

**暑期生产实习报告**

**推荐系统：基于模型的**

**协同过滤算法探究和仿真**

学 院 电子信息工程学院

作 者 何沃洲

学 号 13021264

指导教师 杨晨阳

**2016年8月**

**目 录**

[（一）研究背景 1](#_Toc460281205)

[（二）研究过程 1](#_Toc460281206)

[2.1 数据集的导入 1](#_Toc460281207)

[2.2 基准预测 2](#_Toc460281208)

[2.3 矩阵的SVD分解 2](#_Toc460281209)

[2.4 SVD的随机梯度下降算法 3](#_Toc460281210)

[2.5 测试前的预处理 3](#_Toc460281211)

[（三）研究结果 4](#_Toc460281212)

[3.1 对SVD最佳参数的预估 4](#_Toc460281213)

[3.1.1 隐语义空间维度 4](#_Toc460281214)

[3.1.2 随机梯度下降法参数 5](#_Toc460281215)

[3.2 从SVD到SVD average到SVD+ 6](#_Toc460281216)

[（四）感想与收获 7](#_Toc460281217)

[参考文献 8](#_Toc460281218)

# （一）研究背景

随着电子商务网站的发展，迫切需要出现一种能滤出所有可用的替代品的推荐系统。让客户从纷繁复杂的物品中作出最恰当的选择是困难的，而近年来推荐系统被认为是一种解决信息过载问题的有效工具。设计和开发推荐系统是多个学科共同努力的结果，这得益于计算机科学领域的发展成果，特别是机器学习、数据挖掘、信息检索等等。

其中，推荐系统的协同过滤（CF）算法已经得到了人们的普遍关注并取得了很大的进展，它在Netflix大赛中的核心作用更证明了它的重要性。构成CF有两种主要技术：基于邻域（记忆）的方法和隐语义模型。这个报告主要关注于后者，尝试通过描述产品和用户两种实体在因子上的特征来解释评分，而这些因子是根据用户的反馈自动推断出来的。

报告里的建模主要基于矩阵因子分解的SVD算法，仿真基于参考文献[4]。该领域相关研究使用的数据集主要是Netflix大赛公开的超过一亿条电影评分数据，限于笔记本配置的原因，本报告仿真选择的数据集为movielens数据集[[1]](#footnote-1)，包括了943个用户对1682部电影的十万条评分数据，利用数据集自带的mku.sh工具把数据随机划分出80%作为训练集(*TrainSet*)，剩下的20%作为测试集(*TestSet*)。推荐结果的质量由在测试集上计算得出的均方根误差（*RMSE*）来度量（公式如下，其中、分别表示用户*u*对物品*i*的预测评分值和测试集上的实际评分值）。

(1)

# （二）研究过程

## 2.1 数据集的导入

由于movielens数据集是以若干条某用户对某部电影的评分这样的形式给出的（评分值为1~5的整数），为了便于矩阵分解模型的应用，首先做一个预处理，把评分条的数据填充到UI矩阵里。矩阵的行向量代表某用户*u*对所有物品的打分，列向量代表某物品*i*得到的来自所有用户的评分，缺省的评分将在矩阵相应位置用0表示，由此得到一个巨大的稀疏矩阵。

## 2.2 基准预测

典型的CF数据显示了用户和物品中存在的偏置，也就是说，数据中有某些用户评分比其他用户高、某些物品得到的评分比其它物品高的明显倾向。这里把这些偏置（也就是与用户-物品交互作用无关的因子）在基准预测中建模，把真正代表用户-物品相互作用的那部分数据隔离开来，放到更合适的用户偏好模型中去。

设为总体平均评分，和分别表示用户*u*和物品*i*的评分与平均值的偏差，未知评分的基准综合考虑了用户和物品两个因子：

(2)

一种简单的粗略估计把和的计算分离开：

对于用户*u*， (3)

对于物品*i*， (4)

往后还将讨论通过随机梯度下降和正则化对这些参数进行校正。

## 2.3 矩阵的SVD分解

矩阵的SVD分解把用户和物品两方面的信息映射到一个维度为*f*的联合隐语义空间中，于是每一个用户*u*和每一件物品*i*都分别和一个*f*维向量相关，表示了用户对这些因子的偏好程度或物品拥有这些因子的积极程度。因此用户-物品之间的相互作用被建模为该空间中的内积，它表示了用户对物品的总体兴趣度，加上之前的基准预测可以得到最终的预测评分，即

