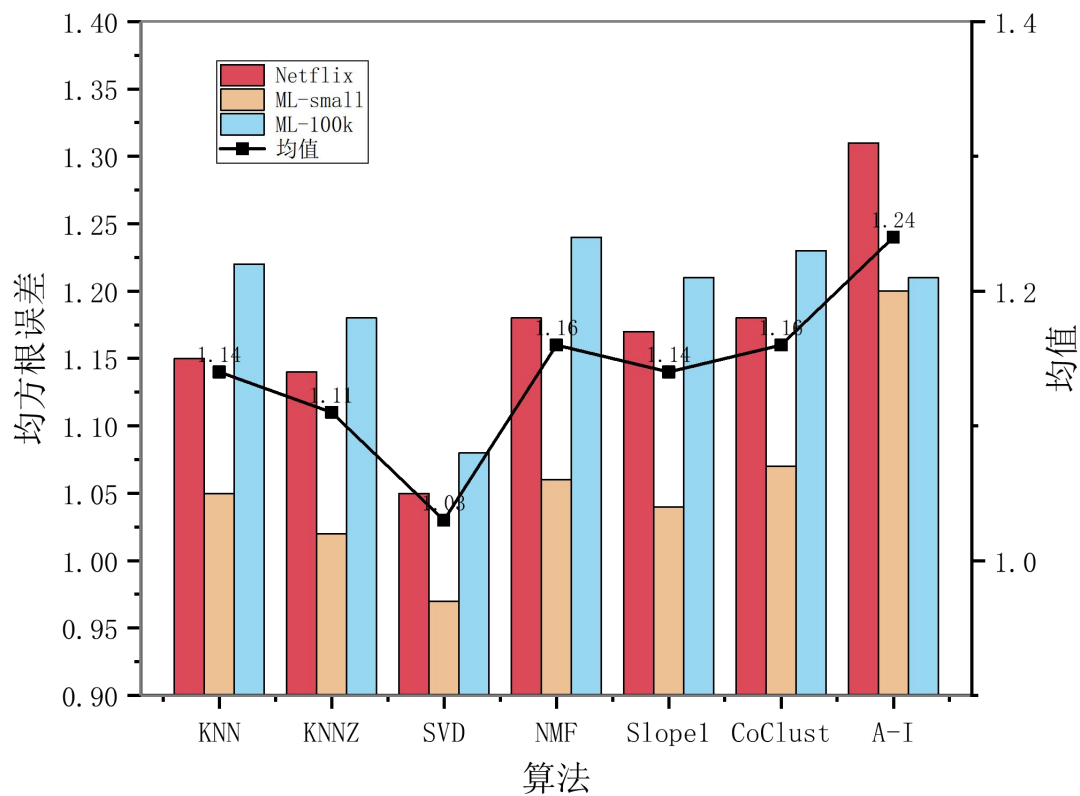


图 4.13 20%训练集 80%测试集均方根误差柱状折线图

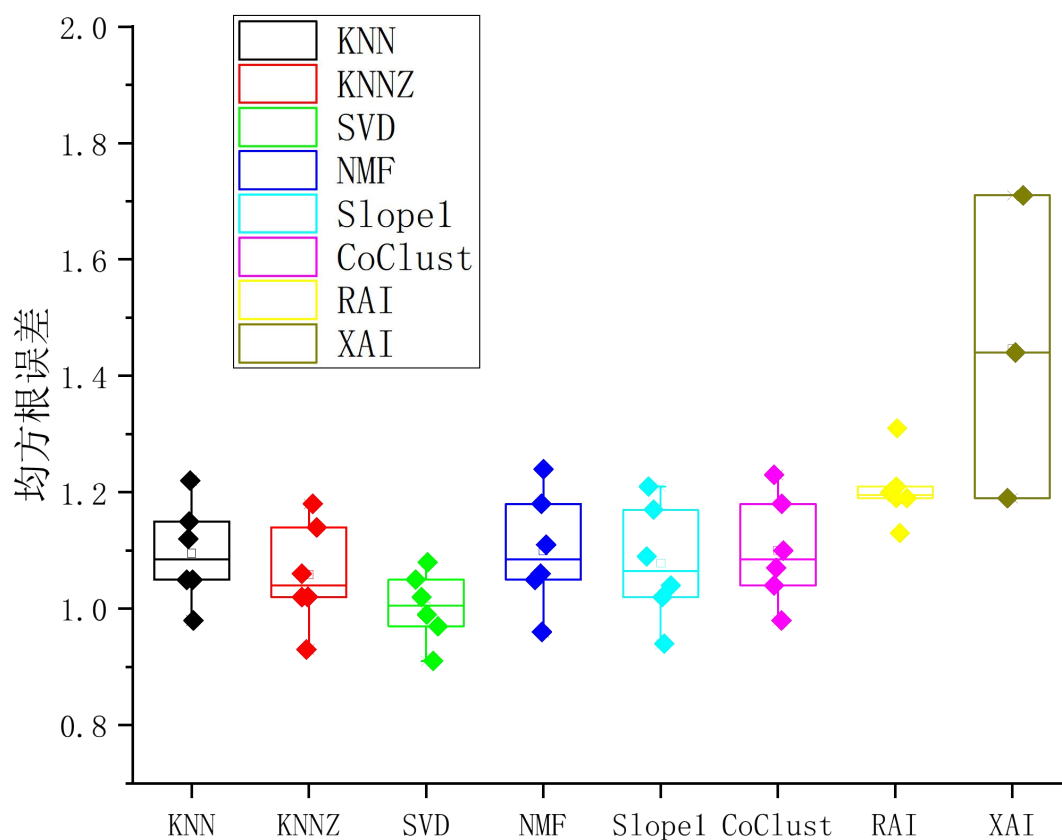


图 4.14 均方根误差箱线图

平均绝对误差(MAE)是另一种衡量预测值与真实值之间差异的方法，指预测评分与真实评分之间的绝对误差取平均值，平均绝对误差的值越小，代表预测结果与真实值之间差距越小，即预测的准确度越高。各算法平均绝对误差柱状折线对比图和箱线图如图 4.15、图 4.16、图 4.17 所示。

