

KDD Cup 2012 Track 1 解题报告

张作柏

17300240035

2019 年 6 月 14 日

目录

1	任务简介	1
1.1	任务描述	1
1.2	数据信息	1
1.2.1	名词定义	1
1.2.2	数据文件	1
1.2.3	注意事项	2
1.3	提交格式	3
1.4	评价方式	3
1.5	为什么使用 Julia?	3
2	数据预处理	4
2.1	会话分析	4
2.2	成对训练	5
3	用户兴趣模型	6
3.1	隐因子模型	6
3.2	成对训练	6
3.3	随机梯度下降	7
3.4	日期时间因素	8

3.5 用户个人信息	8
3.6 间接反馈信息	9
3.7 关键词、标签信息	9
4 用户行为模型	10
4.1 时间度量	10
4.2 概率预测	10
5 集成学习 AdaBoost	11
5.1 模型原理	11
5.2 训练方法	11
6 结果评估	13
6.1 测试模块	13
6.2 测试结果	13
7 结论	14
参考文献	15
致谢	16

1 任务简介

本次 PJ 我选择了 KDD Cup 2012 Track 1 题目¹，其中模型主要参考了 [1] 一文。

1.1 任务描述

近年来，随着像 Facebook、Twitter、腾讯微博等社交平台的发展，在线社交网络引起了广泛的关注。全中国最大的微博系统之一，腾讯微博，已经成为网络社交的重要平台。目前，腾讯微博拥有超过 2 亿的注册用户，每天产生四千万条信息。海量的数据引起了数据挖掘爱好者的注意，如何利用数据信息改善用户的使用体验，成为了一个十分有趣并值得研究的问题。

本任务中，我们需要根据用户的兴趣，预测他是否会关注某个对象 (item)。对象可以是某个组织、个人、群体等等。最终我们要在所有备选推荐中，选择至多三个对象推荐给用户。

1.2 数据信息

1.2.1 名词定义

对象 (item): 对象是腾讯微博中的一个用户，他可以代表组织、个人或群体。数据集中大约有六千个不同的对象。

发微博 (tweet): 发微博是指用户可以在微博系统中发表一条信息，他的关注者会看到这条信息的提醒。

转发 (retweet): 用户可以转发其他用户发表的信息，并在其下添加评论。

评论 (comment): 用户可以在别人的微博下发表评论。

关注者 (follower): 用户可以关注其他用户，若用户 A 关注了 B，则称 A 是 B 的关注者。

1.2.2 数据文件

1. **训练数据集 rec_log_train.txt:** 记录了用户与对象之间的历史推荐结果。

文件格式: (UserId) (ItemId) (Result) (Unix-timestamp)

在 Unix-timestamp 的时间，系统向用户 UserId 推荐了物品 ItemId，得到的结果为 Result。Result 为 1，表示接受；Result 为 -1，表示拒绝。

2. **测试数据集 rec_log_test.txt:** 记录了测试集中用户与对象之间的可能推荐。

文件格式同训练数据集 rec_log_train.txt

区别在于其中 Result 域为 0，需要我们来预测。

¹<https://www.kaggle.com/c/kddcup2012-track1>

3. **用户信息 user_profile.txt**: 记录了用户的详细个人资料。

文件格式: (UserId) (Year-of-birth) (Gender) (Number-of-tweet) (Tag-Ids)

依次表示用户的 Id, 出生日期, 性别, 发表的微博数量和标签。其中, 出生日期将以年份的形式给出; 性别为 0、1、2, 分别代表未知、男性、女性; 标签是由用户选择的代表个人兴趣的关键词, 有的用户没有选择标签, 标签的格式为: tag-id1;tag-id2;...;tag-idN, 其中每个标签用正整数来表示, 如果一个用户没有任何标签的话, 那么他的 tag-id 为 0。

4. **对象数据 item.txt**: 记录对象的属性, 包含它所属的类别与关键词。

文件格式: (ItemId) (Item-Category) (Item-Keyword)