(5)

为了学习模型中的参数，也就是、、、，可以通过最小化以下正则化的平方误差得到：

min (6)

参量控制了正则化的程度，合适的取值有利于避免对训练集的过拟合，尤其是在训练样本不大的时候。它一般通过交叉验证来确定。最小化过程可通过随机梯度下降算法或者交替最小二乘法来实现。下文主要讨论前者。

## 2.4 SVD的随机梯度下降算法

一种简单的随机梯度下降算法由研究者Funk等推广并得到成功的实践。这种算法对训练数据中的所有评分做循环。对于给定的评分，其预测评分记为，则预测误差为

(7)

对于给定的训练集评分，我们通过朝着与梯度相反的方向移动来修正参数，即

(8)

(9)

我们把这种基础模型称为“SVD”。其中*k*代表*f*维空间的第*k*个特征因子。当在Netflix数据集上评估该方法时，建议使用以下参数值：，。在实际应用SVD时使用了early-stop，即当测试集上的*RMSE*不再下降或下降幅度很小的时候便停止迭代。

当考虑基准偏置的学习时，和可以跟同时通过迭代训练得到。参考文献[4]给出的和的训练式子为

(10)

(11)

我们把这种包含了基准学习的分解称为“SVD+”。其中参量的建议值为0.05。

## 2.5 测试前的预处理

经过以上训练得到的重构UI矩阵可用于对原UI矩阵缺省元素的预测。缺省元素的预测评分值绝大部分依然落在[1,5]的范围内，对于少数范围以外的元素，我们把预测值1以下的元素重新赋值为1，把预测值5以上的元素赋值为5。经过测试该方法可在一定程度上减小*RMSE*，并具有一定的合理性。

此外，由于随机划分训练集和测试集时，测试集里存在一些物品没有在训练集里被评分，或者一个用户没有在训练集里对物品评分，那么该物品和该用户的未知评分值便无法在重构的UI矩阵中作出预测，表现为相应位置的元素预测值为未知数。在这种情况下，计算*RMSE*时把该元素排除在外。

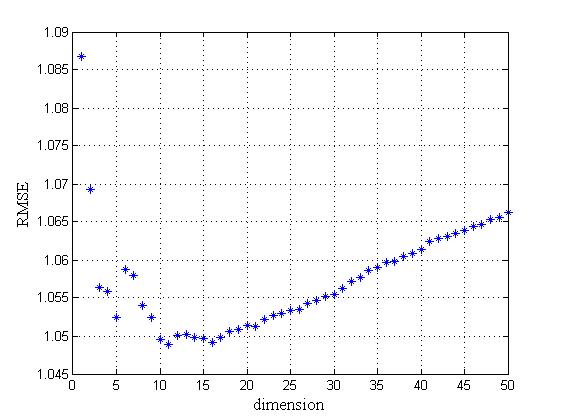
# （三）研究结果

## 3.1 对SVD最佳参数的预估

为了粗测SVD中几个重要的参数，包括映射到隐语义空间的维度(dimension)、训练步长(alpha)和正则化因子(lambda)等等，在此先评估参数值大小对预测效果的影响。

3.1.1 隐语义空间维度

在Netflix大赛中，推荐的SVD维度约为100。由于movielens数据集比较小，在此以1为步长扫描了维度在50以下的SVD效果（依然由在测试集上计算出的*RMSE*评价）。扫描由保留维度数对应的若干个奇异值的直接SVD低阶近似完成，可快速估测出给定条件下的最佳维度，结果如图1。

