京东杯2019第六届泰达创新创业挑战赛-用户对品类下店铺的购买预测解题思路

1）概述

用户对品类下店铺的购买预测的数据包括用户行为、商品评论、商品信息、店铺信息、用户信息5个数据表，需要预测在考察周内用户的购买品类，以及对应品类下店铺。对每个用户的预测包括用户-品类和相应品类下店铺两个方面，评分进行加权。此处可以分成两个任务去进行分别预测，但是出于对简化问题复杂度的考虑，这里直接合并作为一个二分类任务展开。

本次大赛没有直接提供训练集对应标签，需要参赛者根据业务数据的理解进行训练集、验证集以及测试集的构建，赛题具有相当的灵活性，但也增加了赛题的难度。针对需要解决的问题和数据特征，我们主要从四个方面进行处理：数据预处理，标签数据集划分，特征工程，模型选择与训练。

在模型选择上，我们队伍使用XGBoost 作为主要模型，后期进行多模型训练的Bagging融合。比赛前期尝试使用LightBGM,Decision tree等算法进行尝试，由于前期对于特征工程的工作尚未优化好，性能未有明显变化，仰赖于XGBoost对于特征重要性有良好的评估效果，故一开始没有在模型选择上进行纠结。

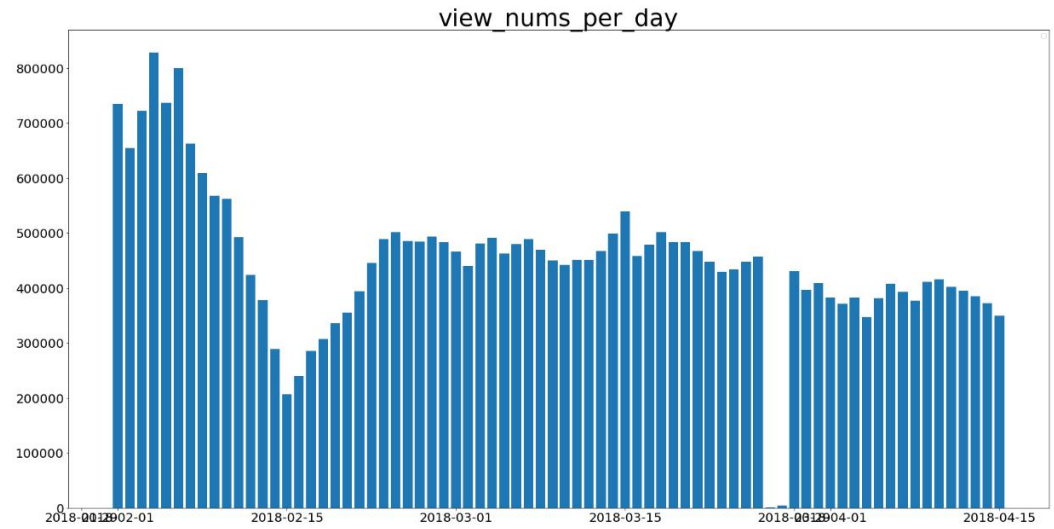
数据预处理上,我们首先对源数据文件进行了相应的EDA (Exploratory Data Analysis)，也就是对数据进行探索性的分析，从而为之后的数据处理和建模提供必要的依据支撑。其次，缺失数据处理主要采用众数或平均值进行填充。最后，由于团队中采用概率模型（树形模型），故没有对特征数据做归一化处理。

工具的使用上，我们采用了python, pandas, numpy, xgboost, sklearn, matplotlib, seaborn等相关工具,特别的, Jupyter notenook的使用让我们的团队协作更加便利。

