山东财经大学

**本科学年论文**

**题目：****基于用户的协同过滤推荐算法实现**

**学 院** 管理科学与工程

**专 业** 大数据管理与应用

**班 级** 2001

**学 号** 202006140455

**姓 名** 李 典

**指导老师** 彭苏勉

山东财经大学教务处制

二Ｏ二三 年 六 月

基于用户的协同过滤推荐算法实现

**摘 要**

本研究旨在利用基于用户相似度的协同过滤推荐算法，对WikiLens数据集中的用户进行产品推荐。WikiLens是一个包含丰富用户行为数据的数据集，其中包括用户对维基百科页面的浏览、收藏、评分等行为。通过分析这些行为数据，我们将尝试构建一个准确可靠的推荐系统，为用户提供个性化的产品推荐。

本论文将从以下几个方面展开：首先，我们将介绍基于用户相似度的协同过滤推荐算法的基本原理和相关研究。接着，我们将详细描述WikiLens数据集的特点和结构，并介绍我们所采用的数据预处理方法。然后，我们将提出并实现基于用户相似度的协同过滤推荐算法，并对其进行实验评估和性能分析。最后，我们将总结研究结果，并对未来的研究方向进行展望。

通过本研究，我们希望能够深入理解基于用户相似度的协同过滤推荐算法在产品推荐中的应用效果，并为推荐系统的改进和优化提供有益的参考。相信这将对提升用户的产品发现体验和满意度，以及促进电子商务的发展具有重要的意义。

关键词：协同过滤；推荐；相似度；WikiLens

Implementation of user based Collaborative filtering recommendation algorithm

ABSTRACT

This research aims to use the Collaborative filtering recommendation algorithm based on user similarity to recommend products to users in the WikiLens dataset. WikiLens is a dataset that contains rich user behavior data, including user browsing, bookmarking, rating, and other behaviors on Wikipedia pages. By analyzing these behavioral data, we will attempt to build an accurate and reliable recommendation system to provide personalized product recommendations for users.

This paper will start from the following aspects: First, we will introduce the basic principles and related research of Collaborative filtering recommendation algorithm based on user similarity. Next, we will provide a detailed description of the characteristics and structure of the WikiLens dataset, and introduce the data preprocessing methods we adopt. Then, we will propose and implement a Collaborative filtering recommendation algorithm based on user similarity, and conduct experimental evaluation and performance analysis on it. Finally, we will summarize the research results and provide prospects for future research directions.

Through this research, we hope to deeply understand the application effect of Collaborative filtering recommendation algorithm based on user similarity in product recommendation, and provide useful reference for the improvement and optimization of recommendation system. I believe this will have significant implications for improving users' product discovery experience and satisfaction, as well as promoting the development of e-commerce.

**Keywords**：Collaborative filtering；recommendation；similarity；WikiLens

目录

[一、 研究背景 1](#_Toc139604825)

[二、 研究意义 1](#_Toc139604826)

[三、 研究现状 1](#_Toc139604827)

[四、 算法原理 1](#_Toc139604828)

[(一) 协同过滤算法 1](#_Toc139604829)

[1. 协同过滤算法介绍 2](#_Toc139604830)

[2. 以用户为基础（User-based）的协同过滤 2](#_Toc139604831)

[3. 皮尔逊相关系数 3](#_Toc139604832)

[4. 预测用户对产品的评分 3](#_Toc139604833)

[5. 优缺点 3](#_Toc139604834)

[(二) 所用评估方法 4](#_Toc139604835)

[五、 WikiLens数据集的特点和结构 4](#_Toc139604836)

[(一) 大规模性 4](#_Toc139604837)

[(二) 多样性 4](#_Toc139604838)

[(三) 语言多样性 4](#_Toc139604839)

[(四) 数据预处理 4](#_Toc139604840)

[(五) 开放性 5](#_Toc139604841)

[六、 算法实现 5](#_Toc139604842)

[(一) 数据预处理 5](#_Toc139604843)

[(二) 用皮尔逊相关系数计算测试集中的用户相似度 5](#_Toc139604844)

[(三) 构建推荐方法 5](#_Toc139604845)

[(四) 推荐结果评估 5](#_Toc139604846)

[七、 结论 5](#_Toc139604847)