依次表示对象的 Id, 对象所属的类别 (以 “a.b.c.d” 的格式给出, 表示所属的子类别), Item-Keyword 是从对象的个人资料中提取的关键词, 以 “id1;id2;...;idN” 的形式给出。

5. **用户行为 user_action.txt**: 记录了用户在微博上的行为, 包括艾特、转发和评论。

文件格式: (UserId) (Action-Destination-UserId) (Number-of-at-action) (Number-of-retweet) (Number-of-comment)

假如用户 A 转发了 B 的微博 5 次, '@'B 共 3 次, 评论 6 次, 则相应的数据为 “A B 3 5 6”。

6. **用户关注行为 user_sns.txt**: 描述用户的关注信息。

文件格式: (Follower-userid) (Followee-userid)

表示前者关注了后者。

7. **用户关键词描述 user_key_word.txt**: 记录了从用户的微博信息中提取的关键词。

文件格式: (UserId) (Keywords)

其中关键词的格式为 “kw1:weight1;kw2:weight2;...kwN:weightN”。关键词是从用户的微博中提取的, 可以用来预测用户的兴趣, 权值表示用户对某个关键词的感兴趣程度, 权值越大, 越感兴趣。每个关键词都用唯一的整数表示, 并与 item 的关键词共用一个字典。

1.2.3 注意事项

以下是一些没有在题目中给出, 但是我在数据处理的过程中发现的问题:

- 对象 item 的 ID 与用户 user 的 ID 是共用一个词典的, 即每个 ItemId 也对应一个 UserId。
- 用户的性别并非只有 0、1、2 三种, 还有性别为 3 的用户。
- 用户填写的出生年中存在非法数据, 格式不正确, 不是年份的形式。

1.3 提交格式

原提交网站上的提交格式说明消失了，我是从别人的博客上看到的，并结合讨论区的评论和自己的一些理解，总结出的以下格式。

测试集 `rec_log_test.txt` 中包含了不重复的 34,910,937 对用户与对象，以及相应的推荐时间，但未给出推荐的结果。文件是按照时间顺序排序的，时间 $< 1,321,891,200$ 的记录将被用于 `public leaderboard`，而时间 $\geq 1,321,891,200$ 的记录将被用于 `private leaderboard` 的最终评测。我们的目标是预测每个集合中推荐给用户的 `item`。

因为腾讯的默认设置是最多推荐三个 `item` 给用户，所以在每个测试集上，我们需要为每个用户推荐至多三个 `item`，且按照顺序排序，这将在评价指标的度量中起到作用。

最终提交时，上传一个 `.csv` 文件，文件总共两列，包括 `UserId` 与 `ItemIds`，分别表示用户的 `Id` 和推荐给他的对象 `Id`。`public leaderboard` 数据在前，`private` 数据在后，分别按照用户 `Id` 升序排列，推荐的对象按照推荐顺序排序。最终提交的文件中必须恰好包含 1,340,127 行。

1.4 评价方式

本次比赛使用的是 MAP(Mean Average Precision) 作为评价指标，计算方式如下：

$$ap@n = \sum_k P(k) / (\text{用户接受总数}), \quad (1)$$

其中 $P(k)$ 表示前 k 个物品中用户接受的个数，若物品 k 未被接受，则规定 $P(k) = 0$ 。

在该任务中，我们至多推荐三个 `item` 给用户，故 $n = 3$ ，`item` 按顺序记为 #1,#2,#3。

假如用户总共接受了 3 个物品，分别是 #1,#3,#4，那么 $ap@3 = (\frac{1}{1} + \frac{2}{3})/3 \approx 0.56$ 。

假如用户总共接受了 4 个物品，分别是 #1,#2,#4，那么 $ap@3 = (\frac{1}{1} + \frac{2}{2})/3 \approx 0.67$ 。

假如用户总共接受了 2 个物品，分别是 #1,#3，那么 $ap@3 = (\frac{1}{1} + \frac{2}{3})/2 \approx 0.83$ 。

1.5 为什么使用 Julia ?