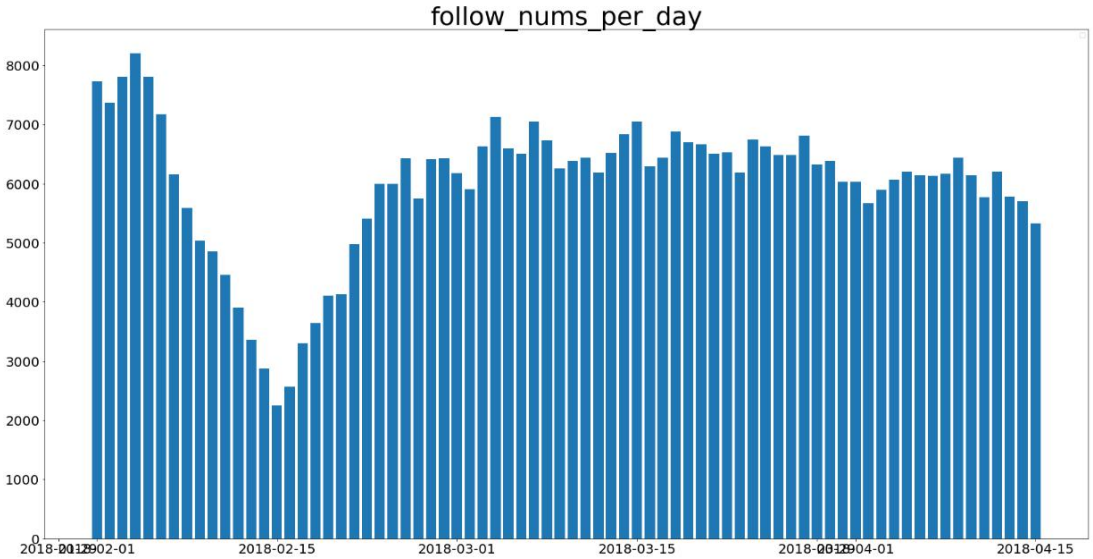
2）数据处理

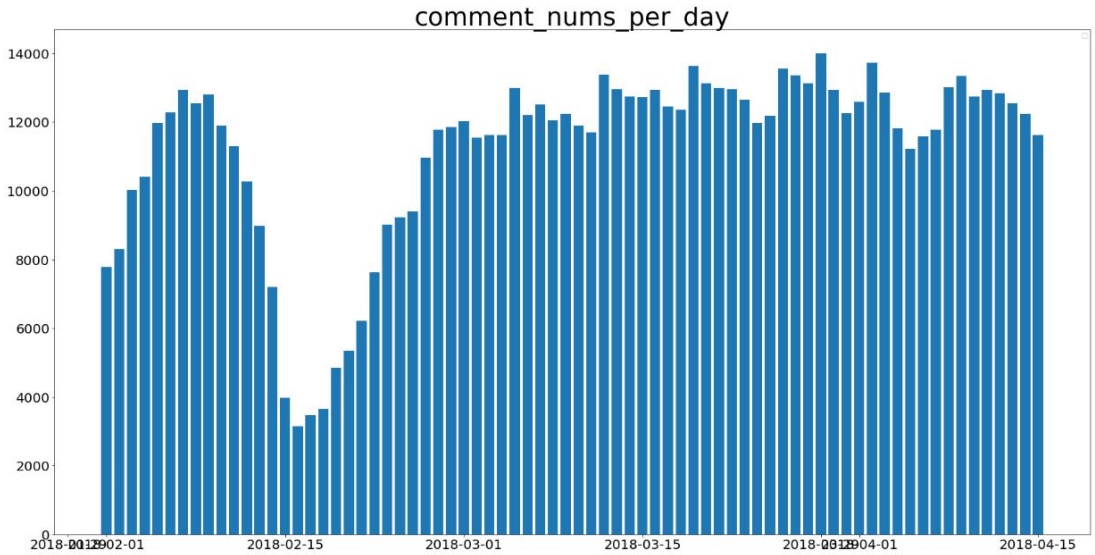
- **EDA探索性分析**

浏览数据：2月份的数据由于处于春节期间，数据波动较大，3.27-3.28两日存在大量缺失（但总体影响不大），其余日期的浏览数量较为平稳。



收藏、评论数据： 除去过节影响，整体数据较为平稳。





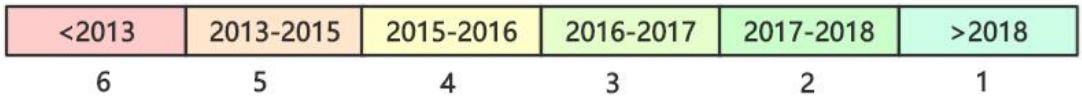
- 处理 **Missing Data**

jdata_user 用户信息表中的 age 采用中位数，sex，city_level，province，city，county，shop_reg_tm，jdata_shop.cate 等采用众数进行填充。

