中国科学技术大学

学士学位论文

基于模型的机器学习方法及其在推荐系统中的应用

作者姓名: 许鑫

学科专业: 统计学

导师姓名: 杨灿 教授　郑泽敏 教授

完成时间: 二〇二二年五月二十一日

University of Science and Technology of China

A dissertation for bachelor's degree

Model-based Machine Learning and Its Application in Recommender Systems

Author: Xin XU

Speciality: Statistics

Supervisors: Prof. Can YANG, Prof. Zemin ZHENG

Finished time: May 21, 2022

中文内容摘要

低秩矩阵近似（Low-rank matrix approximation，LRMA）是指将数据表示为低秩矩阵的一类方法。它在很多领域中都有应用，包括推荐系统、自然语言处理、信号处理等。在许多应用中，我们的目标是在缺少许多观测值时解决 LRMA 问题。它被称为低秩矩阵补全（low-rank matrix completion，LRMC）。这个问题对于推荐系统非常有用。LRMC 面临着几个挑战。首先，算法需要对在高维空间的大量观测值有很好的可扩展性。其次，数据矩阵中能够观察到的元素太过稀疏，以至于没有足够的信息。第三，我们的数据集经常是不均衡的，即数据矩阵的一些行或列有大量的元素被我们收集到，而另外一些行或列可能只有很少的元素被我们收集到。

基于模型的机器学习方法（Model-based machine learning，MBML）是一种将所有机器学习算法视为模型加上推理方法的观点。模型是由一组关于领域问题的假设组成，这些假设被明确的指出来，并且用精确的统计语言表示。推理方法是指获取数据并计算目标变量的计算过程。有了这个观点，我们可以对于不同的问题来定制特定的解决方案，用于达到精确地适应手头的问题的目的。这样，我们就不用将我们手中的问题进行转化以满足一些现有的算法的假设。MBML 框架的最大优势在于：相同的推理方法可以应用于大量的模型。因此，我们可以通过专注于模型的修改来改进我们的结果，而不用担心在不同的模型下怎么去计算我们关心的目标变量。运用 MBML 理念解决问题的例子包括 Microsoft Research 的 Infer.NET 框架、BUGS 和 Stan 等软件。

在本论文中，我们将重点关注推荐系统领域，处理并分析一个实际的数据集：Book-Crossing 数据集。我们的目标是预测用户对不同书籍的评分星级。从 MBML 的观点出发，我们克服了前面所提到的三个挑战，一步一步地得到了该

问题的解决方案。在第一部分中，我们建立了一个统计模型。该模型的复杂度随着样本量线性增长，并且在数据量大并且稀疏的 Book-Crossing 数据集上表现良好。我们发现：我们的解决方法实际上和已有的概率矩阵分解（probability matrix factorization，PMF）殊途同归。在第二部分中，我们解决了由于数据集的不均衡性所引起的所谓的冷启动问题（cold-start problem）。

关键词：低秩矩阵补全；基于模型的机器学习观点；推荐系统；概率矩阵分解；冷启动问题

Abstract

Low-rank matrix approximation (LRMA) refers to methods that represent data as low-rank matrices, with application in various fields including recommender systems, natural language processing, signal processing, and many others. In many applications, we aim to solve the LRMA problem when many observations are missing. It is called the low-rank matrix completion (LRMC) and is extremely useful for recommender systems. There are several challenges in LRMC. First, the algorithm needs to scale up to a large number of observations in the high dimensional setting. Second, the observed data sets are often too sparse to have enough information. Third, the data sets are always imbalanced, namely, some rows or columns have a large number of observed entries, while some might possess very few observed elements.

Model-based machine learning (MBML) is a perspective that views all machine learning algorithms as models plus inference methods. A model consists of a set of explicit assumptions about the problem domain, expressed in a precise statistical language. Inference methods refer to computational processes that take the data and compute targets of interest. With this point of view, we can seek to create a bespoke solution tailored to precisely fit the problem at hand, instead of transforming our problem to suit some existing algorithms. The greatest strength of MBML is that the same inference method can be applied to a large number of models, therefore we can improve our results by focusing on the revision of the model without any concerns about computing the outcomes. Examples of the MBML philosophy include Infer.NET framework at Microsoft Research, BUGS, and Stan.

In this thesis, we will focus on the Book-Crossing data set in the field of recom-

mender systems. The goal is to predict the customers' preferences for different books. We overcome all the challenges mentioned earlier and get our solution step by step from the MBML standpoint. In the first part, we build up a statistical model that scales linearly with the number of observations and performs well on the large, sparse Book-Crossing data set. Our solution actually corresponds to an existing method: probability matrix factorization (PMF), different paths with the same destination. In the second part, we solve a so-called cold-start problem caused by the imbalanced data set.

Key Words: low-rank matrix completion; model-based machine learning; recommender systems; probability matrix factorization; cold-start problem

目 录

第一章　绪论

数据是作为信息来源而收集的测量或观察结果。我们用数据点表示关于研究对象一个实体所收集的所有感兴趣的指标。我们用一个向量 $d_i \in \mathcal{R}^M$ 来表示一个数据点。其中，M 是这个向量的维度，即我们对该实体所感兴趣的指标的个数；下标 i 用于标识不同的实体。数据集是数据点的全体。我们用一个矩阵 $D = [d_1, d_2, ..., d_N]$ 来表示有 N 个数据点的数据集。就拿本文将要分析的数据集为例：我们想研究用户对不同书籍的喜好程度。研究对象的一个实体就是一个用户，一共有 N 个用户；我们所感兴趣的指标是他们对于 M 本不同书籍的评分，于是，d_n 的每一维就表示第 n 个用户对不同的书籍的评分；数据集就可以由 N 个列向量组成的“书籍-用户矩阵” $D = [d_1, d_2, ..., d_N]$ 来表示，我们简称为数据矩阵。

在实际问题中，由于测量误差或者噪声的存在，我们得到的数据集并不是我们真正关心的量。所以，我们通常采用以下方式：$D = X + E$。其中，E 表示测量误差或者噪声；X 是我们真正想要得到的量。还拿“书籍-用户矩阵”来作为说明：D 是我们最终得到的评分结果，X 才是真实的用户对于书籍的喜好程度，E 是在收集评分过程中存在的一些误差。那在这个例子中，E 是由什么引起的呢？Savage (1971) 指出：在调查受试者对于某物品的喜好时，所有的受试者都报告或以其他方式表明，他们不明确知道自己的偏好; 他们都有过在对不同物品的喜好中动摇或者优柔寡断的经历。换言之，用户对书籍的评分本身就是模糊的：很难有理由相信，对《三国演义》评分为 8 分的张三就比对《三国演义》评分为 7 分的李四更喜欢《三国演义》。另外一个可能的误差来源是：尽管受试者在某个试验中表现出对某事物的偏好，但是他可能在另外一些情况下彻底改变对该事物的喜好程度 (Savage, 1971)。例如：张三在经受一系列挂科的折磨之后，心

情十分低落，买了一本心灵治愈小说《摆渡人》。张三读完《摆渡人》之后心灵如久旱逢甘霖般被滋润，立刻给了这本书 10 星好评，但当他人生一帆风顺时再读《摆渡人》顿觉索然无味。

想要直接从数据中恢复出我们所关心的量通常来说是很困难的，因此，我们需要对 X 做一些假设。数据分析的目的就在于“通过现象看本质”：通过对 X 做一些假设，对 D 进行分析，得到我们真正想要的结果。假设 X 是一个低秩矩阵，随之而来的就是一类得到 X 的办法，这类方法就是 LMRA。在书本喜好程度的例子中，这一假设是基于这样一个事实：有相似品味的读者对同一类型的书籍的喜好程度高度相关。有了这一假设，一种通常的做法是将问题用如下数学语言表达：

$$min_X \frac{1}{2}\|D - X\|_F^2,\ s.t.\ rank(X) \leqslant r \tag{1.1}$$

其中，$\|\cdot\|_F$ 表示 Frobenius 范数。优化问题 1.1 的解可以通过对 D 做奇异值分解来得到 (Eckart et al., 1936)，并且有许多很好的理论性质。但是，事情并没有这么简单，数据矩阵 D 中通常会有缺失值，这便是我们后面要介绍的低秩矩阵补全（low-rank matrix completion，LRMC）问题。

第一节　LRMC 和面临的挑战

低秩矩阵近似（Low-rank matrix approximation，LRMA）是通过低秩矩阵 X 来逼近数据矩阵 D 的一类方法。他是机器学习中核心概念之一，有着广泛的应用，诸如自然语言处理、信号处理、数据降维等 (Markovsky, 2012)[chapter 1]。有很多标准来衡量用 X 来近似 D 的好坏程度，其中残差平方和是最自然的一种，最终得到的优化问题如式 1.1 所示。

当只有数据矩阵 D 的一个子集被观测到的时候，这就成了 LRMA 问题的一个拓展：低秩矩阵补全（low-rank matrix completion，LRMC）问题 (Candès et al.,

2009; Mazumder et al., 2010; Nguyen et al., 2019)。LRMC 试图利用数据矩阵 D 的部分观测值来得到 X。为了将其表达成一个优化问题，我们沿用 Candès et al. (2010) 的记号。用 $\Omega \in \{1, ..., M\} \times \{1, ..., N\}$ 表示数据矩阵 D 中可以观察到的元素的下标全体。定义投影算子 $P_\Omega : \mathcal{R}^{M\times N} \to \mathcal{R}^{M\times N}$：

$$(P_\Omega(D))_{ij} = \begin{cases} 0 & (i,j) \notin \Omega \\ D_{ij} & (i,j) \in \Omega \end{cases}$$

从而 LMRC 问题就可以表达成如下优化问题：

$$min_X \frac{1}{2}\|P_\Omega(D - X)\|_F^2, \, s.t. \, rank(X) \leqslant r \tag{1.2}$$

优化问题 1.2 相比于优化问题 1.1 少考虑了观测不到元素的残差平方和，但是问题却更加复杂。Fazel (2002) 指出直接求解优化问题 1.2 是 NP-hard 的。

Zhou et al. (2014) 将求解优化问题 1.2 的算法归纳为两大类：最小化秩方法和矩阵分解方法。

最小化秩方法通常采用一些凸近似，Mazumder et al. (2010); Candès et al. (2010) 进行了如下近似：

$$min_X \frac{1}{2}\|P_\Omega(D - X)\|_F^2 + \lambda\|X\|_* \tag{1.3}$$

其中，$\|\cdot\|_*$ 是矩阵的 nuclear norm，Fazel (2002) 证明了它是矩阵秩的 tightest convex surrogate。有很多优化算法去解决优化问题 1.3，它们大都基于 Cai et al. (2010) 提出的 singular value thresholding (SVT) 算子和定理。计算 SVT 的时候需要奇异值分解，当矩阵的维数很高时时间开销非常大。

矩阵分解方法则是考虑用两个低秩矩阵的乘积来代替优化问题 1.2 中的 X (Rennie et al., 2005)。可以证明：任何秩小于 K 的矩阵 $X \in \mathcal{R}^{M\times N}$ 可以表达成 $X = A^T B$，其中 $A \in \mathcal{R}^{K\times M}$，$B \in \mathcal{R}^{K\times N}$。因此，优化问题 1.2 可以表达成如下的 bi-convex 问题：

$$min_{A\in\mathcal{R}^{K\times M},B\in\mathcal{R}^{K\times N}}\frac{1}{2}\|P_\Omega(D-A^TB)\|_F^2$$

这一优化问题在数据矩阵元素缺失较多的时候是个病态的，一个通常的做法是加上关于 A,B 惩罚项，这就是 Srebro et al. (2004) 提出的形式：

$$min_{A,B}\frac{1}{2}\|P_\Omega(D-A^TB)\|_F^2+\frac{\lambda}{2}(\|A\|_F^2+\|B\|_F^2) \tag{1.4}$$

Srebro et al. (2004) 还证明了：当 $K>r$ 时，该问题与优化问题 1.3 等价。但是采用矩阵分解的方式，可以不用计算奇异值分解，大大减小了时间上的开销。而且在矩阵维数特别高的时候，采用含有奇异值分解的优化算法几乎是不可行的 (Tuzhilina et al., 2021)。

运用 LRMC 的一个著名的例子就是 “Netflix” 竞赛，它的目标是基于用户对已经看过的电影的评分来构建一个电影推荐系统 (Feuerverger et al., 2012)。这一数据集有三个特点：

- 数据维数高：有大约 48 万用户，大约一万八千部电影；数据量大：有超过 10 亿个评分。

- 数据矩阵观测到的元素是稀疏的：大约只有 1.2% 的元素是已知的。

- 数据不均衡：平均上每个用户评了大概两百部电影，但是有些用户对超过一千部电影评过分，而有些用户评分数少于五部电影。

在第二章中可以看到：本文将要分析的数据集也有类似的特点。这些特点给书本推荐系统的构建带来了一些挑战，本文的核心就是围绕如何解决这些挑战展开。

第二节　基于模型的机器学习方法

机器学习算法是计算机所遵循的用于解决一些机器学习问题的一系列指令。基于模型的机器学习方法（Model-based machine learning，MBML）(John et al., 2019) 是一种可以为不同的实际问题构建量身定制的算法的通用方法。MBML 将一切机器学习算法看成由两部分组成：模型和推理方法。其中，模型是指关于领域问题的所有的假设，需要明确地表达出来，通常会有严格的数学表示；推理方法指的是训练模型用的方法，是利用模型输入数据得到我们想要的目标变量的计算过程。以深度学习解决物体检测问题为例：关于图片中物体一些性质的假设就包含在 Krizhevsky et al. (2012) 提出的神经网络构架中。神经网络的架构就是模型，蕴含了对实际问题的一些直观，只不过没有明确的用语言表达出来；而训练网络用的随机梯度下降法就是推理方法。

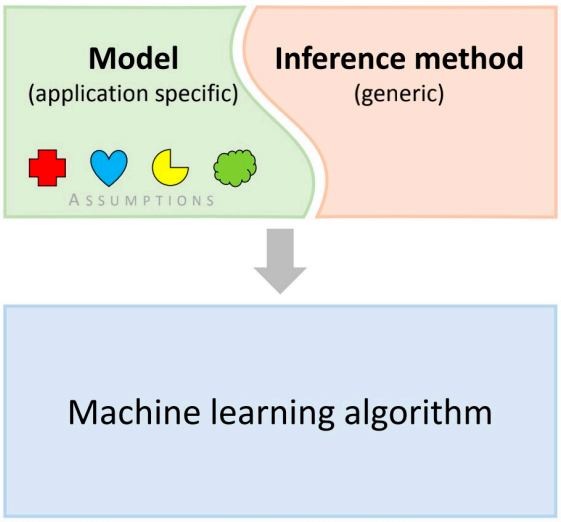

图 1.1　MBML 观点下的机器学习算法 (图片来自 John et al. (2019))

图 1.1 是 MBML 观点下的机器学习算法示意图。我们可以把机器学习算法视作一个输入是数据、输出是目标变量的大盒子。MBML 就是把这个盒子剖分成两部分来处理：模型和推理方法。图中模型那部分的不同形状的图案就代表与特定应用相关的不同假设。保持推理方法不变，修改模型，即修改算法的假设，