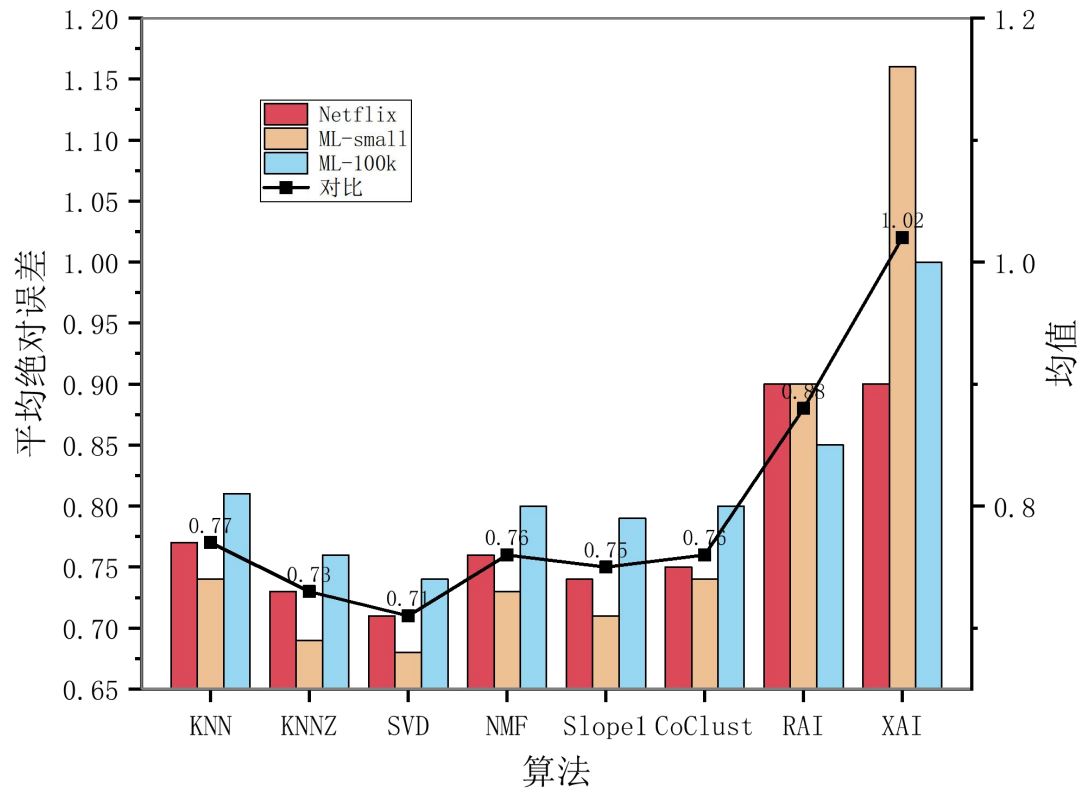


图 4.15 80%训练集 20%测试集平均绝对误差柱状折线图

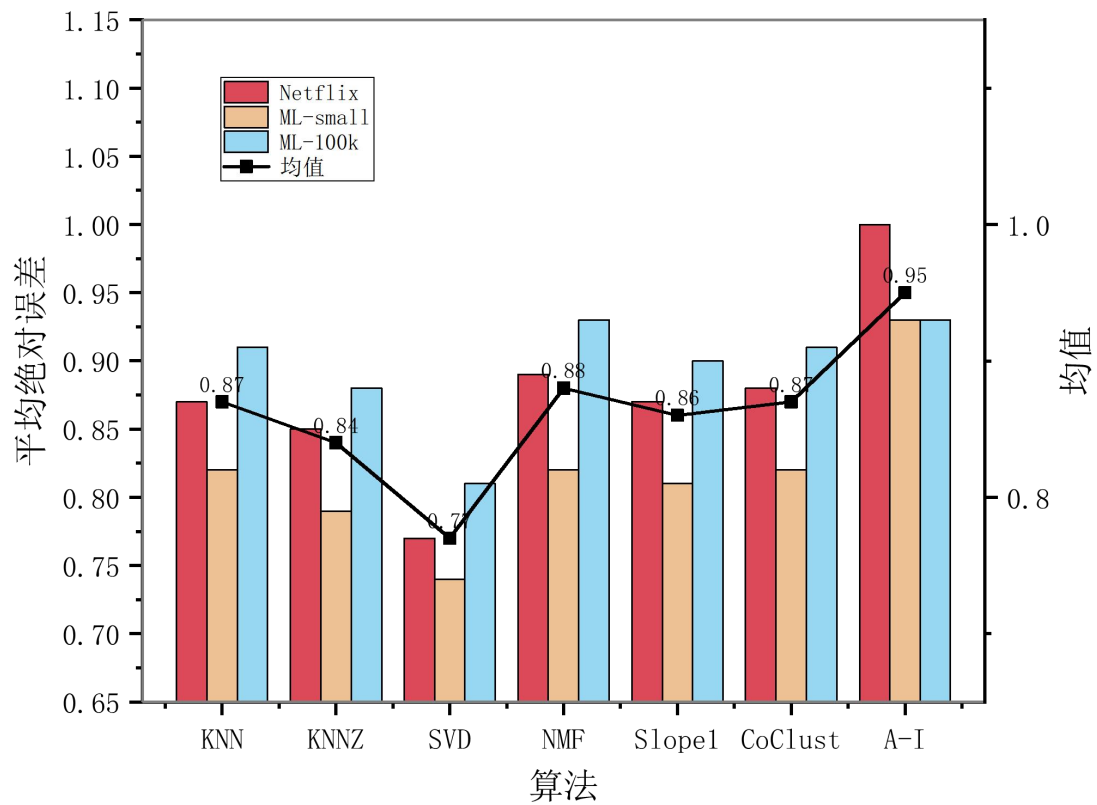


图 4.16 20%训练集 80%测试集平均绝对误差柱状折线图

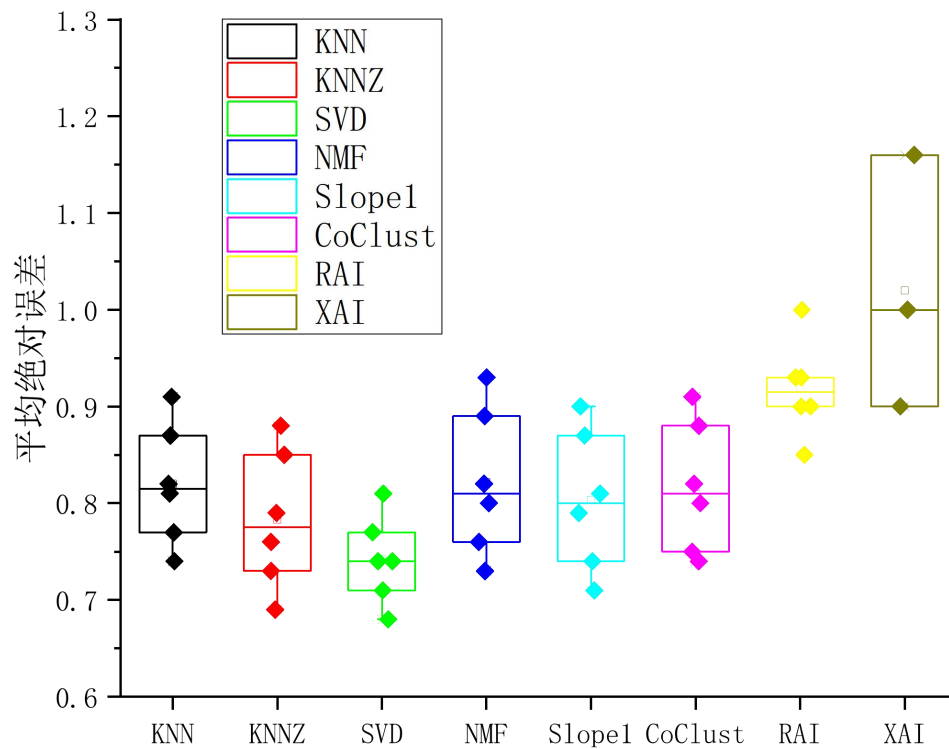


图 4.16 平均绝对误差箱线图

均方根误差和平均绝对误差虽然都是衡量模型预测值与真实值之间误差的

评价指标,但两者在计算方法和使用范围上有所不同。均方根误差对异常值比较敏感,而平均绝对误差则具有一定的鲁棒性;均方根误差更加适用于变量分布较均匀的情况,而平均绝对误差则更加适用于存在离群点或偏态分布的样本数据。因此,在选择使用均方根误差或平均绝对误差指标时应该基于具体的场景和需求。如果模型需要对所有样本都进行准确预测,可以使用均方根误差指标;如果样本数据中存在离群点或偏态分布情况,可以选择使用平均绝对误差指标来评估模型的性能。均方根误差和平均绝对误差是评估推荐算法性能的常用指标,可以直观地反映推荐算法的准确率,从而帮助开发者快速了解推荐算法的性能表现。这两个指标的计算方法简单易懂,具有一定的客观性和可比性,因此被广泛应用于推荐系统的性能评估中。然而需要注意的是,均方根误差和平均绝对误差只能反映推荐算法的整体准确率,不能反映推荐算法的个性化程度。此外,在其他方面的推荐算法性能评估中,如推荐结果的多样性和新颖性等,均方根误差和平均绝对误差则不能发挥作用。此外,由于均方根误差和平均绝对误差对异常值比较敏感可能会对推荐算法的评估产生影响,因此在使用时需要开发者注意。均方根误差和平均绝对误差作为推荐算法性能评估的重要指标,其具有一定的优点和缺点,需要根据具体情况进行选择和应用。

