# 《数据科学与工程算法》项目报告

报告题目： 基于矩阵分解的推荐算法

姓 名： 高宇菲

学 号： 10215501422

完成日期： 2023.7.5

摘要 [中文]：

随着大数据时代的到来，精准的推荐系统算法被广泛使用，并取得了令人满意的效果。本项目使用movielens开源数据集，主要实现了基于矩阵分解算法的推荐系统算法，在测试集上的准确率（MAE）达0.75，具有一定的科学价值和商业价值。本项目使用平方损失函数，并用梯度下降优化损失函数，并使用了正则化防止过拟合。最后，本项目还对特定用户进行了案例分析，结果表明本项目的算法可以较准确地预测用户偏好的电影类型。本项目还可以通过协同过滤、考虑偏置来提升准确率。

Abstract [English]：

With the advent of the era of big data, accurate recommendation system algorithms are widely used and have achieved satisfactory results. This project uses movielens open source dataset and mainly implements a recommendation system algorithm based on matrix decomposition algorithm with an accuracy rate (MAE) of 0.75 on the test set, which has some scientific and commercial value. This project uses a squared loss function and optimizes the loss function with gradient descent, and uses regularization to prevent overfitting. Finally, this project also conducts a case study for a specific user, and the results show that the algorithm of this project can predict the user's preferred movie genre more accurately. This project can also improve the accuracy by collaborative filtering and considering bias.

1. 项目概述

本项目基于MF算法，根据Movielens 100k 开源数据集构建了推荐系统，并且对推荐的性能，如重建损失RMSE、准确性MAE、训练时间做了统计和说明，最后进行了案例分析，为某几个用户推荐他们最喜欢的电影。

本项目具有一定的科学价值。在实际的推荐系统中，用户-项目交互矩阵往往是非常稀疏的，矩阵分解方法能有效地处理这种稀疏性，将大型的用户-项目矩阵分解为两个低秩矩阵，从而使得系统可以基于用户和项目的潜在特征来进行推荐。而且，矩阵分解的算法通常可以很好地扩展到大规模的数据集，因为可以通过随机梯度下降等优化方法进行高效的计算。这使得基于矩阵分解的推荐系统可以应用在实际的大规模推荐任务中。最后，准确和有效的推荐系统在各种商业场景中具有巨大的应用价值，如电子商务、在线广告、信息检索、社交媒体等，通过提升用户体验和满足用户需求，进而提升商业效益。

目前，推荐系统依据做出推荐的方式分为三个主要领域[1]，分别是基于内容的过滤(Content-Based Filtering)，协同过滤(Collaborative Filtering)和混合方案(Hybrid approach)。在基于内容的过滤方法中，用户会被推荐与他们过去喜欢的项目相似的其他项目或产品[2][3]。用于进行推荐的内容可以是图像、音频流、视频流、标签或文本等[4]。然而，内容信息往往不容易被机器分析[5]。基于内容的过滤方法的推荐结果往往受到“过度专业化”问题的困扰[2]，而且很难进行评估[5]。由于这些问题，基于内容的过滤方法在实践中并不被广泛采用。

协同过滤方法基于用户之间或项目之间的相似性来进行推荐。它利用大量用户的历史行为数据，发现用户之间的兴趣相似性或项目之间的关联性。当一个用户的行为与其他用户的行为相似时，协同过滤算法会将其他用户喜欢的项目推荐给该用户，或者当一个项目与其他项目具有相似的用户喜好时，将相似项目推荐给用户。

混合方法结合了基于内容的过滤和协同过滤的优势，以获得更准确和个性化的推荐结果。混合方法可以根据具体情况和需求，将基于内容的过滤和协同过滤进行灵活的组合，以提高推荐系统的效果。

本文使用的矩阵分解方法属于协同过滤范畴。

1. 问题定义

**定义1 个性化推荐问题**

将用户对电影的评分用矩阵表示，通过将矩阵进行分解，将用户兴趣和待推荐电影都映射到一个维的隐空间中。通过比较用户兴趣向量和待推荐电影向量，确定用户对待推荐电影的感兴趣程度大小。

其中，矩阵是一个稀疏矩阵， .

**定义2 矩阵分解**

给定一个用户集合，一个项目集合，以及有用户对项目的评分所构成的评分矩阵。矩阵分解旨在找到两个矩阵和，使得

其中为潜在特征的维数，满足。此外，矩阵中每一行表示用户与特征之间的关联强度，而矩阵中的每一列表示项目与特征之间的关联强度。

1. 方法
2. 数据集

本项目使用的Movielens 100k 开源数据集中，包含了[MovieLens](<http://movielens.org>) 上由610个用户对9742部电影的100836个评分，评分时间是1996.3.29 - 2018.9.24。每位用户至少评价了20部电影，用户是随机选取的。评分范围是0.5星-5.0星。

实验时，选取前300个用户对3000部电影的评分。

1. 数据预处理

首先，考虑到矩阵大小不能太大，用sklearn.preprocessing中 LabelEncoder将电影实际ID 重新映射编号，并保存映射表。

其次，将下载的数据整理成评分矩阵。这是一个高度稀疏的矩阵。中保存了用户对电影的评分。如果未评分，该元素为0。

1. 划分数据集

随机划分训练集、验证集、测试集，比例为64:18:18。使用scipy.sparse找出非零元素坐标，用numpy.random.shuffle函数打乱坐标，并按比例划分坐标点。