我们就可以产生不同的机器学习算法。

机器学习的算法总会做出一些假设，而算法有没有效果、效果好不好，当然在某些意义上取决于应用它的时候数据和这些假设的吻合程度。MBML 的观点就是突出算法的假设，将这些都囊括到模型的部分，并显示地表达出来。这样一来，一方面我们可以更容易地根据手头上问题的特点和数据的需求去选择合适的现有的算法以达到好的效果；另一方面，我们甚至可以从数据的特点和问题的需求出发，提出一系列与之符合较好的假设，再权衡合适推理方法，从而构建出新的效果好的算法。采用 MBML 观点来解决问题会有如下几个优点：

- 假设被明确地写出来：能够让我们更好地理解算法为什么有效，更容易地发现哪些假设可以修改使得更符合手头的问题。
- 一种推理方法可以运用到很多不同的模型：修改模型并不需要改动推理方法，从而可以只专注于模型里假设的修改以提高算法效果。
- 对于同一个模型，可以选择不同的推理方法，从而可以更好地权衡算法的速度与性能。
- 如果用 Infer.NET (Minka et al., 2014) 等采用 MBML 方法的框架，我们只需要修改模型，整个算法的代码就会自动生成，大大减少了工作量。

MBML 是本文构建书本推荐系统的核心工具，我们将用这一观点一步一步地提出需要的假设构建模型（参见第三章）。本文主要关心的是模型部分，不去考虑选用不同的推理方法上的优劣，也不去考虑如何实现自动生成代码问题。

第三节　概率矩阵分解

概率矩阵分解（probability matrix factorization，PMF）是从概率的角度解释 LRMC 中的矩阵分解方法（参见本章第一节），是由 Salakhutdinov et al. (2007) 提

出的。为方便说明，我们以书本推荐系统为例，仍采用之前的记号：$D \in \mathcal{R}^{M\times N}$ 是数据矩阵，$A \in \mathcal{R}^{K\times M}, B \in \mathcal{R}^{K\times N}$，$A$ 每一列都是一本书籍的潜在特征向量，B 每一列都是一个用户的潜在特征向量。PMF 的概率图如图 1.2 所示。首先，定义在已观测到的评分上的条件分布：

$$p(D|A,B,\sigma^2) = \prod_{n=1}^{N}\prod_{m=1}^{M}\left[\mathcal{N}(d_{mn}|a_m^T b_n,\sigma^2)\right]^{I_{mn}}$$

其中，$\mathcal{N}(x|\mu,\sigma^2)$ 是均值为 μ，方差为 σ^2 的正态分布的概率密度函数；I_{mn} 是示性函数：如果第 n 个用户给第 m 本书评过分，那么它就为 1，否则它为 0。然后，再对书本特征向量和用户特征向量引入高斯先验分布：

$$p(A|\sigma_a^2) = \prod_{m=1}^{M}\mathcal{N}(a_m|0,\sigma_a^2\mathbf{I}),\ p(B|\sigma_a^2) = \prod_{n=1}^{N}\mathcal{N}(b_n|0,\sigma_b^2\mathbf{I})$$

通常把 $\sigma^2,\sigma_a^2,\sigma_b^2$ 视作给定的参数。于是，关于书本和用户特征的对数后验为：

$$\begin{aligned}\log p(A,B|D,\sigma^2,\sigma_a^2,\sigma_b^2) = &-\frac{1}{2\sigma^2}\|P_\Omega(D-A^TB)\|_F^2 - \frac{1}{2\sigma_a^2}\|A\|_F^2 - \frac{1}{2\sigma_b^2}\|B\|_F^2\\ &-\frac{1}{2}\left(\left(\sum_{m=1}^{M}\sum_{n=1}^{N}I_{mn}\right)\log\sigma^2 + M\,K\log\sigma_a^2 + N\,K\log\sigma_b^2\right) + C\end{aligned}$$

其中，C 是与参数和变量无关的常数。如果把 A,B 视作参数，对该模型采用 maximum a posteriori (MAP) 估计，最大化上述对数后验等价于最小化下面的损失函数：

$$E = \frac{1}{2}\|P_\Omega(D-A^TB)\|_F^2 + \frac{1}{2}(\lambda_a\|A\|_F^2 + \lambda_b\|B\|_F^2) \tag{1.5}$$

此处，$\lambda_a = \frac{\sigma^2}{\sigma_a^2}$，$\lambda_b = \frac{\sigma^2}{\sigma_b^2}$。这与优化问题 1.4 极其类似，其中 $\lambda = \frac{\sigma^2}{\sigma_a^2} = \frac{\sigma^2}{\sigma_b^2}$，只不过 PMF 的损失函数中正则化系数可以不同。也就是说：PMF 恰好给出了 Srebro et al. (2004) 提出模型的一种概率解释。

如果用 MBML 的观点来看 PMF 的话：模型可以由图 1.2 所示的概率图模型来严格表示，我们不妨称之为 PMF 模型。它蕴含了对问题所作出的假设，比如：

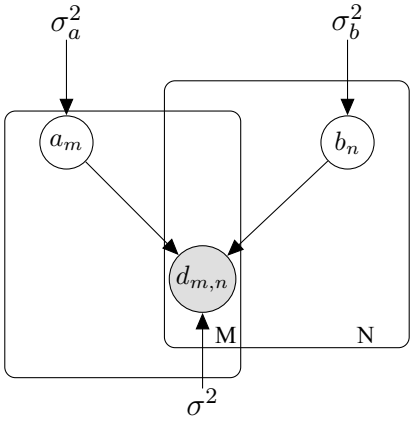

图 1.2　PMF 概率图

各个用户特征向量是独立的；各本书的特征向量也是独立的……只不过这些假设并没有显示地被写出来。如果我们采用上述的 MAP 估计，那么推理方法就是求解 MAP 估计的优化方法。当然，还可以选用其他的推理方法来求解 PMF 模型：Lim (2007) 采用变分贝叶斯的推理方法来求解；Salakhutdinov et al. (2008) 则用一种完全的贝叶斯观点来看待，采用 Markov chain Monte Carlo（MCMC）来求解。不同的推理方法的性能和速度直接各有权衡，因为本文主要注重模型的建立过程，所以在此就不多赘述。

在本文中，我们并不打算将 PMF 模型加上 MAP 推理方法抑或是其他推理方法直接用到 Book-Crossing 数据集 (Ziegler et al., 2005) 上。这是因为目前这些假设都是隐含的，即使运用之后效果好，我们也很难去理解这个算法为什么有效。如果效果不好，我们也很难基于现在的模型去改进。我们将采用 MBML 方法从头建立所有的假设，再进一步修改我们的假设来得到一个定制的效果比较好的算法。

第四节　主要工作和论文框架

本文的目的在于分析实际数据集——Book-Crossing 数据集 (Ziegler et al., 2005) 以构建一个书本推荐系统。用到的核心工具是 MBML（参见本章第二节），

主要关注模型的建立过程，力求从无到有、一步步地提出合理的假设来构建模型。我们发现：这样建立起来的模型实际上就是 PMF （参见本章第三节），也就是说我们找到了 PMF 的一种合理的解释。最后，我们利用 Book-Crossing 数据集 (Ziegler et al., 2005) 中关于用户和书籍的额外信息进一步修改模型，解决了由于数据集的不均衡引起的冷启动问题，我们将最终的模型称之为 warm-start Logistic PMF （WSLPMF）。所有的算法都是用 R 语言 (Team, 2014) 从头实现的，考虑到速度问题，采用了 Rcpp 包 (Eddelbuettel et al., 2011) 对显示的循环进行了加速。算法的实现、服务器上训练模型用的脚本、画图用的代码等等都已经放在 https://github.com/XinXU-USTC/undergraduate-dissertation。

本文框架大体如下：在第二章中，我们简要介绍 Book-Crossing 数据集 (Ziegler et al., 2005)，并对其做一些预处理。在第三章中，我们用 MBML 的观点建立模型，并选用 MAP 估计作为推理方法；我们在模拟生成的数据集上测试了算法的正确性，并将在实际数据集上训练出的结果和“直接用每本书的平均评分作为预测值”的方法（后面我们称这种用平均值预测的方法为 book-avarage ）对比。在第四章中，我们着重解决由于数据的不均衡性所引起的冷启动问题（cold-start problem），即算法对于评分很少的用户预测不准确的问题。在第五章中，我们将进行总结并对算法可以改进的地方进行进一步思考。

第二章　Book-Crossing 数据集

第一节　简介

BookCrossing 社区是为了满足书籍爱好者在世界各地交换书籍、分享经验。Ziegler et al. (2005) 在这个社区里历时四周搜集到 Book-Crossing 数据集。该数据集主要包括三个文件：用户评分，用户信息和书籍信息。图 2.1 中分别展示了这三个文件的前六行：第一行左侧图片显示的是用户评分的信息；第一行右侧的图片显示的是用户相关的信息；第二行的图片显示的是书本相关的信息。为方便起见，以后我们将这三部分信息分别叫做：评分数据集（对应于数据矩阵 D 中可观测到的元素），用户数据和书籍数据。我们将在后面的三节中分别详细介绍这三部分数据。

	User.ID	ISBN	Book.Rating
1	276725	034545104X	0
2	276726	0155061224	5
3	276727	0446520802	0
4	276729	052165615X	3
5	276729	0521795028	6
6	276733	2080674722	0

	User.ID	Location	Age
1	1	nyc, new york, ...	*NA*
2	2	stockton, calif...	18
3	3	moscow, yuko...	*NA*
4	4	porto, v.n.gaia...	17
5	5	farnborough, ...	*NA*
6	6	santa monica, ...	61

	ISBN	Book.Title	Book.Author	Year.Of.Publication	Publisher	Image.URL.S	Image.URL.M
1	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.am...	http://images
2	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.am...	http://images
3	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	http://images.am...	http://images
4	0374157065	Flu: The Story of the G...	Gina Bari Kolata	1999	Farrar Straus Giroux	http://images.am...	http://images
5	0393045218	The Mummies of Uru...	E. J. W. Barber	1999	W. W. Norton & Co...	http://images.am...	http://images
6	0399135782	The Kitchen God's Wife	Amy Tan	1991	Putnam Pub Group	http://images.am...	http://images

图 2.1　Book-Crossing 数据集中三个文件的部分示意

第二节　评分数据集

评分数据集每一行是一个（用户编号，书籍 ISBN 号，评分星级）三元组，分别标识了评分在数据矩阵 D 中的列号、行号和值。我们不妨以后称这个三元组为样本。评分数据集共有 1,149,780 个样本，涉及 105,283 个用户（涉及最大

的用户编号为 278,854 ）和 340,556 本书籍。

根据 Ziegler et al. (2005)，评分星级 0 表示用户并没有对书本评分，但是浏览过或者点击过这本书的介绍，即隐式的反馈。我们这里只考虑显式的反馈。去掉所有的 0 评分样本后得到拥有 433,671 个样本的数据集，涉及 77,805 个用户（涉及最大的用户编号仍为 278,854 ）和 185,973 本书籍。我们仍把去掉隐式反馈样本的数据集叫“评分数据集”。以后，“评分数据集”就是指只含显示反馈样本的数据集。我们画出了评分数据集上评分星级的条形图，如图 2.2 左侧图形所示。从图形中可以看出：评分星级为 $1-10$ 的整值。整个图形明显向右侧偏斜，表示用户更倾向于给书籍评高的星级。这一点也是比较符合实际情况的，因为人们只能对看过的书籍做出评分，而读者一般只看自己喜欢的书。

进一步地，我们如下划分训练集和验证集：对每个用户，70% 的样本用来训练，30% 的样本用来验证。同时，我们去除验证集中包含训练集未出现的书籍的所有样本。最终，得到拥有 334,954 个样本包含 153,859 本书籍的训练集和拥有 64,857 个样本包含 30,550 本书籍的验证集。以后，将训练集和验证集中的样本分别称为训练样本和验证样本。

类似地，我们画出了训练集和验证集上评分星级的条形图，分别如图 2.2 中间和右侧图形所示。从图 2.2 中可以看出，训练集和验证集上评分星级的条形图都与整个评分数据集上评分星级的条形图形状相仿。这说明：我们划分训练集和验证集并未改变样本在不同评分星级上的占比，这样的划分是合理的。

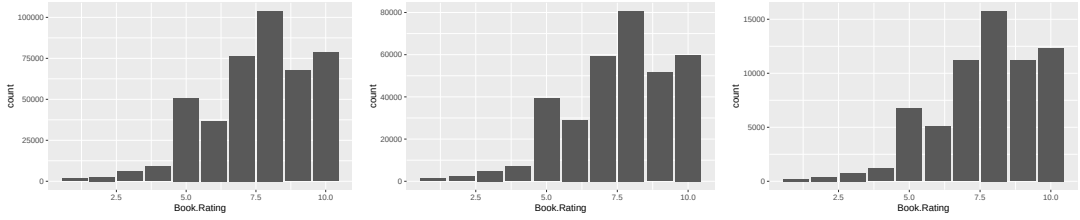

图 2.2　评分星级条形图: 左侧为整个评分数据集; 中间为训练集；右侧为验证集

第三节　用户数据

用户数据每一行包括：用户编号、用户位置和用户年龄。用户数据中共包含 278,858 条用户信息。其中，共有 57,339 个不同的地点。由于每个地点平均只有不到六个用户，并且里面存在大量的乱码，因此我们后面不考虑用户的位置信息。为方便，在第四章中，我们仅考虑使用用户的年龄作为额外信息。

用户的年龄中存在约 39.7% 的缺失值，去除缺失值后做出用户年龄的条形图如图 2.3 左侧图形所示。可以看出：年龄为三十岁左右的用户居多，用户年龄大概集中在 $10-70$ 之间。但是其中存在一些异常数据：有一些超过一百岁的用户，在零岁附近有一个小的峰值。异常数据大概占 0.46%。我们直接将这些异常数据当成缺失数据处理，得到的新的条形图如 2.3 右侧图形所示。顺带一提，我们当然可以有更优雅的方法处理异常数据，但是考虑到这些异常数据占比很小，所以直接把它们当成缺失数据处理也是无伤大雅的。

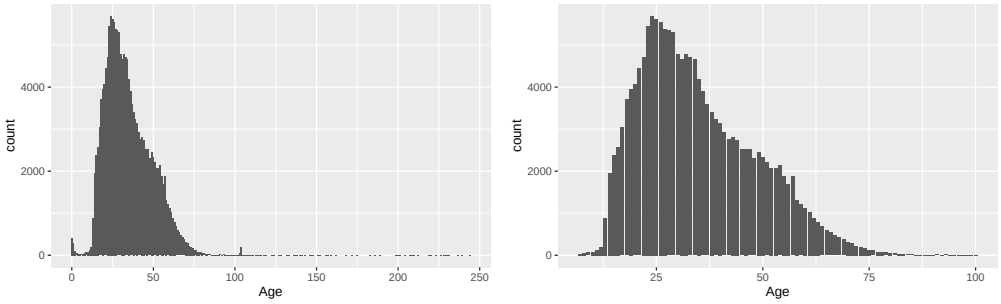

图 2.3　用户年龄条形图：左侧为原始数据；右侧为去除异常值后的数据

第四节　书籍数据

书籍数据每一行包括：书籍的 ISBN 号，名称，作者，出版年份，出版社和图片链接。书籍数据中共包含 271,360 本书籍的信息，其中涉及 242,135 个书籍名称、102,024 位作者、16,808 个出版社和 117 个不同的出版年份。类似地，为方便，在第四章中，我们仅考虑使用书籍的出版年份作为额外信息。