Julia 是一种用于科学计算的新兴语言，其最大的特点是速度快，号称兼具 Python 的便捷性与 C 的速度。在本次项目中，我使用了 Julia 语言，一是因为想要熟悉一下这门新的语言，并在后续的研究中使用，二是因为 Python 的效率太低，且对于本任务并无太大用处，所以不妨使用新的 Julia 语言。

因为在这次的 PJ 中，我的训练过程是手写的，未使用自动求导的模块，所以其中涉及到大量的矩阵、向量运算。而 Julia 对于这些线性代数操作，支持许多快速的算法，具有与 MATLAB 比肩的数值分析运算能力。这也是我使用 Julia 的主要原因之一。

2 数据预处理

本节主要介绍在数据预处理过程中使用的方法，主要方法与 [1] 一文相同，有一些小细节进行了改动。

2.1 会话分析

训练集中共有 73,209,277 个训练数据，其中涉及 1,392,873 个用户和 4,710 个对象。其中，包含 5,253,828 个正例和 67,955,449 个负例。可以看到，负例的个数远超过正例的个数，那实际上所有的负例都是有效的吗？其实不然。这些记录是从现实生活中的用户浏览记录中提取的，而实际中我们自己在浏览微博时，更多是关注于微博的内容，而不是微博的推荐内容。所以用户没有接受一个推荐，并不一定说明用户真的对这个对象不感兴趣，而有可能是因为用户并没有意识到这个推荐。因此，为了便于我们对用户的兴趣建模，我们需要先剔除一部分的噪声数据。

这里，我认为论文中的方法不是很正确，且实现后发现得到的数据与论文中给出的并不一致，所以我没有完全照搬论文中的方法。

因为每个用户的浏览习惯是不同的，这一点主要体现在浏览时长上。对于每个用户，我们对将所有对他的推荐，按时间顺序排序，然后计算相邻两个推荐时刻之间的间隔，即

$$\Delta t_s = t_{s+1} - t_s, \quad (2)$$

其中， t_s 和 t_{s+1} 分别对应第 s 个推荐时刻和第 $s+1$ 个推荐时刻。这里需要注意，因为会有对象在同一时刻被推荐，所以这里要计算相邻两个时刻，而非相邻两个对象的时间间隔。

我们需要将用户的推荐记录划分为若干个时间段，时间段之间是互相独立的，因此需要先计算一个阈值 τ ，当相邻时间间隔大于等于 τ 时，则说明可以将两边划分为两个互不影响的时间段。那么，为了避免过大的时间间隔的影响，我们在确定阈值时，只考虑相邻间隔不超过一个小时的数值，即如果 $\Delta t_s < 3600$ 的话，我们将其记为 $\ddot{\Delta t}_s$ ，只考虑这一部分时间间隔。对所有的时间间隔取平均，得到阈值

$$\tau = \frac{1}{2} \times (\tau_0 + \frac{\sum_s \ddot{\Delta t}_s}{|\ddot{\Delta t}|}). \quad (3)$$

其中， $|\ddot{\Delta t}|$ 表示不超过一小时的时间间隔个数。取 $\tau_0 = 90$ ，计算得到阈值 τ ，对于大于 τ 的时间间隔，我们把它分为两个时间段。

现在，我们将第 k 个时间段中的样本集合记为 Φ_k ，其中的正样本记为 Φ_k^+ 。记出现最早的正样本的下标记为 σ_- ，最晚出现的记为 σ_+ ，那么对于下标为 σ_s 的样本，我们进行如下过滤：

1. 若该时间段出现的正样本过多，则说明可能这是噪声数据，将这个时间段中的数据全部剔除，即只选择 $0 < \frac{|\Phi_k^+|}{|\Phi_k|} \leq \epsilon$ 的时间段，其中取 $\epsilon = 0.86$ 。

2. 我们在下标区间 $\sigma_- - \pi_-$ 到 $\sigma_+ + \pi_+$ 中选取样本, 即样本 σ_s 需要满足 $\sigma_- - \pi_- \leq \sigma_s \leq \sigma_+ + \pi_+$ 。这一限制是基于假设: 用户可能在意识到第一个正样本之前, 一直没有关注到推荐, 可能在接受最后一个正样本之后, 就没有再留意推荐信息。这里, 为了减少样本数量, 提高训练效率, 我们取 $\pi_+ = \pi_- = 1$ 。

