统计信号处理大作业报告

2015011011

万子牛

无 52

目录

一、数学基础

本次大作业我选择的是用交替最小二乘来对稀疏矩阵进行ALS分解，即

$$M = UV^T$$

其中M是$m \times n$的待分解的稀疏矩阵，可以理解为是m个用户对n个产品的评分矩阵。其中大于 0 的值代表用户对产品的评分，等于 0 的值代表用户未对该产品作出评分。U是一个$m \times k$的矩阵，其中$k \ll m$，可以理解为m个用户对k种类别的产品的喜爱程度。V是一个$n \times k$的矩阵，可以理解被评分的n个产品属于k种类别的程度大小，比如一部电影可以同时被分为动作片，惊悚片，犯罪片。

考虑目标损失函数：

$$C(U,V) = \left|\left|W * (M - UV^T)\right|\right|_F^2 + \lambda\left(\left|\left|U\right|\right|_F^2 + \left|\left|V\right|\right|_F^2\right) \dots\dots\dots\dots\dots (1)$$

则分解的过程可以等效为对目标损失函数的优化过程，即：

$$\min_{U,V}\left|\left|W * (M - UV^T)\right|\right|_F^2 + \lambda(\left|\left|U\right|\right|_F^2 + \left|\left|V\right|\right|_F^2)$$

其中，W是标志矩阵，$\mathrm{W}(i,j) = 1$表示用户对电影已经打过分了，$W(i,j) = 0$表示未打分，$*$表示矩阵对应元素相乘。添加参数λ的目的是为了引入偏置而避免最小二乘得到结果出现过拟合。下标F代表取得矩阵的$Frobenius$范数，假设有矩阵$A = \left[a_{ij}\right]_{m\times n}$，则其$F$范数定义为：

$$||A||_F = \sqrt{tr(A^T A)} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}}$$

则目标损失函数可以被写为

$$C(U,V) = \sum_{i,j,s.t.M_{ij}\neq 0}\left(\sum_a U_{ia}V_{ja} - M_{ij}\right)^2 + \lambda(\sum_{i,a}(U_{ia})^2 + \sum_{j,a}\left(V_{ja}\right)^2)$$

则对于矩阵 U 的任意元素U_{ia}，有

$$\frac{\partial C(U,V)}{\partial U_{ia}} = 2 \times \sum_{j,s.t.M_{ij}\neq 0}[V_{ja}\left(U_{ia}V_{ja} - M_{ij}\right)] + 2\lambda U_{ia}$$

$$= 2 \times \sum_{j,s.t.M_{ij}\neq 0}(U_{ia}V_{ja}^2 - M_{ij}V_{ja}) + 2\lambda U_{ia}$$

$$= 2 \times \left(U_{ia}(W_i^T * V_{.a})^T(W_i^T * V_{.a}) - U_{ia}M_i\left(W_i^T * V_{.a}\right)\right) + 2\lambda U_{ia}$$

其中，W_i代表矩阵W的第i行；M_i代表矩阵M的第i行；$V_{.j}$代表矩阵V的第j列。

要找到局部最优，则令$\frac{\partial C(U,V)}{\partial U_{ia}} = 0$，得到：

$$U_{ia} = \frac{M_i(W_i^T * V_{.a})}{(W_i^T * V_{.a})^T(W_i^T * V_{.a}) - \lambda}$$

为方便后续计算，不妨设$n \times k$的矩阵W_I：

$$W_I = [\underbrace{W_i^T\ \ W_i^T\ \ \dots\ \ W_i^T}_{k个}]$$

则对于矩阵 U 的任意一行U_i，当$\frac{\partial C(U,V)}{\partial U_i} = 0$时，有：

$$U_i = M_i(W_I * V)[(W_I * V)^T(W_I * V) - \lambda I]^{-1} \dots\dots\dots\dots\dots (2)$$

其中，I代表$k \times k$维的单位矩阵。此时，$\frac{\partial C(U,V)}{\partial U_i} = 0$，即$C(U,V)$取得局部最优。因此，在固定$V$的条件下，我们可以通过此公式逐个更新每一个$U_i$即$U$的每一行，最终得到$U$。

同理，对于固定的U，我们可以令$\frac{\partial C(U,V)}{\partial V_j} = 0$，得到：

$$V_j = M_{.j}^T(W_{.J} * U)[(W_{.J} * U)^T(W_{.J} * U) - \lambda I]^{-1} \dots\dots\dots\dots\dots (3)$$

