

队伍编号	dsa2101058
题号	B

在线少儿教育平台用户消费行为价值分析与营销建议

摘要

本文对购买在线教育平台体验课的用户属性及浏览行为数据进行分析，通过机器对数据进行深度学习从而预测用户的购买行为并给出营销建议。

本文首先对数据进行了预处理和分析，初步掌握了数据分布和概况，在初步尝试各类模型效果后决定采用以 XGBoost、RandomForest 为第一层分类器，Logistic Regression 为第二层分类器的融合模型，将各类数据联合处理后进行超参数调整与训练，最终准确率为 99%，f1 值为 85%，达到预期标准。

研究搭建了准确率超过 85% 的销售预测模型，该模型有助于企业判别高质量的用户和渠道，优化营销成本。(1)高效识别有效客户，并采用精细化的客户运营；(2)对自身软件设计进行反思，完善自身功能；(3)发现运营中的重点环节，针对性营销以降低成本提高利润率。

关键词：深度学习；模型融合；分类器；销售预测模型

目录

一、问题分析.....	1
二、数据侧写.....	2
2.3 缺失值处理.....	5
2.4 异常值处理.....	6
2.5 冗余属性处理.....	6
2.6 编码调整.....	7
2.7 数据平衡.....	7
三、可视化分析.....	7
3.1 针对登录情况的可视化分析.....	7
3.2 针对城市情况的可视化分析.....	9
四、模型构建.....	12
4.1 模型评估指标选择.....	12
4.2 模型选择.....	12
五、第一层分类器设计.....	13
5.1 方案 A：一分类.....	13
5.2 方案 B：二分类.....	13
六、第二层分类器设计.....	16
6.1 模型调参.....	16
6.2 模型效果分析与解释.....	17
七、分析与建议.....	18
7.1 针对用户方向的建议.....	18
7.2 针对平台的建议.....	21
附录-核心代码.....	23