[附录 7](#_Toc139604848)

[(一) 原始数据 7](#_Toc139604849)

[(二) 源程序 7](#_Toc139604850)

# 研究背景

随着互联网的快速发展，用户在线上的活动越来越多，其中之一就是通过社交网络和电子商务平台来分享和推荐产品。然而，传统的基于内容的推荐算法已经难以满足用户的需求，因为用户的兴趣和行为是多样化和不断变化的。因此，基于用户相似度的协同过滤推荐算法应运而生。它可以帮助用户过滤和筛选出个性化的推荐结果，提供更加精准和有针对性的推荐，从而减少用户的选择困难和信息过载。

# 研究意义

基于用户相似度的协同过滤算法可以应用于各种领域，如电子商务、社交媒体、音乐和电影推荐等。通过深入研究和改进该算法，可以提高推荐系统的准确性和效果，增强用户对推荐结果的满意度。此外，该算法还可以为企业提供更好的商业机会，通过个性化推荐来增加销售和用户忠诚度。

# 研究现状

基于用户相似度的协同过滤推荐算法已经在社交网络和电子商务平台中得到广泛应用，并取得了一定的效果。具体来说，这种算法主要是根据用户之间的相似度，来推荐用户可能感兴趣的产品或服务。

在社交网络中，基于用户相似度的协同过滤推荐算法已经被广泛应用，如Facebook的Recommended Posts算法，Twitter的Trending Topics算法等。这些算法可以根据用户之间的相似度，来推荐用户可能感兴趣的内容。

在电子商务平台中，基于用户相似度的协同过滤推荐算法也已经被广泛应用，如Amazon的Recommendation Engine算法，eBay的Shopping Recommendations算法等。这些算法可以根据用户之间的相似度，来推荐用户可能感兴趣的产品。

总的来说，基于用户相似度的协同过滤推荐算法已经取得了一定的应用效果，但是也存在一些问题，如用户间的相似度难以准确衡量，用户对推荐结果的满意度有待提高等。因此，未来的研究重点应该是如何提高算法的准确性和用户满意度。

# 算法原理

1. 协同过滤算法

本次实验报告所使用的推荐算法是基于用户相似度的协同过滤推荐算法。

1. 协同过滤算法介绍

协同过滤推荐（Collaborative Filtering recommendation）是在信息过滤和信息系统中正迅速成为一项很受欢迎的技术。与传统的基于内容过滤直接分析内容进行推荐不同，协同过滤分析用户兴趣，在用户群中找到指定用户的相似（兴趣）用户，综合这些相似用户对某一信息的评价，形成系统对该指定用户对此信息的喜好程度预测。

协同过滤是迄今为止最成功的推荐系统技术，被应用在很多成功的推荐系统中。电子商务推荐系统可根据其他用户的评论信息，采用协同过滤技术给目标用户推荐商品。

协同过滤算法主要分为基于启发式和基于模型式两种。

其中，基于启发式的协同过滤算法，又可以分为基于用户的协同过滤算法（User-Based）和基于项目的协同过滤算法（Item-Based）。

启发式协同过滤算法主要包含3个步骤：

1. 收集用户偏好信息；
2. 寻找相似的商品或者用户；
3. 产生推荐。

“巧妇难为无米之炊”，协同过滤的输入数据集主要是用户评论数据集或者行为数据集。这些数据集主要又分为显性数据和隐性数据两种类型。其中，显性数据主要是用户打分数据，譬如用户对商品的打分，五分制的1分，2分等。

但是，显性数据存在一定的问题，譬如用户很少参与评论，从而造成显性打分数据较为稀疏；用户可能存在欺诈嫌疑或者仅给定了部分信息；用户一旦评分，就不会去更新用户评分分值等。

而隐性数据主要是指用户点击行为、购买行为和搜索行为等，这些数据隐性地揭示了用户对商品的喜好。

隐性数据也存在一定的问题，譬如如何识别用户是为自己购买商品，还是作为礼物赠送给朋友等。

1. 以用户为基础（User-based）的协同过滤

用相似统计的方法得到具有相似爱好或者兴趣的相邻用户，所以称之为以用户为基础（User-based）的协同过滤或基于邻居的协同过滤(Neighbor-based Collaborative Filtering)。

|  |  |  |
| --- | --- | --- |
| 协同过滤的推荐算法 | 基于记忆的协同过滤(Memory-based CF) | 基于用户(User-based) |
| 基于物品(Item-based) |
| 基于模型的协同过滤(Model-based CF) | 以隐因子模型、朴素贝叶斯分类为代表 |