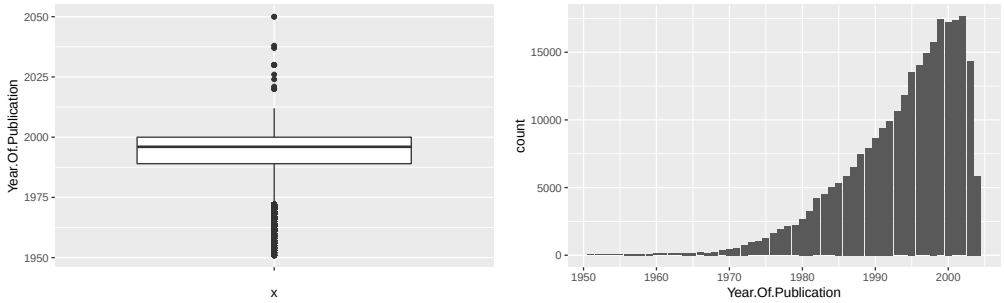

图 2.4　书籍出版年份示意图：左侧为原始数据的箱图；右侧为除去异常值的条形图

书籍数据的出版年份大约有 1.7% 的缺失值。图 2.4 左侧图形是书籍出版年份的箱图，为了更清晰的显示，图中仅展示出版年份在 1950 年以后的书籍。从图中可以看出：大部分书籍的出版年份大概在二十世纪九十年代。因为 Ziegler et al. (2005) 收集到的数据是截止到 2004 年，所以出版年份在其之后的为异常值。大约有 0.026% 的异常数据，去除它们之后书籍出版年份的条形图如 2.4 右侧图形所示。图中也仅展示出版年份在 1950 年以后的书籍，可以看出：2000 年左右出版的书籍是最多的。

第三章　构建书本推荐系统

第一节　简介

本章主要运用第一章第二节中的 MBML 方法处理第二章第二节中的评分数据集，将完整地展现如何一步步提出合理的假设来构建书本推荐系统。本章结构大体如下：我们会在第二节中详细介绍模型的建立过程，指出与 PMF 模型的关联性（参见第一章第三节）；在第三节中出算法实现的细节；第四节展示如何在模拟数据集上测试算法和代码的正确性；在第五节中给出在评分数据集的效果。

第二节　用 MBML 构建算法

一、刻画书本和用户

首先，我们需要考虑如何去刻画书本。一个直观的想法是：书本是有不同的特点对的，比方说：这本书到底是偏现实一些的作品还是偏虚幻的呢？抑或是：这本书到底是感性偏多还是理性占主要成分呢？每一对这样的特点可以用一个随机变量来刻画，而每一本书在这一对特点上的表现就是该随机变量的一次实现，即一个实数。这个实数的符号指示了这本书具有特点对中的哪一个特点，它的绝对值大小则表达该书具有的相应特点的强烈程度。比如：用随机变量 X 来表示“虚幻-现实”特点对，如果某本书在这对特点上的表现是 x，那么 $x > 0$ 意味着这本书是现实的，$|x|$ 值越大这本书就越现实；$x < 0$ 意味着这本书是虚幻的，$|x|$ 值越大这本书就越虚幻。假设我们想去挖掘 K 对这样的特点对，于是可以用一个 K 维随机向量 $A_* = (A_{*1}, ..., A_{*K})$ 来表示这些特点对。我们把这个随机向量 A_* 称为是书本潜在特征，称它的取值范围为书本潜在特征空间。于是，每一本书对应于书本潜在特征的一个实现，即书本潜在特征空间中的一个点。

我们可以用类似的想法对用户的喜好做出假设。对于之前提到的特点对，我

们可以用一个随机变量来刻画用户对这一对特点的喜好，而每一个用户的喜好就是该随机变量的一次实现，即一个实数。这个实数的符号指示了用户更喜欢特点对中的哪一个特点，它的绝对值大小则表达用户对相应特点的喜爱程度。比如：用随机变量 Y 来表示用户对“虚幻-现实”特点对的喜好，如果某个用户的喜好是 y，那么 $y > 0$ 意味着用户喜欢现实的书籍，$|y|$ 值越大表示用户越喜欢现实类型的书；$y < 0$ 意味着用户喜欢虚幻的书籍，$|y|$ 值越大表示用户越喜欢虚幻类型的书。相应于书本潜在特征 A_*，我们也可以得到用户潜在特征 $B_* = (B_{*1}, ..., B_{*K})$，并称它的取值范围为用户潜在特征空间。于是，每一个用户对应于用户潜在特征的一个实现，即用户潜在特征空间中的一个点。

到目前为止，为了刻画书本和用户，我们已经提出了两条假设：

1. 书籍可以由书本潜在特征来刻画，每一本书对应于书本潜在特征空间中的一个点。

2. 用户喜好可以由用户潜在特征来刻画，每一个用户对应于用户潜在特征空间中的一个点。

其中，第一条假设看起来比较合理，因为理论上我们可以让书本潜在特征的维度足够大，使得它足以刻画不同的书籍。在书本潜在特征的维度比较小的时候，第一条假设可能不那么完美，但也是够用的。在第二条假设中，我们认为一个读者的品味可以由用户潜在特征空间中单一的一个点描述，但有些时候情况并非如此：有些读者既喜欢看儿童读物又喜欢看一些充满暴力血腥的书籍，这时候，这位读者可能同时占据了用户潜在特征空间多个点。但是，我们可以认为这类读者比较少，因此第二条假设也比较合理。这两条假设可以用图 3.1 中第一行左侧的因子图来表达。以后，“用户”、“读者”既可以指实体也可以指由实体抽象出的用户潜在特征空间中的一个点；“书”、“书本”、“书籍”既可以指实体也可以指由实体抽象出的书本潜在特征空间中的一个点。具体含义视语境而定，若有混淆，

我们会加以说明。

为了更好地阐释书本特征和用户特征，我们用一个编造的例子来阐释。在图 3.2 中，我们给出了一个二维的潜在特征空间：横轴的正负半轴分别描述“现实”和“虚幻”，纵轴的正负半轴分别描述“感性”和“理性”。我们在这个潜在特征空间中标注了四大名著和三位读者。书籍在书本潜在特征空间中的位置标识了书籍的特点，比如：《三国演义》在第四象限，说明它是一本充满理性的、比较现实的书。还能注意到一些书比较靠近坐标轴，则它们对某些特点对是没有表现出倾向性，或者说：很难根据某些特点对来将这本书分类。比方说，我们很难划定《西游记》到底是一本理性的书还是感性的书。类似地，用户在用户潜在特征空间的位置则标识了他的品味：王五在第四象限，说明他喜欢看理性的、现实的书；而李四则喜欢虚幻的书，但对理性或者感性的书不显示出任何的偏好。图 3.2 还能给我们一些直观的想法：当一个读者和一本书在潜在特征空间的位置特别靠近的时候，是否能说明他很可能可以给这本书一个高的评分星级呢？也就是说，王五是不是很可能会喜欢《三国演义》呢？对这个问题的思考自然地启发我们对“用户与书本关系”做出进一步的假设，后面我们会对此详细地展开。

图 3.2 还能给出第一条假设中关于书本潜在特征空间维度作用的一些直观感受。如果我们只考虑“虚幻-现实”这一对特点，则我们很难将《三国演义》和《水浒传》区分开来。类似地，如果只考虑“理性-感性”这一对特点，则很难将《红楼梦》和《水浒传》区分开来。如果同时考虑这两对特点，则能很好的区分四大名著在潜在特征空间的位置。也就是说：增加书本潜在特征空间的维度可以更好地刻画不同书籍的特点。这一点在后续实验部分（本章第五节）会得到证实。

二、刻画书本和用户的关联

假设我们有一本书和一个用户，分别对应于书本潜在特征 A_* 和用户潜在特征 B_* 的一次实现：$a = (a_1, ..., a_K)$, $b = (b_1, ..., b_K)$，那我们需要怎么去衡量他们

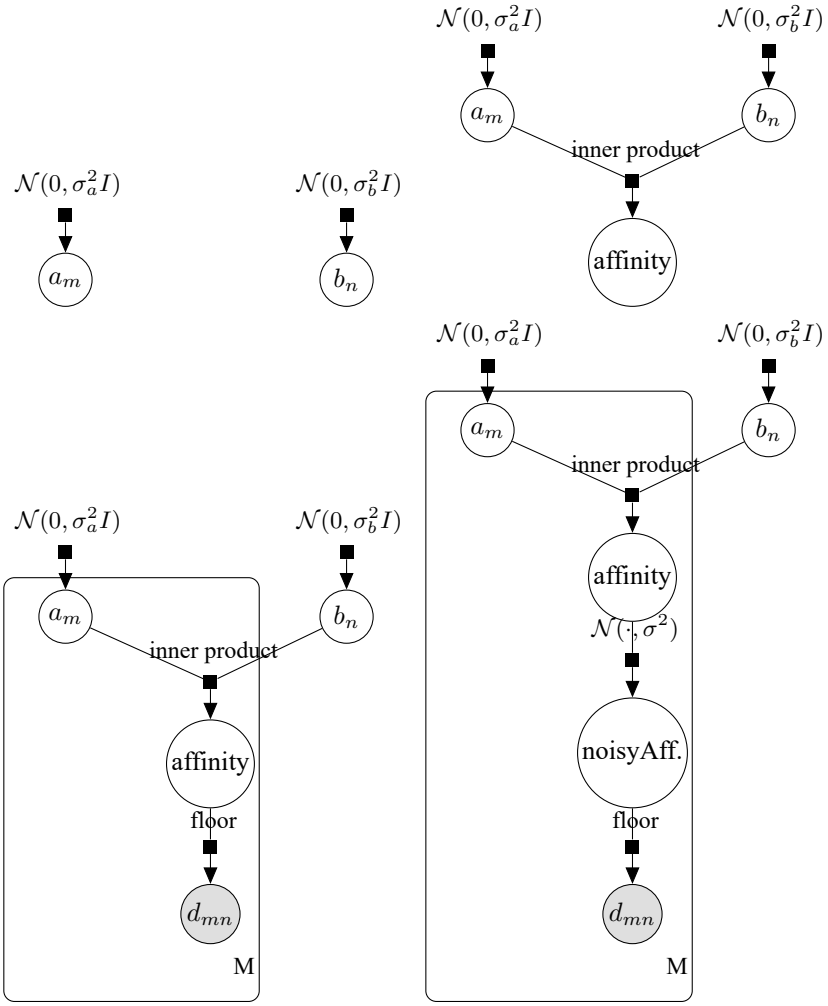

图 3.1　模型构建过程示意图

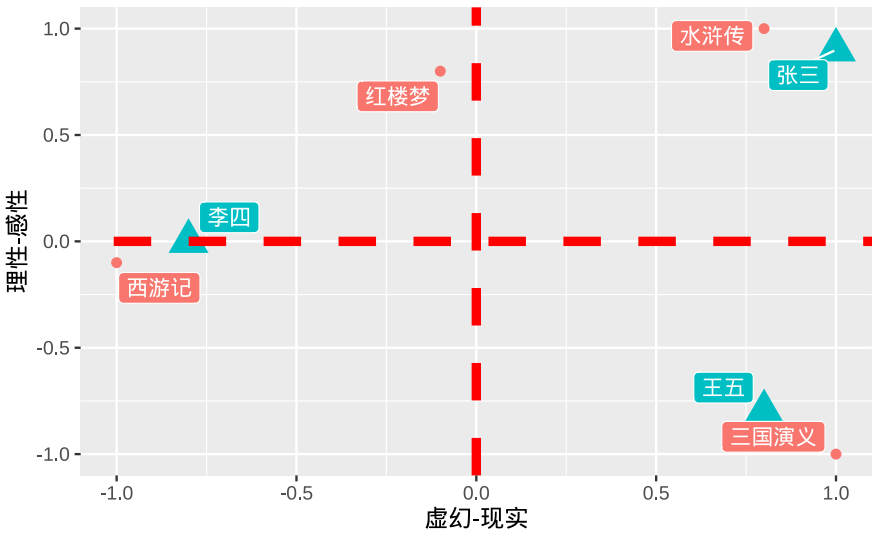

图 3.2　潜在特征空间示意图

之间的匹配度呢？图 3.2 给了我们一个直观：用 a 和 b 在潜在特征空间的位置关系来度量，距离越近匹配度越高。但由于书本潜在特征空间和用户潜在特征空间尺度不一样，用距离来度量书本和用户的关系不太合理。况且，我们前两条假设里还蕴涵着这样的关系：固定 a 的某个分量 a_k，且不妨 $a_k > 0$, 当 $b_k > 0$ 且绝对值增大，那么这个用户对这本书在第 k 个特点对上的匹配程度就越好；当 $b_k < 0$ 且绝对值增大，那么这个用户对这本书在第 k 个特点对上的匹配程度就越差。这种关系恰可以由逐分量乘积 $a \cdot b = (a_1 b_1, ..., a_K b_K)$ 来表达。

于是，我们可以用书本 a 和用户 b 的逐分量乘积 $a \cdot b = (a_1 b_1, ..., a_K b_K)$ 来表达书本和用户在各个特点对上的匹配程度。我们称之为“逐分量亲和度”。从逐分量亲和度可以看出每一对特点上用户和书籍的匹配度。如果进一步假设逐分量亲和度的各分量之间互不影响，那么逐分量亲和度的各分量之和 $\sum_{k=1}^{K} a_k b_k$ 就可以反应用户对书本的匹配度，我们称之为“总亲和度”。

为了刻画用户和书本的关联性，我们又新提出了一条假设：

3. 用户和书本在各对特点上的匹配程度可以由逐分量亲和度描述，且逐分量亲和度各分量间互不影响，于是总亲和度可以刻画用户和书本的匹配程度。

其中，“逐分量亲和度各分量间互不影响”在实际情况中可能并不成立。比如：某些读者可能喜欢第一人称叙事的科幻小说，但是讨厌第一人称叙事的历史小说。于是，“第一人称-第三人称”和“科幻-历史”这两对特点会互相影响。但注意到图 3.1 第一行左侧的图形中我们引入的正态先验协方差矩阵是对角阵，这说明：我们已经将互相关联的特点对排除在考虑范围之内了。于是，这第三条假设也是相对合理的。其实，“总亲和度”就是用户和书本的内积，我们可以用图 3.1 第一行右侧的因子图来表达前三条假设。

三、从总亲和度到评分星级

有了总亲和度，为了得到评分星级，一个自然的假设就是：

4. 用户对书本的总亲和度越高，那么他就越可能给这本书更高的评分星级。

我们怎么去达到这个假设的要求呢？第二章我们提到评分星级范围是 $1-10$ 内所有整值。于是，一个简单的做法是直接将总亲和度映射到评分星级：

$$\text{star} = \begin{cases} 1 & \text{affinity} < 1 \\ floor(\text{affinity}) & 1 \leqslant \text{affinity} \leqslant 10 \\ 10 & \text{affinity} > 10 \end{cases} \tag{3.1}$$

有了这个映射之后，对于一个给定的用户，我们可以把包含他的样本拿来训练。于是，对于他没评过分的书籍，利用训练好的模型和推理方法就可以进行预测。这样所得到的模型可以由图 3.1 第二行左侧的因子图来表达。

