统计信号处理第二次大作业

最小二乘法应用

姓名:　刘　前
学号：2014011216
班级：　无 47
日期：2017-06-24

目录

摘要

协同滤波是推荐系统中常用的推荐算法之一。本次大作业基于协同滤波这一问题背景，尝试使用加权正则化交替最小二乘法解决了协同滤波中的填充矩阵问题，其基本思想是使用交替最小二乘法优化二次误差函数。除此之外，本文还实现了正则奇异值分解的方法，但是未利用最小二乘的思想，因而不在报告中详细介绍。加权正则化交替最小二乘法是本次大作业重点呈现的算法，有效解决了协同滤波问题。本文着重对该算法超参数的选择、性能评价 (测试集上的均方误差) 和算法的收敛速度进行了分析与讨论，最终得到了性能较优的结果。

1 问题背景

推荐系统是目前非常活跃的研究方向之一，主要目的是利用用户的评分记录等信息对用户进行推荐。推荐系统包含了很多推荐算法，其中最常用的算法是协同滤波。

协同滤波经典背景是：假设有 n_u 个用户和 n_i 部电影 (也可以是商品、美食等其他物品)，用户 u 对电影 i 的打分记作 M_{ij}，由此可以形成一个大小为 $n_u \times n_i$ 的评分矩阵 M。实际生活中每个用户不一定对所有的电影都打过分，所以矩阵 M 不会被完全填充，因而协同滤波所要解决的问题也可以归纳为填充矩阵问题，即: 根据现有的打分信息，估计得到整个评分矩阵 M[1]。

2 最小二乘法简介

最小二乘法应用十分广泛，尤其在函数逼近、数据拟合和回归分析等领域，是最重要的方法之一。首先，从狭义上理解，最小二乘法主要可以概括为：给定二次误差函数 $L(\mathbf{y}, \hat{\mathbf{y}}) = ||\mathbf{y} - \hat{\mathbf{y}}||_2^2$ 下，利用该在样本点上预测值与样本值间的误差总和 $L = \sum_{i=1}^{N} L(\mathbf{y}_i, \mathbf{y}(\hat{\mathbf{x}_i}))$ 最小确定数学模型 $\hat{\mathbf{y}}(x)$。

需要注意的是，最小二乘法的使用前提是：给定数学模型 $\hat{\mathbf{y}}(x)$ 的函数形式 $f(\mathbf{x}, \theta)$，当此函数相对于模型参数 θ 是非线性函数时，此问题即为非线性最小二乘问题，否则即为有闭式解的线性最小二乘问题。

近年来，最小二乘问题又有很多扩充内容，一类是针对模型参数 θ 进行约束或者限制，比如：等式约束、稀疏性约束、能量最小化约束等等。也有对拟合模型 $\hat{\mathbf{y}}(x)$ 进行约束的，比如光滑性约束等，这类问题可以转化为普通最小二乘问题，或者利用迭代重加权最小二乘方法求解。另一类是优化参数具有多重线性性，可以使用交替最小二乘方法求解。

3 加权正则化交替最小二乘法

3.1 算法目标

加权正则化交替最小二乘法 (Alternating-Least-Squares with Weighted- Regularization，后文简称为 ALS-WR) 是最小二乘法的一种变体。对于填充矩阵问题，ALS-WR 算法的思路和目标如下。

将所要填充的矩阵表示为：$\mathbf{M} = \mathbf{U}\mathbf{V}^{\mathrm{T}}$，其中 $M_{ij} = \mathbf{U}(i,:)\mathbf{V}(j,:)^{\mathrm{T}}$，表示用户 i 对电影 j 的打分是用户 i 的隐变量 $\mathbf{U}(i,:)$ 与电影 j 的隐变量 $\mathbf{V}(j,:)$ 的内积。考虑到用户类型和电影类型有限，因而可以认为评分矩阵 $\mathbf{M}$ 是低秩的。由于矩阵的秩是矩阵奇异值向量的 0 范数，可以将其放缩到 1 范数 (矩阵的核范

数)。矩阵的核范数满足:

$$||\mathbf{X}||_* = \min_{\mathbf{U},\mathbf{V}|\mathbf{X}=\mathbf{U}\mathbf{V}^{\mathrm{T}}} \frac{1}{2}(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2) \quad (1)$$