- **处理 Time 类数据**

针对 time 类数据的分布广泛且无法作为数值型数据直接处理，故进行 time 数据分箱，有两种划分方案：一种是等值划分（按照值域均分），另一种是等量划分（按照样本数均分）。我们这里选择等值划分,将特征按照时间大小均匀地划分为 6 个区间，即离散化为 1~6。

需要进行时间分箱操作的字段有：user_reg_tm, shop_reg_tm, market_time, 具体分箱细节如下图：



另为方便后续特征工程中时间滑窗工作，故将用户操作时间字段 action_time 转换成精确到天的 Date 类型。

- **处理 Outlier**

在离群值的处理上，通过 EDA 的分析，我们发现 3 月前数据存在着较大的不稳定性，故训练集不采用这部分样本。

- **转换某些 Categorical Variable 的表示方式**

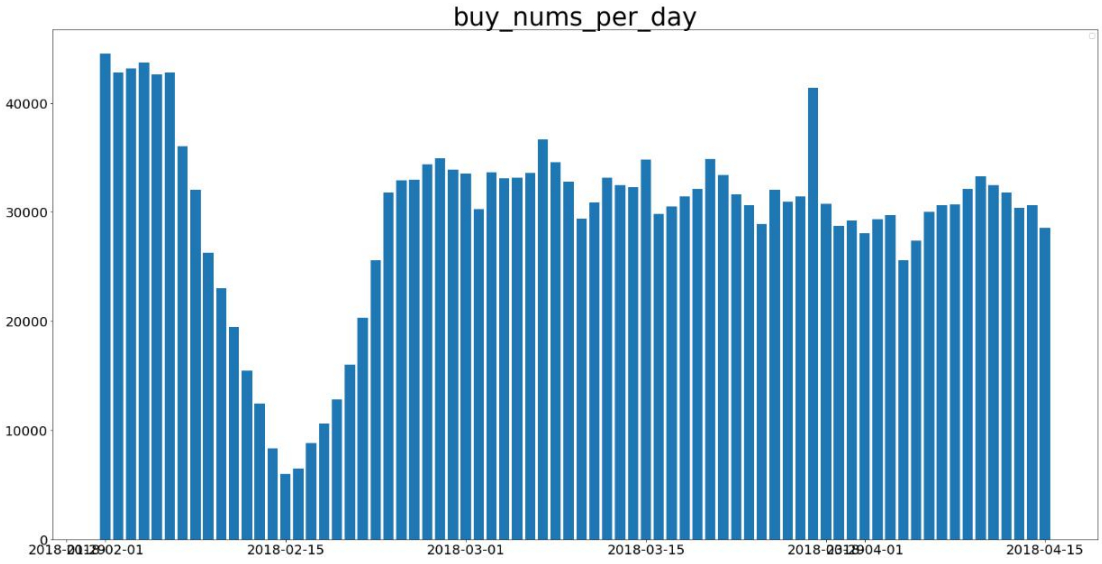
由于离散特征在数值上没有具体含义，直接参与训练可能会影响分类性能，故我们采用了 One-Hot 编码，得到了 01 特征，尽管针对于 XGBoost 来说可能不太需要 One-Hot 编码, 但鉴于我们的类别范围较小，也取得了不错的效果。进行 One-Hot 的特征有 sex，city_level，provinceuser_reg_tm, shop_reg_tm,market_time。