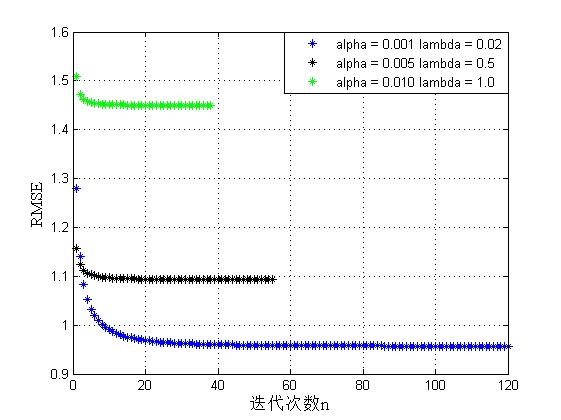


**图1 movielens数据集上SVD映射维度与预测效果(*RMSE*)的关系**

由散点图可以看出，当维度比较小（10以下）时，*RMSE*呈现了较大的波动且普遍偏大，预测效果不好。维度在15以上时，*RMSE*与维度之间呈现明显的正相关，意味着在计算量加大的同时预测效果也在变差，此时对训练集有过拟合的趋势，表现为一些随机的噪声对模型产生了干扰。在10到15的区间内，*RMSE*较为稳定对维度变化不敏感并保持在较低的水平。综合考虑计算量与预测效果的平衡，决定使用映射维度数为10进行仿真和测试。

3.1.2 随机梯度下降法参数

训练步长和正则化因子是随机梯度下降算法里的两个重要参数，它们的选择将直接影响到计算复杂度（即运算时间）和结果的误差及收敛性。在测试中发现，步长和正则化因子的改变所产生的影响较为相似，所以在此放在一起讨论。为了比较不同参数值选择对结果的影响，在此以SVD（暂不考虑偏置，即公式10和11）对三组参数进行了研究，结果如图2。

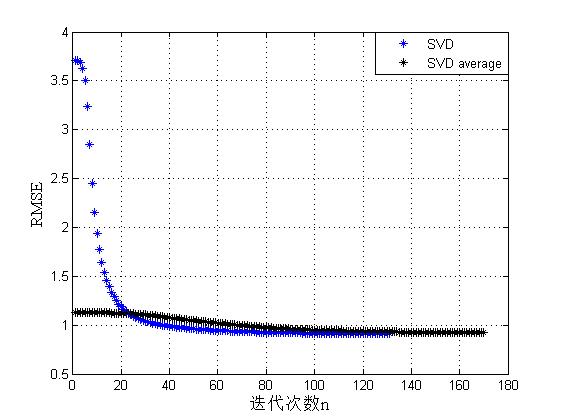


**图2 随机梯度下降法不同的参数选择对预测效果(*RMSE*)及收敛性的影响**

由结果可以看出，在early-stop条件下，步长alpha和正则化因子lambda选择得越大，*RMSE*降到其最低点越快，而停止迭代时的*RMSE*也越大。经过权衡，决定选择alpha = 0.001，lambda = 0.02进行仿真，此时运算时间不致于太长，而预测效果也能达到较理想的水平，这组参数值同时也是在Netflix数据集上建议选择的。

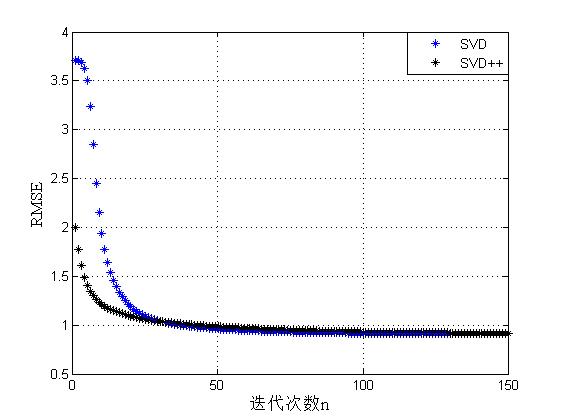
## 3.2 从SVD到SVD average到SVD+

对于SVD的基础模型，以上已经通过参数的合理选择使其优化。为了更大程度地发挥SVD的效能，在训练前可先把评分的总体均值分离开，使对样本训练时集中学习用户与物品相互作用的隐含维度，我们把这种方法称为“SVD average”。由图3的效果对比可以看出，SVD average收敛速度有所下降，但它可以获得更小的初始误差，在迭代次数少时有较大的优势，在某些运算条件受限的情况下可以得到应用。