最后, 经过数据清洗, 我们得到 3,794,928 个正例和 11,193,905 个负例, 总共约一千五百万的训练数据, 数据量缩减五倍, 这大大提升了训练的效率。

2.2 成对训练

不同于寻常的二分类问题, 在这一问题中, 我们需要对所有的推荐物品进行排序, 这更像是一个学习排名的问题 [2]。实践证明 [3] [4] [5], 以分类问题中常使用的 MSE (Mean Square Error) 作为目标函数进行训练的效果不是很好。在这一问题中, 我们使用更加适合排名问题的成对训练方法。这一方法在推荐系统中的应用比较广泛, 且非常适合排名问题。

成对训练的基本思路是, 将一个正例与一个负例组合, 通过训练使得正例的概率尽可能的高于负例的概率。如何构造成对的组合呢? 对于每个用户, 因为负例的数量较多, 所以我们对于它的每一个负例, 随机挑选一个正例与它匹配, 最终, 我们将得到 11,193,905 个训练样本。

3 用户兴趣模型

本节中，我们主要介绍我们的核心模型——用户兴趣模型。根据用户之前的推荐记录和用户的信息，推测用户的喜好，借此来预测用户对推荐物品接受与否。

3.1 隐因子模型

隐因子模型 (*Latent Factor Model*) 是推荐系统中常使用的一种算法 [6]。算法利用矩阵分解的思想，对用户的喜好进行建模，是一种协同过滤 (*Collaborative Filtering*) 的常用方法 [7]。

给定一个用户对物品的评分矩阵，如何衡量用户对物品的喜爱程度？LFM 的一个基本假设是存在一个用户与物品属性的低维表示，可以用于衡量用户与物品的密切程度。通俗地讲，就是每个物品可能存在若干个属性，而每个用户对这些属性分别有相应的喜爱程度，最终这些属性的加权平均即为用户对物品的评分。这里的属性即对应算法中的隐因子 (*Latent Factor*)。

隐因子模型也可被理解为矩阵低秩分解问题——用两个低秩矩阵的乘积近似原矩阵。矩阵分解的技术常被用于估计用户与物品的隐因子向量，其中最受欢迎的便是奇异值分解 (*Singular Value Decomposition*) [8]。然而，由于矩阵分解的复杂度过高 [9]，本问题中的矩阵维度过大，所以并不适用。

本问题中，我们用 \mathbf{q}_i 和 \mathbf{p}_u 分别代表物品的属性向量与用户对属性的喜好向量，令 \hat{r}_{ui} 表示用户 u 关注物品 i 的预测概率，则预测值可表示为：

$$\hat{r}_{ui} = f(b_{ui} + \mathbf{q}_i^\top \mathbf{p}_u) \quad (4)$$

其中函数 $f(\cdot)$ 是 sigmoid 函数，即 $f(x) = \frac{1}{1+e^{-x}}$ ，用于将实数值映射到 $(0, 1)$ 区间表示概率。而 b_{ui} 是偏置项，可定义为：

$$b_{ui} = \mu + b_u + b_i \quad (5)$$

这个方程中的 μ 表示所有的平均得分， b_u 和 b_i 分别表示用户和物品的偏置项。

我们的目标是用这些向量近似矩阵，故等价于下面的最优化问题：

$$\min_{\mathbf{p}_*, \mathbf{q}_*, b_*} \sum_{(u,i) \in \Omega} (r_{ui} - f(b_{ui} + \mathbf{q}_i^\top \mathbf{p}_u))^2 \quad (6)$$

我们可使用迭代方法取学习其中参数的值，效率比矩阵分解要高许多。但是，实验结果证明直接套用该模型并不能取得好的结果。在下一节中，我们将介绍如何将成对训练的模式与 LFM 模型结合。