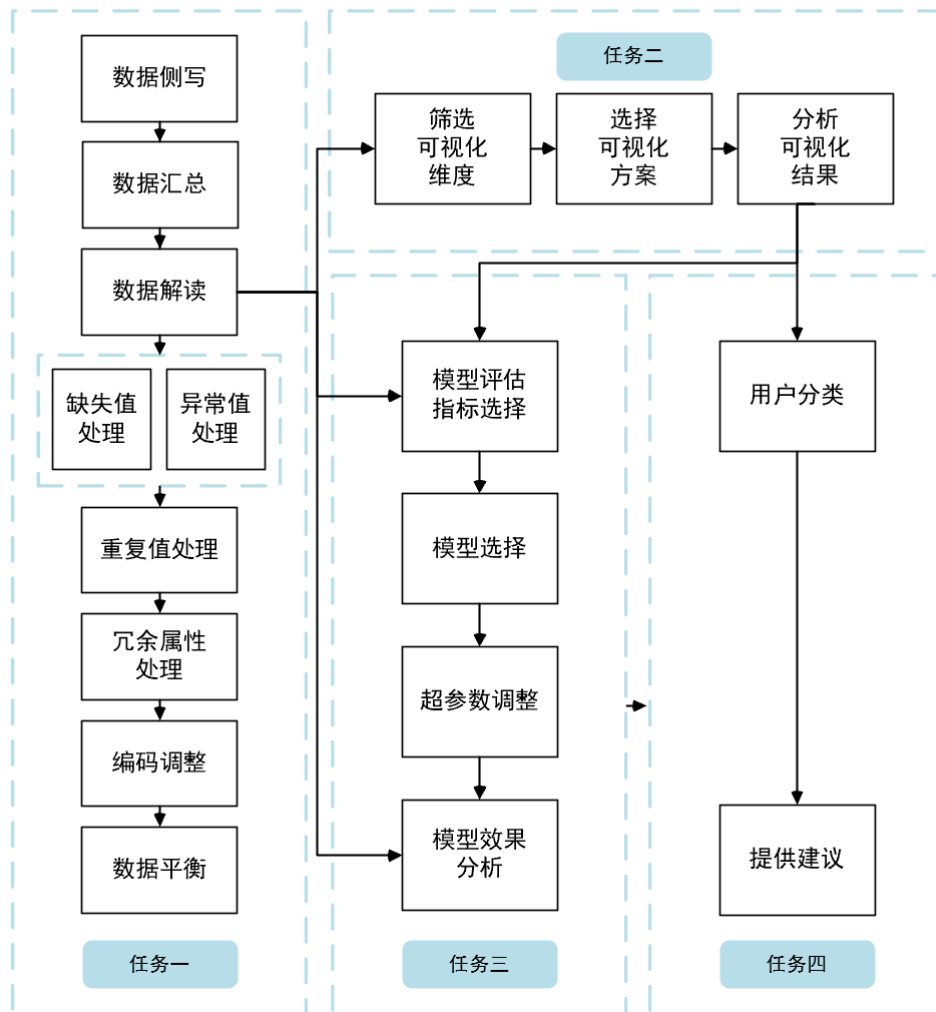
一、问题分析

对于在线教育平台，研究用户消费行为以进行数据化运营是非常有价值的，对于进行客户关系管理、用户漏斗模型分析并优化运营策略有着极大的帮助。数据化运营是飞速发展的数据挖掘技术、数据储存技术等大数据环境下各种数据分析技术推动的结果，是互联网行业推动企业成长的利刃。互联网行业相对于其他行业，较大的一点区别就在于其用户在产品上的每一个行为都可被记录并加以分析。本文围绕着该在线教育平台的用户数据，针对客户的特征属性进行描述性统计，并使用数据挖掘技术搭建出用户消费行为价值模型。因此，本文的主要目标就是充分分析企业的用户行为数据，从中提炼出有用的变量，并利用这些变量建立一套合理有效的用户购买行为预测模型。

根据此模型能够对用户的购买行为进行预测，结合模型中高度相关的变量进行可行有效的营销实操，帮助企业提高销售额和用户转化率。此外，该模型也有助于企业搭建客户关系管理系统，提高利润率。

本文总体思路如下：

图表 1 研究思路路线图



二、数据侧写

由题意可知，本题目为典型的二分类问题，因此提前对数据进行侧写，从宏观上把握数据的整体分布、摸清各属性之间的联系和特征有利于分类精确度的提升。由于附件中所提供的各表包含的信息与最终用户是否购买情况息息相关，本节将所提供的数据进行总体观察。使用 `pandas_profiling` 和 `pandasgui` 库对附件所提供数据的分布情况进行总体分析结果如下



The screenshot shows the pandasgui interface for the file 'user_info.csv'. It has three tabs: 'Overview' (selected), 'Reproduction', and 'Warnings' (with 10 warnings). The 'Overview' tab displays two tables: 'Dataset statistics' and 'Variable types'.

Dataset statistics	
Number of variables	8
Number of observations	135968
Missing cells	28209
Missing cells (%)	2.6%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	8.3 MiB
Average record size in memory	64.0 B

