

## Project 2 个性化推荐

### 1. 问题描述

之前的课上，我们了解到个性化推荐问题是大数据的一个典型应用。个性化推荐，就是利用已知的用户浏览历史推荐新的信息：给定用户行为矩阵 $X_{m \times n}$ ，其中 $m$ 是用户数， $n$ 是需要推荐的内容数量， $X$ 中的元素 $X_{ij}$ 表示用户 $i$ 对某个电影 $j$ 的打分。因此，所谓的推荐任务就转化成，当我们已知 $X$ 中的一部分值时，如何对未知值进行预测。本次 Project，大家的任务是利用视频推荐网站 Netflix 的数据集完成推荐任务。

### 2. 数据集

我们使用的是 Netflix 推荐竞赛的一个子集，包含 10000 个用户和 10000 个电影。用户行为数据包含用户对电影的打分，分数的取值范围是 1-5。我们选取行为数据的 80%作为训练集，其余的 20%作为测试集。具体的文件格式如下：

(1) 用户列表 users.txt

文件有 10000 行，每行一个整数，表示用户的 id，文件对应本次 Project 的所有用户。

(2) 训练集 netflix\_train.txt

文件包含 689 万条用户打分，每行为一次打分，对应的格式为：用户 id 电影 id 分数 打分日期  
其中用户 id 均出现在 users.txt 中，电影 id 为 1 到 10000 的整数。各项之间用空格分开

(3) 测试集 netflix\_test.txt

文件包含约 172 万条用户打分，格式与训练集相同。

(4) 电影名称 movie\_titles.txt

文件对应每部电影的年份和名称，格式为：电影 id，年份，名称

各项之间用逗号分隔

备注：数据集中的打分时间和电影题目等信息，只有之后的选作内容才会使用到。

### 3. 实验内容

(1) 数据预处理。

大家首先要体会在大数据处理过程中不得不经历的一个步骤：数据清洗及格式化。对应到我们的问题，数据已经清洗过，因此只需我们将输入文件整理成维度为用户\*电影的矩阵  $X$ ，其中 $X_{ij}$ 对应用户  $i$  对电影  $j$  的打分。对于分数未知的项，可以采取一些特殊的处理方法，如全定为 0 或另建一个矩阵进行记录哪些已知哪些未知。

鉴于大家计算机的计算能力有区别，如果发现处理 10000\*10000 的矩阵有困难，可以选取部分用户和部分电影构成的子集，并在实验报告中说明选取规则（不得少于2000\*2000）。

注：经测试，即使使用全量数据时，所需的内存可控制在不超过4GB（前提是数据、算法处理得当）。

这一步的输出为两个矩阵， $X_{train}$ 和 $X_{test}$ ，分别对应训练集与测试集。

(2) 协同过滤

协同过滤(Collaborative Filtering)是最经典的推荐算法之一，包含基于 user 的协同过滤和基于 item 的协同过滤两种策略。本次，大家需要实现基于用户的协同过滤算法。算法的思路非常简单，当我们需要判断用户  $i$  是否喜欢电影  $j$ ，只要看与  $i$  相似的用户，看他们是否喜欢电影  $j$ ，并根据相似度对他们的打分进行加权平均。用公式表达，就是：

$$\text{score}(i,j) = \frac{\sum_k \text{sim}(X(i),X(k))\text{score}(k,j)}{\sum_k \text{sim}(X(i),X(k))}$$

其中， $X(i)$ 表示用户  $i$  对所有电影的打分，对应到我们的问题中，就是 $X_{train}$ 矩阵中第  $i$  行对应的 10000 维的向量（未知记为 0）。

$\text{sim}(X(i),X(k))$  表示用户  $i$  和用户  $k$ ，对于电影打分的相似度，可以采用两个向量的余弦相似度来表示，即：

$$\cos(x,y) = \frac{x \cdot y}{|x| \cdot |y|}$$

通过上面的公式，我们就可以对测试集中的每一条记录，计算用户可能的打分。

注：在上面的公式中，我们没有严格注明求和项下标 $k$ 的范围。请大家自行思考，应该如何选择 $k$ 的范围。

本次Project，我们采用 RMSE(Root Mean Square Error，均方根误差)作为评价指标，计算公式为：

$$\text{RMSE} = \sqrt{\frac{1}{|Test|} \left( \sum_{(i,j) \in Test} (X_{ij} - \tilde{X}_{ij})^2 \right)}$$

其中  $Test$  为所有测试样本组成的集合， $|Test|$ 为集合的大小（即测试样本的数量）， $\tilde{X}_{ij}$ 为预测值， $X_{ij}$ 为实际值。

(3) 基于梯度下降的矩阵分解算法

之前的课程中，我们介绍了矩阵分解在推荐算法中的应用。对给定的行为矩阵 $X$ ，我们将其分解为 $U, V$ 两个矩阵的乘积，使 $U$ 和 $V$ 的乘积在已知值部分逼近 $X$ ，即： $X_{m \times n} \approx U_{m \times k} V_{n \times k}^T$ ，其中 $k$ 为隐空间的维度，是算法的超参数。基于行为矩阵的低秩假设，我们可以认为 $U$ 和 $V$ 是用户和电影在隐空间的特征表达，它们的乘积矩阵可以用来预测 $X$ 的未知部分。

本实验，我们使用梯度下降法优化求解这个问题。我们的推荐算法的目标函数是：

$$J = \frac{1}{2} \|A \odot (X - UV^T)\|_F^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2$$