- **数据分散在几个不同的文件中，需要 Join 起来**

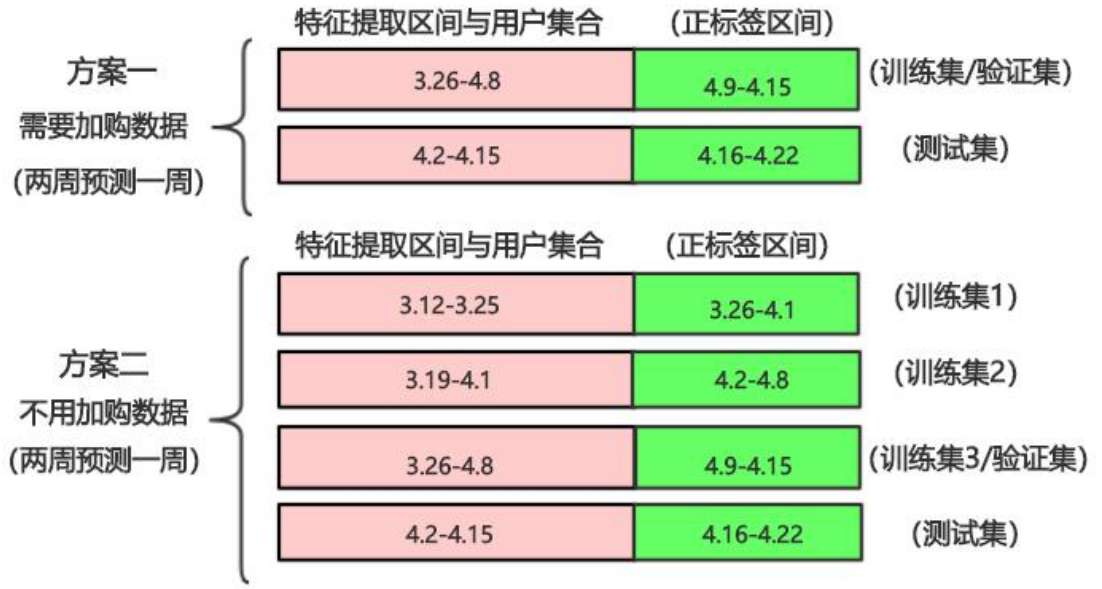
由于比赛中没有给出相应训练集的 label,所以对于我们需要将行为数据 jdata_action 和商品数据 jdata_product 以 sku_id 为键进行 join,取其相应考察周（1 周）内 action_type=2 的记录数，新建 label 列，值为 1。再用 label 为 1 的数据集和考察周之前一段时间(2 周)内产生过任意行为的 user_cate_shop 数据进行 left join 操作，同时对于 join 操作后 label 列产生的 NAN 值填充 0。这样，一个完整的训练/验证数据标签就完成了。

- **数据的选择与原因**

在数据的选择上，官网提供了 2018-02-01 到 2018-04-15 用户集合 U 中的用户，对商品集合 S 中部分商品的行为、评价、用户数据我们首先分析了购买行为在时间维度上的分布，如下图所示。



从图中我们发现用户购买行为中 2 月份的数据存在较大波动，故为了排除春节（02-16）/元宵（03-02）节假日效应的余波，我们选择数据时尽量选择离考察周最近的一些数据进行训练与预测，另外对行为数据中的浏览、关注、评论、加购数据进行分析，不幸的发现，2018-04-08 之前的加购数据全部缺失，加购数据存在天数只有 2018-04-08 到 2018-04-15，这里我们分析了两种方案，结合模型的训练耗时和实际测试的效果，我们的总体方向都是使用考察周前两周的数据进行训练，如下图所示。



在需要考虑加购数据时，我们能够利用的加购物车数据仅为 4 月 8 日一天的加购记录。对应的在测试集中的特征构建时我们也仅使用 4 月 15 日的加购数据。其中抽样少量做线下验证。

在不考虑加购物车数据的情况下（购物车数据过少），我们构造了三个训练集，其中前期训练时使用前两个训练集分别训练，使用第三个训练集做验证集，由此测试确定需要提交的正样本个数所对应的的概率阈值，后续将第三个训练集也加入到训练中。