4.4 运行效率

推荐系统中的运行效率主要指的是系统对用户进行个性化推荐时所需的时间和计算资源。具体来说,它由以下几个方面组成:数据处理效率指系统对原始数据进行处理和分析的速度和效果,包括数据清洗、去重、归一化、特征提取等;模型训练效率指系统在对用户行为数据进行建模和训练时所需的时间和计算资源,包括模型选择、参数调整、训练算法等;推荐效率指系统在实际推荐过程中所需的时间和计算资源,包括向量相似度计算、推荐列表生成、排序等;并发性能指系统在面对大量并发请求时所能承受的压力和处理能力,包括负载均衡、缓存机制、分布式部署等。通过优化这些方面,可以提高推荐系统的运行效率和性能,提高用户对系统的满意度和使用体验。根据运行时间得出运行效率变化,是通过对某个系统或过程的运行时间的测量,来推断运行效率的变化情况。这种方法可以对系统或过程的运行效率进行评估和提高,以达到优化系统或过程的目的。

本文在 Rago 研究的基础上优化算法^[6],在相同机器测试下,对于 Netflix

数据集, Rago 设计的 Aspect-Item 算法原先运行时间为 2100 秒, 如图 4.17 所示, 在本文的实验中优化至 1912 秒, 缩减了 9%的运行时间, 如图 4.18 所示。XAI 算法相较于 RAI 算法, 在提高运行效率的同时, 并且将 XAI 算法推广到了 Movielens 数据集上, 在 Movielens-latest-small 数据集上运行时间为 46127 秒, 在 Movielens-100k 数据集上运行时间为 1885 秒, 如图 4.19 与图 4.20 所示。

```
netflix
NR USERS 4772
Begin prefs
End actors
End directors
End genres
Number of user-items pairs: 86180
Accuracy: 0.7701206776514272
RMSE: 1.316157159360712
MAE: 0.973578556509631
Precision: 0.824300600294484
Recall: 0.8415894537517526
F1: 0.8328553138835445
time: 2100.3702986240387
```

图 4.17 RAI 算法 Netflix 数据集

```
netflix
NR USERS 4772
Begin prefs
End actors
End directors
End genres
Number of user-items pairs: 86180
Accuracy: 0.770201902993734
RMSE: 1.3158441439217112
MAE: 0.9733812949640288
Precision: 0.8243162901307967
Recall: 0.8417484569462714
F1: 0.8329411764705882
time: 1912.2780219730126
```

图 4.18 XAI 算法 Netflix 数据集

```
small
NR USERS 610
Begin prefs
End actors
End directors
End genres
Number of user-items pairs: 16060
Accuracy: 0.7280199252801992
RMSE: 1.7084326784182695
MAE: 1.1572229140722292
Precision: 0.8941925218774861
Recall: 0.9470579974722652
F1: 0.9198663302189184
time: 46127.11631512642
```

图 4.19 XAI 算法 Movielens-small 数据集

```
100k
NR USERS 943
Begin prefs
End actors
End directors
End genres
Number of user-items pairs: 23210
Accuracy: 0.7734597156398104
RMSE: 1.4445070450462612
MAE: 1.0049116760017234
Precision: 0.8669173281344288
Recall: 0.939011513902157
F1: 0.9015253910021239
time: 1885.5215673446655
```

图 4.19 XAI 算法 Movielens-100k 数据集

5 总结与展望

5.1 总结

本文探究了基于用户偏好参数的交互式电影推荐系统的设计与优化，旨在为用户提供更加便利、精确和人性化的推荐服务。本文通过整合多种算法，设计并实现 XAI 算法，并对推荐系统进行了大量的实验测试和分析。本文主要从算法对比实验、提高推荐精度和减少系统运行时间三个方面来阐述本文的研究成果。

算法对比实验是提高推荐系统性能的重要手段之一。本文通过整合 Aspect-Item、KNN、Z-KNN、SVD、NMF、Slope One 及 CoClustering 等算法，对数据进行预处理，并设计并实现了 XAI 算法。通过相关算法的对比实验，根据准确率、精确率、召回率、调和均值、均方根误差、平均绝对误差和运行效率等指标考量了系统性能。实验结果表明，相较于 Rago 的 Aspect-Item 算法，XAI 算法不仅具有更高的适用性，而且运行时间还减少了 9%。这也为本文在算法设计和数据预处理方面获取了新的思路和经验，为后续工作提供了重要的理论参考和实践借鉴。