其中矩阵W_J的定义为：

$$W_I = [\underbrace{W_{.j}\ \ W_{.j}\ \ \dots\ \ W_{.j}}_{k个}]$$

在固定U的条件下，我们可以通过此公式逐个更新每一个V_j即V的每一行，最终得到V。

二、算法概述

1. 初始化U和V。
   U的初始化值服从均值为μ，方差为 1 的高斯分布。其中μ是训练集的80000组数据的平均值；
   V的初始化值服从参数为0.5的二元分布，取值为 0 或 1，代表一部电影属于某一类或不属于某一类。
2. 迭代，对于iter = 1,2,3, …，每次迭代使用公式(2)更新U的值，再使用公式(3)更新V的值。
3. 通过公式(1)计算该阶段的目标损失，保存在向量Cost中。
4. 当$iter$达到预设的上限，或者，$\mathrm{Cost(iter)} - \mathrm{Cost(iter-1)} < \epsilon$，代表目标损失函数收敛，停止迭代。

三、实现情况

1. 实现环境

Matlab R2018a.

2. 预测误差和参数选择

本次实验使用均方误差对预测结果进行评价：

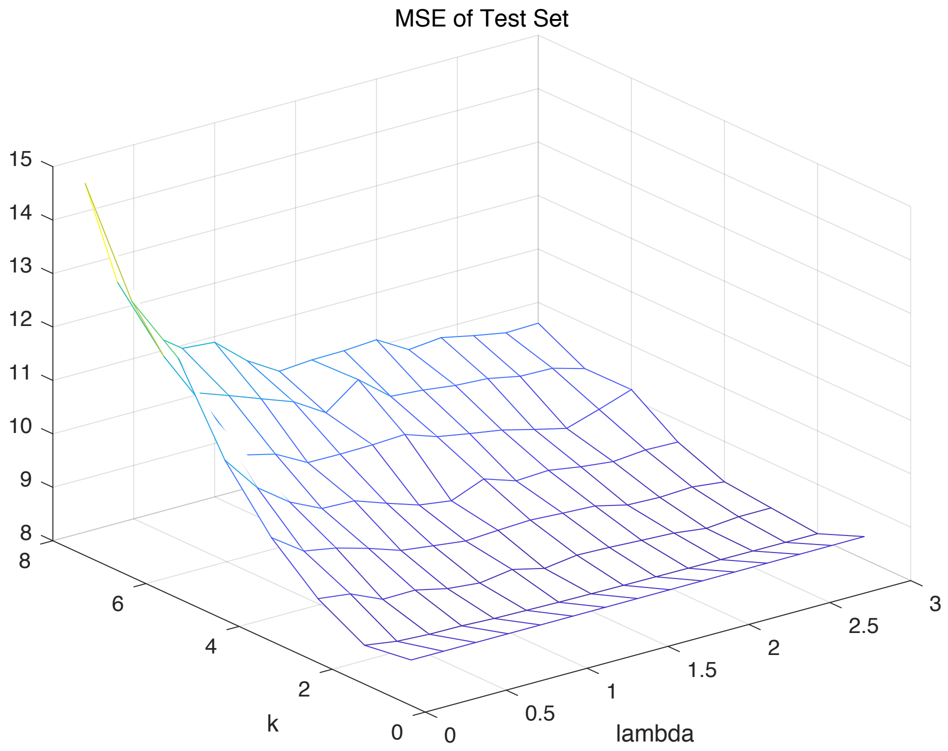
$$MSE = \frac{1}{|S|}\sum_{(i,j)\epsilon S}\left|\left|M(i,j) - X(i,j)\right|\right|^2 \dots\dots\dots\dots\dots (4)$$

对于不同的参数$\lambda\ \epsilon\ [0.2, 3]$，和$k\ \epsilon\ [1,8]$值，预测的$MSE$值分布如下：

λ\K	1	2	3	4	5	6	7	8
0.2	8.41	8.29	8.76	9.50	10.55	12.05	12.72	14.52
0.4	8.41	8.20	8.68	9.08	9.87	10.83	11.52	12.51
0.6	8.41	8.16	8.42	8.97	9.43	9.91	10.68	11.38
0.8	8.41	8.16	8.42	8.80	9.15	9.77	10.46	10.94
1.0	8.41	8.13	8.29	8.60	9.13	9.44	10.24	10.89
1.2	8.41	8.12	8.23	8.53	8.92	9.44	10.02	10.39
1.4	8.41	8.11	8.32	8.45	8.72	9.45	9.65	10.02