因而, 上述问题可以转换为求解优化下面的目标函数:

$$\min_{\mathbf{U},\mathbf{V}} = ||\mathbf{W}*(\mathbf{M}-\mathbf{U}\mathbf{V}^{\mathrm{T}})||_F^2+\frac{\lambda}{2}(||\mathbf{U}||_F^2+||\mathbf{V}||_F^2) \quad (2)$$

其中, 需要说明的是: λ 是控制矩阵低秩程度的超参数; $\mathbf{W}$ 是标志矩阵, $W(i,j)=1$ 表示用户 i 对电影 j 已经打过分, $W(i,j)=0$ 表示未打分, * 表示矩阵对应元素相乘。本次大作业 ALS-WR 算法针对于目标函数进行优化求解。

3.2 算法原理

目标函数 (式2) 的优化问题可以使用交替最小二乘法进行求解。其核心思路是先固定其中一个变量, 另一个变量作为优化变量使用最小二乘方法求解。比如, 先固定 $\mathbf{V}$, 以 $\mathbf{U}$ 作为优化变量, 使用最小二乘法进行更新; 然后再固定 $\mathbf{U}$, 以 $\mathbf{V}$ 作为优化变量, 如此交替, 直至一定次数或者判断收敛 [2]。

根据算法目标, 希望找到一个低秩矩阵 $\mathbf{X}$ 来逼近矩阵 $\mathbf{M}$, 其中 $\mathbf{M}=\mathbf{U}\mathbf{V}^{\mathrm{T}}$。设矩阵 $\mathbf{U}$ 为 $m\times d$ 的矩阵, $\mathbf{V}$ 为 $n\times d$ 的矩阵。其中 d 表示特征的个数, 若矩阵 $\mathbf{M}$ 的秩为 r, 则一般有 $d<<r$。为方便后文推导, 不妨记 $\mathbf{U}=[u_1,u_2,\cdots,u_m]^{\mathrm{T}}$, $\mathbf{V}=[v_1,v_2,\cdots,v_n]^{\mathrm{T}}$。

根据标志矩阵 $\mathbf{W}$ 的作用, 不妨设矩阵 $\mathbf{M}$ 中的所有非零元素构成集合 Ω, 则公式2可以转换为

$$L = \sum_{i,j,(i,j)\in\Omega}(M_{ij}-u_i^{\mathrm{T}}v_j)^2+\frac{\lambda}{2}(||u_i||^2+||v_j||^2) \quad (3)$$

下面使用交替最小二乘法使得损失函数 (式3) 最小。先固定 $\mathbf{V}$, 以 U 作为优化变量。将损失函数 (式3) 对 u_i 进行求导, 则

$$\frac{\partial L}{\partial u_i} = \sum_{j,(i,j)\in\Omega} 2(M_{ij}-u_i^{\mathrm{T}}v_j)(-v_j)+2\times\frac{\lambda}{2}u_i = 0 \quad (4)$$

$$\sum_{j,(i,j)\in\Omega} M_{ij}v_j = \frac{\lambda}{2}u_i + \sum_{j,(i,j)\in\Omega} u_i^T v_j v_j \quad (5)$$

将公式5中的求和改写成向量的形式, 则得到

$$V_{i*}^{\mathrm{T}}M_{i*} = \frac{\lambda}{2}Iu_i + V_{i*}^{\mathrm{T}}V_{i*}U_{i\cdot} \quad (6)$$

$$U_{i\cdot} = (\frac{\lambda}{2}I + V_{i*}^{\mathrm{T}}V_{i*})^{-1}V_{i*}^{\mathrm{T}}M_{i*} \quad (7)$$

根据公式7即可得到 V 固定时 U 的更新公式。同理, 当 U 固定时, 以 V 为优化变量的求解公式为:

$$V_{j\cdot} = (\frac{\lambda}{2}I + U_{*j}^{\mathrm{T}}U_{*j})^{-1}U_{*j}^{\mathrm{T}}M_{*j} \quad (8)$$

其中, 矩阵 I 表示 $d\times d$ 的矩阵, $V_{i*}=[v_j,(i,j)\in\Omega]$, 表示用户 i 评价过的电影的得分向量构成的矩阵, $U_{*j}=[u_i,(i,j)]\in\Omega$, 表示第 j 个电影的评分用户的评分向量构成的矩阵。M_{i*} 表示用户 i 评过的分数, M_{*j} 表示电影 j 得到的分数。