3.2 成对训练

在第2.2节中，我们已将数据按照正负例进行了分组，每组由一对正例和负例构成。我们的目标是通过学习用户的喜好向量，来使得用户选择正例的概率要大于负例的概

率。因此，我们将训练的损失函数调整为：

$$g(\hat{r}_{ui}, \hat{r}_{uj}) = (1 + \exp(-(b_j - b_i + (\mathbf{q}_j - \mathbf{q}_i)^\top \mathbf{p}_u)))^{-1} \quad (7)$$

其中样本 $\hat{r}_{ui} \in \Omega^-$, $\hat{r}_{uj} \in \Omega^+$ 。

需要注意：

- 这里只给出了一个样本的损失函数，整体的损失函数可视为所有样本之和。实际上，因为使用随机梯度下降算法，所以只需要用到一个样本的损失函数。
- 这里是需要最大化损失概率，只需要在学习时更改梯度更新方向即可。
- 偏置项中的 b_u 与 μ 被抵消了，因为两个样本取自同一用户。这一现象还会出现在后续的模型扩展中，尽管原文并未言明。
- 因为 loss function 的变化，一些参数变得无法学习，自然也就会被踢出模型。因此，要根据可学习的参数调整预测的方式。

正如最后一条所言，我们需要调整在最后预测推荐物品的方式。原本我们只需要给出用户接受概率最高的三个物品，但是现在因为 b_u 等参数变得不可学习了，所以无法预测用户对物品的接受概率。好在我们只需要比较用户对物品的相对喜爱程度即可，由方程 (7) 可知，我们只需要选择 $b_i + \mathbf{q}_i^\top \mathbf{p}_u$ 最大的三项物品即可。

3.3 随机梯度下降

随机梯度下降 (Stochastic Gradient Descent) [10] 是机器学习中最常用的优化算法之一。它常被用于迭代地优化某个函数，在凸优化问题中扮演这尤其重要的角色。其基本思路是在每一步选择函数值下降最快的方向，迈出一小步。在随机梯度下降中，我们每次取一个样本损失函数的梯度作为整个损失函数梯度的近似，沿该方向迈一小步。写成更新式即为：

$$x \leftarrow x - \eta \cdot \frac{\partial L}{\partial x} \quad (8)$$

其中 η 表示学习率，用于调节更新的步长。

随机梯度下降的好处是，每次迭代只需要考虑一个样本，所以时空复杂度非常低。唯一需要注意的一点就是如何计算 loss 对参数的梯度。因为是我自己手写的梯度更新，所以在此写出几个主要参数的梯度推导过程。

令 $L = g(\hat{r}_{ui}, \hat{r}_{uj}) = f(b_j - b_i + (\mathbf{q}_j - \mathbf{q}_i)^\top \mathbf{p}_u)$ ，其中 sigmoid 函数 $f(\cdot)$ 的导数为：

$$\frac{\partial f(x)}{\partial x} = f(x) \cdot (1 - f(x)) \quad (9)$$

因此，我们可以得到

$$\frac{\partial L}{\partial b_j} = L \cdot (1 - L) \quad (10)$$

$$\frac{\partial L}{\partial \mathbf{q}_j} = L \cdot (1 - L) \cdot \mathbf{p}_u \quad (11)$$

$$\frac{\partial L}{\partial \mathbf{p}_u} = L \cdot (1 - L) \cdot (\mathbf{q}_j - \mathbf{q}_i) \quad (12)$$

利用这些梯度，我们就可以进行参数学习了。后面的几个小节中，会引入新的可学习参数，不过其推导过程与梯度形式都与上式类似，就不一一推导了。

3.4 日期时间因素

时间因素在衡量用户行为中起到了比较关键的作用。比如，白天的时间用户常常会忙于工作，而不会关注微博，而晚上关注的行为就多了些。另外，物品的受欢迎程度也因时间而异。

首先，我们添加三个时间因子：hour、second 和 day，其偏置计算方式如下：

$$b_{hour} = b_{Bin(t)} \quad (13)$$

$$b_{day} = \frac{t - d^-}{d^+ - d^-} b_{day}^- + \frac{d^+ - t}{d^+ - d^-} b_{day}^+ \quad (14)$$

$$b_{sec} = \frac{sec(t) - s^-}{s^+ - s^-} b_{sec}^- + \frac{s^+ - sec(t)}{s^+ - s^-} b_{sec}^+ \quad (15)$$