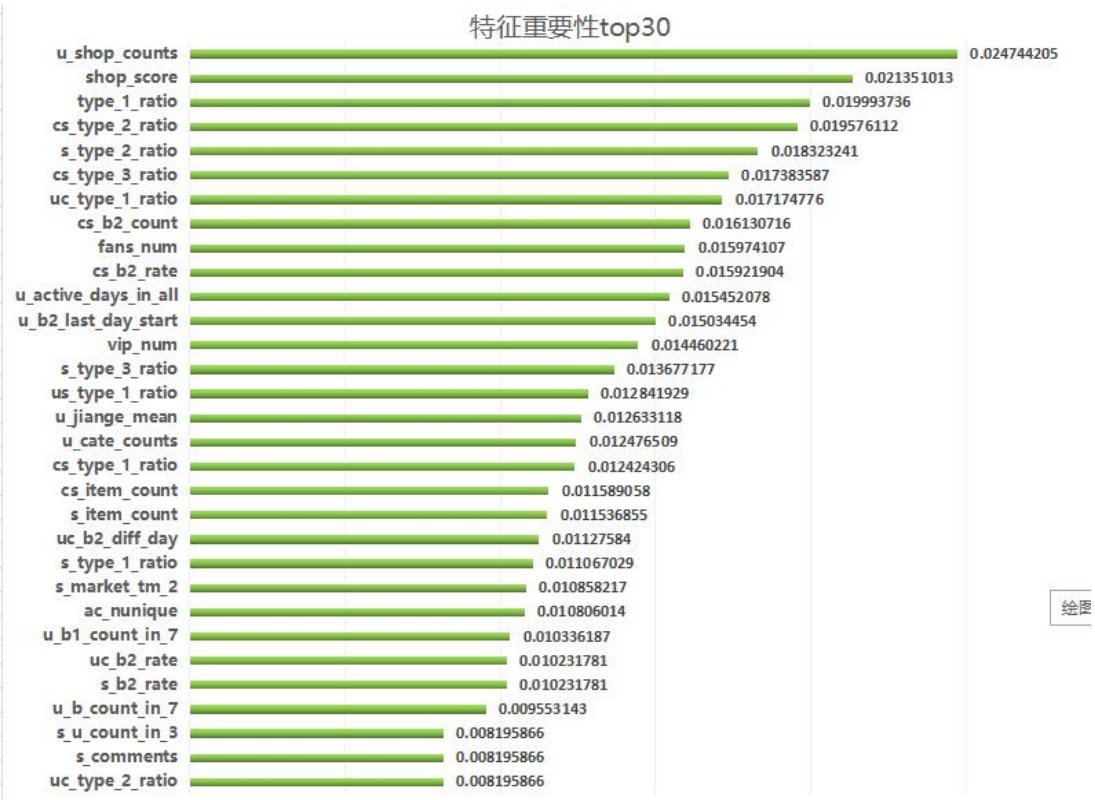
3）特征选择与获取

A. 哪些特征是关键特征？

根据前期的头脑风暴，我们构造了大量的特征。从模型可解释性出发构建特征是我们一开始的出发点，从业务特征上来说，我们构建了 7 类特征，分别为用户特征 U 类，类别特征 C 类，店铺特征 S 类，用户-类别组合 U_C 类，用户-店铺组合 U_S 类，用户-类别-店铺组合 U_C_S 类。

特征类别	特征说明
用户特征 U 类	（1）用户在考察周前一段时间做出行为的店铺数量 （2）用户在考察周前一段时间做出行为的品类数量 （3）用户在考察周前一段时间内做出行为的天数 （4）用户在考察周前的购买频次 （5）用户的平均购买时间间隔 （6）用户的点击购买转化率 （7）用户最近一次购买距离考察周的时间间隔
类别特征 C 类	（1）品类的收藏购买转化率 （2）品类的购买评论转化率 （3）品类操作购买转化率
店铺特征 S 类	（1）店铺的评分 （2）店铺的购买评论转化率 （3）店铺的收藏购买转化率， （4）店铺商品的数量 （5）店铺的会员数，粉丝数
用户-类别组合 U_C 类	（1）用户对品类的点击购买转化率， （2）用户-类别对上一次购买距考察周的时间 （3）用户对品类的操作购买转化率
用户-店铺组合 U_S 类	（1）用户对店铺的点击购买转化率， （2）用户-店铺对上一次购买距考察周的时间 （3）用户-店铺对上一次点击距考察周的时间 （4）用户对品类的操作购买转化率
品类-店铺组合 C_S 类	（1）品类店铺下的购买评论转化率 （2）品类-店铺组合被购买的频次 （3）品类-店铺组合的点击购买转化率 （4）品类-店铺组合的收藏购买转化率 （5）品类下店铺新品的数量
用户-类别-店铺组合 U_C_S 类	（1）用户对品类下店铺的最近一次点击距离考察周的时间 （2）用户对品类下店铺的最近一次购买距离考察周的时间 （3）用户对品类下店铺在考察周前 n 天的行为总量