同时提高推荐精确率和召回率是优化推荐系统的主要目标之一。本文引入了论解释的概念，通过给予解释，向用户明确了推荐决策的依据和理由，使得用户可以更好地理解推荐结果背后的原因。这不仅有助于提高用户的信任度，还能够提高推荐系统的精确率和召回率，因为用户在理解推荐决策的过程中能够提供更加准确的反馈信息。同时，本文采用多种算法的组合，包括 Aspect-Item、KNN、Z-KNN、SVD、NMF、Slope One 及 CoClustering 算法，以此提高推荐决策的透明度与精确率和召回率的调和均值，使得用户能够得到更加满意的推荐结果。

减少系统运行时间是优化推荐系统的另一个重要方面。本文使用多种优化方法，包括利用数据预处理技术、使用高效的算法实现以及优化算法参数等手段，以此达到提高系统性能和用户体验的目的。在实验中，相较于 Rago 的 Aspect-Item 算法，本文设计的 XAI 算法不仅运行时间减少 9%，而且在 Movielens 数据集上表现更加出色。通过减少运行时间，本文不仅提高了推荐系统的效率，还可以为用户提供更为快捷和便利的服务，增强用户体验。

综合实验结果表明，本文设计的交互式电影推荐算法不仅具有较高的系统性

能和精确率，而且通过给予解释呈现了清晰的推荐机制，从而获得了友好的用户体验。这也为后续基于用户偏好参数的推荐系统的研究和发展提供了重要的理论和技术支持，为优化推荐系统的实现提供了新的思路和方法。

5.2 展望

交互式可解释的混合推荐系统是一种结合了多个推荐算法的系统，具有较高的推荐效果和用户满意度。在这个系统中，用户可以与算法进行交互，并获得算法的解释，从而更好地理解并接受对他们的推荐结果。交互式推荐系统是结合用户的历史行为和即时反馈等，进行个性化的推荐，并让用户与系统进行交互来提高推荐效果和满意度。可解释性推荐系统则是为了提高推荐结果的透明度和可接受度，采用决策树和规则等方法，以帮助用户理解推荐依据和策略。混合推荐系统则是将多个推荐算法结合起来，以提高推荐效果和精确度。

随着移动互联网和社交媒体的兴起，推荐系统显得越来越重要。传统的推荐系统通常采用协同过滤、基于内容的推荐和基于知识的推荐等算法。这些算法大多是无法解释的，难以让用户理解算法是如何给出其推荐结果的。此外，这些算法缺乏对用户的反馈和适应性、灵活性不够，不能及时调整推荐策略，导致用户的体验和满意度较低。因此，交互式推荐系统作为新型推荐方法正在逐渐成为研究热点。交互式推荐系统是指通过与用户的交互行为来改善推荐结果的系统。它可以结合用户的历史行为和个人兴趣以及即时反馈等，从而进行个性化的推荐。而且，因为用户可以与系统进行交互，他们可以自己调整推荐策略、更好地理解算法是如何给出其推荐结果的，从而提高用户的满意程度。

传统的推荐系统往往缺乏对其结果的解释，导致许多用户难以接受其给出的推荐结果。特别是在一些涉及到安全和隐私问题的场景下，用户对推荐结果的透明度要求越来越高，这就需要推荐系统具有可解释性。具有可解释性的推荐系统是指通过给用户提供解释来增加推荐系统的透明度和可接受度。此类系统将复杂的推荐算法结果转化为人类可读的形式，以帮助用户理解和接受它们。可解释性推荐系统通常采用类似于决策树和规则的方法，以描述系统根据何种规则或标准来做出推荐决策。解释可以采取文本、图片或图形的形式展现，从而使用户更清楚地了解系统的推荐依据和推荐策略，从而提高用户对推荐结果的理解。随着用户对透明度和可解释性的要求越来越高，可解释性推荐系统将会成为未来推荐