其中 hour 表示一天中的时刻 (取值范围为 1~24)，sec 表示一天中的第几秒 (取值范围为 0~86400)，day 表示的是记录中的第几天 (取值范围为 1318348785~1322668798)。这里 b_{day}^- , b_{day}^+ , b_{sec}^- , b_{sec}^+ 是四个可学习的参数。 d^- , d^+ , s^- , s^+ 分别表示整个数据集时间的开始与结束和一天时间的开始和结束，此处分别设置为 1318348785, 1322668798, 0, 86400。

同理，我们可以在 item 属性向量中添加 \mathbf{z}_{day} 和 \mathbf{z}_{sec} 两个属性向量，计算方法同上。

3.5 用户个人信息

为了综合考虑用户的各种信息，用户的年龄与性别也可能会影响用户对物品的喜好。所以，我们可以将这些因素作为喜好向量 $\mathbf{y}_{age(u), gender(u)}$ 添加到用户的喜好向量 \mathbf{p}_u 中。除此之外，我们还可以把用户发表的微博数量作为一个因素，因为微博的数量也可以衡量用户的活跃程度。

注意到，item 和用户是共用一个字典的，也就是说 item 也有性别、年龄等因素。不同的人对不同年龄段、不同性别的对象的喜好是不同的，不同的对象对不同年龄段、不同性别的人的吸引程度也是不同的，所以我们可以把这两个因素，作为偏置加入到 b_{ui} 项中。

接下来，我们讨论如何处理性别、年龄、微博数量三个属性。

对于性别属性，数据描述中声称有三种：男、女、未知，但实际操作中我发现还有第四种性别，因为无法解释，所以我就只好添加了一种情况，即性别对应 1、2、3、4 四种类别。

对于年龄属性，信息中给定的是出生年份，我们要将其转化为用户的年龄：

$$\text{age}(u) = \begin{cases} 0 & x < 1950 \\ \text{ceil}((x - 1950)/2) + 1 & 1950 \leq x < 2004 \\ 28 & x \geq 2004 \\ 29 & x \text{ is illegal} \end{cases} \quad (16)$$

对于微博数量这一属性，因为其分布过于稀疏，我们通过取 \log 将其压缩到 $[0, 15]$ 的区间范围内。

3.6 间接反馈信息

3.7 关键词、标签信息

最终，汇总了所有的信息后，我们得到最新的属性向量为：

$$\begin{aligned} \ddot{b}_{ui} = & b_i + b_{\text{hour}} + b_{\text{day}} + b_{\text{sec}} + b_{u, \text{gender}(i)} \\ & + b_{u, \text{age}(i)} + b_{\text{gender}(u), i} + b_{\text{age}(u), i} + b_{kw(u), i} + b_{\text{tag}(u), i} \end{aligned} \quad (17)$$

$$\ddot{\mathbf{q}}_i = \mathbf{q}_i + \mathbf{z}_{\text{day}} + \mathbf{z}_{\text{sec}} \quad (18)$$

$$\begin{aligned} \ddot{\mathbf{p}}_u = & \mathbf{p}_u + \mathbf{y}_{\text{age}(u), \text{gender}(u)} + \mathbf{y}_{\text{tweetnum}(u)} \\ & + |S(u)|^{\alpha_1} \sum_{k \in S(u)} \mathbf{y}_k + |A(u)|^{\alpha_2} \sum_{l \in A(u)} \mathbf{y}_l \\ & + \sum_{m \in K(u)} W(u, m) \cdot \mathbf{y}_{kw(m)} + |T(u)|^{-\frac{1}{2}} \sum_{n \in T(u)} \mathbf{y}_{\text{tag}(n)} \end{aligned} \quad (19)$$

