基于正负反馈矩阵的 SVD 推荐模型①

吴 扬, 林世平

(福州大学 数学与计算机科学学院, 福州 350108)

摘 要: 矩阵奇异值分解技术已经被广泛应用在个性化推荐系统之中. 通过矩阵奇异值分解可以提高个性化推荐的准确度. 传统的奇异值分解模型对整个矩阵进行分解, 得到 user 和 item 两个特征矩阵, 然后进行评分预测, 并未考虑不同范围的评分包含的不同信息. 通过计算评分中的临界值, 把评分矩阵拆分成两个矩阵, 称为正反馈矩阵和负反馈矩阵. 再基于两个反馈矩阵的特征来完成对评分的预测. 在实验数据方面, 使用 MovieLens 的数据集, 对传统的奇异值分解模型(SVD)和基于超图的奇异值分解模型(HSVD)进行改进. 实验结果表明, 引入偏好区分概念的模型 PSVD、PHSVD, 其推荐效果都优于原模型.

关键词: 推荐系统; 协同过滤; 矩阵分解; 正(负)反馈矩阵; SVD 模型.

SVD Recommendation Model Based on Positive and Negative Feedback Matrix

WU Yang, LIN Shi-Ping

(School of Math and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: Singular value decomposition technique has been widely used among the personalized recommendation system. By matrix singular value decomposition can improve the accuracy of personalized recommendations. The traditional model only do the singular value decomposition of the matrix is decomposed into user feature matrix and item feature matrix, and the prediction score is not considered different information containing a different range of scores. By calculating scores critical value, the scoring matrix split into two matrices, called positive feedback matrix and negative feedback matrix. Then two feedback matrices based on the feature to complete the scoring in the prediction. In the experimental data, as used herein MovieLens data sets, the traditional model of singular value decomposition (SVD) and based on hypergraph singular value decomposition model (HSVD) to improve it. Experimental results show that the effects of PSVD, PHSVD model are better than the original model.

Key words: recommended system; collaborative filtering; matrix factorization; positive (negative) feedback matrix; SVD model.

1 引言

随着科技的发展, 互联网上的信息量日益增大, 用户很难从中筛选出有用的信息, 因此, 通过推荐系统把用户可能喜欢的东西推荐给他们, 在当今的大数据背景下非常有意义. 推荐系统的诞生旨在提高企业收益, 例如亚马逊的推荐系统为其带来了占公司营收比 20%的收益. 个性化推荐作为推荐系统中的一部分, 目的是为用户提供个性化服务, 推荐不同领域的产品, 解决营销中出现的“长尾问题”.

个性化推荐的一个重要任务是从用户的历史评分记录中发现用户偏好. 因为不同的用户具有不同的评分习惯和喜恶评价标准, 例如有的用户习惯给高分, 而有的习惯给低分. 能否准确挖掘出用户的每一个特征, 关系到个性化推荐算法结果是否准确.

协同过滤算法是个性化推荐中的重要算法, 主要可以分为基于记忆的方法和基于模型的方法[3]. 其中基于记忆的方法包括 user-based 方法和 item-based 方法, 其主要思想是寻找相似用户(或物品), 并用相似用户(或

① 收稿时间:2014-10-10;收到修改稿时间:2014-11-28

物品)的评分完成推荐. SVD 模型算法是基于模型方法的一种, 它的推荐效果显著高于基于记忆的方法. SVD 模型的主要思想是从评分矩阵 R 中分解出表示用户特征的矩阵 P、和表示物品特征的矩阵 Q, 再以 $P*Q$ 为基础构造数学模型来预测评分. 但传统的 SVD 模型对评分矩阵不做任何处理, 直接把原始的评分矩阵用两个特征向量矩阵来表示. 后人为了改进预测结果, 在评分预测函数中加入了用户偏倚(user-bias)和物品偏倚(item-bias)[4], 然而这两种偏倚是直接作用在最终评分数值之上的, 只是在评分函数上添加了两个待训练的参数, 而对于为何会导致预测结果产生偏移没有深入探讨, 因此该方法实际上并没有抓住预测结果会产生偏倚的根本原因. 本文通过划分用户的偏好, 设置用户评分临界值对原始评分矩阵进行分割, 再分别进行奇异值分解和参数训练, 得到了较好的实验结果.