系统的必要组成部分。大量研究表明，提供相关的解释信息可以明显提高用户对推荐结果的满意程度和信任度。

混合推荐系统是指结合多种推荐算法来进行推荐的系统。混合推荐系统是针对单一推荐算法的缺点而设计的，能够提高推荐效果和精确度。例如，基于协同过滤的推荐系统在处理稀疏数据时效果较差，而基于内容的推荐系统则可以填补这一空白。混合推荐系统可以充分利用不同算法之间的交叉优势，从而为用户提供更加准确的推荐结果。近年来，随着机器学习和深度学习技术的发展，混合推荐系统也变得更加复杂和智能化。例如，基于深度学习的混合推荐系统可以在同时考虑用户行为和物品特征的情况下，进行准确的推荐。此外，在混合推荐系统中，交互式 and 可解释性技术也得到了不断的应用，更好地满足了用户的需求。

在未来的研究中，可以继续探索推荐系统的可个性化、可解释化、智能化和私密化等方面问题。例如，结合关系图谱和知识图谱等技术，将外部信息融入到推荐计算中，从而更好地刻画用户的兴趣和行为。同时，借助深度学习和强化学习等技术，深化推荐算法的应用，提高推荐结果的质量和效率。此外，在保护用户隐私和个人信息的前提下，加强推荐系统私密性保护，为用户提供更加安全和可靠的推荐服务。交互式可解释的混合推荐系统是未来推荐系统研究的重要方向。在此基础上，可以继续探索推荐系统的可个性化、可解释化和智能化等方面问题，从而让推荐系统更好地服务于人们的生产和生活。交互式、可解释性和混合推荐系统是推荐系统研究中的重要方向，它们促进了推荐系统的发展，并取得了显著的成果。未来的研究将会在这些方面不断深入，从而让推荐系统更好地服务于人类的生产和生活。

结束语

作为在疫情期间上了三年网课的大学生，毕业设计对于我来说是一个非常宝贵和重要的机会。它让我不再只停留于理论知识的运用，而是能够将理论与实践相结合，进而开发具体实际项目的能力也随之得到了提升。

经过数个月的文献阅读和学习补足，我顺利研究并实现了交互式电影推荐系统。虽然这个过程中我遇到了很多困难，包括对该领域不熟悉和自身能力不足等问题，但通过导师的指导和同学的帮助，我成功克服了所有难关。我把所学的基础知识和设计方法都应用到整个过程中，从而充分展现了我的综合应用能力。

这次毕业设计的经历让我深刻地感受到科研的艰辛，虽然只是刚接触科研工作，但思考创新点和调试代码等过程已经让我意识到科研并不是一帆风顺的。除了扎实的基础知识，还需要足够的毅力、定力、拼劲和韧劲，在思考和实现创新点的过程中坚持不懈地追求最终胜利。对于初出茅庐的人来说，编程时总会遇到各种各样的错误，但只有那些具备足够毅力和拼劲的人才能在这条路上一路坚持到底。

大学四年带给我的不仅仅是学术成就，更是生活的沉淀与人生的感悟。这些将成为我以后路上前行的精神支柱，让我在每一个决策和选择面前都能够胸怀信心和勇气。在毕业之后，我将会勇往直前，踔厉奋发，迎接新的挑战 and 机遇。

致 谢

花开花落万物道，聚散离别终有时。总以为来日方长，却不知时光匆匆，青葱四年，落笔为终，已经写到论文的最后一章，意味着我的大学本科生涯也即将结束。始于 2019 年初秋，终于 2023 年盛夏，是结束，亦是开始。所有的相遇，于我皆是礼物，所有的经历，于我皆是宝藏，在此感恩每一次相遇与经历。

涓涓师恩，铭记于心。首先要衷心感谢我的导师谭立兴老师，带我进入计算机协会核心组这个大家庭，也开启了我的竞赛和科研生涯。于学习方面，从考核进组到竞赛培训，从论文选题到撰写修改，谭老师都给予了我悉心的指导与意见。谭老师严谨的科研态度与开阔的学术视野深深地影响着我、引领着我；于思想生活，谭老师的思想与话语总能给予我勇气与力量，关心帮助着我。第一次参加竞赛时懵懂无知的我侥幸取得全省二等奖里的最高分，距离国奖一步之遥，幻想着下一次便可圆梦。在有幸被选为核心组领队时，我便感受到另一份别样的责任感，从那一刻起，我的目标不止有国奖，需要将更多心血放在学弟学妹们的成长。我想要将核心组发展得更好，回报谭老师对我的信任。在竞赛失利时第一个想到的便是对不起谭老师对我的期望，辜负了所有对我寄予厚望的朋友与老师，一度陷入自我怀疑是否是个合格的领队。感谢谭老师对我的开导，也感谢核心组的每一位伙伴的劝慰。毕竟遗憾组成了生活，惋惜编织了人生，人生总有种种失利，只有在失败中涅槃重生，才值得享受胜利的凯歌，哪怕雨雪霏霏也要去追寻阳光。