4 用户行为模型

4.1 时间度量

4.2 概率预测

5 集成学习 AdaBoost

本节中我们将组合方法融入模型中，组合分类器 (*ensemble*) 是一个复合模型，由多个分类器组合而成，个体分类器投票，组合分类器基于投票返回类标号预测。组合分类器往往比成员分类器更准确。

5.1 模型原理

自适应提升算法 (*Adaptive Boosting*) 是一种机器学习算法，由 Yoav Freund 和 Robert Schapire 于 1997 年首次提出 [11]。其基本思想是：前一个分类器分错的样本会被用来训练下一个分类器，相对于大多数其他学习算法而言，不容易出现过拟合现象。

AdaBoost 可被应用于人脸识别 [12]，网络入侵检测 [13]，预测蛋白质结构 [14] 等机器学习问题。此外，它还可以用于处理不平衡数据 [15]，非常适合处理我们这道题中正样本较少的情况。

关于 AdaBoost 的原理可参考教材 [16] 第 380 页和这篇博客²。图 1 描述了 AdaBoost 的基本过程³。其基本原理如下：

给定数据集 \mathcal{D} ，它包含 d 个类标记的元组 $(\mathbf{X}_1, y_1), (\mathbf{X}_2, y_2), \dots, (\mathbf{X}_d, y_d)$ ，其中 y_i 是元组 \mathbf{X}_i 的类别标号。初始时，AdaBoost 对每个训练元组赋予相等的权重 $1/d$ 。为了组合 k 个基分类器，我们将进行 k 轮迭代。每次迭代过程如下：

1. 按训练元组的权重从 \mathcal{D} 进行有放回抽样，得到训练集 \mathcal{D}_i 。
2. 使用训练集 \mathcal{D}_i 训练分类器 M_i 。
3. 计算分类器 M_i 在 \mathcal{D}_i 上的分类误差。若误差大于 0.5，则重新训练该模型。
4. 对于每个分类正确的元组，调整其权重，将其权重乘以 $\frac{\text{error}(M_i)}{1-\text{error}(M_i)}$ ，其中 $\text{error}(M_i)$ 表示分类器 M_i 的分类误差。
5. 规范化训练元组的权重。

当使用组合分类器对新元组 \mathbf{X} 分类时，设置模型 M_i 的投票权重为 $\log \frac{1-\text{error}(M_i)}{\text{error}(M_i)}$ ，最后返回最大权重的类。

5.2 训练方法

不同于普通的分类问题，本问题是一个 Learning to Rank 的模型，不能简单地套用 AdaBoost 算法。且我们在训练时使用的是 pairwise training 方法，而在预测时需要对所有推荐物品排序，这其中的差异使得 AdaBoost 的应用并不方便。