看起来我们现在已经得到了模型和它的严格的数学表现形式，但实际上图 3.1 第二行左侧的因子图所并没有很准确地表达我们的模型。原因在于该因子图的模型确定性太大：用户和书本之间的总亲和度越高，那么用户对该书的评分星级就越高（这与我们的第四条假设还是有一些差别的）。当我们的训练样本都满足这一条件的时候，这一因子图所表达的模型才能被训练出来。但实际上，会存在一些样本在该因子图所表达的模型下的生成概率是零，即按照该因子图所表达的模型生成数据的方式，有些样本没有办法被生成。举一个极端的例子来说明：我们有两本凡尔纳科幻名著系列里的书，一本是《神秘岛》，另一本是《海底两万里》。这两本书在书本潜在特征空间中位置非常接近，但是张三对《神秘岛》打了 10 分，给《海底两万里》打了 1 分。此时，用户潜在特征空间的任何一点都无法满足要求。在考虑到数据本身存在的噪声（参见第一章），所以图 3.1 第二行左侧的因子图在训练的时候很容易崩溃。

一种解决方案是：在得到总亲和度之后我们先加上一些噪声，再将加噪声之后的总亲和度利用式 3.1 映射到评分星级。于是得到图 3.1 第二行右侧的因子图。由于高斯噪声的存在，用户和书本总亲和度高的不是一定就评分星级高。我们可

以控制高斯噪声的方差来使得：用户和书本之间的总亲和度越高，那么用户对该书的评分星级就越可能高。于是，这样的因子图能更好地抽象出第四条假设。

四、协同过滤对

如果我们按照图 3.1 第二行右侧的因子图，每次对一个用户来训练模型，那我们将会学习到该用户所评过分的书籍的潜在特征。由于我们的评分数据集相对于数据矩阵 D 来说是稀疏的，所以，每次对一个用户来训练的数据利用率很低。注意到，从四条假设里可以看出：书本潜在特征并不会因为用户的不同而改变。因此，我们可以将所有的用户放在一起考虑，这样可以至少为很多书籍提供了一些数据。这种方法叫做协同过滤对（Collaborative Filtering，CF）(Goldberg et al., 1992)，得到的模型可以用如图 3.3 左侧因子图来表示。在第一个推荐系统 Tapestry 中，CF 是用来处理邮件文件的，Goldberg et al. (1992) 解释了这个术语的来源：“people collaborate to help one another perfoem filtering by recording their reactions to documents they read”。在我们这里，就是运用其他用户对某一书本的评分来辅助对某一用户对该书籍评分的预测，这样可以大大提高数据的利用率，解决了摘要部分提到的第二个挑战：数据矩阵 D 可观测到的元素是稀疏的，算法需要有效地利用样本。

五、完整的算法

目前，我们已经展现了如何通过直观一步步建立起模型，即如何提出的一系列假设。我们将这些假设总结到表 3.1 中，其中我们还加入了一条假设。简单来说，这条假设说的是：凡是前四条假设没涉及到的内容都对我们的研究问题没有影响。这一般是所有模型都隐含的一条假设，因为模型是对现实问题的简化，总会有取舍。前文中已经对前四条假设的合理性做出了分析，同样地，我们也需要讨论一下第五条假设。实际上，读者是否喜欢某本书可能还与一些其他因素有

关，比如：看书的时间，看书时的感情状态等。Zhang et al. (2015) 就考虑过将一些其他的因素加入电影推荐系统中。本文就不去过多纠结其他因素的影响，只是指出在这里模型是可以进一步改进的。容易发现假设中的不足并可以进一步改进模型，这也是运用 MBML 方法的好处之一。

表 3.1　PMF 模型——一些假设

1. 书籍可以由书本潜在特征来刻画，每一本书对应于书本潜在特征空间中的一个点。
2. 用户喜好可以由用户潜在特征来刻画，每一个用户对应于用户潜在特征空间中的一个点。
3. 用户和书本在各对特点上的匹配程度可以由逐分量亲和度描述，且逐分量亲和度各分量间互不影响，于是总亲和度可以刻画用户和书本的匹配程度。
4. 用户对书本的总亲和度越高，那么他就越可能给这本书更高的评分星级。
5. 用户对书籍的评分星级的高低之于书本潜在特征有关，与其他的东西无关。

如果我们将图 3.3 左侧因子图转化成概率图的话，那么结果恰好就是图 1.2。可以说：我们用 MBML 方法发掘出了 PMF 概率图所隐含的所有假设，并显示地表达出来了。这个模型和 Salakhutdinov et al. (2007) 提出的 PMF 模型的数学表达形式完全一致，只不过在 MBML 的观点下，模型偏重于表 3.1 中的五条假设。我们不妨把用 MBML 方法建立起来的模型也叫做 PMF 模型。总的来说，PMF 模型就是表 3.1 中的五条假设，它的数学表达形式如图 3.3 左侧因子图或者图 1.2 中的概率图所示。

算法可以分解为模型和推理方法（参见第一章第二节）。为简便，推理方法采用 MAP 估计，与第一章第三节介绍的完全相同。后面我们称 PMF 模型加上

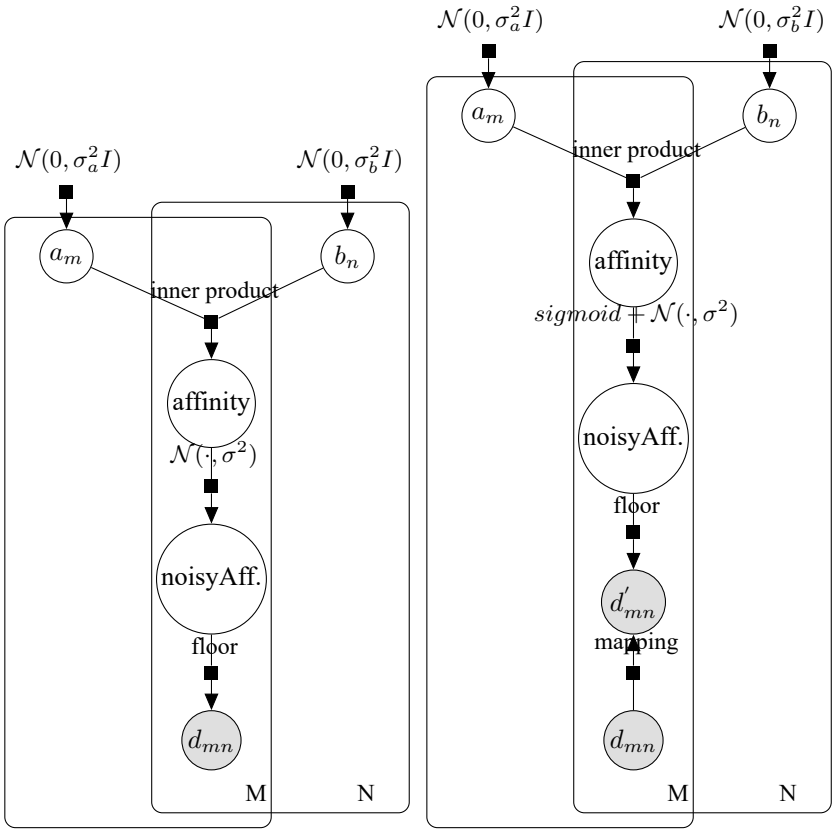

图 3.3　完整的因子图：左侧为 PMF 因子图；右侧为 LPMF 因子图

MAP 估计为 PMF 算法。

六、评分“溢出”

前文中，为了满足表 3.1 中第四条假设，我们将总亲和度加上噪声后送入式 3.1。由于我们对总亲和度在 1 和 10 之外进行了截断，这样会损失一部分信息：总亲和度加噪声后为 0 和为 -1，最后预测出来的评分星级都是 1，但两者书本和用户的匹配程度是不一样的。于是，这种简单的处理方式并不能很好地表达第四条假设，对模型的准确率会产生一定的影响。问题在于加噪声后总亲和度的范围比评分星级的范围大，我们不妨把它叫做评分“溢出”问题。

为解决评分“溢出”问题，一个自然的想法是把总亲和度和评分星级做一个变换，使得它们范围一致。我们把总亲和度通过 sigmoid 函数 $\sigma(x) = \dfrac{1}{1+e^{-x}}$ 变换到 $[0,1]$ 区间；再把 i 评分星级变换为 $\dfrac{i-1}{10-1}$, $i = 1, 2, ..., 10$。最后的因子图是

图 3.3 右侧图形，与 Salakhutdinov et al. (2007) 提出的 Logistic PMF （LPMF）是一样的。

值得一提的是，图 3.3 中的两个因子图都可以作为表 3.1 中模型的数学表达形式，只不过两者在第四条假设的表示上有所区别。为了加以区分，我们把表 3.1 中的假设连同图 3.3 中的左侧因子图叫做 PMF 模型；把表 3.1 中的假设连同图 3.3 中的右侧因子图叫做 LPMF 模型。它们加上 MAP 的推理方法后分别称为 PMF 算法和 LPMF 算法。在本章第五节我们将看到：LPMF 算法的预测效果确实比 PMF 算法好。

第三节　算法实现细节

本节将详细介绍算法实现细节。我们本节以 LPMF 算法为例进行说明，将 PMF 算法实现的细节放在附录 A 中。LPMF 算法是对模型采用 MAP 估计，对于固定的 $\sigma^2, \sigma_a^2, \sigma_b^2$，只需优化如下损失函数：

$$E = \frac{1}{2}\|P_\Omega(D - \sigma(A^T B))\|_F^2 + \frac{1}{2}(\lambda_a \|A\|_F^2 + \lambda_b \|B\|_F^2) \tag{3.2}$$

此处，$\lambda_a = \frac{\sigma^2}{\sigma_a^2}$，$\lambda_b = \frac{\sigma^2}{\sigma_b^2}$；$\sigma(x) = \frac{1}{1+e^{-x}}$ 是逐分量的。我们采用 stochastic gradient descent with momentum (Rumelhart et al., 1986) 对其进行优化，算法框架如算法 3.1 所示。

用得到的数据是评分数据集的训练集，需要输入固定的参数：$\sigma^2, \sigma_a^2, \sigma_b^2$，最后的输出是书本潜在特征矩阵 $A = [a_1, ..., a_M]$ 和用户潜在特征矩阵 $B = [b_1, ..., b_N]$。在初始化时，将 A, B 进行零初始化；并给它们的每一列一个相同的小初始动量，具体地：从标准差为 0.01 均值为零的正态分布中抽取。当损失函数收敛或者达到预先设定的最大迭代次数 $max.iteration$ 时算法停止，收敛准则为：损失函数的相对变化 $\Delta = |\frac{E-E'}{E'}| < \epsilon$，其中 E 是当前的损失函数值，E' 是上一步的损失函数值，ϵ 是一个很小的常值。

Data: User/Book/Rating training set

Input: $\sigma^2, \sigma_a^2, \sigma_b^2$

Output: A, B

initialization;

while $\#iterations \leqslant max.iteration$ **do**

  for *each batch* **do**

    calculate gradient of A, B;

    update A, B;

    evaluate loss E;

  end

end

算法 3.1: LPMF 算法框架

每一次循环中，我们要对每一个 batch 进行处理。我们首先需要计算损失函数 E 关于 A, B 的梯度：

$$\frac{\partial E}{\partial a_m} = \sum_{n=1}^{N}\sum_{m=1}^{M} I_{mn}(\sigma(a_m^T b_n) - D_{mn})\sigma(a_m^T b_n)(1-\sigma(a_m^T b_n))b_n + \lambda_a a_m$$

$$\frac{\partial E}{\partial b_n} = \sum_{n=1}^{N}\sum_{m=1}^{M} I_{mn}(\sigma(a_m^T b_n) - D_{mn})\sigma(a_m^T b_n)(1-\sigma(a_m^T b_n))a_m + \lambda_b b_n$$

其中，I_{mn} 为一个示性函数，当训练集中存在第 n 个用户对第 m 本书的评分星级时，它取 1；否则，它的值为 0。于是，对于每一个 batch，其梯度计算方法如算法 3.2 所示。有了当前 A, B 的梯度 $grad_A, grad_B$ 后，我们通过下式更新 A, B：

$$V_A = \mu V_A + t * grad_A$$

$$V_B = \mu V_B + t * grad_B$$

$$A = A - V_A$$

$$B = B - V_B$$

我们称 μ 为 decay factor，称 t 为步长。

可以看出计算梯度、计算损失函数时时间复杂度与训练样本数量成线性关系。即在算法 3.1 中第 4 行和第 6 行都是 $\Theta(\#training\,examples)$，于是整个算法的时间复杂度也是与训练样本数量成线性关系的。由于训练集比较大，直接在 R (Team, 2014) 中用显示循环比较耗时。在实现过程中，凡是需要遍历整个训练集所有样本的部分我们都用 Rcpp (Eddelbuettel et al., 2011) 进行加速。至此，我们解决了摘要部分提到的第一个挑战：数据集包含大量的样本，我们的算法需要对样本数有很好的可扩展性。

Input: User/Book/Rating training set, A, B, λ_a, λ_b, batch size:$\#batch$, number of training examples: nrow(train)

Output: gradient of A, B : $grad_A, grad_B$

1 $grad_A = \lambda_a * A * nrow(train)/\#batch$;

2 $grad_B = \lambda_b * B * nrow(train)/\#batch$;

3 **for** *each training example:* (m, n, r) **do**

4 $\quad g = \frac{1}{1+e^{-a_m^T b_n}}$;

5 $\quad grad_{a_m} += g * (1-g) * (g-r) * b_n$;

6 $\quad grad_{b_n} += g * (1-g) * (g-r) * a_m$;

7 **end**

算法 3.2: LPMF 梯度计算的伪代码

有了训练结果之后，我们可以采用算法 3.3 对整个验证集进行预测并评估算法的预测准确度。我们用 mean absolute error （MAE）和 root mean square error（RMSE）来评估预测的准确性。类似地，我们也用 Rcpp (Eddelbuettel et al., 2011) 进行了加速。

Input: User/Book/Rating validation set, A, B

Output: predicted stars; MAE; RMSE

for *each validation example:* (m, n) **do**

  $predictied\,star = floor(9 * a_m^T b_n)$;

end

calculate MAE, RMSE;

算法 3.3: LPMF 预测的伪代码

第四节　测试算法正确性-模拟数据集

为了测试算法和代码的正确性，我们将在一个小型的模拟数据集上进行实验。我们通过如下方式产生数据：从标准正态分布中抽取 $A \in \mathcal{R}^{K\times M}, B \in \mathcal{R}^{K\times N}$，并令 $M = \sigma(A^T B) + E$，其中 E 从 $\mathcal{N}(0, 0.01^2)$ 中抽取，$\sigma(x) = \frac{1}{1+e^{-x}}$ 是 sigmoid 函数。通过如下映射，得到模拟的评分数据矩阵 D：

$$D_{mn} = \begin{cases} 1 & M_{mn} < 0 \\ 1 + floor(9 * M_{mn}) & 0 \leqslant M_{mn} \leqslant 1 \\ 10 & M_{mn} > 1 \end{cases}$$

然后我们抽取 D 中 10% 的元素作为观测数据，称之为模拟评分数据集。将模拟评分数据集中的 70% 的样本作为训练集，30% 的样本作为验证集，并分别称之为模拟训练集和模拟验证集。从图 3.4 可以看出模拟训练集与模拟验证集上的评分星级的条形图形状类似，所以划分是合理的。