Variable types	
num	4
CAT	3
BOOL	1

图 1 user_info.csv

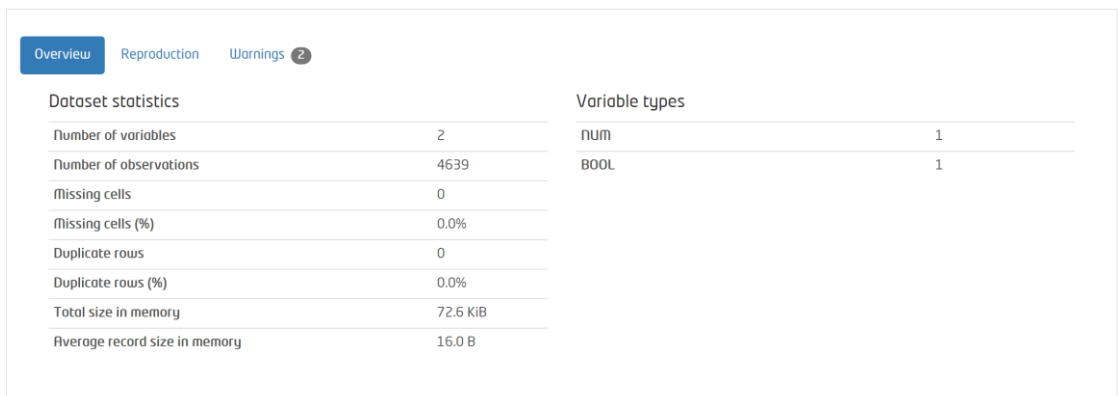


The screenshot shows the pandasgui interface for the file 'Logindays.Csv'. It has three tabs: 'Overview' (selected), 'Reproduction', and 'Warnings' (with 15 warnings). The 'Overview' tab displays two tables: 'Dataset statistics' and 'Variable types'.

Dataset statistics	
Number of variables	16
Number of observations	135617
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	16.6 MiB
Average record size in memory	128.0 B

Variable types	
num	11
BOOL	5

图 2 Logindays.Csv



The screenshot shows the pandasgui interface for the file 'Result.csv'. It has three tabs: 'Overview' (selected), 'Reproduction', and 'Warnings' (with 2 warnings). The 'Overview' tab displays two tables: 'Dataset statistics' and 'Variable types'.

Dataset statistics	
Number of variables	2
Number of observations	4639
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	72.6 KiB
Average record size in memory	16.0 B

Variable types	
num	1
BOOL	1

图 3 Result.csv

Overview		Reproduction	Warnings 30
Dataset statistics		Variable types	
Number of variables	26	NUM	25
Number of observations	135617	BOOL	1
Missing cells	0		
Missing cells (%)	0.0%		
Duplicate rows	0		
Duplicate rows (%)	0.0%		
Total size in memory	26.9 MiB		
Average record size in memory	208.0 B		