²<https://www.cnblogs.com/pinard/p/6133937.html>

³图源自<https://www.datacamp.com/community/tutorials/adaboost-classifier-python>

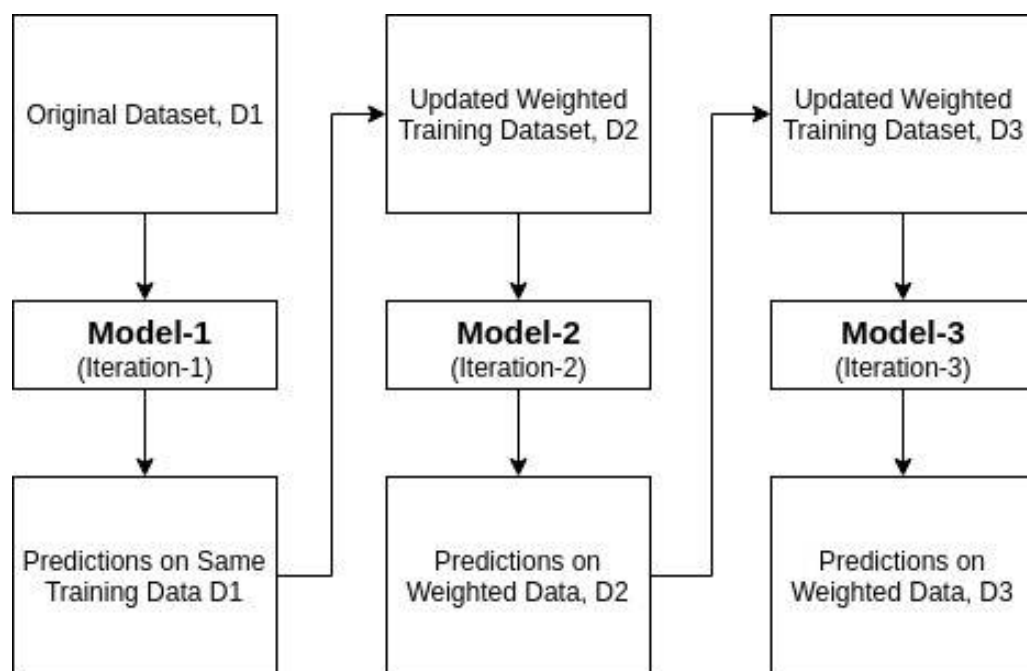


图 1: AdaBoost 流程

我做的一个尝试是:在训练时仍采用抽样-pairwise training-更新权重的方法。pairwise training 中每个元组可视为一个二分类问题,目标是使得正样本的出现概率高于负样本。计算分类器误差时,若某一元组的概率大于 0.5,则将其视为正确分类的样本。在分类时,将 K 个分类器的权重进行加权平均,后比较所有物品的权重,取其中最大的三个作为最终的推荐结果。该方法最终取得了比较好的效果。

参数设置 受系统内存大小限制,我们无法使用较多的模型进行组合,故仅设置模型数量 K 为 5。对于每个模型,设置其参数维度 $d = 90$,学习率为 0.00025。每次训练的过程中,从整个训练样本中有放回的抽样 10,000,000 个样本。整个训练过程在服务器上持续了一个星期,也足以说明这个问题的规模之大。

6 结果评估

6.1 测试模块

6.2 测试结果

7 结论

参考文献

- [1] Yunwen Chen, Zuotao Liu, Daqi Ji, Yingwei Xin, Wenguang Wang, Lu Yao, and Yi Zou. Context-aware ensemble of multifaceted factorization models for recommendation prediction in social networks. In *KDD-Cup Workshop*, 2012.
- [2] Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.
- [3] Johannes Fürnkranz and Eyke Hüllermeier. Pairwise preference learning and ranking. In *European conference on machine learning*, pages 145–156. Springer, 2003.
- [4] Steffen Rendle and Christoph Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 273–282. ACM, 2014.
- [5] Amit Sharma and Baoshi Yan. Pairwise learning in recommendation: experiments with community recommendation on linkedin. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 193–200. ACM, 2013.
- [6] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.
- [7] Yelong Shen and Ruoming Jin. Learning personal+ social latent factor model for social recommendation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1303–1311. ACM, 2012.
- [8] Frank Kleibergen and Richard Paap. Generalized reduced rank tests using the singular value decomposition. *Journal of econometrics*, 133(1):97–126, 2006.
- [9] Virginia Klema and Alan Laub. The singular value decomposition: Its computation and some applications. *IEEE Transactions on automatic control*, 25(2):164–176, 1980.
- [10] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.
- [11] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.
- [12] Bo Wu, Haizhou Ai, Chang Huang, and Shihong Lao. Fast rotation invariant multi-view face detection based on real adaboost. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, pages 79–84. IEEE, 2004.

- [13] Weiming Hu, Wei Hu, and Steve Maybank. Adaboost-based algorithm for network intrusion detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(2):577–583, 2008.
- [14] Bing Niu, Yu-Dong Cai, Wen-Cong Lu, Guo-Zheng Li, and Kuo-Chen Chou. Predicting protein structural class with adaboost learner. *Protein and peptide letters*, 13(5):489–492, 2006.
- [15] Paul Viola and Michael Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *Advances in neural information processing systems*, pages 1311–1318, 2002.
- [16] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

致谢

感谢张宇恒同学与我交流论文复现中的细节！