本文的篇章结构如下: 第二节介绍相关模型的背景知识, 包括 SVD 模型和 HSVD 模型; 第三节介绍本文提出的算法; 第四节进行实验并对比分析实验结果; 第五节对全文进行总结.

2 背景

2.1 SVD 模型

SVD 模型是一个最优化控制模型, 它把基准预测模型 (baseline predictor) 和矩阵分解模型 (matrix factorization model)融合到了一起[5].

2.1.1 基准预测模型

在评分矩阵中的, 不同用户的分值可能包含不同的意义. 例如有的用户习惯于给物品偏高(或偏低)的分数. 基准预测模型正是为了解决这个问题而提出的. 基准预测模型的预测公式如下:

$$r_{ui}' = u + b_u + b_i \tag{1}$$

其中 u 表示预测基准(一般取评分矩阵的平均值), b_u 和 b_i 分别表示第 u 个用户和第 i 个物品的评分偏倚值.

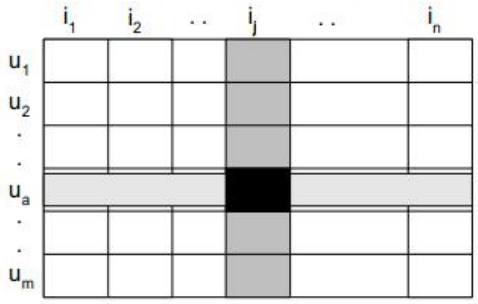
把 b_u 和 b_i 作为参数, 考虑如下最优化问题:

$$min \sum_{(u,i)\in T} \left(r_{ui} - u - b_u - b_i\right)^2 + \lambda\left(b_u{}^2 + b_i{}^2\right) \tag{2}$$

基准预测模型先计算最优化问题(2), 得出 b_u 和 b_i, 再根据(1)计算预测评分 r_{ui}'.

2.1.2 矩阵分解模型

我们可以把所有用户的评分数据看成是一个矩阵, 其中每一行代表每个用户的评分, 每一列代表每个物品得到的评分.



假设评分矩阵 R 的维度是 $m*n$, 那么 R 的奇异值分解可以写成如下形式:

$$R = U * S * V^T \tag{3}$$

其中 S 是由 R 的奇异值组成的对角矩阵, U 和 V 分别是 m*m 和 n*n 的酉矩阵.

注意到, (3)式可以改写为如下形式:

$$R = P_u * Q_i^T \tag{4}$$

其中 $P_u = \left(U * \sqrt{S}\right)$, $Q_i = V * \sqrt{S}^T$.

这样, 就把评分矩阵分解成用户特征矩阵 P_u 和物品特征矩阵 Q_i 相乘的形式. 用 P_u 表示矩阵 P_u 的第 u 行, 用 Q_i 表示矩阵 Q_i 的第 i 行, 则第 u 个用户对第 i 个物品的预测评分可以表示为:

$$r_{ui}' = p_u * q_i^T$$

如果评分矩阵的规模很大, 可以通过选取 S 中最大的 k 个特征值进行近似计算, 以达到降维的目的.

记评分矩阵 R 的 k 阶近似矩阵为 R', 则 R'的计算方法如下:

$$R' = P_{uk} * Q_{ik}^T$$

其中 $P_{uk} = \left(U_k * \sqrt{S_k}\right)$, $Q_{ik} = V_k * \sqrt{S_k}^T$, S_k 表示 S 中最大的 k 个特征值组成的 k 阶对角矩阵, U_k 和 V_k 分别是 $m*k$ 和 $n*k$ 阶的矩阵, 表示的是与 S 中最大的 k 个特征值对应的特征向量. 假设 φ 是通过降维处理之后的矩阵 R'所保留的原始矩阵 R 的信息量. 则

$$\varphi = \sum_{i=1}^{k} \lambda_i \Big/ \sum_{i=1}^{n} \lambda_i$$

其中 k 是降维后的维度, n 是原始维度, 这里的λ_i表示原始矩阵的特征值, 且

$$\lambda_i \geq \lambda_j, \text{if } i < j$$

通常推荐算法使用的数据集是稀疏性很大的矩

阵, 因此选取的 k 不需要太大就可以基本保证原始矩阵 R 的大部分信息, 故即使 k 比 n 小很多, 降维操作带来的误差也不会很大. 通常取 $\varphi=80\%$ 或 $\varphi=90\%$.