根据 XGBoost 特征重要性分析，排名前 30 的特征如下图：



B. 特征是如何想到和获取的？

（1）由于用户行为对购买的影响随时间减弱，根据分析，用户在两周之前的行为对考察周是否购买的影响已经很小，故而只考虑距考察周两周以内的特征数据。

（2）由于数据来源于垂直电商，其特点是线上购买线下消费，猜测其购买行为具有一定的周期性，进一步猜测行为周期为两个星期，同时统计其前1/3/5/7/10 天的各种行为。

（3）由于问题已被明确为 U-C-S 是否发生购买行为（标记 label 取｛0，1]）的分类问题，最终的特征数据均要合并到生成以 U-C-S 为 index（key）的样本集上来。进一步地，如要考虑所有可能的 U-C-S ，必将面临组合爆炸的问题，所以这里只关注在距考察周两周以内出现过 U-C-S 。

（4）针对当前业务背景，考虑从 user、item、item_category 三大基本维度及其组合入手进行特征构建，简称 U、I、C。

- **用户属性特征：**用于考察用户个人身份与其消费行为习惯之前的关系，类别型特征进行了分箱独热。
- **品类特征：**重点考察每个类别在所有商品类别中的江湖地位、影响力与受欢迎程度。
- **店铺特征:**重点考察店铺的受欢迎程度，评价情况，店铺中新品上市情况。
- **组合类特征：**分别对以上三种类别特征在业务关联上组合求出数值型特征、频次型特征、时间型特征。
- **时间滑窗特征：**结合 2 周训练数据进行以上 7 类特征分别进行考察周前 1、3、5、7、10 天的数据特征汇总。

- **派生特征：**利用时间滑窗特征生成的统计型特征对其多个时间范围内的统计量做减法操作，得到时间间隔区间统计特征。

C. 特征之间是否有相关关系？

在特征选择过程中，我们从两个维度进行，一个是皮尔森相关系数的判断，另一个就是信息增益的角度。相关系数侧重的是相关性，而信息增益和基尼指数可以挖掘深度的相关性，比如某个特征在一定量其他特征划分之后才起到比较好的划分作用。比如我们使用皮尔森相关系数进行特征的相关系分析时发现，某些特征在 XGBoost 中特征重要性很高，然而在相关性分析中却接近于 0，故在特征提取过程中，不能单纯依靠一个指标进行判断，应结合具体场景联合分析。

D. 特征是否经过处理？

由于采用的是树形模型，XGBoost 不依赖于数值型特征具体范围，故没有对特征进行归一化处理，仅对于离散化的类别特征进行了独热处理。通过特征重要性分析，发现部分统计型特征重要性较低，但是经过特征之间的加减之后构造出新特征，新特征的重要性大大提高，对模型也有不少的提升。估对于类别过多且相关性较低的离散特征和重要性较低的数值型特征进行了加减法后再删减。

4）模型选择与训练

A. 为什么选择这个模型？

基于树的算法通常抗噪能力更强,我们很容易对缺失值进行处理。除此之外，基于树的模型对于 categorical feature 也更加友好。Tree boosting（树提升）已经在实践中证明可以有效地用于分类和回归任务的预测挖掘。

B. 模型的训练方式？

在前期的尝试中，由于对加购物车数据的执念，一直坚持只使用了方案一进行训练，在训练过程中随机划分了 20%左右样本进行作为验证集，采用'binary:logistic'方式输出目标预测概率，评估指标采用'auc'，训练轮次 500 次。A 榜结束前一周成绩一直徘徊在 0.055 左右，采用 grid search 对模型参数进行微调，鉴于效果没有明显提升以及线上提交次数的限制，转而团队继续对数据进行分析挖掘更多的特征。由于后续几个强特（如购买时间间隔等）的加持，A 榜结束时，团队成绩上升到 0.0583 左右，B 榜提交相同文件，成绩大概 0.06 左右，我们猜测 B 榜数据量相对多一些。进入 B 榜后，由于在交流群中讨论时意外地发现一些细节：前排有大佬竟完全未使用购物车数据也能有很好的训练效果，故我们决定抛开执念，重新构造训练集，由于没有购物车数据的困扰，数据可选择范围变开阔，我们重新构造了方案二的训练集，但是此时 B 榜的提交验证机会只剩两次，兼顾保守与自信，我们并没有完全摒弃之前的数据集与模型，而是采用了模型融合的方式。