综合上述推导, 即可以根据公式7和8使用交替最小二乘方法更新得到矩阵 $\mathbf{U}$ 和 $\mathbf{V}$, 最终得到的近似矩阵 $\mathbf{X}=\mathbf{U}\mathrm{V}^{\mathrm{T}}$。

3.3 算法描述及性能评价

经过之前的推导, 加权正则最小二乘法的思路可以归纳如下。首先使用随机数生成函数初始化矩阵 $\mathbf{V}$, 再使用公式7更新矩阵 $\mathbf{U}$, 接着使用公式8更新 $\mathbf{U}$, 直到迭代一次的次数或者均方误差 (MSE) 收敛。因此, ALS-WR 可以概括为算法1。

3.3.1 算法描述

Algorithm 1 $ALS-WR$ 算法

输入: 用户的评分矩阵 $\mathbf{M}$，特征个数 d，迭代次数 Iterations，超参数 λ

初始化: 使用随机数生成函数 (rand()) 初始化矩阵 $\mathbf{V}$

for $n = 1 : Iterations$

1. $U_{i\cdot} = (\frac{\lambda}{2}I + V_{i*}^{\mathrm{T}}V_{i*})^{-1}V_{i*}^{\mathrm{T}}M_{i*}$
2. $V_{j\cdot} = (\frac{\lambda}{2}I + U_{*j}^{\mathrm{T}}U_{*j})^{-1}U_{*j}^{\mathrm{T}}M_{*j}$

end

输出: $\mathbf{U}, \mathbf{V}, \mathbf{X} = \mathbf{U}\mathbf{V}^{\mathrm{T}}$

3.3.2 数据及算法评价

按照算法1所示的流程，本文使用作业提供的数据对该算法进行实现和性能方面的测试。

本次大作业中给出的数据尺寸为 943×1682，其中有 90000 个数据已经给出，需要将这 90000 个数据拆分为 80000 和 10000 两部分，利用 80000 个数据作为训练数据，另外 10000 个作为测试数据测试算法的效果。

算法评价的指标是均方误差，可以表示为公式9，其中 S 表示测试样本构成的集合。

$$MSE = \frac{1}{|S|}\sum_{(i,j)\in S}||M(i,j) - X(i,j)||^2 \quad (9)$$

3.4 参数的选择

根据 ALS-WR 算法 (算法1)，算法中涉及的参数主要包括特征个数 d、防止过拟合的超参数 λ 以及迭代次数 Iterations，每个参数都算法的性能或算法的运行时间有直接的影响，因而需要分析不同的参数，并通过尝试和调整选择得到合适的数值，使得既能保证算法的性能 (测试数据的 MSE 较低)，同时也要尽可能降低程序的运行时间[1]。

本部分将分别对三个参数的影响进行分析，暂时假定三个参数对算法结果的影响是独立的，只需找到各自参数在其他参数固定时使得 MSE 最小的值，理论上即可得到算法的最佳效果。

3.4.1 特征个数 d

改变选择的特征个数，其他参数固定 ($\lambda = 0.20$, Iterations = 5[2])。d 的取值分别从 5 取到 200，间隔为 5，运行得到的结果如2所示。