2.1.3 SVD 模型

SVD 模型结合了上述两个模型, 它的评分预测函数如下:

$$r_{ui}'=p_u*q_i^T+b_u+b_i \tag{5}$$

最优化问题的目标函数为:

$$\begin{aligned}&min\sum_{(u,i)\in T}\left(r_{ui}-p_u*q_i^T-b_u-b_i\right)^2\\&+\lambda\left(b_u{}^2+b_i^2+p_u{}^2+q_i^2\right)\end{aligned} \tag{6}$$

模型通常使用随机梯度下降法(SGD)求解问题(6), 再根据(5)完成评分预测.

2.2 HSVD 模型

与 SVD 模型不同, HSVD 模型中用于提取用户特征的矩阵不是评分矩阵, 而是用户-物品的关联矩阵. HSVD 模型把关联矩阵进行加权处理, 并把它看成一个超图的邻接矩阵[2], 再根据图的谱理论进行特征聚类, 聚类方法基于超图邻接矩阵的奇异值分解结果. 通过特征预测评分. HSVD 的评分预测函数使用了非对称 SVD 模型的思想, 即把物品特征矩阵作为已知量, 以此训练用户特征矩阵[6].

HSVD 模型的预测函数如下:

$$r_{ui}^{'}=p_u*q_i^T+b_u+b_i \tag{7}$$

HSVD 模型的参数优化目标函数如下:

$$\begin{aligned}&min\sum_{(u,i)\in T}\left(r_{ui}-p_u*q_i^T-b_u-b_i\right)^2\\&+\lambda\left(b_u{}^2+b_i^2+p_u{}^2\right)\end{aligned} \tag{8}$$

3 算法

3.1 算法背景和定义

在现实生活中, 商品通常包含了多个属性. 在用户对一件商品做出评价时, 商品所包含不同属性会影响评分的高低. 例如商品包含用户喜欢的属性则容易获得高分, 而包含用户不喜欢的属性则可能获得低分.

在这里我们认为用户评分高的物品包含的用户喜欢的属性占主导地位, 用户评分低的物品则相反. 因此根据用户的这一特点, 可以把用户评分矩阵拆分成两个矩阵:

$$R=R_p+R_n$$

其中 R_p 表示用户所喜欢的物品的评分矩阵, 称为正反馈评分矩阵; R_n 表示用户不喜欢的物品的评分矩阵, 称为负反馈评分矩阵. 把正反馈评分矩阵中的每个评分称为评分矩阵的正反馈项, 负反馈评分矩阵中的每个评分称为评分矩阵的负反馈项. 把我们的算法称为 PSVD 算法.

3.2 PSVD 算法描述

3.2.1 计算反馈评分矩阵

首先, 算法需要得到正、负两个反馈评分矩阵. 这就必须判断每个评分所反应出的主要是用户的喜欢还是厌恶. 根据基准预测模型(baseline predictor)的思想, 评分之中存在用户偏倚和物品偏倚, 因此仅用一个基准值 μ 来区分正、负反馈项并不合理. 一种改进的方法是找到为每个评分 r_{ui} 找到一个基准值 μ_{ui}, 当 $r_{ui}<\mu_{ui}$ 时, 该评分表示用户厌恶, 将其归入负反馈评分矩阵; 反之, 则将其归入正反馈评分矩阵.