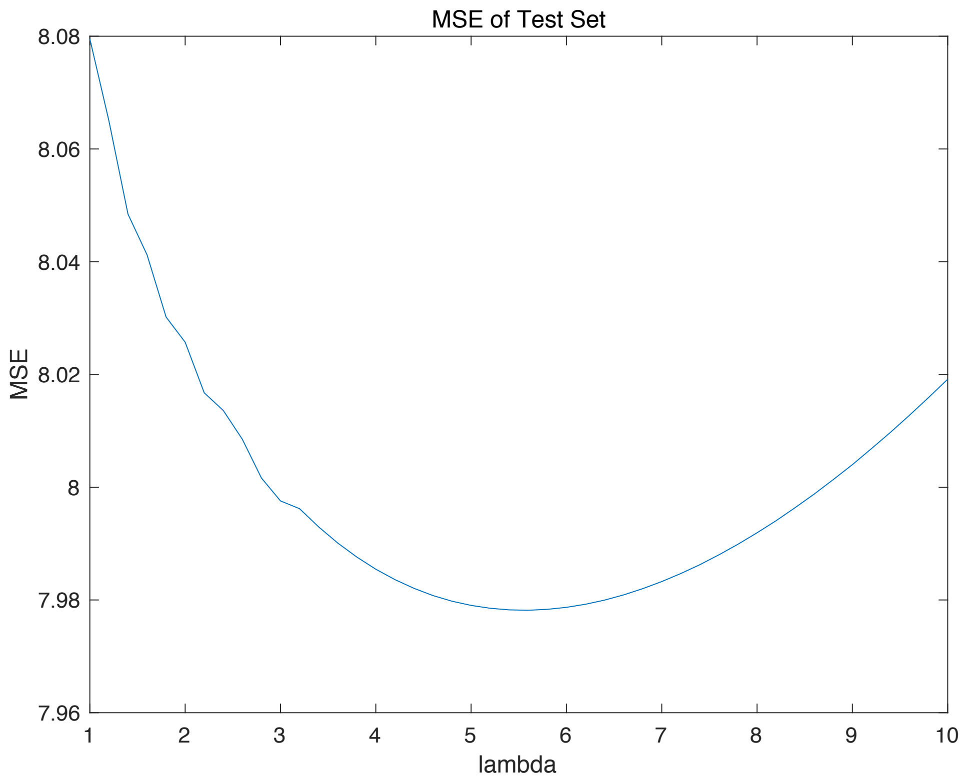
1.6	8.41	8.10	8.18	8.49	8.64	9.48	10.1	10.07
1.8	8.41	8.09	8.26	8.51	8.88	9.27	9.63	10.08
2.0	8.41	8.09	8.24	8.48	8.68	9.17	9.58	10.12
2.2	8.42	8.09	8.21	8.45	8.72	9.07	9.50	9.77
2.4	8.42	8.08	8.18	8.30	8.62	8.92	9.47	9.83
2.6	8.42	8.08	8.10	8.28	8.55	8.77	9.35	9.71
2.8	8.42	8.08	8.16	8.31	8.61	8.96	9.33	9.60
3.0	8.42	8.08	8.16	8.24	8.59	9.17	9.16	9.61

对应的三维分布图：



可以看到，在k值固定的条件下，在一定范围内提高λ，则测试集的MSE渐渐减小。在λ值固定的条件下，当k值大于 2 后，随着增大k值，测试集的MSE有明显的恶化。显然，预测表现最好的参数k为$k=2$。因此，考虑固定$k=2$，寻找最佳的参数λ。