其中， $A$ 是指示矩阵， $A_{ij} = 1$ 意味着 $X_{ij}$ 的值为已知，反之 $A_{ij} = 0$ 。 $\odot$ 是哈达玛积（即矩阵逐元素相乘）。 $\|\cdot\|_F$ 表示矩阵的Frobenius范数，计算公式为 $\|A\|_F = \sqrt{\sum_{i,j} A_{ij}^2}$ 。在目标函数 $J$ 中，第一项为已知值部分， $U$ 和 $V$ 的乘积逼近 $X$ 的误差。后面的两项是为防止过拟合加入的正则项， $\lambda$ 为控制正则项大小的参数，由我们自己定义。

当目标函数取得最小值时，算法得到最优解。首先，我们对 $U$ 和 $V$ 分别求偏导，结果如下：

$$\frac{\partial J}{\partial U} = (A \odot (UV^T - X))V + 2\lambda U$$

$$\frac{\partial J}{\partial V} = (A \odot (UV^T - X))^T U + 2\lambda V$$

之后，我们迭代对 $U$ 和 $V$ 进行梯度下降，具体算法如下：

```
Initialize U and V
Loop until convergence
    Calculate  $\frac{\partial J}{\partial U}, \frac{\partial J}{\partial V}$ 
    Update  $U = U - \alpha \frac{\partial J}{\partial U}, V = V - \alpha \frac{\partial J}{\partial V}$ 
End loop
```

算法中 $\alpha$ 为学习率，通常根据具体情况选择一个较小的数。在本次作业中，可以尝试 $1e-5$ 至 $1e-1$ 之前的实数值。算法的收敛条件，可以选择目标函数  $J$  的变化量小于某个阈值，或设置一个最大的迭代次数。

（上述是最经典的批量梯度下降，同学们也可自行学习并尝试一些其他的优化器，如SGD、Adam等）

至此，我们就完成了在给定的 $k$ 和 $\lambda$ 下，对目标函数 $J$ 进行优化求解的全过程。同学们可以将每次迭代的目标函数值

和测试集上的RMSE值plot出来，观察算法收敛过程中这两个指标的变化。此外，我们还需要尝试不同的 $k, \lambda$ 值，找到算法的最佳参数。

#### 4. 报告提交要求

本次 Project 共 100 分，大家只需要提交**报告和代码**，请**不要提交中间结果和数据文件**。报告需要包含如下内容：

(1) 数据整理阶段的简单说明，以及最后是否选用的全量数据。（20 分，未能使用全量数据的酌情扣 1-5 分）

(2) 基于用户的协同过滤实现。包含代码的简单介绍，最终的 RMSE，算法的时间消耗。（20 分）

注意，由于本实验中X矩阵非常稀疏，因此使用matlab等对稀疏矩阵有特别优化的工具时，请尽可能使用矩阵运算，避免使用for循环。我们的公式如何转化为矩阵形式的运算，请同学们自行推导。

(3) 矩阵分解算法结果。（40分）报告需要包含：

a) 对于给定 $k=50, \lambda=0.01$ 的情况，画出迭代过程中目标函数值和测试集上RMSE的变化，给出最终的RMSE，并对结果进行简单分析。

b) 调整 $k$ 的值（如10,50,100）和 $\lambda$ 的值（如0.001,0.01,0.1,1），比较最终RMSE的效果，选取最优的参数组合。

(4) 将(2) (3)的结果进行对比，讨论两种方法的优缺点。（20 分）

#### 5. 选作内容

本次 Project 包含 20 分的选作内容，供对推荐问题感兴趣的同学进行深入研究：本次数据集中，我们提供了用户打分行行为的日期，电影名称等额外的数据。这些数据为我们评估打分行为的相似度和电影的相似度提供了额外的信息来源。如何利用这些信息改进我们的推荐算法，使结果进一步提升？

对选作内容感兴趣的同学，直接将提出的方法以及相应的实验结果写入实验报告即可。

注：若只提供一个笼统的思路，比如“考虑不同电影的年份信息”，而无具体算法和实验支持，则不会加分。

#### 6. 提交要求：

(1) 报告可以用中文或英文，不超过8页A4纸。

(2) 请将所有代码放在一个文件中。注意，我们并不会执行大家的代码，而仅作为查重使用。不要提交任何中间结果或原始数据。

(3) 所有的提交内容应打包成一个压缩文件上传。压缩文件名，以及所有的报告、代码，均需以学号命名，如 2020001002.pdf, 2020001002.py, 2020001002.zip. 请仔细检查（因为我们将用一个代码查重软件进行查重，不合规的命名只能手动修改）

(4) 如果不符合上述提交要求，会扣除5%的分数。

(5) 迟交标准：每迟交一周，将会减去20%的分数。计算周数将向上取整，例如，若迟交一天，将按照 $\lceil 1/7 \rceil = 1$ 周，即20%标准扣分。请尽早开始！

#### 一些提示：

(1) 请按照本次作业的实际背景和RMSE的物理含义，判断RMSE取值的合理范围。比如，如果我们采用随机猜(random guess)作为一个基线方法，它的RMSE大概是多少？这将帮助判断，你的程序是否执行正确。

(2) 请注意，矩阵中未出现的值代表用户未看过的电影，而非用户对该电影打了0分。这也是矩阵分解中我们加入指示矩阵 $A$ 做哈达玛积的原因。

(3) 在矩阵分解算法中， $U$ 和 $V$ 的初始化也会产生影响。本次作业的实际背景下， $U/V$ 的物理含义是什么？它们的初始化应该在什么范围？