基于记忆的协同过滤主要分为：

相似统计方法得到具有相似兴趣爱好的用户。

基于模型的协同过滤主要分为：

先用历史数据得到一个模型，在用此模型进行预测。基于模型的推荐广泛应用使用的技术包括神经网络技术、潜在语义LFM分析、贝叶斯网络聚类算法模型、分类算法模型、回归算法模型、矩阵分解模型、神经网络模型图模型、隐语义模型。

1. 皮尔逊相关系数

在协同过滤中，一个重要的环节就是如何选择合适的相似度计算方法，常用的两种相似度计算方法包括皮尔逊相关系数(Pearson Correleation Coefficient)和余弦相似度等。据有关研究表明，Pearson相关系数在基于用户的相似度计算方面的表现要优于其他算法，因此，本次报告中我使用的相似度计算方法为皮尔逊相关系数。

皮尔逊相关系数（Pearson Correlation）是衡量向量相似度的一种方式。输出范围为-1到+1，其中0代表无相关性，负值代表负相关，正值代表正相关。皮尔逊相关系数在欧几里德距离上做出了优化，对向量的值做了中心化处理，即对两个向量中的所有维度都减去元素的平均值，中心化后所有维度的平均值基本为0；然后对中心化结果求余弦距离，但余弦距离的计算要求每个向量中所有的值都必须非空，若两个向量v1=(3,2,4)、v2=(-1,2,null)，则无法进行余弦距离计算的。皮尔逊相关系数把向量中所有null维度赋值为0，再对结果进行余弦计算。

皮尔逊相关系数的计算公式如下：

通常通过以下取值范围判断向量的相关程度：

1. 0.8-1.0 极度相关
2. 0.6-0.8 强相关
3. 0.4-0.6 中等程度相关
4. 0.2-0.4 弱相关
5. 0.0-0.2 极弱相关或无相关
6. 预测用户对产品的评分

基于用户的协同过滤子引擎，通过下面的公式来计算出用户对推荐产品的评分：

1. 优缺点
2. 协同过滤优点：

实现快

对商品和物品没有要求

效果有保障

1. 协同过滤缺点：

冷启动

马太效应

推荐解析模糊

1. 所用评估方法

取20%的数据作为测试集 ，n=测试集里用户i所对应的产品个数，评价分数初始为0/n。 然后利用算法为训练集中的用户i生成推荐列表，如果一个商品在推荐列表里并且也在用户i的测试集里，那么评价分数变成 (0+(1-abs(测试集中用户i对此产品的真实评分-推荐列表中预测的用户对此产品的评分))/用户i对所有产品评分的均值)/n，遍历推荐列表里的每件产品，最后得出评估分数m/n。评估分数处于0~1之间，越接近1推荐系统效果越好。

# WikiLens数据集的特点和结构

WikiLens数据集是一个大规模的语言数据集，旨在为机器学习和自然语言处理任务提供训练数据。以下是其主要特点和结构：

1. 大规模性

WikiLens数据集包含了大量的文本数据，其中包括英文、中文、日文、法文和西班牙文等多种语言的文本数据。该数据集的总容量超过了10亿个文档，是目前最大的语言数据集之一。

1. 多样性

WikiLens数据集包括了各种类型的文本数据，包括新闻报道、文章、评论、学术论文、博客文章等。这些数据来自于不同的来源，如新闻机构、学术机构、博客网站等，这些数据的多样性使得WikiLens数据集更加接近于真实的语言使用情况。

1. 语言多样性

WikiLens数据集包括了多种语言的文本数据，包括英文、中文、日文、法文和西班牙文等多种语言。这些语言的多样性使得WikiLens数据集更加适合用于国际化的自然语言处理任务。