3.2.2 建立评分预测模型

其次, 算法通过矩阵分解模型来预测评分. 在进行矩阵分解之前, 先对正负反馈评分矩阵做预处理, 令 $R_p'=R_p-U*I_p$, $R_n'=R_n-U*I_n$. 其中 U 表示基准值 μ_{ui} 组成的矩阵, I_p 和 I_n 表示 R_p 和 R_n 的标示矩阵(即当 $R_p(i,j)\neq 0$ 时, $I_p(i,j)=1$). 运用 SVD 模型思想, 假设

$$\begin{aligned}R_p'&=P_{u1}*Q_{i1}^T+b_{u1}+b_{i1}\\R_n'&=P_{u2}*Q_{i2}^T+b_{u2}+b_{i2}\end{aligned}$$

评分预测公式如下:

$$\begin{aligned}r_{ui}'&=\mu_{ui}+p_{u1}*q_{i1}^T+b_{u1}+b_{i1}\\&+p_{u2}*q_{i2}^T+b_{u2}+b_{i2}\end{aligned} \tag{9}$$

PSVD 算法的步骤:

第一步: 计算出评分矩阵 R 每行的平均值 $\overline{\mu}_u$ 和每列的平均值 $\overline{\mu}_i$, 再通过 $\mu_{ui}=rate*\overline{\mu}_u+(1-rate)*\overline{\mu}_i$ 算出每个评分 r_{ui} 的基准值. 其中 $rate$ 是参数, 且 $rate\in[0,1]$.

第二步: 计算 R_p' 和 R_n', 并通过奇异值分解计算出 P_{u1}、Q_{i1}^T 和 P_{u2}、Q_{i2}^T.

第三步: 求解最小二乘问题

$$min \sum_{(u,i)\in T} (r_{ui} - \mu_{ui} - p_{u1} * q_{i1}^T$$
$$-b_{u1} - b_{i1} - p_{u2} * q_{i2}^T - b_{u2} - b_{i2})^2$$
$$+\lambda(b_{u1}^{\ 2} + b_{i1}^{\ 2} + p_{u1}^{\ 2} + q_{i1}^{\ 2}$$
$$+b_{u2}^{\ 2} + b_{i2}^{\ 2} + p_{u2}^{\ 2} + q_{i2}^{\ 2})$$

其中 p_{u1}、q_{i1}^T、p_{u2}、q_{i2}^T 的初始值取 R_p'和 R_n'奇异值分解的结果，b_{u1}、b_{i1}、b_{u2}、b_{i2} 的初始值取 0.

记

$$e_{ui} = r_{ui} - \mu_{ui} - p_{u1} * q_{i1}^T - b_{u1}$$
$$-b_{i1} - p_{u2} * q_{i2}^T - b_{u2} - b_{i2}$$

最小二乘问题目标函数中的各参数按照梯度下降法进行优化，即:

$$b_{u1} = b_{u1} + l_1(e_{ui} - \lambda * b_{u1})$$
$$b_{i1} = b_{i1} + l_1(e_{ui} - \lambda * b_{i1})$$
$$p_{u1} = p_{u1} + l_1(e_{ui} * q_{i1} - \lambda * p_{u1})$$
$$q_{i1} = q_{i1} + l_1(e_{ui} * p_{u1} - \lambda * q_{i1})$$
$$b_{u2} = b_{u2} + l_2(e_{ui} - \lambda * b_{u2})$$
$$b_{i2} = b_{i2} + l_2(e_{ui} - \lambda * b_{i2})$$
$$p_{u2} = p_{u2} + l_2(e_{ui} * q_{i2} - \lambda * p_{u2})$$
$$q_{i2} = q_{i2} + l_2(e_{ui} * p_{u2} - \lambda * q_{i2})$$

其中 l_1、l_2 表示正、负反馈矩阵的学习率(梯度下降的步长).

第四步: 通过本节中的公式(9)预测用户评分.