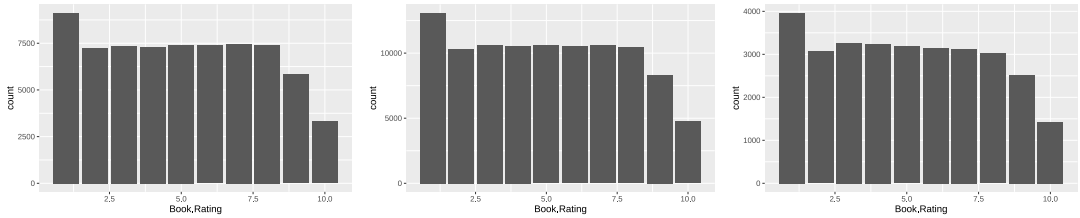

图 3.4　评分星级条形图：左侧为模拟训练集；中间为模拟评分数据集；右侧为模拟验证集

我们固定潜在特征空间维度 $K = 4$，将模拟训练集分别喂入 PMF 算法和

LPMF 算法，并在模拟训练集上计算损失函数 E，在模拟验证集上记录预测评分星级的 MAE 和 RMSE。结果如图 3.5 所示。从左到右纵坐标依次为：MAE、对数损失函数、RMSE。图中的横坐标都表示 Epoch 数，算法 3.1 中的 3 – 7 行循环一次就是一个 Epoch。这里，我们运行 80 个 Epoch 就终止算法。图 3.5 中左侧图形和右侧图形中的红色水平虚线为直接用每本书评分星级的平均值作为预测的相应指标（后面我们称用书本评分星级平均值作为预测值的方法为 book-average）。可以看出：PMF 算法和 LPMF 算法在 MAE 和 RMSE 的标准下对 book-average 都有一定的提高。这说明代码没有问题，算法基本也是正确的。我们还可以注意到：图 3.5 中间图形两个损失函数下降的趋势和左右两侧的图形上 MAE 和 RMSE 的下降趋势大致相同。这说明了选取式 1.5 和式 3.2 作为损失函数的合理性，即采用 MAP 推理方法的正确性。

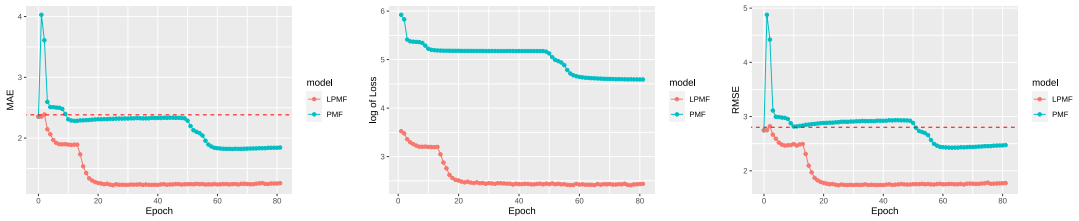

图 3.5　模拟数据集算法表现

另外，从图 3.5 两侧图形中可以看出：LPMF 算法在模拟数据集上的表现比 PMF 算法要好：预测得更准确，并且达到一个较好的预测结果所用的 Epoch 数更少。这个结果是自然的，因为模拟数据集的产生机制是按照图 3.3 右侧的因子图进行 ancestral sampling (Bishop et al., 2006)[Chapter 10]。需要注意的是，这个结果还不能说明在评分数据集上 LPMF 算法表现更好，尽管我们在第二章中通过理论分析得出：LPMF 算法能更好地反应假设。

第五节　实验结果-实际数据集

一、实验设定

对于 stochastic gradient descent with momentum，我们固定 decay facotr $\mu = 0.9$，每一个 batch 包含 30,000 个训练样本。至于步长，PMF 算法中设定 $t = 0.001$，LPMF 算法中设定 $t = 0.3$。我们可以通过调整潜在特征空间的维度 K 和式 1.5、式 3.2 中的 λ_a, λ_b 来控制模型复杂度。对于潜在特征空间维度 K 的讨论我们将在第四小节进行。至于 λ_a, λ_b，我们通过调参来选定。最终，PMF 算法中选定 $\lambda_a = \lambda_b = 3 \times 10^{-6}$；LPMF 算法中选定 $\lambda_a = \lambda_b = 1 \times 10^{-5}$。在调参过程中，我们发现：当 λ_a, λ_b 在最终选定的值附近（相差十倍范围左右）变动时，算法最终结果几乎不会改变；但当 λ_a, λ_b 离最终选定的值很远时，算法最终会陷入“常值预测”的困境，结果很差。

二、PMF v.s. LPMF

我们固定潜在特征空间维度 $K = 2$，按照第一小节中的设定，将训练集分别喂入 PMF 算法和 LPMF 算法，并在训练集上计算损失函数 E，在验证集上记录预测评分星级的 MAE 和 RMSE。我们运行 50 个 Epoch 后终止算法并保留 MAE 上表现最好的模型。结果如图 3.6 所示。从左到右依次为：MAE-Epoch 图、对数损失函数的相对变化-Epoch 图、RMSE-Epoch 图。其中，对数损失函数的相对变化为 $log\,\Delta = log\,|\frac{E-E'}{E'}|$，$E$ 是当前损失函数值，E' 是上一个 Epoch 损失函数值。

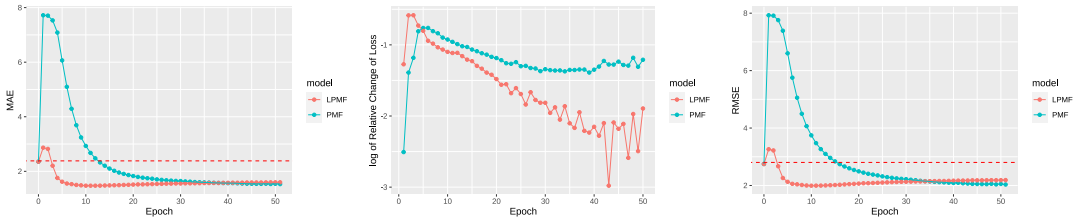

图 3.6　Book-Crossing 数据集算法表现（$K = 2$）

图 3.6 中左侧图形和右侧图形中的红色水平虚线为利用 book-average 的相应

指标。可见，PMF 算法和 LPMF 算法在 MAE 和 RMSE 的标准下对 book-average 都有一定的提高。PMF 算法对 book-average 在 MAE 和 RMSE 上分别提高了 3.4% 和 0.33%；LPMF 算法则分别提高了 7.1% 和 1.6%。正如第 二 节所分析的那样，LPMF 算法更贴合模型的假设，表现更好。而且从图 3.6 中间图形的趋势可以看出：LPMF 算法将更快地收敛。我们后面主要考虑对 LPMF 算法的改进。

三、实验结果解释

训练完成后，我们可以得到书本潜在特征矩阵 $A = [a_1, ..., a_M]$，每一列表示一本书。由本章第二节的讨论，我们可以知道：这些列向量的一个分量表示书本在某一对特点对上的倾向性。但是，我们并不知道每一个分量到底表示什么含义，这也是“潜在”的意义所在。当潜在特征空间的维度 $K = 2$ 时，我们可以很容易地将书本潜在特征矩阵 A 的某些列向量可视化，然后对其每一个分量进行解释。为此，我们选出了六本书作为例子来进行说明：《Little Altars Everywhere》、《Waiting to Exhale》、《Adressat unbekannt》、《Henry der held》、《Soul Survivor》和《Auf der Suche nach dem verlorenen Gl$\ddot{u}$ck》[①]。图 3.7 画出了它们在潜在特征空间中的位置。根据这几本书的特点，我们可以做出如下解释：横纵坐标分别表示“现代-历史”和“理性-感性”这两个特点对。

我们还可以注意到图 3.7 第四象限中的三本书非常地靠近，在附录 B 中可以看到它们有着类似的特点，粗糙地讲，这三本书都有真实的历史背景，利用主角的故事来阐明一定的信念或道理。这说明：学习到的书本潜在特征矩阵 A 可以反应书本之间的相似性。更具体地，书本在潜在特征空间很靠近则它们就比较相似。这对于做一些 item-specific 的推荐很有用，比如在淘宝上经常看到的“买了这件物品的人还喜欢……”。

评分数据集相对于整个数据矩阵 D 来说是稀疏的，所以大部分书籍拥有的

[①]我们将对这六本书的简要介绍放在附录 B 中

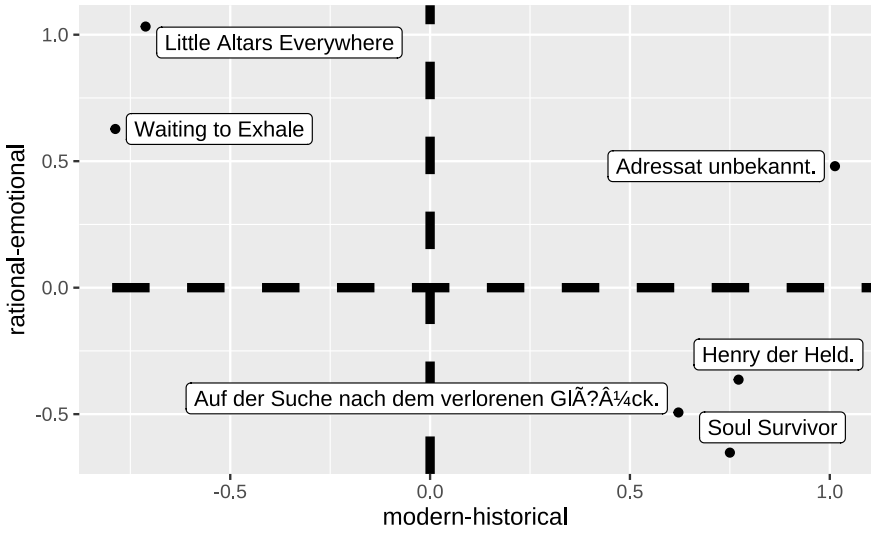

图 3.7　书本潜在特征示例（$K = 2$）

评分数量很少。拥有不同评分数量书籍的数目条形图如图 3.8 右侧图形所示。为了更清晰地展示，我们只画出了拥有 $3 - 30$ 个评分的书本。可以看到：绝大多数书本拥有的评分数目小于 10。这对我们最终得到的书本潜在特征矩阵 A 有什么影响？我们先从理论上进行分析：算法 3.2 中，LPMF 算法在计算书本潜在特征矩阵 A 的梯度的时候，训练样本 (m, n, r) 只对 a_m 的梯度有贡献。对于除了 a_m 之外的列向量，在算法 3.1 中更新时起主要作用的是 V_A。由于我们对 V_A 的初始值很小（参见本章第三节），所以训练样本中涉及到的次数较少的书本几乎不被更新，它们每一个分量的值就接近零（初始化时的值）。于是，大部分书本在潜在特征空间中到原点的距离应该很小，而且随着书本拥有评分数目的增多，到原点的距离大体上会增大。为了查看实验结果是否与理论分析一致，我们按照书本拥有的评分数目对其进行分组，并在组内求到原点的平均距离，得到如图 3.8 右侧所示的散点图。在图中，横纵坐标大致呈现出正相关性，这与理论分析一致。由对称性可知：用户潜在特征矩阵 B 也是类似的，得到的图形如图 3.9 所示。

直观上，一本书拥有的评分数目越少，或者一个用户评过书籍的数目很少，我们对它潜在特征向量的估计越不准确。换言之，我们对 MAP 估计的结果越没把握。其实，我们刚刚在用书本或者用户在潜在特征空间到原点的距离来衡量对

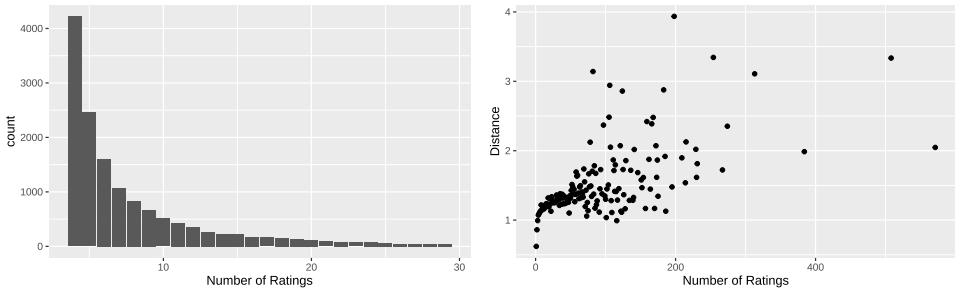

图 3.8　关于拥有不同评分数量的书籍的示意图：左侧为拥有不同评分数量的书籍的数目分布条形图；右侧为到原点平均距离关于评分数量的散点图

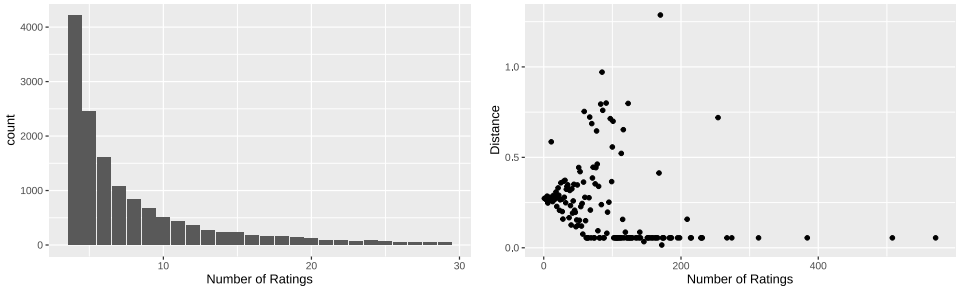

图 3.9　关于评过不同数量书籍的用户的示意图：左侧为评过不同数量书籍的用户分布条形图；右侧为到原点平均距离关于评分数量的散点图

MAP 估计的把握程度。但这种刻画并不准确，因为在图 3.8 右侧散点图中，拥有评分很多的书籍的到原点的平均距离有时候比拥有评分比较少的书本还要小。这一现象在图 3.9 右侧散点图更为明显。这是因为有些书籍在某些特点对上本身就没有倾向性，有些读者对书籍的某些特点对没有偏好。很自然地，对估计的把握程度可以由后验方差来度量。遗憾的是，我们所采用的推理方法没有办法给出后验方差。对此，可以改用其他的推理方法，我们将在第五章中进一步讨论。

四、潜在特征空间维度

我们在第一小节中提到，可以通过调整潜在特征空间的维度 K 来控制模型的复杂度。在这一小节中，我们将在 $K = 2, 4, 8, 16$ 时分别进行训练，以研究潜在特征空间的维度 K 对实验结果的影响。对每一个不同的维度，我们都是采用算法 3.1 进行训练，保留在 MAE 指标下性能最好的结果。其中，实验参数的设

定如第一小节。我们在图 3.10 左侧和右侧折线图中分别展示了 MAE 和 RMSE 随潜在特征空间的维度 K 的变化。可以看出：随着潜在特征空间的维度 K 的增加，算法最终的效果总体趋势是在变好，这也与我们在本章第二节对表 3.1 中第一条假设的分析是相符的。

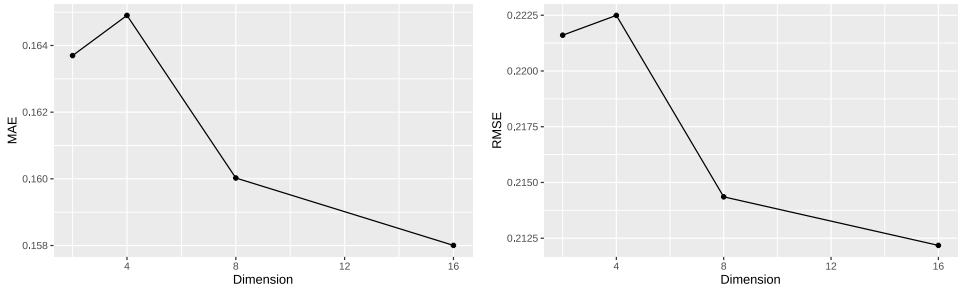

图 3.10　算法在潜在特征空间维度不同时的表现

第四章　冷启动问题

第一节　简介

从图 3.8 和图 3.9 的左侧条形图可以看出：很多书籍拥有的评分数量很少，很多用户评过的书籍的数量也很少。在第三章第二节第三小节中提到：对于这些书籍和用户，我们对书本潜在特征矩阵 A 和用户潜在特征矩阵 B 的估计把握程度很小。也就是说：我们很难确切地知道它们在潜在特征空间中的位置。这自然地会对预测结果产生影响。