1. 数据预处理

WikiLens数据集已经过预处理，包括了分词、词性标注、命名实体识别等处理。这些预处理操作使得该数据集更加适合于机器学习和自然语言处理任务。

1. 开放性

WikiLens数据集是开放的，可以通过多种方式进行访问和使用。该数据集提供了API接口和多种数据预处理工具，以便用户可以方便地使用该数据集进行研究和开发。

总之，WikiLens数据集具有大规模、多样性、语言多样性、数据预处理和开放性等特点，使其成为了机器学习和自然语言处理领域的重要数据集之一。

# 算法实现

1. 数据预处理

首先读入数据，并划分训练集和测试集，将训练集和测试集保存为txt文件，然后格式化数据集。去除数据集中的空格与换行符并以用户的id建立字典，方便后续相似度的计算。

1. 用皮尔逊相关系数计算测试集中的用户相似度

定义Euclid函数。先取出两个用户购买的商品，储存两个用户都购买过的商品id。然后遍历寻找两个用户都购买过的商品，若两个用户都购买过的商品数为0，则皮尔逊相关系数为0。随后根据皮尔逊相关系数的计算公式计算用户对共有商品评价的均值和皮尔逊系数。最后遍历用户，计算某个用户与其他用户的相似度比对。

1. 构建推荐方法

定义recommend函数。取相关系数大于0的用户,储存其商品信息，记录相似用户购买的且user未购买的评分大于等于3的商品，将这些商品推荐给用户。然后根据用户的协同过滤子引擎公式来计算出用户对推荐商品的喜好程度。

1. 推荐结果评估

运用所定义的评估方法，对推荐效果进行评估。最后得出模型的评估分数为0.7588。

# 结论

当使用皮尔逊相关系数作为用户相似度的度量时，模型的准确率可以得到0.76，这已经是一个比较高的准确率了。但是，仍然有一些改进的地方可以考虑，以进一步提高模型的准确率。

数据预处理：在使用皮尔逊相关系数作为用户相似度的度量时，数据预处理对于模型的准确率提高非常重要。因此，可以考虑使用更高级的数据预处理技术，如基于正则表达式的文本分类技术等，以提高数据质量和准确性。

更多的数据集：使用更多的数据集可以提高模型的泛化能力，从而使模型在新的数据上表现更好。因此，可以考虑使用更多的语言数据集，如中文、日文、法文和西班牙文等，以提高模型的准确率。

更多的模型优化：可以考虑使用更多的模型优化技术，如基于支持向量机的推荐算法、基于深度学习的推荐算法等，以提高模型的准确率和效率。

更好的特征提取：可以考虑使用更好的特征提取技术，如基于注意力机制的文本分类技术、基于时间序列的特征提取技术等，以提高模型的准确率和效率。

总之，基于用户相似度的协同过滤推荐算法已经得到了广泛应用，并取得了一定的效果。但是，仍然有很多地方需要改进，以进一步提高模型的准确率和效率。

# 附录

1. 原始数据

[wikilens-ratings\out.wikilens-ratings](wikilens-ratings/out.wikilens-ratings)

1. 源程序

#读入数据

content = []

with open('./wikilens-ratings/out.wikilens-ratings') as fp:

content = fp.readlines()

content.remove(content[0])

content

#划分训练集，测试集

import pandas as pd

from sklearn.model\_selection import train\_test\_split

rnames = ['user\_id', 'product\_id', 'rating', 'timestamp']

data = pd.read\_csv('./wikilens-ratings/out.wikilens-ratings',sep = '\t', names=rnames)

data = data.drop(0)

trn\_data, val\_data, \_, \_ = train\_test\_split(data, data, test\_size=0.2)

trn\_data = trn\_data.sort\_values(by='user\_id',ascending=True)

trn\_data.to\_csv('trn\_data.txt',sep='\t', index=False,header=False)

val\_data = val\_data.sort\_values(by='user\_id',ascending=True)

val\_data.to\_csv('val\_data.txt',sep='\t', index=False,header=False)

data

#格式化数据集

content\_trn = []

content\_val = []

with open('./trn\_data.txt') as fp:

content\_trn = fp.readlines()

trn\_data\_dict = {}

for line in content\_trn:

line = line.strip().replace('\n','').split('\t')

if not line[0] in trn\_data\_dict.keys():

trn\_data\_dict[line[0]] = {line[1]:line[2]}

else:

trn\_data\_dict[line[0]][line[1]] = line[2]

with open('./val\_data.txt') as fp:

content\_val = fp.readlines()

val\_data\_dict = {}

for line in content\_val:

line = line.strip().replace('\n','').split('\t')

if not line[0] in val\_data\_dict.keys():

val\_data\_dict[line[0]] = {line[1]:line[2]}

else:

val\_data\_dict[line[0]][line[1]] = line[2]