1. 损失函数

用预测评分和实际评分之间的差距的平方作为损失函数。

其中，K是评分矩阵中所有非零元素的坐标，是评分矩阵行列的元素。是用户矩阵的第行，是项目矩阵的第列。是正则化系数，取0时，没有正则化约束。

1. 求解梯度

对损失函数求关于某行、某列的梯度：

1. 更新公式

其中，

1. 训练

首先，随机初始化P，Q矩阵。其次，按照以上更新公式**逐向量**更新。

每遍历一遍非零元素坐标，就计算一个loss值。

当相邻两次loss相差小于0.005时，认为收敛，结束训练。

1. 调参

首先令正则化系数为0，得到合适的步长。使得步长既不过小，又不会计算溢出。

其次，调整正则化系数，防止过拟合。

考虑到训练速度和准确性，结合数据集中的电影分类信息，隐空间大小选择18或32。

1. 实验结果

下表记录了不同超参数下的 训练集RMSE，验证集RMSE。

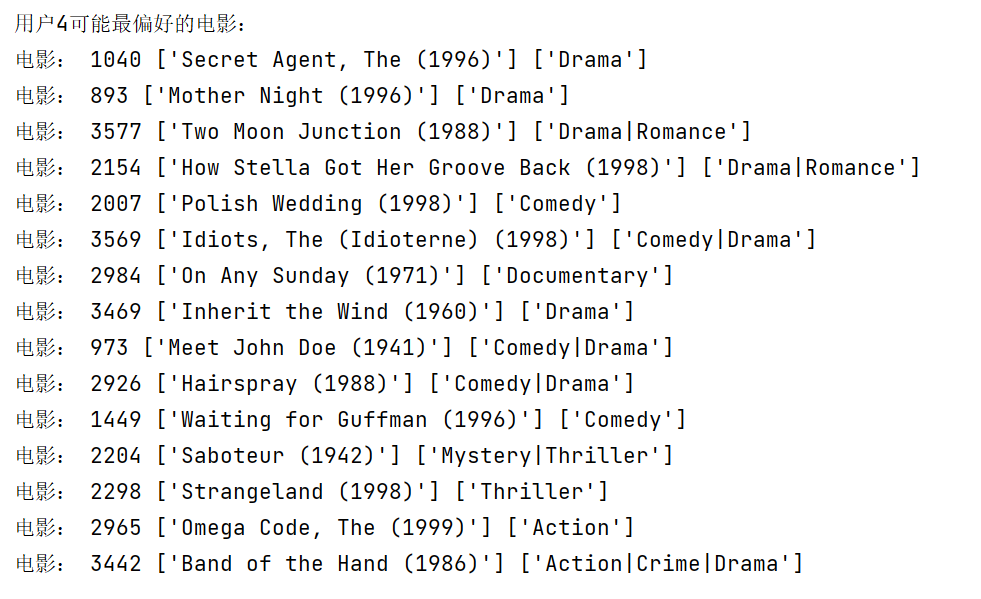
（训练集RMSE，验证集RMSE）

|  |  |  |
| --- | --- | --- |
| 隐空间维度K  正则化系数lamda | 18 | 32 |
| 0.1 | （0.737, 1.402） | （0.735, 1.450） |
| 0.12 | （0.800, 1.450） | （0.798, 1.500） |
| 0.14 | （0.850, 1.500） | （0.850, 1.540） |

可以看出，隐空间维度为18，正则化系数为0.1时，泛化程度较好。在这组超参数下，训练集训练时间23分22秒，测试集RMSE=1.414，MAE= 0.758。

下面对某用户进行案例分析。

用P矩阵的第4行乘以Q矩阵，对结果从高到低排序，找出得分最高的10部电影，就是应当为用户4推荐的10部电影。用预处理时的电影编号映射表打印出这十部电影的名称和类型：



可以观察到，用户4偏好喜剧、戏剧。再打印用户4实际给出了5分的电影：



可以看出该用户确实偏好喜剧、戏剧，与预测类型重合度较高。

1. 结论

本项目成功预测了用户的偏好。结果表明，基于矩阵分解具有高效和有效的优点，正则化有效提升了模型的泛化能力，使用这些技术可以有效搭建推荐系统。

本项目使用了多种方法提升了训练速度，使得其更有实际应用价值。比如使用了逐向量更新而不是逐元素更新，再如使用了scipy.sparse定位非零元素。

未来，本项目还可以考虑偏置和协同过滤，进一步提升准确率。

参考文献：

[1] A Review on Matrix Factorization Techniques in Recommender Systems

[2] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions”, in IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734-749, 2005

[3] C.C. Aggarwal, “Content-Based Recommender Systems, In Recommender Systems, Springer International Publishing, pp. 139-166, 201

[4] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutirrez, “Recommender systems survey, Knowledge-Based Systems, Elsevier Journal, vol. 46 no.1, pp.109-132, 2013.

[5] F. Cacheda, V. Carneiro, D. Fernndez, and V. Formoso, “Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems, ACM Transactions on the Web (TWEB), vol.5, no. 1, pp. 1-33, 2011