图 4 Visit_info.csv

同时为了检查数据完整情况，使用 `seaborn` 绘制数据缺失分布图,其中含有缺失数据的数据集展示如下，其中蓝色部分代表没有缺失的数据，黄色部分代表缺失数据：

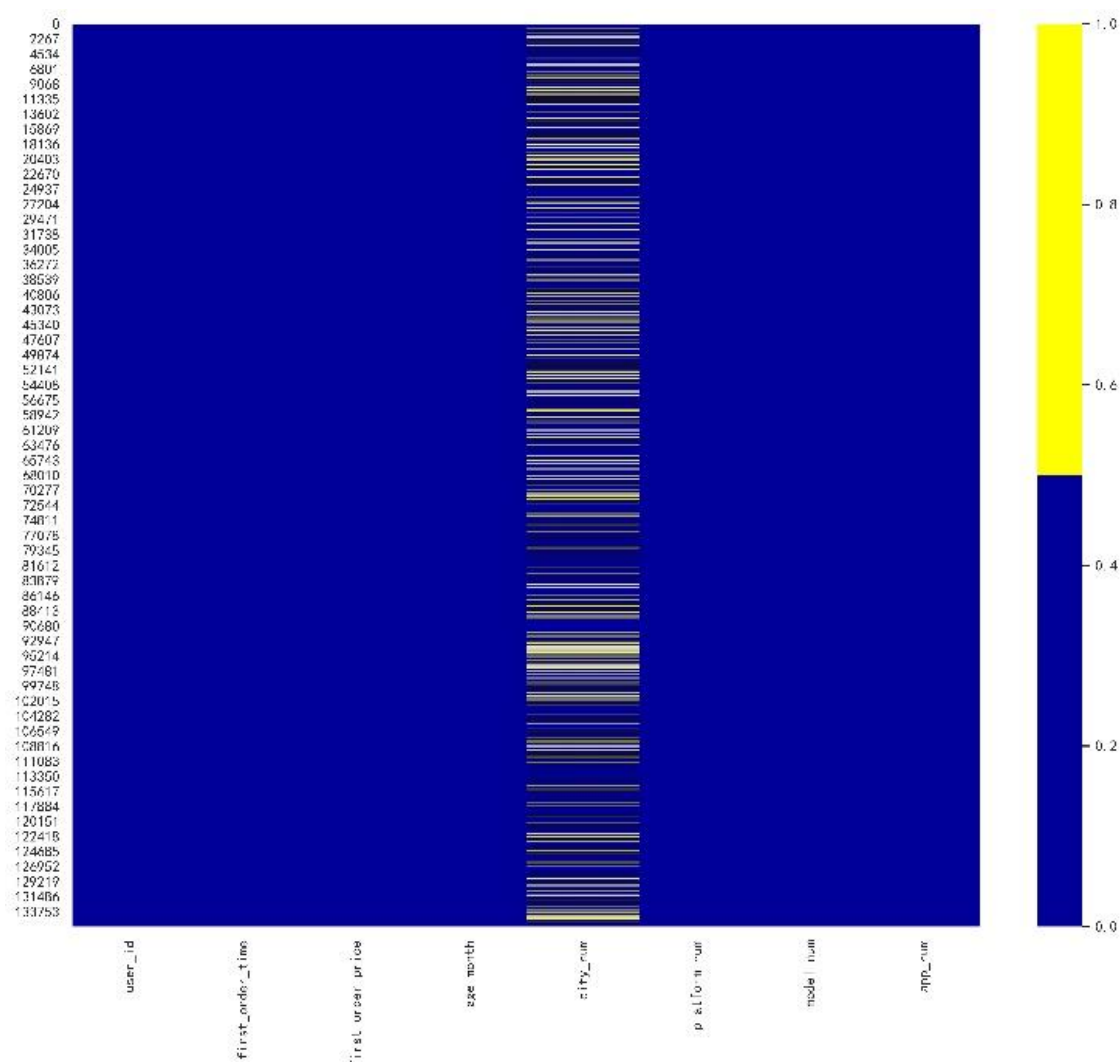


图 5 数据缺失分布图

为展示各变量之间的相关性关系，使用 `python` 绘制相关性散点图，部分展示如下：

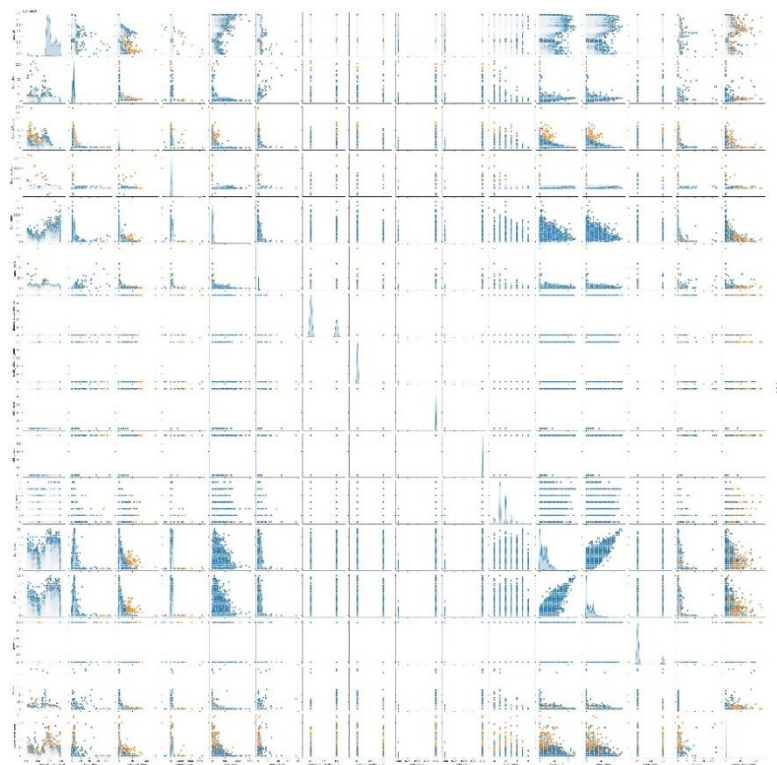


图 6 数据汇总

因为要根据用户的各种行为和属性预测最终的购买意愿，用户的信息维度需要尽量全面，同时在数据侧写中可以观察到各表中用户 id 大致可以一一对应。

因此以用户 id 作为目标匹配属性，将 login_day,user_info,visit_info 三个表进行连接，并添加 result 表中用户的最终购买属性，令购买用户属性值为 1，其余未购买用户属性值为 0，合并成总表 datamerge.csv。

