1 程序执行说明

执行 code 文件夹下 main.sh（./main.sh）即可获得预测文件。程序在 Red Hat 4.8.2-16 和 Ubuntu 5.4.0 均测试过；Python 版本为 3.6，没有需要额外安装的包。

2 方案简介

2.1 主要思路

当前方案是一个初步尝试，仅将精力放在了“召回”阶段。给定某用户 u，召回的途径主要有两种：

（1）对用户 u 的历史感兴趣商品列表排序，取前 K（K<50）个商品作为预测列表的一部分；

（2）从商品流行度列表挑选 50-K 个商品（这些商品不属于用户 u 曾感兴趣的商品类别），填补剩余的预测列表。

途径（1）可保证召回率，途径（2）可保证新颖性。本方案选择最后 1 天为验证集，依据 Recall*0.15+Novel*0.85（与线上评价近似）评价效果，从而确定 K 的取值（K=27）。

2.2 实施细节

（1）赛题指明数据的时间戳是依据一个周五的 00:00 做的偏移，故假设数据是从 2019 年 7 月 5 号开始，为数据添加了相关时间信息。

（2）设置行为的权重。’pv’（浏览）权重为 1，’fav’（收藏）权重为 2，’cart’（加购物车）权重为 3，’buy’（购买）权重为 4。用户购买完商品后，可能会查询物流情况，从而触发新点击，故给’buy’设置的权重较高。

（3）按照 Temporal Recommender Systems 的思想，基于时间信息调整行为权重，距待预测时间越近权重越高。

（4）区分用户性别，分别进行商品流行热度列表统计。

（5）只依据 testA 和 testB 做统计、预测。曾尝试合并 train 和 testA 进行统计分析，但效果近似，故只选用测试集。

2.3 其他尝试

（1）建模预测用户是否会操作历史感兴趣产品。效果尚可，复赛会用于拓展本方案。

（2）依据用户信息对用户分组，为每个组分别统计商品流行度列表。线下效果较好，但线上结果格式不正确（生成的商品预测列表包含 testA 中商品）。复赛更换榜单时会注意。