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*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}表示用户对某个电影的打分。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，就是需要将我们的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）。

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：

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

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 矩阵中第 i 行对应的 10000 维的向量(未知记为 0)。

$sim\big(X(i),X(k)\big)$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 cos 相似度来表示，即：$\cos(\mathrm{x,y}) = \frac{x\cdot y}{|x|\cdot|y|}$。

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。

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：

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\Big(\sum_{<i,j>\in Test}(X_{ij}-\tilde{X}_{ij})^2\Big)}$$

其中 Test 为所有测试样本组成的集合，X_{ij}为预测值，$\tilde{X}_{ij}$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 UV 的乘积在已知值部分逼近 X，即：$\mathrm{X_{m*n}} \approx \mathrm{U_{m*k}}{V_{n*k}}^T$，其中 k 为隐空间的维度，是算法的参数。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：

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

其中，A 是指示矩阵，$\mathrm{A_{ij}} = 1$意味着$\mathrm{X_{ij}}$的值为已知，反之亦然。$\circ$是阿达马积（即矩阵逐元素相乘）。$\|\cdot\|_F$表示矩阵的 Frobenius 范数。计算公式为$\|A\|_F = \sqrt{\sum_i\sum_j a_{ij}^2}$。在目标函数 J 中，第一项为已知值部分，UV 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，λ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：

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

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 (very small random value);

Loop until converge:

$$U = U - \alpha \frac{\partial J}{\partial U};$$

$$V = V - \alpha \frac{\partial J}{\partial V};$$

End loop

算法中α为学习率，通常根据具体情况选择 0.0001 到 0.1 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。

至此，我们就完成了在给定的 k 和λ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，λ值，找到算法的最佳参数。

4. 报告提交要求

本次 Project 共 25 分，大家最终只需要提交**报告和代码**，请不要提交中间结果和数据文件。其中，报告需要包含的内容如下：

(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。（5 分，未能使用全量数据的酌情扣 1-3 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。 （5 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。 (10 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 20,50）和λ的值（如 0.001,0.1），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。 （5 分）

5. 选作内容

本次 Project 包含 5 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。