为了说明这一点，我们将用户按照评过的书籍数量进行分组，一共分为九组，各组用户的评分数量为：$4-6$，$7-9$，$10-13$，$14-17$，$18-19$，$20-23$，$24-50$，$51-100$，>100。各用户组的用户数量如图 4.1 中间条形图所示。潜在特征空间维度 $K=2$ 下由 LPMF 算法训练出的结果在不同用户组预测表现如图 4.1 左侧和右侧图形所示。可以看出：对于评过书籍较少的用户，我们对他们的预测不准确。类似地，对于拥有评分数较少的书籍，我们对它们的预测也不准确。

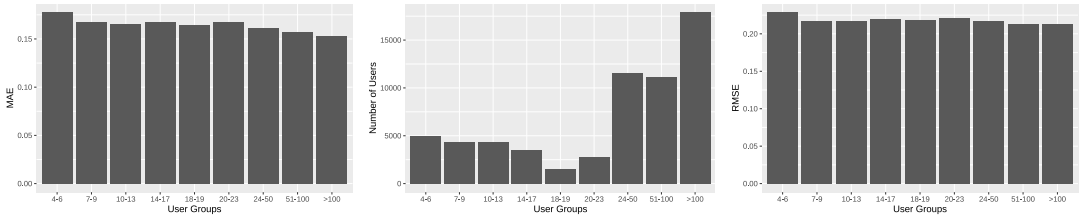

图 4.1　LPMF 在评过不同数量书籍的用户上的表现（$K=2$）

算法在训练样本涉及较少的对象上表现不好，我们把这一问题叫做冷启动问题（cold-start problem）。在实际中，这些在训练集中较少出现的对象往往更重要。比方说，在书本推荐系统中，训练集中出现较少的往往是新用户或者新出版的读物。对他们的预测不准确，可能会导致新用户的流失或者新出版读物的滞

销。在本章中，我们致力于解决冷启动问题。本章的结构大体如下：在第二节中，我们将利用第二章中提到的书本和用户额外信息给出解决方法：WSLPMF 算法。至于用 MBML 推导它的过程，我们放在附录 C 中；在第三节中，我们将会给出实验结果，并对结果进行分析。

第二节 WSLPMF

一、额外信息和特征

当书本潜在特征向量或用户潜在特征向量由于训练样本的缺乏而估计不准确时，我们可以考虑利用书本或用户的额外信息来辅助在潜在特征空间中定位书籍或用户。我们需要把额外信息转化为可以利用的形式，这就用到特征的概念。所谓特征，就是作用于原始数据得到可以利用的数据形式的函数，我们把特征作用于每一条原始数据的结果称为一个特征向量。把特征向量按照列拼凑可以得到书本特征矩阵和用户特征矩阵 $F_A = [f_{a_1}, ...f_{a_M}] \in \mathcal{R}^{L_A \times M}, F_B = [f_{b_1}, ...f_{b_N}] \in \mathcal{R}^{L_B \times N}$。其中，$L_A, L_B$ 分别是书本特征向量和用户特征向量的维度。

注意区分前文提到的“潜在特征向量”和这里“特征向量”的区别。潜在特征向量是我们需要估计的，是未知的。而特征向量是可以直接由数据经过计算得到的，通常是已知的（表 4.1假设六）。

二、WSLPMF 模型

有了特征后，我们可以在表 3.1 中加入几条额外假设将特征融入模型，得到如表 4.1 的模型。我们把它称作 warm-start LPMF（WSLPMF）模型。它可以由概率图 4.2 来表示。其中，$W_A = [W_{A_1}, ...W_{A_K}] \in \mathcal{R}^{L_A \times K}, W_B = [W_{B_1}, ...W_{B_K}] \in \mathcal{R}^{L_B \times K}$ 是表 4.1 第八条假设中提到的权重按列组成的权重矩阵。它们的先验分布为：

$$p(W_A|\sigma^2_{W_A}) = \prod_{k=1}^{K} \mathcal{N}(W_{A_k}|0, \sigma^2_{W_A}\mathbf{I}),\ p(W_B|\sigma^2_{W_B}) = \prod_{k=1}^{K} \mathcal{N}(W_{B_k}|0, \sigma^2_{W_B}\mathbf{I})$$

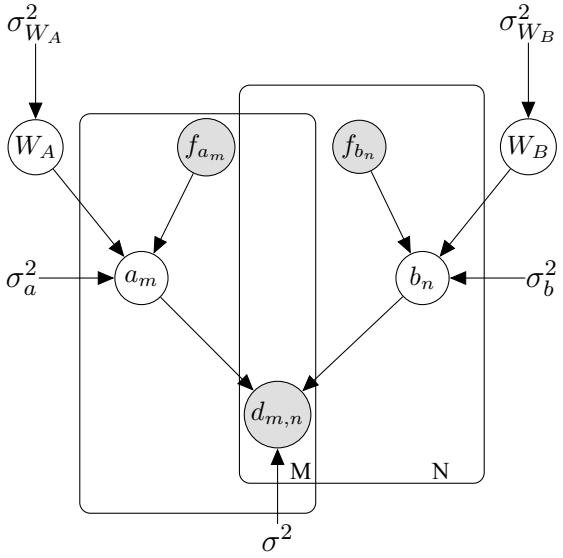

图 4.2　WSLPMF 概率图

定义在已观测到的评分上的条件分布仍为为:

$$p(D|A,B,\sigma^2)=\prod_{n=1}^{N}\prod_{m=1}^{M}\left[\mathcal{N}(d_{mn}|a_n^Tb_m,\sigma^2)\right]^{I_{mn}}$$

给定相应特征矩阵和权重矩阵后，书本和用户潜在特征矩阵有如下分布:

$$p(A|F_A,W_A,\sigma_a^2)=\prod_{m=1}^{M}\mathcal{N}(a_m|W_A^Tf_{a_m},\sigma_a^2\mathbf{I}),\ p(B|F_B,W_B,\sigma_b^2)=\prod_{n=1}^{N}\mathcal{N}(b_n|W_B^Tf_{b_n},\sigma_b^2\mathbf{I})$$

三、WSLPMF 算法

通常把 $\sigma^2,\sigma_a^2,\sigma_b^2,\sigma_{W_A}^2,\sigma_{W_B}^2$ 视作给定的参数。于是，关于书本和用户特征的对数后验为:

$$\begin{aligned}\log p(A,B|D,\sigma^2,\sigma_a^2,\sigma_b^2)=&-\frac{1}{2\sigma^2}\|P_\Omega(D-A^TB)\|_F^2-\frac{1}{2\sigma_a^2}\|A-W_A^TF_A\|_F^2\\&-\frac{1}{2\sigma_b^2}\|B-W_B^TF_B\|_F^2-\frac{1}{2\sigma_{W_A}^2}\|W_A\|_F^2-\frac{1}{2\sigma_{W_B}^2}\|W_B^T\|_F^2\\&-\frac{1}{2}\left(\left(\sum_{m=1}^{M}\sum_{n=1}^{N}I_{mn}\right)\log\sigma^2+M\,K\log\sigma_a^2+N\,K\log\sigma_b^2\right)\\&-\frac{1}{2}K\left(L_A\log\sigma_{W_A}^2+L_B\log\sigma_{W_B}^2\right)+C\end{aligned}$$

表 4.1　WSLPMF 模型

1. 书籍可以由书本潜在特征来刻画，每一本书对应于书本潜在特征空间中的一个点。

2. 用户喜好可以由用户潜在特征来刻画，每一个用户对应于用户潜在特征空间中的一个点。

3. 用户和书本在各对特点上的匹配程度可以由逐分量亲和度描述，且逐分量亲和度各分量间互不影响，于是总亲和度可以刻画用户和书本的匹配程度。

4. 用户对书本的总亲和度越高，那么他就越可能给这本书更高的评分星级。

5. 用户对书籍的评分星级的高低之于书本潜在特征有关，与其他的东西无关。

6. 对于所有的书本和用户，他们的特征向量总是可以计算的。

7. 通过特征可以预判书本和用户在潜在特征空间的大概位置。

8. 对于潜在特征空间的第 k 维，有一个权重 w_k 向量来表示特征每一个维度对潜在特征向量位置预判的贡献程度。当特征第 i 维增长 x 时，特征该维度的贡献值改变 $(w_k)_i \times x$，且特征每一个维度贡献互不影响。

9. 特征只通过假设 $6-8$ 影响相应的潜在特征向量，不以其他方式影响。

其中，C 是与参数和变量无关的常数。我们仍采用 MAP 估计，最大化上述对数后验等价于最小化下面的损失函数：

$$E = \frac{1}{2}\|P_\Omega(D - A^T B)\|_F^2 + \frac{1}{2}(\lambda_a\|A - W_A^T F_A\|_F^2 + \lambda_b\|B - W_B^T F_B\|_F^2 + \lambda_{W_A}\|W_A\|_F^2 + \lambda_{W_B}\|W_B\|_F^2)$$

此处，$\lambda_a = \frac{\sigma^2}{\sigma_a^2}$，$\lambda_b = \frac{\sigma^2}{\sigma_b^2}, \lambda_{W_A} = \frac{\sigma^2}{\sigma_{W_A}^2}, \lambda_{W_B} = \frac{\sigma^2}{\sigma_{W_B}^2}$。类似地，我们把 WSLPMF 算法加上 MAP 估计和优化方法称为 WSLPMF 算法，具体实现细节放在附录 D 中。

第三节　实验结果及分析

一、特征选取

在第二章我们介绍过：数据集还包含书本的出版年份和用户的年龄。我们可以利用这部分额外信息来提取特征。在这之前，我们首先需要确认额外信息有助于预测。直观上，不同年代的书籍有着不同的风格，不同年龄的用户有着不同的品味。为了进一步证实这一想法，我们对书籍和用户分别按照出版年份和年龄分组，然后对不同组的评分分别进行方差分析，得到的检验统计量分别为 7.334 和 26.35，p 值均小于 2×10^{-16}。这说明：有充分的证据表明，不同年份出版的书籍所获的平均评分有显著差异；不同年龄的用户平均评分也有显著差异。注意，我们这里并未去验证方差分析的假设是否满足，但这对我们确认额外信息对预测是否有帮助无伤大雅。因为我们最终关心的是加上额外信息修正后的算法结果是否有改善，即使不做方差分析，我们也可以直接加上额外信息去预测，然后观察实验结果是否有改善。

为了更直观地感受不同出版年份的书籍和不同年龄的用户评分的差异，我们分别选取了 $2000-2004$ 年出版的书籍和 $10, 20, ..., 50$ 岁的用户，将他们的平均评分和一倍标准差展示在图 4.3 中。左侧图形为不同年份出版书籍平均评分和一倍标准差示意图，右侧则为不同年龄的用户的图形。其中，右侧图形中横坐标

的括号内注明了不同用户组相应的用户数量。可以看出：书籍的平均评分随出版年份抑或用户的评价评分随年龄都不是线性增长的。因此，我们在构造特征时并不能将它们表示成数值变量，而是应该用属性变量刻画。另外，我们对所有书本和用户都引入了固定的 1 特征以捕获不同书本或用户的个体差异。对于出版年份有缺失的书籍和年龄有缺失的用户，我们只有固定的 1 特征，对于他们的估计就回到了 LPMF 算法的估计。

我们将一些含有数量较少书籍的组和含有用户较少的组进行了合并，最后对不同组进行编号。为了避免热独码带来的高维矩阵相乘，我们采用如下的特征向量：$f_{a_m} = [1, \text{age group index}], f_{b_n} = [1, \text{year group index}]$。其中，书本和用户特征向量都是二维的，第二个维度分别表示各自组的编号。当然，代码也需要一定的改动，详细代码参见https://github.com/XinXU-USTC/undergraduate-dissertation。

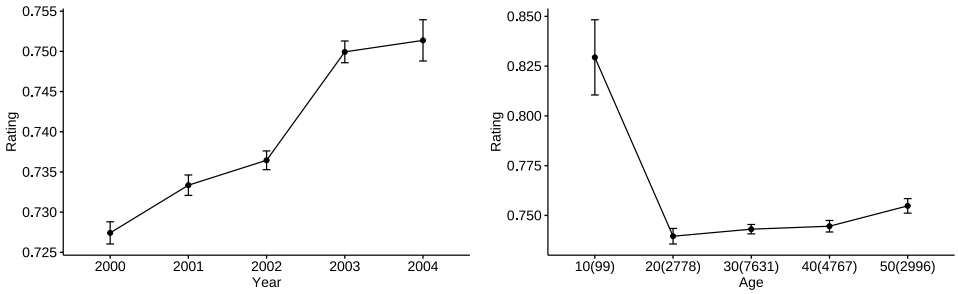

图 4.3　特征选取

二、实验设定

对于 stochastic gradient descent with momentum，我们固定 decay facotr $\mu = 0.9$，每一个 batch 包含 180,000 个训练样本。至于步长，我们选用 $t = 0.9$。至于 λ_a, λ_b，我们通过调参来选定。最终选定的参数为 $\lambda_a = \lambda_b = 1 \times 10^{-8}, \lambda_{W_A} = \lambda_{W_B} = 1 \times 10^{-11}$。与之前类似，在调参过程中我们发现：在我们最终选定的参数一定范围内，实验结果对参数的变化不敏感。

三、结果对比

我们在训练了 200 个 Epoch 时终止算法。最终的 MAE 和 RMSE 相比于 book-average 分别提高了 12.6% 和 10.0%; 相比于 LPMF 分别提高了 30.5% 和 8.6%。我们在评过不同数目书籍的用户组间比较 book-average、LPMF 和 WSLPMF，得到如图 4.4 所示的图形。可以看出：在每一组用户当中 WSLPMF 的效果都优于 LPMF，并且在评过较少书的用户组上 WSLPMF 有明显优势。

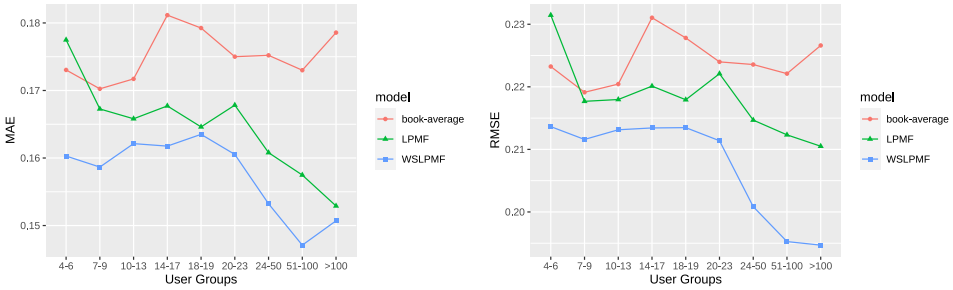

图 4.4　book-average、LPMF、WSLPMF 的对比（$K = 2$）

第五章　总结与展望

本文首先介绍了 LRMC 问题及其面临的挑战和一种用于定制算法的方法：MBML 方法。根据 Book-Crossing 数据集 (Ziegler et al., 2005) 来构建一个书本推荐系统便是一个 LRMC 问题。我们用 MBML 方法一步一步地构建了适用于该数据集的算法，发现它和现有的 PMF 算法和 LPMF 算法 (Salakhutdinov et al., 2007) 是一致的。我们详细介绍了它的实现细节和实验结果。后来，我们发现 LPMF 算法存在着冷启动问题。为此，我们再次运用 MBML 方法，成功地将书本和书籍的额外信息融入模型，构建了 WSLPMF 算法，具有较好的结果。

我们已经在第三章第二节和附录 C 中分别对 LPMF 模型和 WSLPMF 模型（即相应的假设：表 3.1 和表 4.1）的合理性做出了解释，同时还讨论过一些的改进可能性。此处，我们指出一些其它可以改进的地方：

- 选择更好的推理方法。在第三章第五节第三小节中我们提到：采用 MAP 估计并不能推断后验方差，不能给出对估计值的把握程度。为此，我们可以采用近似推断，像 Minka (2013) 提出的 Expectation Propagation；我们还可以用 MCMC 进行推理 (Salakhutdinov et al., 2008)。

- 改进刻画表 3.1 和表 4.1中第四条假设的方式。在第三章第二节中，从总亲和度到星级的映射是提前确定的。我们还可以假设事先不知道某星级对应哪一段总亲和度区间。第四条假设要求：星级越高对应的总亲和度区间在数轴上应该越靠右。于是，我们可以从训练集中学习出这个信息。相对于直接设定星级对应的总亲和度阈值，这种刻画更好些。

- 用一些其他的评判准则来衡量算法的表现。本文中我们一直用 MAE 和 RMSE 来衡量算法表现。但是对于推荐系统，预测准确性可能并不能满足用户需求 (Cosley et al., 2002; Herlocker et al., 2004)。

参 考 文 献

Savage L J. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 1971, 66(336): 783–801.