3.3 PSVD 算法扩展

PSVD 算法的思想同样可以应用在 HSVD 算法上，我们称该算法为 PHSVD 算法. 算法的步骤和 PSVD 算法相似. 下面给出参数优化过程中的目标函数，以及最终的用户评分预测公式:

$$min \sum_{(u,i)\in T} (r_{ui} - \mu_{ui} - p_{u1} * q_{i1}^T$$
$$-b_{u1} - b_{i1} - p_{u2} * q_{i2}^T - b_{u2} - b_{i2})^2$$
$$+\lambda\left(b_{u1}^{\ 2} + b_{i1}^{\ 2} + p_{u1}^{\ 2} + b_{u2}^{\ 2} + b_{i2}^{\ 2} + p_{u2}^{\ 2}\right)$$
$$r_{ui}' = \mu_{ui} + p_{u1} * q_{i1}^T + b_{u1} + b_{i1}$$
$$+p_{u2} * q_{i2}^T + b_{u2} + b_{i2}$$

在这里，只有 p_{u1}、b_{u1}、b_{i1}、p_{u2}、b_{u2}、b_{i2} 是参数，q_{i1}、q_{i2} 是常量，参数的训练方法和 3.2 节相同.

4 实验

本实验数据采用了 MovieLens 数据集(详细信息 http://www.grouplens.org/node/73).

该数据包含了 10 万条用户评分信息，评分范围在 1~5 分之间. 实验目的是对比本文提出的模型 PSVD、PHSVD 与 SVD、HSVD 模型的推荐效果. 实验把数据分成训练集和测试集，训练集用来得到模型中的相关待定参数，测试集用来对比预测结果是否与实际结果相符合.

本实验包括两个部分. 第一部分使用的训练集和测试集是随机抽取. 本部分实验使用 MAE 和 RMSE 两个指标来对比 PSVD 和 SVD 模型评分预测准确度的优劣. 第二部分针对冷启动的情形选择训练集和测试集. 由于 HSVD 算法的提出主要针对的是冷启动问题，因此本部分实验通过准确率和召回率对比冷启动情形下 PHSVD 和 HSVD 模型的最终推荐结果的优劣.

4.1 PSVD 和 SVD 对比实验

本部分实验的数据采用随机抽取的形式，将随机抽取数据的 80%作为训练集，剩余的 20%作为测试集. 设定参数 $rate = 0.8$，学习率 $l_1 = 0.04$、$l_2 = 0.01$，正则化参数 $\lambda = 0.15$.

通过改变特征向量长度 k 和迭代步数 t 来观察实验结果变化，结果如下:

表 1 mae、rmse 与 *k* 的关系表(t=30)

k	5	20	50	100	200
psvd-mae	0.7469	0.7449	0.7413	0.7412	0.7411
psvd-rmse	0.8831	0.8717	0.8648	0.8644	0.8644
svd-mae	0.7812	0.7771	0.7768	0.7766	0.7762
svd-rmse	0.9446	0.9345	0.9339	0.9336	0.9334

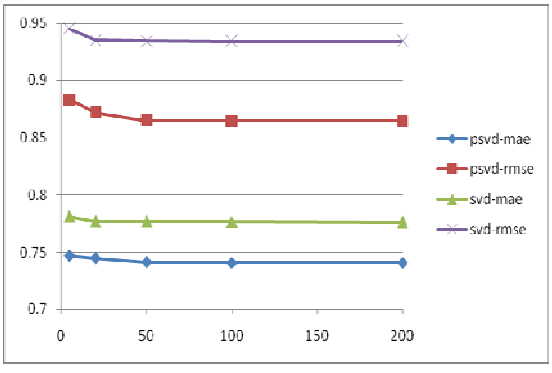


图 1 mae、rmse 与 *k* 的关系图(t=30)

表 2 mae、rmse 与 *t* 的关系表(*k*=50)

t	5	10	20	30	50
psvd-mae	0.7508	0.7469	0.7426	0.7413	0.7417

psvd-rmse	0.8894	0.8765	0.8681	0.8648	0.8649
svd-mae	0.7854	0.7793	0.7772	0.7768	0.7769
svd-rmse	0.9486	0.9367	0.9342	0.9339	0.9339

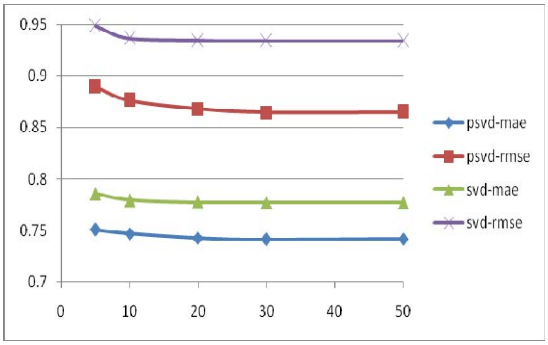


图 2 mae、rmse 与 t 的关系图(k=50)