使用 seaborn 绘制合并后的数据缺失分布图如下所示，其中蓝色部分代表没有缺失的数据，黄色部分代表缺失数据。可以发现各表之间缺失数据较少，有汇总的价值。

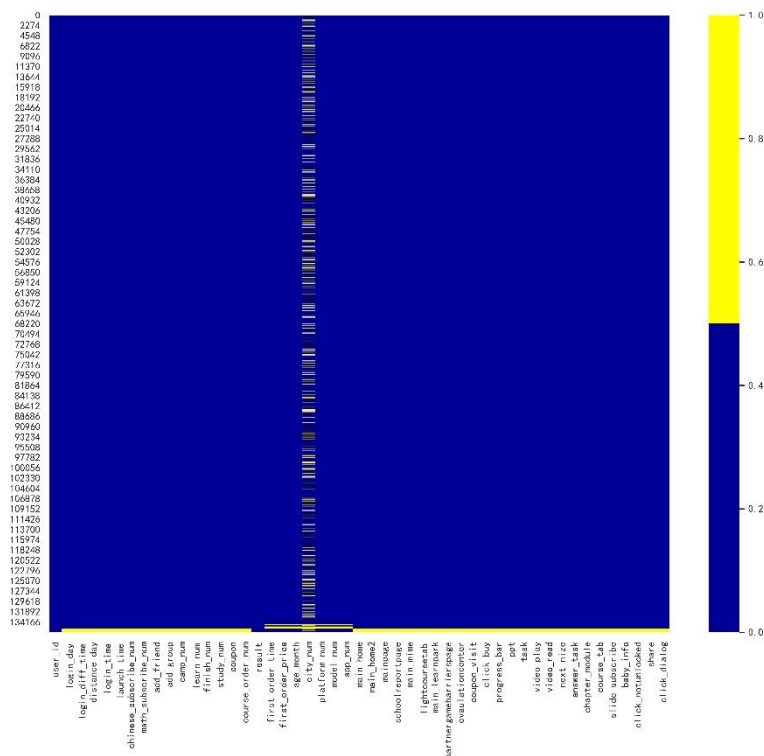


图 7 数据解读

详细分析数据的规律，我们对数据源可以得出以下模型假设：

1. 用户均在中国境内，且其基本属性、用户行为与是否购买之间的关系具有一致性，可以用单一模型（组）统一预测，不需要针对性建模。
2. 登录间隔“-1”值，含义为“注册后从未登录的不活跃用户”
3. 由于大量登陆时长数据超过 24，因此认为登陆时长单位为分钟
4. 由于大量年龄值超过人类平均年龄，且该属性数值除以 12 后，大多分布在普遍受教育阶段，因此认为年龄属性单位为月。
5. 由于存在少数“访问数、领券数量”大大超出正常值的用户，推测该数据集内可能含有刷单用户或管理员测试用账户。在后续操作中需要进一步筛查。
6. 距离期末时间含有负值，推测是用户在采集数据时间节点过后再次产生数据导致，非异常值。

2.3 缺失值处理

根据缺失值图，可以发现 20.7%的用户缺失了“city_num”属性，为保证数据的全面性，对此类用户属性统一补充为“未知”

同时极少量用户缺失了大部分的属性，由于占比大约为 0.9%，且丢失属性过多，不具备预测的价值，因此这部分数据可以直接丢弃。保险起见，计算待丢弃用户的购买期望值，并与总体期望值对比，发现大致符合，即该类用户非特殊群体，可以放心删除。