Eckart C, Young G M. The approximation of one matrix by another of lower rank. *Psychometrika*, 1936, 1(3): 211–218.

Markovsky I. Low rank approximation - algorithms, implementation, applications// Communications and Control Engineering: volume 906. Springer, 2012.

Candès E J, Recht B. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 2009, 9(6): 717–772.

Mazumder R, Hastie T J, Tibshirani R. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research : JMLR*, 2010, 11: 2287–2322.

Nguyen L T, Kim J, Shim B. Low-rank matrix completion: A contemporary survey. *IEEE Access*, 2019, 7: 94215–94237.

Candès E J, Tao T. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 2010, 56(5): 2053–2080.

Fazel M. Matrix rank minimization with applications. PhD thesis, Stanford University, 2002.

Zhou X, Yang C, Zhao H, et al. Low-rank modeling and its applications in image analysis. *ACM Computing Surveys (CSUR)*, 2014, 47(2): 1 – 33.

Cai J F, Candès E J, Shen Z. A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.*, 2010, 20(4): 1956–1982.

Rennie J D M, Srebro N. Fast maximum margin matrix factorization for collaborative prediction. *Proceedings of the 22nd international conference on Machine learning*, 2005: 713–719.

Srebro N, Rennie J D M, Jaakkola T. Maximum-margin matrix factorization//NIPS: volume 17. 2004.

Tuzhilina E, Hastie T J. Weighted low rank matrix approximation and acceleration. *ArXiv*, 2021, abs/2109.11057.

Feuerverger A, He Y, Khatri S. Statistical significance of the netflix challenge. *Statistical Science*, 2012, 27(2): 202–231.

John W, Christopher M B, Thomas D, et al. Model-based machine learning. 2019.

Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 2012, 60: 84 – 90.

Minka T, Winn J, Guiver J, et al. Infer.net 2.6. *Microsoft Research Cambridge*, 2014.

Salakhutdinov R, Mnih A. Probabilistic matrix factorization//NIPS: volume 20. 2007.

Lim Y J. Variational bayesian approach to movie rating prediction. *Proceedings of KDD Cup and Workshop*, 2007, 7: 15–21.

Salakhutdinov R, Mnih A. Bayesian probabilistic matrix factorization using markov chain monte carlo//ICML '08. 2008: 880–887.

Ziegler C N, McNee S M, Konstan J A, et al. Improving recommendation lists through topic diversification//WWW '05. 2005: 22–32.

Team R C. R: A language and environment for statistical computing. *MSOR connections*, 2014, 1.

Eddelbuettel D, François R. Rcpp: Seamless r and c++ integration. *Journal of Statistical Software*, 2011, 40: 1–18.

Goldberg D, Nichols D A, Oki B M, et al. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 1992, 35(12): 61–70.

Zhang C, Gartrell M, Minka T P, et al. Groupbox: A generative model for group recommendation. Microsoft Research, 2015.

Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors. *Nature*, 1986, 323(6088): 533–536.

Bishop C M, Nasrabadi N M. Pattern recognition and machine learning: volume 4. Springer, 2006.

Minka T P. Expectation propagation for approximate bayesian inference. *ArXiv*, 2013, abs/1301.2294.

Cosley D, Lawrence S, Pennock D M. Referee: An open framework for practical testing of recommender systems using researchindex//VLDB'02. Elsevier, 2002: 35–46.

Herlocker J L, Konstan J A, Terveen L G, et al. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 2004, 22(1): 5–53.

附录 A　PMF 算法

本附录将详细介绍 PMF 算法实现细节。PMF 算法是对模型采用 MAP 估计，对于固定的 $\sigma^2, \sigma_a^2, \sigma_b^2$，只需优化如下损失函数：

$$E = \frac{1}{2}\|P_\Omega(D - A^T B)\|_F^2 + \frac{1}{2}(\lambda_a \|A\|_F^2 + \lambda_b \|B\|_F^2)$$

此处，$\lambda_a = \frac{\sigma^2}{\sigma_a^2}$，$\lambda_b = \frac{\sigma^2}{\sigma_b^2}$。我们采用 stochastic gradient descent with momentum (Rumelhart et al., 1986) 对其进行优化，算法框架如算法 A.1 所示。

Data: User/Book/Rating training set
Input: $\sigma^2, \sigma_a^2, \sigma_b^2$
Output: A, B

```
initialization;
while #iterations ⩽ max.iteration do
    for each batch do
        calculate gradient of A, B;
        update A, B;
        evaluate loss E;
    end
end
```

算法 A.1: PMF 算法框架

用得到的数据是评分数据集的训练集，需要输入固定的参数：$\sigma^2, \sigma_a^2, \sigma_b^2$，最后的输出是书本潜在特征矩阵 $A = [a_1, ..., a_M]$ 和用户潜在特征矩阵 $B = [b_1, ..., b_N]$。在初始化时，将 A, B 进行零初始化；并给它们的每一列一个相同的小初始动量，具体地：从标准差为 0.01 均值为零的正态分布中抽取。当损失函数收敛或者达到预先设定的最大迭代次数 $max.iteration$ 时算法停止，收敛准则为：损失函数的相对变化 $\Delta = |\frac{E-E'}{E'}| < \epsilon$，其中 E 是当前的损失函数值，E'

是上一步的损失函数值，ϵ 是一个很小的常值。

每一次循环中，我们要对每一个 batch 进行处理。我们首先需要计算损失函数 E 关于 A, B 的梯度：

$$\frac{\partial E}{\partial a_m} = \sum_{n=1}^{N}\sum_{m=1}^{M} I_{mn}(a_m^T b_n - D_{mn})b_n + \lambda_a a_m$$

$$\frac{\partial E}{\partial b_n} = \sum_{n=1}^{N}\sum_{m=1}^{M} I_{mn}(a_m^T b_n - D_{mn})a_m + \lambda_b b_n$$

其中，I_{mn} 为一个示性函数，当训练集中存在第 n 个用户对第 m 本书的评分星级时，它取 1；否则，它的值为 0。于是，对于每一个 batch，其梯度计算方法如算法 A.2 所示。有了当前 A, B 的梯度 $grad_A, grad_B$ 后，我们通过下式更新 A, B：

$$V_A = \mu V_A + t * grad_A$$

$$V_B = \mu V_B + t * grad_B$$

$$A = A - V_A$$

$$B = B - V_B$$

我们称 μ 为 decay factor，称 t 为步长。

可以看出计算梯度、计算损失函数时时间复杂度与训练样本数量成线性关系。即在算法 A.1 中第 4 行和第 6 行都是 $\Theta(\#training\,examples)$，于是整个算法的时间复杂度也是与训练样本数量成线性关系的。由于训练集比较大，直接在 R (Team, 2014) 中用显示循环比较耗时。在实现过程中，凡是需要遍历整个训练集所有样本的部分我们都用 Rcpp (Eddelbuettel et al., 2011) 进行加速。至此，我们解决了摘要部分提到的第一个挑战：数据集包含大量的样本，我们的算法需要对样本数有很好的可扩展性。

有了训练结果之后，我们可以采用算法 A.3 对整个验证集进行预测并评估算法的预测准确度。我们用 mean absolute error（MAE）和 root mean square error（RMSE）来评估预测的准确性。类似地，我们也用 Rcpp (Eddelbuettel et al., 2011) 进行了加速。

Input: User/Book/Rating training set, $A,\ B,\ \lambda_a,\ \lambda_b$, batch size:$\#batch$, number of training examples: nrow(train)

Output: gradient of $A,\ B$: $grad_A, grad_B$

$grad_A = \lambda_a * A * nrow(train)/\#batch$;

$grad_B = \lambda_b * B * nrow(train)/\#batch$;

for *each training example:* (m, n, r) **do**

    $g = a_m^T b_n$;

    $grad_{a_m} += (g - r) * b_n$;

    $grad_{b_n} += (g - r) * a_m$;

end

算法 A.2: PMF 梯度计算的伪代码

Input: User/Book/Rating validation set, $A,\ B$

Output: predicted stars; MAE; RMSE

for *each validation example:* (m, n) **do**

    if $1 \leqslant a_m^T b_n \leqslant 10$ **then**

        $predictied\,star = floor(a_m^T b_n)$;

    else

        if *predicted star < 1* **then**

            $predictied\,star = 1$;

        else

            $predictied\,star = 10$;

        end

    end

end

calculate MAE, RMSE;

算法 A.3: PMF 预测的伪代码

附录 B　六本书的简介

本附录将对第三章第二节第三小节提到的六本书做一个简要介绍。这些简介是我们从 Amazon 上搜索到的书籍简介和评论中提炼的。由于只是做一个简单的示例，这里的简介可能从文学的角度来讲不严格甚至是在胡言乱语，我们不去过分纠结它。图 B.1 列出了这六本书的 ISBN 号、书名、作者和出版年份。其中，ISBN 号是我们对原本的 ISBN 字符串进行编号后对应的序号。

ISBN	Book.Title	Book.Author	Year.Of.Publication
8	Little Altars Everywhere	Rebecca Wells	2003
9	Waiting to Exhale	Terry McMillan	1995
24	Adressat unbekannt.	Kathrine Kressmann...	2002
25	Henry der Held.	Roddy Doyle	2001
118	Soul Survivor	Christopher Golden	1999
146	Auf der Suche nach dem verlo...	Jean Liedloff	1999

图 B.1　六本书的信息

《Little Altars Everywhere》讲的是成长在一个失调的家庭的年轻女孩的生活经历，捕捉到了童年的纯真，反映了二十世纪六十年代美国小镇上养育子女所面临的冲突。我们关注的关键点在：二十世纪六十年代、失调的家庭生活。《Waiting to Exhale》讲述了四个女人一年忙碌而充满压力的生活。通过彼此，她们找到了支持和指导，发展了友谊。这本书可谓描述现代郊区生活方式和温暖友谊的杰作。我们关注的关键点在：现代生活方式、温暖友谊。《Adressat unbekannt》叙述的是二战时期两位朋友在纳粹政权影响下友谊破裂的故事。我们关注的关键点在：二战时期、友谊破裂。《Henry der held》讲的是爱尔兰独立战争时期英雄亨利所经历的一系列变化。我们关注的关键点在：爱尔兰独立战争时期。《Soul Survivor》围绕一个正在重新体验二战战斗机飞行员过去生活的婴儿展开，他和父母之间的感人故事证实了那些已经相信来世的人的信仰。我们关注的关键点在：二战、相信来世。《Auf der Suche nach dem verlorenen Glück》讲的是一位年

轻的美国人在印第安人部落呆了两年半，发现了这些人如何对待他们的孩子的根源，并展示了关于幼儿的原始需求的长期隐藏知识。我们关注的关键点在：印第安人部落、幼儿的原始需求。

结合上述简介、我们所指出的关键点和图 3.7 中六本书的位置，我们不难粗糙地抽象出两个坐标轴的含义。于是得到了如第三章第二节第三小节的解释。

附录 C　用 MBML 建立 WSLPMF

本附录中，我们将给出运用 MBML 方法解决冷启动问题的详细过程。

第一节　从额外信息到特征

正如第四章第二节：我们可以将额外信息通过特征转化为特征向量，然后一切都从特征向量出发。因此我们做出如下假设：

6. 对于所有的书本和用户，他们的特征向量总是可以计算的。

在第二章中，我们提到额外信息中存在一些缺失值和异常值。我们将它们统一填充为零值，可以将它们对应到特殊的特征向量，也是可以计算的。

本文涉及的额外信息只有书本的出版年份和用户的年龄，特征比较好提取。为方便，我们这里也不过多地涉及特征工程，只是假设对任何书本和用户，我们都知道如何计算特征向量。于是，我们得到了如图 C.1 左侧所示的因子图。

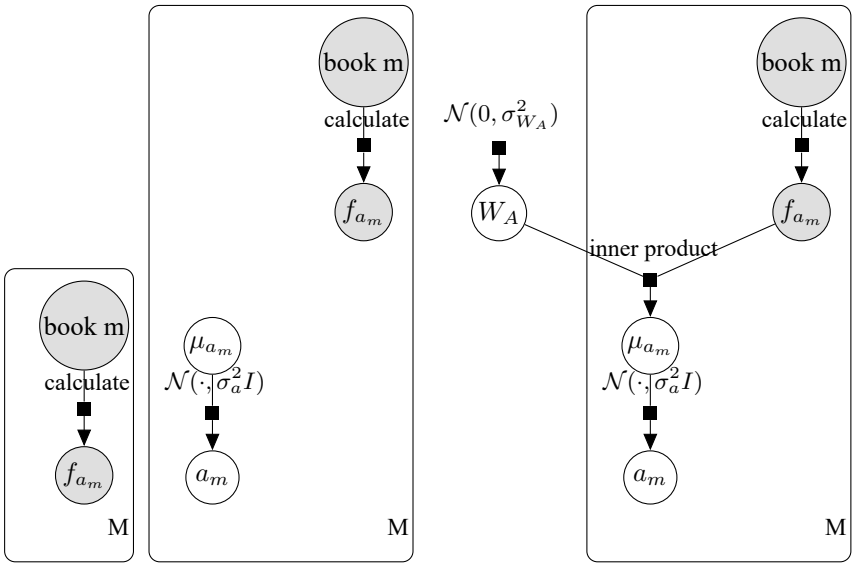

图 C.1　WSLPMF 模型构建示意图

第二节　将特征融入潜在特征

有了特征矩阵之后，我们需要考虑：特征矩阵是如何影响最终的预测的。表 3.1 的第五条假设说：最终的评分星级完全由潜在特征决定。因此，特征只能通过影响潜在特征，进而影响对评分星级的预测。由表 3.1 的第一条假设：书本潜在特征蕴含的是书本的风格特点。而书本特征蕴含的是书本额外信息。一个自然的想法是：知道了书本的额外信息，我们可以提前猜测出书本的大概风格特点。对用户也是类似的。于是，我们得到了如下假设：