从实验结果可以看出， 不论 k 的取值大小, PSVD 模型的评分预测误差均明显优于 SVD 模型. 并且在收敛速度上, 二者也没有明显的差异. 表 2 中可以看出, 当 t 值过大时, 评分预测的误差反而升高, 这种现象是算法的过拟合造成的, 通过改变正则化参数 λ 可以解决这一问题.

4.2 PHSVD 和 HSVD 对比实验

本部分实验的数据模拟了系统中出现新用户的情形. 训练集的抽取方式如下: 先抽取 80%用户的完整评分, 再对剩余的 20%用户每人抽取 10 条评分记录, 合并这两部分的数据作为实验的训练集, 把剩余部分作为实验的测试集. 在计算准确率和召回率时,

本实验假设评分不低于 4 分的记录表示用户喜欢, 即向用户推荐的物品在测试集里必须不低于 4 分, 才认为该项推荐是准确的, 另外, 推荐结果使用 top-N 推荐, 即把评分预测矩阵中, 每行的 N 个最高评分物品作为推荐结果.

本部分实验参数取值和 4.1 相同, 同时设定了特征向量长度 $k=50$, 迭代步数 $t=30$, 推荐个数 $N=10$. 并且采用多次随机抽样取平均值的方式进行, 结果如下:

表 3 冷启动下 PHSVD、HSVD 推荐结果

模型	Hsvd(%)	phsvd(%)
recall	15.26	15.61
precision	16.72	17.38

从实验结果中可以看出, PHSVD 模型的推荐结果在准确率和召回率两个指标上均优于 HSVD 模型.

5 总结

本文根据用户评分的数值, 引入了正负反馈矩阵的概念, 对评分矩阵进行预处理. 实验表明, 使用正负反馈矩阵双向提取用户特征, 能够得到比传统 SVD 模型更好的评分预测结果和推荐结果. 另一方面, PSVD 模型继承了 SVD 模型的优点, 能够较好地解决矩阵稀疏性的问题, 在用户共同评分很少的情况下也能得到较好的实验结果.

此外, PSVD 算法还有改进的空间. 算法仅用一个基准值来划分正负反馈矩阵, 并认为正(负)反馈矩阵中反映出的完全是用户喜欢(讨厌)的特征. 事实上这种划分方式并不合理, 因为一个接近基准值的评分很可能包含了一些更复杂的信息, 这一缺陷可以通过改变正负反馈矩阵的划分方式来改进. 例如可以使用基准区间代替基准值, 把原始的评分矩阵划分为正反馈矩阵、负反馈矩阵和边界反馈矩阵三个部分, 其中边界反馈矩阵包含了那些属于基准区间内的评分. 再通过模糊聚类的方法, 计算边界反馈矩阵的评分关于反馈矩阵的隶属度, 从而更精确地划分评分矩阵, 以改进原有划分方法. 总之, 如何通过改进划分正负反馈矩阵的方法来得到更好的推荐结果, 是今后的工作重点和方向.

参考文献

1 Koren Y, Bell R. Advances in collaborative filtering. Recommender Systems Handbook. Springer US, 2011: 145–186.

2 Pu L, Faltings B. Understanding and improving relational matrix factorization in recommender systems. Proc. of the 7th ACM conference on Recommender systems. ACM. 2013. 41–48.

3 Su X, Khoshgoftaar TM. A survey of collaborative filtering techniques. Advances in Artificial Intelligence, 2009, 2009: 4.

4 Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer, 2009, 42(8): 30–37.

5 Koren Y. Factor in the neighbors: Scalable and accurate collaborative filtering. ACM Trans. on Knowledge Discovery from Data (TKDD), 2010, 4(1): 1.

6 Paterek A. Improving regularized singular value decomposition for collaborative filtering. Proc. of KDD Cup and Workshop. 2007. 5–8