#用皮尔逊相关系数计算测试集中的用户相似度

from math import \*

def Euclid(user1,user2):

#取出两个用户购买的商品

user1\_data = trn\_data\_dict[user1]

user2\_data = trn\_data\_dict[user2]

#储存两个用户都购买过的商品id

item\_set = []

global user1\_item\_value

user1\_item\_value = 0

user2\_item\_value = 0

#遍历寻找两个用户都购买过的商品

for key in user1\_data.keys():

if key in user2\_data.keys():

item\_set.append(key)

user1\_item\_value += float(user1\_data[key])

user2\_item\_value += float(user2\_data[key])

#若两个用户都购买过的商品数为0，则皮尔逊相关系数为0

if len(item\_set) == 0:

pearson\_cc = 0

return pearson\_cc

#计算用户对共有商品评价的均值

user1\_item\_mean = user1\_item\_value/len(item\_set)

user2\_item\_mean = user2\_item\_value/len(item\_set)

#计算皮尔逊系数

m = 0

n,n1,n2 = 0,0,0

for i in item\_set:

m += (float(user1\_data[i])-user1\_item\_mean)\*(float(user2\_data[i])-user2\_item\_mean)

n1 += (float(user1\_data[i])-user1\_item\_mean)\*\*2

n2 += (float(user2\_data[i])-user2\_item\_mean)\*\*2

n = sqrt(n1)\*sqrt(n2)

if n == 0:

#特殊情况下的pearson\_cc计算方式

pearson\_cc = (1-abs(n1-n2))/float(max(len(user1\_data.keys()),len(user2\_data.keys())))

else:

pearson\_cc = m/n

return pearson\_cc

#计算某个用户与其他用户的相似度比对

def top\_similar(user):

res = []

for userid in trn\_data\_dict.keys():

#排除当前用户

if not userid == user:

similar = Euclid(user,userid)

res.append((userid,similar))

res.sort(key = lambda val:val[1],reverse=True)

return res

#构建推荐方法

def recommend(user):

#取相关系数大于0的用户,储存其商品信息

top\_user = top\_similar(user)

items = {}

for i in range(len(top\_user)):

if top\_user[i][1] == 0 or top\_user[i][1] < 0:

top\_user = top\_user[:i]

break

else:

u = top\_user[i][0]

#记录相似用户购买的且user未购买的评分大于等于3的商品，将这些商品推荐给用户

for j in range(len(trn\_data\_dict[u].keys())):

if float(trn\_data\_dict[u][list(trn\_data\_dict[u].keys())[j]]) >= 3 and list(trn\_data\_dict[u].keys())[j] not in trn\_data\_dict[user].keys():

if not u in items.keys():

items[u] = {list(trn\_data\_dict[u].keys())[j]:trn\_data\_dict[u][list(trn\_data\_dict[u].keys())[j]]}

else:

items[u][list(trn\_data\_dict[u].keys())[j]] = trn\_data\_dict[u][list(trn\_data\_dict[u].keys())[j]]

dict\_top\_user = dict(top\_user)

#基于用户的协同过滤子引擎，计算用户对推荐商品的喜好程度

m = {}

n = {}

score = {}

for i in items.keys():

for j in items[i].keys():

if not j in m.keys():

m[j] = (float(items[i][j])\*dict\_top\_user[i])

n[j] = (dict\_top\_user[i])

else:

m[j] += (float(items[i][j])\*dict\_top\_user[i])

n[j] += (dict\_top\_user[i])

for i in m.keys():

score[i] = m[i]/n[i]

score = sorted(score.items(), key=lambda x:x[1], reverse=True)

score = dict(score)

return score

#推荐结果评估

m = 0

n = len(val\_data\_dict[user].keys())

for i in val\_data\_dict[user].keys():

if i in recommend(user).keys():

m += (1-abs(float(val\_data\_dict[user][i])-recommend(user)[i])/user1\_item\_value)

print(m/n)