C. 是否进行了模型融合？模型的融合方式？

采用了多 XGBoost+Bagging 的结果融合方式，在融合之前我们采用方案二中训练集 1,2,分别训练，并使用训练集 3 做验证得到每个模型恰当的概率阈值和正样本个数，然后分别用训练集 1,2,3,2+3 和方案一中训练集分别训练模型并预测考察周的购买情况以及输出正样本个数，最后采用 Bagging 投票器思想对 5 个预

测文件进行少数服从多数（至少 3 票）投票输出最终的提交文件,线上最终得分 0.0673, 相对方案一单模型最好成绩提升 4 个千分点。

5） 说明文档

A. 描述编译/运行预测代码需要的资源和库以及版本备注；

我们的运行环境 Centos7, RAM 128G

软件库版本： Python3.7, pandas:0.24.2, numpy:1.15.4, matplotlib:3.0.2, sklearn:0.20.3, xgboost:0.90, pickle:4.0

B．代码使用说明：如何才能运行提供的预测代码。

工程目录说明：

本团队整个代码的工程目录如下：

```
优生 801_Rank6_Code
|__ code
|       |__data_preprocessing.py
|       |__make_label_2_1.py
|       |__make_features_1_4.py
|       |__make_features_2_4.py
|       |__make_features_3_4.py
|       |__make_features_test.py
|       |__make_features_a_4.py
|       |__make_features_a_test.py
|       |__model_train_1_4.py
|       |__model_train_2_4.py
|       |__model_train_3_4.py
|       |__model_train_a_4.py
|       |__model_train_23_4.py
|       |__model_fusion.py
|
|__ data
    |__ original_data:
    |__ processsed_data
    |__ output
    |   |__feature_importance
    |   |__featureMap
    |   |__model
    |   |__predict
    |__ submit
    |__ features
        |__features_1_4
        |__features_2_4
```

```
|__features_3_4
|__features_a_4
|__features_test
|__features_a_test
```

代码目录说明

- code

存放该工程的所有代码文件，该目录下包含六部分的代码文件，分别为数据预处理的代码文件、构造标签集的代码文件、有加购的特征构造的代码文件、无加购的特征构造的代码文件、模型训练的代码文件以及模型融合的代码文件。

-- data_preprocessing.py

数据预处理,原始的数据文件包括行为数据（jdata_action），评论数据（jdata_comment），商品数据（jdata_product），商家店铺数据（jdata_shop），用户数据（jdata_user）。分别对这五个文件进行缺失值填充，时间分桶等处理，然后把经过预处理的文件存放到 data 目录下的 processsed_data 目录。

-- make_label_2_1.py

标签集构造，该代码文件主要用于无标签数据打标签，通过时间划窗得到不同的标签集，用于构造不同的训练集。

-- make_features_a_4.py
-- make_features_a_test.py

有加购的特征工程，针对当前业务背景，考虑从 user、cate、shop_id 三大基本维度及其组合入手进行特征构建,简称 U、C、S。这里将所需构建的特征分为七大类：U、C、S、UC、US、CS、UCS，对每类分别结合行为次数、时间、排序等视角设计特征。考虑到样本规模，特征数量不宜太少，这里我们设计了 200 多个特征来进行该比赛的数据任务。

-- make_features_1_4.py
-- make_features_2_4.py
-- make_features_3_4.py
-- make_features_test.py

无加购的特征工程，针对当前业务背景，考虑从 user、cate、shop_id 三大基本维度及其组合入手进行特征构建,简称 U、C、S。这里将所需构建的特征分为七大类：U、C、S、UC、US、CS、UCS，对每类分别结合行为次数、时间、排序等视角设计特征。考虑到样本规模，特征数量不宜太少，这里我们设计了 200 多个特征来进行该比赛的数据任务。

-- model_train_1_4.py
-- model_train_2_4.py

-- model_train_3_4.py
-- model_train_a_4.py
-- model_train_23_4.py

模型训练，5 个训练文件对应 5 个单模型，（1）把特征工程文件构造的七大维度特征拼接成训练集和测试集；（2）分别对不同的训练集进行模型训练，这里主要有 5 个单漠型，对应不同的训练集，输出不同的预测结果；（3）根据（2）训练的五个模型预测出来的结果进行投票融合，遵循少数服从多数原则。