对于固定的$k=2$，$\lambda \epsilon [1,10]$，测试集的MSE分布如下：

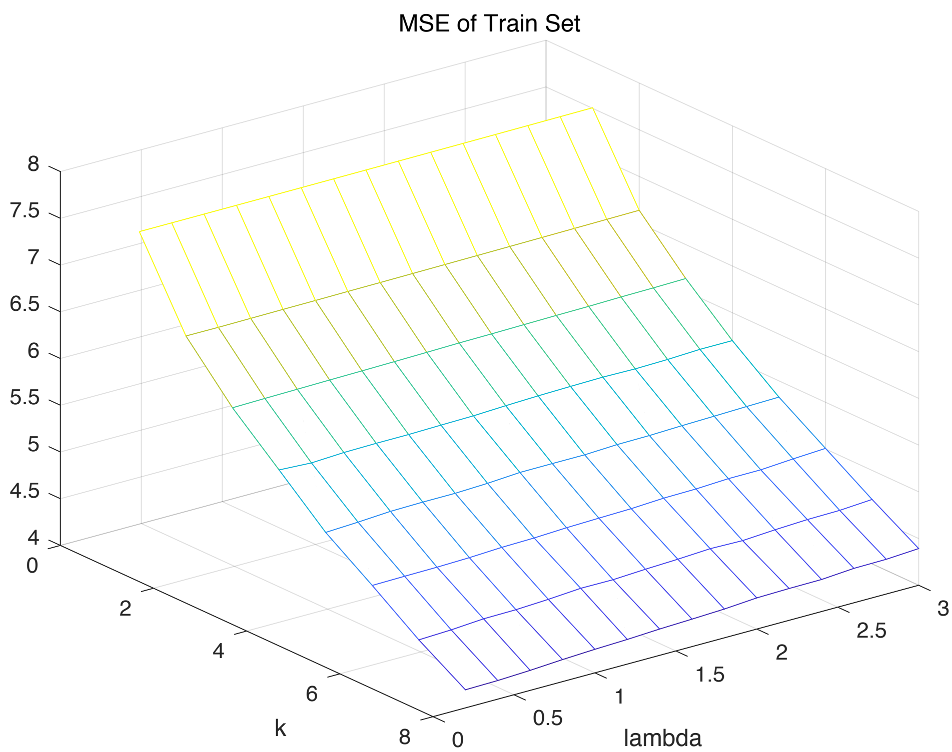


预测表现最好的一组参数是$\lambda = 5.4, k = 2$。

对应该参数的$X = UV^T$存储在$X.mat$文件中。

3. 预测分析

对于不同的参数$\lambda \epsilon$ [0.2, 3]，和$k \epsilon$ [1,8]，训练集的MSE分布图如下：



分析不同参数对应的预测误差和训练集误差，我们可以发现以下现象：

1） 在λ值固定的条件下，在 k 值大于 2 后，随着增大 k 值，测试集的 MSE 有明显的恶化。但与此同时，训练集的 MSE 则是随着 k 值的增大越来越小。可以分析这是由于 k 值增大意味着对产品的细分，这导致过拟合的现象越来越严重。

2） 在 k 值固定的条件下，在一定范围内提高λ，则测试集的 MSE 渐渐减小。与此同时，训练集的 MSE 则是随着λ值的增大而恶化。显然这是因为参数λ的作用是防止过拟合。

4. 训练集优化

由于数据集矩阵非常稀疏，故如果完全随机地选择 80000 个数据，则训练集矩阵会出现大量的空行和空列。故在选择 80000 个数据的时候应该有规则地挑选，使得挑选出的训练集尽量不出现空行和空列。做法是：在数据集中挑选时设定每行每列至少应该纳入训练集的样本点个数L。如果在原数据集中某一行或某一列的样本点数本来就小于L，那么就将该行或者该列的全部数据都选到训练集中。最后，如果以该法选出的训练集样本点数少于 80000 个点，则在剩下的点中随机选取来补齐 80000 个点。这样做使得训练结果在测试集上的MSE下降了10%左右，达到 7.0 左右。该值设置越高，越容易过拟合。在该实验中，我粗略设置了几个值，最终得到当$L = 35$时效果最好，在测试集上的MSE为$\mathbf{6.83}$。

得到的 80000 个训练集和 10000 个测试集保存在$\mathbf{data_set.mat}$中。

5. 收敛速度

k 越大，收敛需要迭代的次数越大。原因是 k 越大，自由度就随之越高，能影响目标损失函数的参数随之越多，收敛速度自然就更慢。在最优参数的情况下，收敛需要的迭代次数为146次。

6. 测试误差

训练拟合的参数在测试集上的MSE为6.83，得到的在测试集上的MSE值保存在$\mathrm{MSE.mat}$中。

四、文件清单

1. $./Code/ALS.m$：主程序，用于生成$\lambda = 5.4, k = 2$情况下的预测矩阵X。
2. $./Code/costFunction.m$：函数，

   传入参数：

   U：分解结果U，

   V：分解结果V

   M：训练集矩阵M

   W：M的标志矩阵

   λ：防止过拟合的参数λ

   返回值：

   通过公式(1)计算得到的目标误差值。

   在主程序中被调用。
3. $./Code/MSE.m$：函数，

   传入参数

   X：通过交替最小二乘法得到的预测矩阵$X = UV^T$

   S：测试集

返回值：通过公式(4)计算得到的预测矩阵和测试集之间的均方误差值。
在主程序中被调用。

5. $./Code/Data\_Shuffle.m$：用于生成$L = 35$的参数下得到的数据集分割结果。
7. $./Result/MSE.mat$：保存在$k = 2, \lambda = 5.4$的参数下，得到的填充矩阵X在测试集上的MSE。
8. $./Result/X.mat$：保存943×1682的矩阵预测矩阵X。
9. $./Result/data\_set.mat$：保存在$L = 35$的参数下，得到的数据集分割结果。