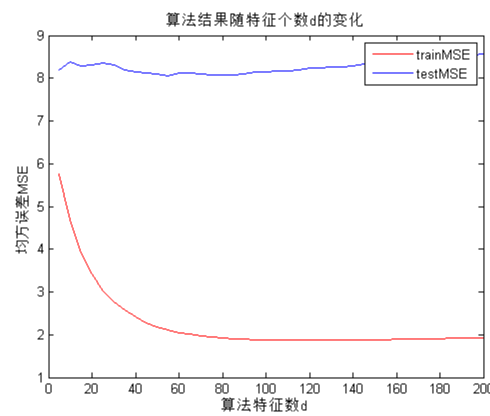

图 1: 算法结果随特征个数 d 的变化

根据图2，可以看出当 d 取值在 40 至 60 时，在训练数据和测试数据上均体现出较低的 MSE。在从算法复杂度的角度考虑，选择 d 的数值为 40。

[1]本次大作业涉及到的运行时间是基于 4GB 内存的笔记本电脑、在 MATLAB R2014 软件环境下得到。

[2]以便快速得到 MSE。实际上 5 次迭代一般不能保证已经收敛到最小值，下同。

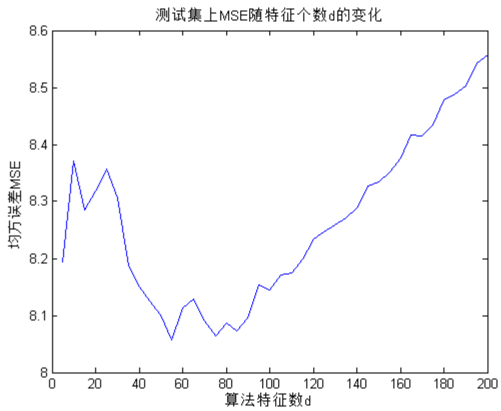

图 2: 测试集上的 MSE 随特征个数 d 的变化

3.4.2 超参数 λ

超参数 λ 是控制低秩程度的参数，也具有防止过拟合的作用，不同的 λ 会对最终的 MSE 结果产生直接的影响。改变 λ 的数值，其他参数固定 (d = 40, Iterations = 5)。λ 的取值分别从 0.01 取到 0.50，间隔为 0.01，运行得到的结果如3所示。

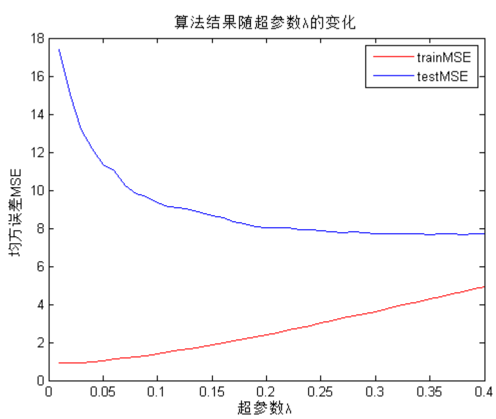

图 3: 算法结果随超参数 λ 的变化

根据图3，可以看出当 λ 取值在 0.40 左右时，算法的性能较好。

3.4.3 迭代次数 Iterations

一般来说，随着迭代次数的增加，MSE 会逐渐收敛到最小值。但是迭代次数过多不仅会造成时间复杂度的增加，还有可能导致过拟合现象的发生。因而，选择合适的迭代次数也对算法性能有非常重要的影响。

图4展示了算法在测试集和训练集上的 MSE 随着迭代次数增加的变化情况。从图中可以看出迭代次数大概在 30 至 40 次时已经收敛，此时迭代次数增加反而可能导致 MSE 增加。

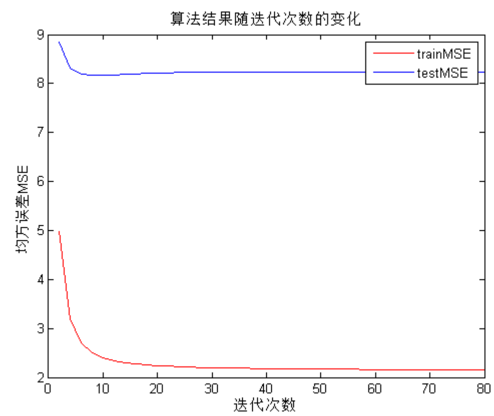

图 4: 算法结果随迭代次数的变化

3.5 算法收敛时间

3.5.1 理论时间复杂度

假设矩阵 $\mathbf{M}$ 中评分点的个数为 n_r，由先验可得矩阵 $\mathbf{M}$ 高度稀疏，所以可以得到 $n_r << m \times n$。不妨记用户个数为 m，电影数目为 n，选择的特征个数为 d，算法迭代次数为 n_i。

对于 ALS-WR 算法，每次更新矩阵 $\mathbf{U}$ 的时间复杂度为 $O(d^2(n_r + d \times m))$，每次更新矩阵 $\mathbf{V}$ 的时间复杂度为 $O(d^2(n_r + d \times n))$。根据总的迭代次数为 n_i，则总的时间复杂度为

$O(d^2(n_r + d \times (m+n)))$。因此可以得到，当算法使用的特征个数 d、迭代次数 n_i、用户数 m、电影数 n 固定时，时间复杂度主要取决于矩阵 $\mathbf{M}$ 中已有评分数据的个数。

3.5.2 最终参数选择

根据上文**参数的选择**部分的分析，暂时选择的参数为：特征个数 d 为 40 左右，超参数 λ 为 0.40 左右，迭代次数 Iterations 为 30 左右。选择这些参数的假设是各参数对算法 MSE 的作用是独立的，实际可能并不满足这一假设。但是通过手动微调，发现这一参数组合性能比较稳定[3]，MSE 结果浮动不大，因而将这一结果作为最终选择的参数。