2.4 异常值处理

对各个维度数据绘制箱线图，针对其中的长尾数据进行分析，发现存在属性存在长尾分布和异常值，部分可视化展示如下：

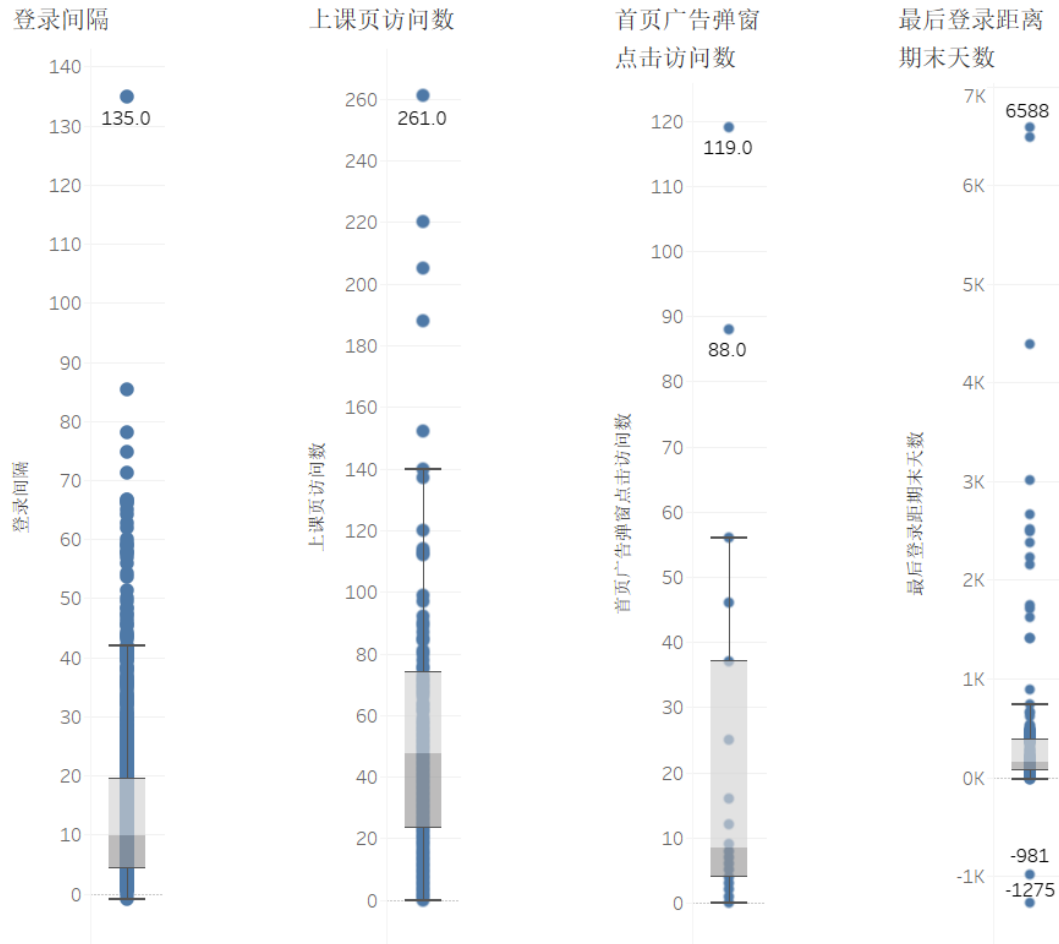


图 8 长尾数据分析

因此在可视化时，需要对具有明显异常值的属性进行筛选，对于难以区分特殊值和异常值的属性，需要进一步做分箱化处理，其中将过于超出正常范围的个体归入“非正常”类别。

而对于模型预测来说，如果使用基于决策树的模型，具有长尾分布的属性中，特殊数据也是显著的特征，盲目丢弃或分箱会造成精度下降，因此对于用来预测数据中的异常值不作清洗。

2.5 冗余属性处理

根据平台总览结果，`app_num` 结果为全 1，即所有用户都激活了 `app`，因此该属性和最终是否购买之间没有任何关联，对可视化和模型训练均无贡献，需要删除。同样，`click_buy` 属性经过异常值处理后也为全 0，需要删除。