7. 通过特征可以预判书本和用户在潜在特征空间的大概位置。

此时，因子图如图 C.1 中间图形所示。假设中的“大概”反映在因子图里就是高斯噪声，也就是：通过特征预判潜在特征后需要加上噪声。否则，会出现图 3.1 第二行左侧因子图类似的问题。

接下来我们要考虑特征预判潜在特征的具体计算过程。我们先在潜在特征空间的某个维度（不妨设为第 k 维）上考虑，于是问题转化成：如何用潜在特征向量去预测表示某特点对上倾向性的一个实数？自然地，我们可以用一个权重向量 w_k 来表示特征每一个维度对预测该实数的贡献，并且它们的贡献互不影响。于是，我们得到如下假设：

8. 对于潜在特征空间的第 k 维，有一个权重 w_k 向量来表示特征每一个维度对潜在特征向量位置预判的贡献程度。当特征第 i 维增长 x 时，特征该维度的贡献值改变 $(w_k)_i \times x$，且特征每一个维度贡献互不影响。

这一假设说明：从特征到潜在特征的计算是个线性模型。得到如图 C.1 右侧的因子图。当然，我们可以修正这条假设，让它包含不同特征之间的交互作用的影响。但对于书本和用户，本文都只考虑了一个特征，因此不需要去考虑交互作用。值得一提的是：从特征到潜在特征的计算还可以是多项式的形式，甚至更复杂的形式。这些都是模型可以改进的地方。

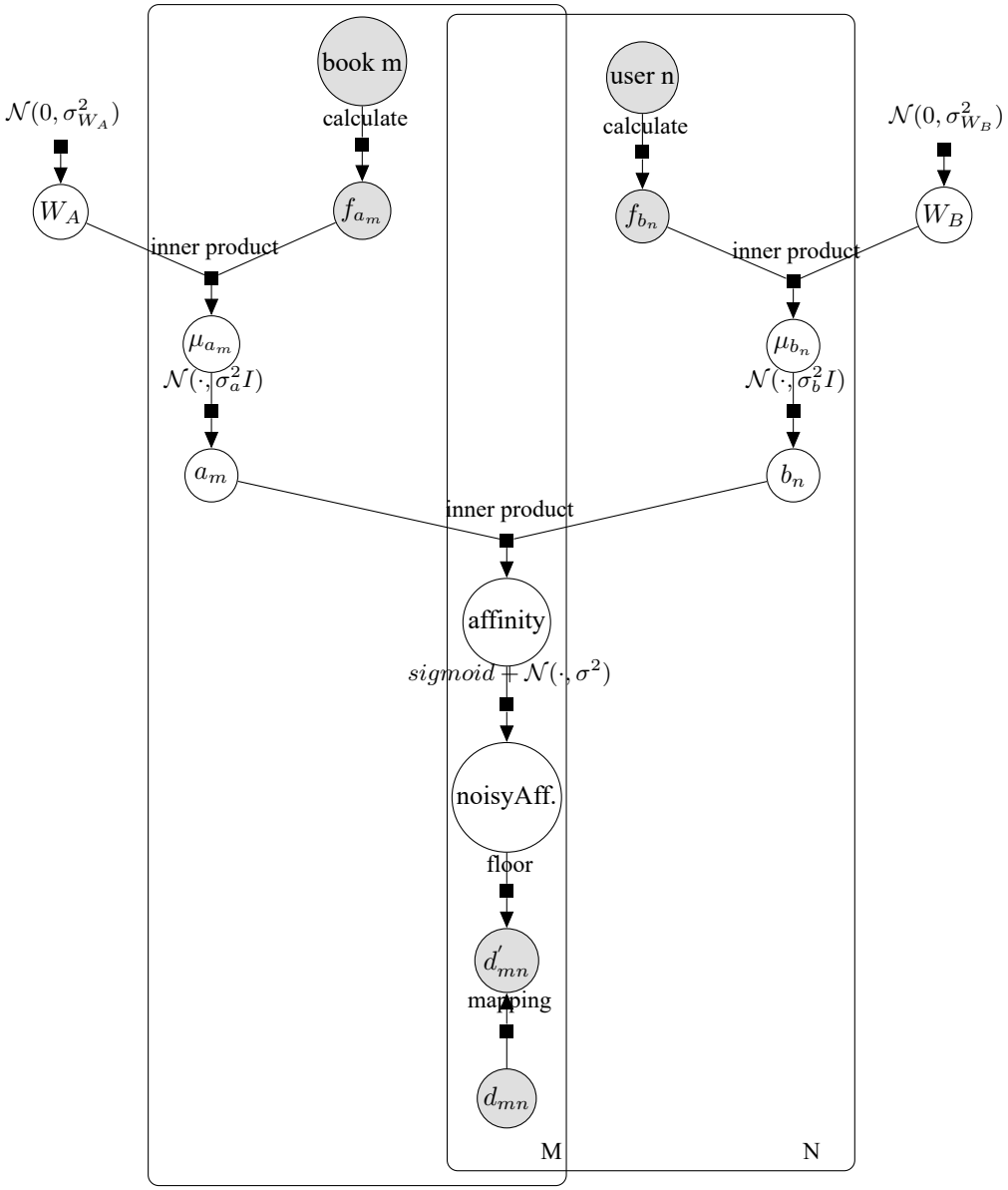

图 C.2　WSLPMF 因子图

第三节　完整模型

我们将刚刚提出的假设和表 3.1 中的假设结合起来得到的模型如表 4.1 所示。其中，我们还加入了第九条假设。它与第五条假设类似，通常都是隐含的假设，我们这里将它显示地写出来。第五条和第九条假设一起说明：凡是假设中没提到的对评分星级的预测都没有影响。我们在第三章第二节已经讨论过这种类型的假设，在此就不赘述。最终，表 4.1 中的模型可以由因子图 C.2 来描述，它对应的概率图就是图 4.2。

附录 D　WSLPMF 算法

本附录将详细介绍 PMF 算法实现细节。WSLPMF 算法是对模型采用 MAP 估计，对于固定的 $\sigma^2, \sigma_a^2, \sigma_b^2, \sigma_{W_A}^2, \sigma_{W_B}^2$，只需优化如下损失函数:

$$E = \frac{1}{2}\|P_\Omega(D - A^T B)\|_F^2 + \frac{1}{2}(\lambda_a\|A - W_A^T F_A\|_F^2 + \lambda_b\|B - W_B^T F_B\|_F^2 + \lambda_{W_A}\|W_A\|_F^2 + \lambda_{W_B}\|W_B\|_F^2)$$

此处，$\lambda_a = \frac{\sigma^2}{\sigma_a^2}$，$\lambda_b = \frac{\sigma^2}{\sigma_b^2}, \lambda_{W_A} = \frac{\sigma^2}{\sigma_{W_A}^2}, \lambda_{W_B} = \frac{\sigma^2}{\sigma_{W_B}^2}$。我们采用 stochastic gradient descent with momentum (Rumelhart et al., 1986) 对其进行优化，算法框架如算法 D.1 所示。

Data: User/Book/Rating training set
Input: $\sigma^2, \sigma_a^2, \sigma_b^2, \sigma_{W_A}^2, \sigma_{W_B}^2$
Output: W_A, W_B

```
initialization;
while #iterations ⩽ max.iteration do
    for each batch do
        calculate gradient of A, B;
        update A, B;
        calculate gradient of W_A, W_B;
        update W_A, W_B;
        evaluate loss E;
    end
end
```

算法 D.1: WSLPMF 算法框架

用的数据是评分数据集的训练集，需要输入固定的参数:$\sigma^2, \sigma_a^2, \sigma_b^2, \sigma_{W_A}^2, \sigma_{W_B}^2$，最后的输出是书本权重矩阵 $W_A = [W_{A_1}, ...W_{A_K}]$ 和用户潜在特征矩阵 $W_B =$

$[W_{B_1}, ...W_{B_K}]$。在初始化时，将 A, B, W_A, W_B 进行零初始化；并给它们的每一列一个相同的小初始动量，具体地：从标准差为 0.01 均值为零的正态分布中抽取。当损失函数收敛或者达到预先设定的最大迭代次数 $max.iteration$ 时算法停止，收敛准则为：损失函数的相对变化 $\Delta = |\frac{E-E'}{E'}| < \epsilon$，其中 E 是当前的损失函数值，E' 是上一步的损失函数值，ϵ 是一个很小的常值。

每一次循环中，我们要对每一个 batch 进行处理。我们首先需要计算损失函数 E 关于 A, B, W_A, W_B 的梯度：

$$\frac{\partial E}{\partial a_m} = \sum_{n=1}^{N}\sum_{m=1}^{M} I_{mn}(\sigma(a_m^T b_n) - D_{mn})\sigma(a_m^T b_n)(1-\sigma(a_m^T b_n))b_n + \lambda_a a_m$$

$$\frac{\partial E}{\partial b_n} = \sum_{n=1}^{N}\sum_{m=1}^{M} I_{mn}(\sigma(a_m^T b_n) - D_{mn})\sigma(a_m^T b_n)(1-\sigma(a_m^T b_n))a_m + \lambda_b b_n$$

$$\frac{\partial E}{\partial W_{A_k}} = -\lambda_a \sum_{m=1}^{M}(A_{mk} - (W_A^T f_{a_m})_k) f_{a_m} + \lambda_{W_A} W_{A_k}$$

$$\frac{\partial E}{\partial W_{B_k}} = -\lambda_b \sum_{n=1}^{N}(A_{nk} - (W_A^T f_{b_n})_k) f_{b_n} + \lambda_{W_B} W_{B_k}$$

其中，I_{mn} 为一个示性函数，当训练集中存在第 n 个用户对第 m 本书的评分星级时，它取 1；否则，它的值为 0。于是，对于每一个 batch，其梯度计算方法如算法 D.2 所示。有了当前 A, B 的梯度 $grad_A, grad_B, grad_{W_A}, grad_{W_B}$ 后，我们通

过下式更新 A, B, W_A, W_B：

$$V_A = \mu V_A + t * grad_A$$

$$V_B = \mu V_B + t * grad_B$$

$$A = A - V_A$$

$$B = B - V_B$$

$$V_{W_A} = \mu V_{W_A} + t * grad_{W_A}$$

$$V_{W_B} = \mu V_{W_B} + t * grad_{W_B}$$

$$W_A = W_A - V_{W_A}$$

$$W_B = W_B - V_{W_B}$$

我们称 μ 为 decay factor，称 t 为步长。

可以看出计算梯度、计算损失函数时时间复杂度与训练样本数量成线性关系。即在算法 D.1 中第 4 行、第 6 行和第 8 行都是 $\Theta(\#training\,examples)$，于是整个算法的时间复杂度也是与训练样本数量成线性关系的。由于训练集比较大，直接在 R (Team, 2014) 中用显示循环比较耗时。在实现过程中，凡是需要遍历整个训练集所有样本的部分我们都用 Rcpp (Eddelbuettel et al., 2011) 进行加速。至此，我们解决了摘要部分提到的第一个挑战：数据集包含大量的样本，我们的算法需要对样本数有很好的可扩展性。

有了训练结果之后，我们可以采用算法 D.3 对整个验证集进行预测并评估算法的预测准确度。我们用 mean absolute error（MAE）和 root mean square error（RMSE）来评估预测的准确性。类似地，我们也用 Rcpp (Eddelbuettel et al., 2011) 进行了加速。

Input: User/Book/Rating training set, A, B, W_A, W_B, λ_a, λ_b, λ_{W_A}, λ_{W_B}, batch size:$\#batch$, number of training examples: nrow(train)

Output: gradient of A, B, W_A, W_B : $grad_A, grad_B, grad_{W_A}, grad_{W_B}$

$grad_A = \lambda_a * A * nrow(train)/\#batch$;

$grad_B = \lambda_b * B * nrow(train)/\#batch$;

$grad_{W_A} = \lambda_{W_A} * W_A * nrow(train)/\#batch$;

$grad_{W_B} = \lambda_{W_B} * W_B * nrow(train)/\#batch$;

for *each training example:* (m, n, r) **do**

  $g = \frac{1}{1+e^{-a_m^T b_n}}$;

  $grad_{a_m} += g * (1-g) * (g-r) * b_n$;

  $grad_{b_n} += g * (1-g) * (g-r) * a_m$;

end

$aux_A = A - W_A^T F_A$;

for m *from 1 to* M **do**

  for k *from 1 to* K **do**

    $grad_{W_{A_k}} -= \lambda_{W_A}(aux_A)_{km}$;

  end

end

$aux_B = B - W_B^T F_B$;

for n *from 1 to* N **do**

  for k *from 1 to* K **do**

    $grad_{W_{B_k}} -= \lambda_{W_B}(aux_B)_{kn}$;

  end

end

算法 D.2: WSLPMF 梯度计算的伪代码

Input: User/Book/Rating validation set, W_A, W_B, F_A, F_B

Output: predicted stars; MAE; RMSE

for *each validation example:* (m, n) **do**

  $predictied\,star = floor(9 * (W_A^T f_{a_m})^T (W_B^T f_{b_N}))$;

end

calculate MAE, RMSE;

算法 D.3: WSLPMF 预测的伪代码

致　谢

本文是在杨灿老师和郑泽敏老师的悉心指导下完成的，在此特别感谢他们对我巨大的支持和无私的帮助。

衷心感谢杨老师给我细致耐心的指导和无微不至的关心。杨老师在我们的毕业设计上花了不少时间和精力。他给我们提供了很多可以选择的课题，并鼓励我们“feel free to choose”，为我们打开了新世界的大门。他给我们的相关文献和阅读材料都很有趣并且切中问题要害，让我们在短时间内拥有提升自己的契机。他及时为我们指引方向、答疑解惑，并鼓励我们独立思考，嘱咐我们“minds-on and hands-on”。正是有了杨老师的指点和帮助，我得以在毕业设计中学习到了很多新知识，在动手实践过程中对问题有了更深刻的理解。在此再次感谢杨老师。

另外，感谢大学四年的每一位任课老师，他们的无私付出给我打下了坚实的基础。感谢陈卿老师，他的每一堂数学分析课都让我如沐春风；感谢将我引入统计大门的胡太忠老师，“泊松过程，从每一个时刻看未来，过程的演化规律都是一样～的”仍回响在耳畔；感谢在我极度低落鼓励并接纳我的杨亚宁老师，他的话就像他主页上的“Life is a supermartingale”一样耐人寻味……

非常感谢亲人给我的莫大的鼓励与温暖。感谢父母默默的付出和关心。无论我做出怎样的选择，他们总是坚定的站在我身后。感谢哥哥一直以来对我的指引和帮助。他总像无明灯一样指引我前行，又像是一把伞为我遮风挡雨。

感谢一直与我共同生活学习的小伙伴们。他们的帮助让我度过难关；他们的陪伴让平平无奇的日子也能充满欢声笑语；和他们的交流学习让我受益良多……

最后，感谢我生命中遇到的每一个人。感谢你们，因为生活里需要美丽，需要惊喜。