3.5.3 算法收敛时间及均方误差

在选定的参数组合下，最后达到的均方误差为 7.80 左右浮动 (图5)，MATLAB 运行时间在 38s 左右浮动 (图6)。

testMSE	7.8006
trainMSE	4.6876
U	*943x40 double*
V	*1682x40 double*

图 5: ALS-WR 均方误差 (最佳)

函数名称	调用	总时间	自用时间*	总时间图 (深色条带 = 自用时间)
ALS_optimal	1	38.778 s	0.093 s	
ALS_WR	1	38.607 s	36.532 s	
calcMSE	31	2.153 s	2.153 s	

图 6: ALS-WR 算法 MATLAB 运行时间

[3]由于整体的训练集和测试集是随机选择的，因而算法每次运行的结果不固定，可能会有其他的参数组合在某次运行中得到个别的更小的 MSE，但是不够稳定。

4 正则化奇异值分解

本文重点呈现了最小二乘法在填充矩阵问题中的应用。在推荐系统中，还有一个广泛应用的算法为奇异值分解 (Singular Value Decomposition, SVD) 算法。本文参考了文献 [3] 和 [4]，也基于作业提供的数据实现了改进版的 SVD 算法——正则化奇异值分解 (Regularized Singular Value Decomposition, SVD)。因算法实现结果在时间长度和 MSE 上均不如 ALS-WR 算法的性能，因而不在文中具体阐述 (具体步骤详见代码)。通过与 ALS-WR 算法类似的算法分析 → 算法实现 → 参数选择等过程，同样可以在测试集上达到 8.0 左右的均方误差 (MSE)，略微高于 ALS-WR 算法的 7.80。

5 总结

本文主要使用加权正则化交替最小二乘法解决了协同滤波中的填充矩阵问题。同时还尝试实现了正则奇异值分解算法，同样能够解决了协同滤波问题。在文中，笔者对 ALS-WR 算法超参数的选择、性能评价 (测试集上的均方误差) 和算法的收敛速度分别进行了细致地分析与讨论，最终确定了参数，得到了性能较优的结果，保存得到填充矩阵。

6 附录

6.1 代码清单

6.1.1 ALS-WR 算法代码

文件名	文件功能
ALS_WR.m	ALS-WR 算法函数代码
ALS_WR_Iter.m	ALS-WR 算法调整版 (研究迭代次数)
ALS_d.m	研究特征个数 d 的影响
ALS_lambda.m	研究超参数 λ 的影响
ALS_Iteration.m	研究迭代次数的影响
ALS_optimal.m	参数确定后的实现代码
calcMSE.m	计算训练数据或测试数据上的均方误差 (MSE)

表 1: ALS-WR 算法代码清单

6.1.2 RSVD 算法代码

文件名	文件功能
RSVD.m	RSVD 算法实现代码
mean_calc.m	函数: 计算训练数据均值
bias_calc.m	函数: 计算用户和电影的属性值

表 2: RSVD 算法代码清单

6.2 程序运行说明

本文实现的目前最小 MSE 为 7.80，相应的矩阵 $\mathbf{X}$ 存储在 X.mat 中。其中，ALS-WR 算法请直接运行 ALS-WR 文件内的 ALS_optimal.m 脚本文件，等待约 40s 即可输出得到结果；RSVD 算法请直接运行 RSVD 文件内的 RSVD.m 脚本运行，时间稍长，需要约 3 分钟输出结果。

如果老师/助教在运行中出现任何问题，请联系 liuqian14@mails.tsinghua.edu.cn，谢谢！

参考文献

[1] Zhou Y, Wilkinson D, Schreiber R, et al. Large-Scale Parallel Collaborative Filtering for the Netflix Prize[C]. Algorithmic Aspects in Information and Management, International Conference, Aaim 2008, Shanghai, China, June 23-25, 2008. Proceedings. DBLP, 2008:337-348.

[2] 李改, 李磊等. 基于矩阵分解的协同过滤算法 [J]. 计算机工程与应用, 2011, 47(30):4-7.

[3] Ma, Chih Chao. A Guide to Singular Value Decomposition for Collaborative Filtering[J]. 2008.

[4] Koren Y. The bellkor solution to the netflix grand prize[J]. Netflix Prize Documentation, 2009.