2.6 编码调整

根据总览结果可以发现，city_num 属性为文本数据，first_order_time 属性为时间序列数据，均难以直接用于模型预测，因此将 city_num 属性转化为 onehot 编码，first_order_time 转化为时间戳，即相对于格林威治时间 1970 年 01 月 01 日 00 时 00 分 00 秒，北京时间 1970 年 01 月 01 日 08 时 00 分 00 秒的总秒数。

2.7 数据平衡

根据总览结果，最终购买数的用户和未购买用户比例为 28 比 1，属于严重偏态数据。在这种情况下，即使全部预测为不购买，也有 96.6% 的准确率，因此使用合成少数类过采样技术进行平衡。

其基本思想是通过对少数类样本进行散布分析，并根据分布规律人工合成新样本数据添加到数据集中，以达到平衡分布的目的。因为主要使用 KNN 的方法合成新样本，而不是简单的进行随机复制，因此其结果更具有代表性。

其算法伪代码表示为：

当数据集中数据不满足设定条件时：

 随机取数据集中的 n 个少类样本

 对于 n 个目标点中的每个点：

 使用 KNN 方法分别找出离该点最近的 m 个少类样本

 在这 m 个少类样本中任取一个点，与该点进行随机线性插值

 将所生成的点加入原数据集

为增加随机性，使训练的模型更具有普适性，并保证测试集的数据均为真实数据，本文中所有数据平衡的地方均只针对训练集做处理，且均在训练前生成随机数据。

三、可视化分析

3.1 针对登录情况的可视化分析

根据下方箱线图能够得出结论：购买的用户相对于未购买的用户在软件上投入了更少的时间。对于这部分时间寻找对应的用户行为，根据以下访问数对比条形图可以得以体现，首页访问数直接表明用户在软件内的探索情况。未购买的用户访问了人均访问了超过 56 次首页，而购买了的用户却只访问了不到 48 次。

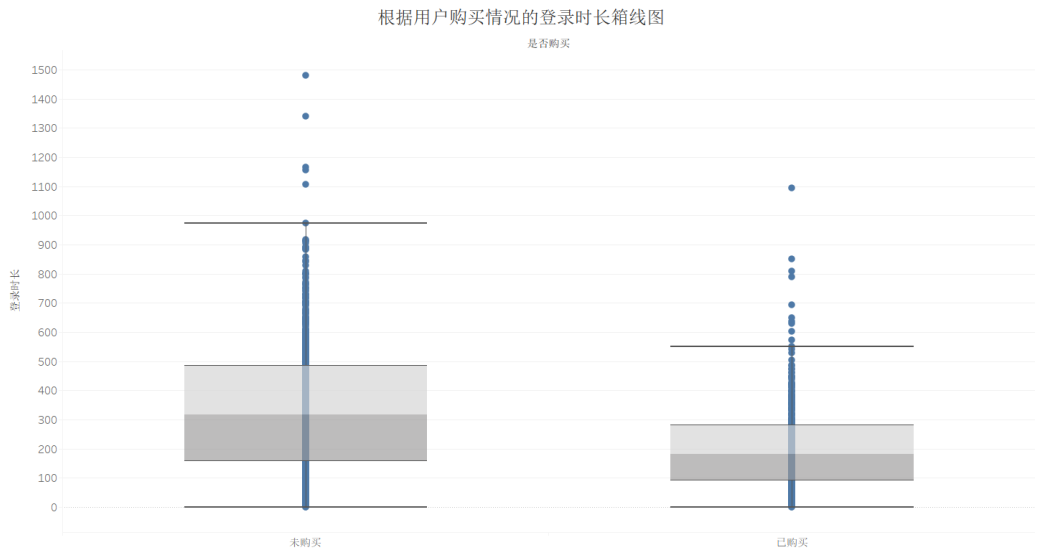


图 9 根据用户购买情况的登录时长箱线图

而在用户具体页面的访问行为上：

用户在【课程】这一系列功能上使用最多，包括拖动进度条、PPT 下一步、界面继续、视频跟读、识字等。这一些列访问行为构成了用户的基本需求，也是软件本身的核心内容。其次发现用户的其他访问行为相对于基础需求大量不足，如【反馈】系列功能所包含的测评中心、课程计划等，又如【增长】系列功能所包含的分享、领券等等。