春晖寸草，无以回报。感谢我的父母二十余年的悉心养育，感谢他们在我求学路上的无私付出与无条件支持。他们给予了我无尽的爱与包容，守护了我的梦想与追求，正是因为有他们作为我的避风港，我才有机会、有勇气去大胆地眺望这个更精彩的世界，去开拓自己的眼界与格局，愿父母万事如意，平安健康。

弓背霞明剑照霜，秋风走马出校园。最后，我要感谢自己，虽然时常遇到挫折却一直坚持向前，感谢自己一直保持对生活的热爱，对美好的期待。

执笔至此，感慨万千，青山隐隐水迢迢，秋尽江南草未凋。山水相逢，终有一别。言辞有尽，谢意难全。愿不忘初心，追风赶月莫停留，平芜尽处是春山。

参 考 文 献

- [1] Zheng L, Noroozi V, Yu P S. Joint deep modeling of users and items using reviews for recommendation[C]. New York: ACM Press, 2017: 425-434.
- [2] Seo S, Huang J, Yang H, et al. Interpretable convolutional neural networks with dual local and global attention for review rating prediction[C]. New York: ACM Press, 2017: 297-305.
- [3] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead[J]. Nature Machine Intelligence, 2019, 1(5).
- [4] Cataldo Musto, Fedelucio Narducci, Pasquale Lops, Marco de Gemmis, Giovanni Semeraro. Linked open data-based explanations for transparent recommender systems[J]. International Journal of Human - Computer Studies, 2018, 121.
- [5] 王攸妍, 孙康高, 汤颖. 基于局部模型融合的交互式电影推荐系统[J]. 数据与计算发展前沿, 2021, 3(04): 54-69.
- [6] 饶超. 面向交互式推荐的深度强化学习推荐算法研究[D]. 武汉: 华中科技大学, 2021.
- [7] Yongfeng Zhang, Xu Chen. Explainable Recommendation: A Survey and New Perspectives[J]. Foundations and Trends® in Information Retrieval, 2020, 14(1).
- [8] Thomas M. Cover, Peter E. Hart. Nearest neighbor pattern classification. [J]. IEEE Trans. Information Theory, 1967, 13(1).
- [9] Lee D D, Seung H S. Learning the parts of objects by non-negative matrix factorization. [J]. Nature, 1999, 401(6755).
- [10] Xin Luo, Mengchu Zhou, Yunni Xia, Qingsheng Zhu. An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems. [J]. IEEE Trans. Industrial Informatics, 2014, 10(2).
- [11] Daniel Lemire, Anna Maclachlan. Slope One Predictors for Online Rating-Based Collaborative Filtering[J]. CoRR, 2007, abs/cs/0702144.
- [12] Dhillon I S. Co-clustering documents and words using bipartite

- spectral graph partitioning[C]. New York: ACM Press, 2001: 269-274.
- [13] Baroni Pietro, Toni Francesca, Verheij Bart. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games: 25 years later[J]. *Argument & Computation*, 2020, 11(1-2).
- [14] Mengnan Du, Ninghao Liu, Xia Hu. Techniques for interpretable machine learning[J]. *Communications of the ACM*, 2019, 63(1).
- [15] Balog K, Radlinski F, Arakelyan S. Transparent, scrutable and explainable user models for personalized recommendation[C]. New York: ACM Press, 2019: 265-274.
- [16] Rago A, Cocarascu O, Bechlivanidis C, et al. Argumentative explanations for interactive recommendations[J]. *Artificial Intelligence*, 2021, 296: 103506.