**图3 SVD与SVD average的预测效果(*RMSE*)随迭代次数的变化对比**

下面继续对SVD average中引入的偏置校正进行拓展。在2.2中已经阐述了基准偏置的粗略估计方法，为了得到更为合理的用户及物品偏执度和，我们在随机梯度下降训练和同时对和进行并行的学习和调整（公式10和11），即“SVD+”。由图4的结果可以看出，SVD+不但获得更小的初始误差而且兼备较快的收敛速度，相比SVD有明显的优势。



**图4 SVD与SVD+的预测效果(*RMSE*)随迭代次数的变化对比**

**表1 SVD、SVD average、SVD+在笔记本上运行的指标数据**

|  |  |  |  |
| --- | --- | --- | --- |
| 指标 | **SVD** | **SVD average** | **SVD+** |
| 运算时间 **/** s | 106.7 | 143.2 | 132.8 |
| 停止迭代的*RMSE* | 0.91432 | 0.94372 | 0.91762 |

（软硬件环境: intel corei7处理器2.1GHz、8G内存、win10操作系统、MATLAB R2014a）

# （四）感想与收获

这次生产实习虽然时间不长，在暑假一些课程和比赛之间断断续续地完成了仿真，但我从中得到的收获却是很丰富的。首先感谢杨老师的指导，感谢学长在百忙之中抽出时间跟我讨论、给我意见。这次实习我做的课题是“推荐系统”。刚领到这个题目时我也感觉有点纳闷，虽然它和移动大数据有联系，但毕竟是个偏计算机科学的领域。但慢慢地学着做着，我开始发现它跟我的第一感觉不大一样，这是一个新兴的领域，里面涉及到的诸多算法都是跨学科的，跟现实需求的紧密联系赋予了它丰富的内涵和外延，超越了纯粹数理推导的枯糙。我觉得在其中收获最多的两点应该就是阅读文献和发现问题、探究问题的能力了。

一开始拿到四篇长文的时候真的感觉有点无从下手，我以做英语阅读题的习惯开着电子词典逐字逐句地看，然而这样的效率实在是太低了，不仅看的慢，有时候好不容易看懂了下文发现又把前面的内容给忘记了。于是我关掉了词典，每段都只找关键句仔细读，其它的知道了大意之后再确定信息量最大的重点部分慢慢读，感觉适应多了，后来当我在某些段落找标示的参考文献去读的时候也不觉得那么可怕了。还有的时候，文章看着觉得差不多都懂了，但是上了仿真，结果却一塌糊涂，这就是没有及时地发现问题、把握重点信息。我刚开始仿真的时候就犯了不少这样的错误，例如忽略了初始条件、忽略了预处理步骤，连函数里计算*RMSE*的总体都弄错了，这一方面是因为对数学语言没理解、数学直觉不敏感，另一方面是自己确实还没好好消化好文章的核心和作者的思想。这时候也不能光想着求助，应该好好再琢磨琢磨，把问题分解开来分析，好好想想哪里掉了线，虽然可能会多耗点时间，但这个过程对发现问题、解决问题这种能力的培养却是必要的。

以后学习的道路还很漫长，对自己学习、自己探究的需求也更加突出了，在这个过程里不但要提高能力，还要培养信心，相信只要肯下功夫就没有什么困难是克服不了的。

# 参考文献

[1] M. Ekstrand, J. Riedl, and J. Konstan. “Collaborative Filtering Recommender Systems”. *Found. Trends Hum.-Comput. Interact.* 2011.

[2] Y. Shi, M. Larson, and A. Hanjalic. “Collaborative Filtering beyond the User-Item Matrix- A Survey of the State of the Art and Future Challenges”. *ACM Comput. Surv.* 2014.

[3] Y. Koren, R. Bell, and C. Volinsky. “Matrix factorization techniques for recommender systems”. *Computer.* 2009.

[4] A. Paterek. “Improving regularized singular value decomposition for collaborative filtering”. Proceedings of KDD cup and workshop. 2007.

[5] 推荐系统：技术、评估及高效算法[M]. 弗朗西斯科等著. 李艳民等译. 机械工业出版社. 2015年7月第一版.

1. 数据来源：*MovieLens Datasets* http://grouplens.org/datasets/movielens/ [↑](#footnote-ref-1)