数据目录说明

- data

该目录存放原始数据、预处理后数据、提取的特征、训练输出的文件以及最终提交的文件。

-- original_data
该目录存放原始的数据。

-- processsed_data
该目录存放经过预处理后生成的数据。

-- output
|__feature_importance # 存放生成的特征重要性文件
|__featureMap # 存放生成的特征图
|__model # 存放训练好的模型
|__predict # 存放预测概率的输出文件

-- features
|__features_1_4 # 存放模型 1 生成的特征文件和对应的训练集
|__features_2_4 # 存放模型 2 生成的特征文件和对应的训练集
|__features_3_4 # 存放模型 3 生成的特征文件和对应的训练集
|__features_a_4 # 存放模型 5 生成的特征文件和对应的训练集
|__features_test # 存放生成的无加购的测试集特征文件
|__features_a_test # 存放生成的有加购的测试集特征文件

-- submit
该目录存放最终提交的数据文件。

代码运行步骤：

1. 首先将原始数据解压到 data 目录下 original_data 目录，original_data 目录下直接对应 5 个原始文件。
2. 数据预处理和特征工程
a、先运行 data_preprocessing.py 对缺失值和时间进行处理
b、步骤 a 运行完后，接着运行 make_label_2_1.py 构造对应的标签集。

c、步骤 b 运行完后，同时运行 make_features_1_4.py，make_features_2_4.py，make_features_3_4.py，make_features_a_4.py，make_features_test.py，make_features_a_test.py，生成对应模型的特征文件。

注：生成的文件都在 data/features 目录对应的目录文件下；为了节省时间，等步骤 b 运行完后，构造特征的六个文件可以同时运行，结果会输出到对应的模型特征文件目录下；

3. 模型训练

上述代码全部运行结束后，依次运行 model_train_1_4.py，model_train_2_4.py，model_train_3_4.py，model_train_a_4.py，model_train_23_4.py 这 5 个模型训练文件，代码将先对 5 个模型对应的七大维度特征文件进行拼接，生成对应的训练集和测试集文件；接着分别对 5 个模型进行训练，模型训练结束后输出对应的预测概率文件；

4、模型融合

5 个模型文件全部训练结束后，接着运行 model_fusion.py 文件，最后根据投票融合的方式，对五个模型输出的预测结果进行融合，得到最终的要提交的预测结果，最终输出提交的文件存放在 data/submit/目录，文件名为 predict_fusion.csv

6）有趣的发现

A. 使用的小技巧？

“特征工程决定了模型准确性的上限，而参数调优只是不断逼近这个上限。”由于深度学习（神经网络）对机器性能的要求较高，故我们只能在特征工程上多花功夫，除了对数据集提供的特征做相关组合挖掘，我们也借鉴了诸多论文、博客对于相关业务领域的分析，争取在原有理解基础上派生出更加深层次的特征，诚然，我们在这个过程中确实得到了很多的启发。

C． 您觉得您最突出的优势是什么？

在模型的选择上，我们并没有投入太多的时间，可能错过了更有效的模型，但是这可能恰恰是我们的优势，我们投入了大量的时间在特征的构建和特征的选择上，我们对于业务场景的知识做了深入的学习和理解，这也是我们能够找到强特征并提升分数的前提。最后，在模型的融合上，我们进行了 XGBoost 多模型+Bagging 投票器的融合也使得我们的模型更加稳定。

D． 每次成绩提升改进说明。

每次成绩的显著提升，基本上都是在找到新的强特之后，若以分数作为参考系，提升进程大概如下：

版本	线上分数	改进说明
V1	0.054	加入基础的 7 类特征（用户属性 U，品类 C，店铺 S 以及三者的轻度交叉组特征合）

V2	0.0583	继续围绕 U,C,S,UC,US,UCS 这七个维度的特征进行优化和扩展
V3	0.063	新加入 U、UC、US、UCS 点击购买距离考察周时间间隔以及平均购买周期等时间型特征，构造了验证集和验证函数，确定模型较为恰当的阈值
V4	0.0673	对方案 1 和 2 的 XGBoost 预测结果进行投票融合