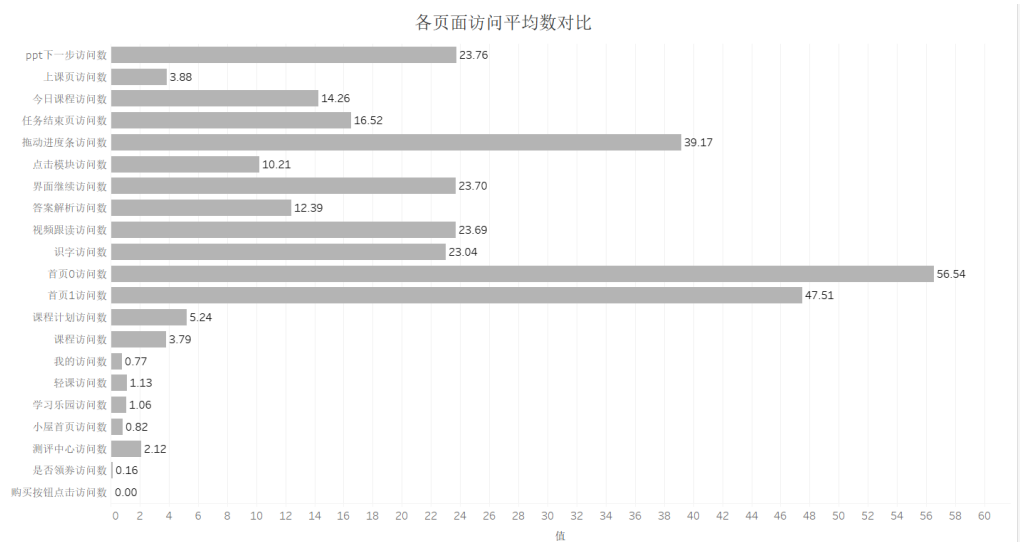


图 10 各页面访问平均数对比

另一方面，由下图所示，最终购课的用户相对于未购课用户的重复学习行为显著,可以认为这是两个群体的关键差异行为。

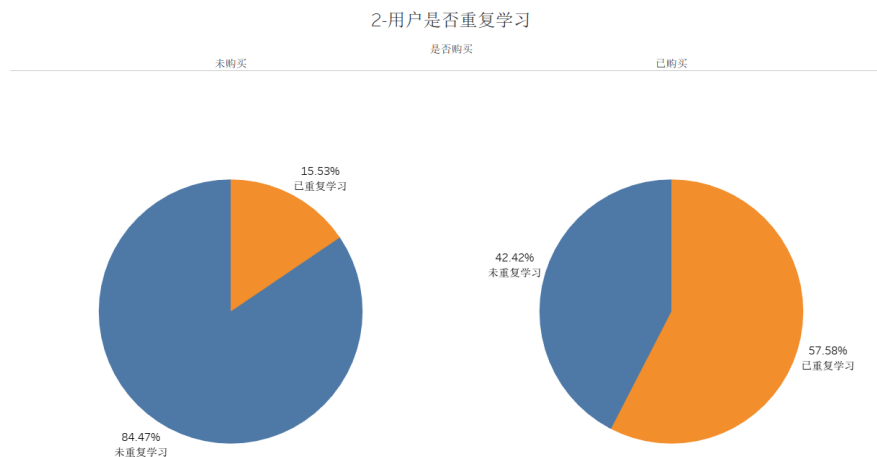


图 11 用户重复学习情况

综上所述，有以下推测：

购买的用户更多关注于【课程】相关功能，而未购买的用户较多地浏览了 APP 的其他页面，所以会访问更多的首页。而 APP 本身可能在课程设计上没有太大问题，但在功能设计上或许没有充分考虑用户的使用习惯，存在分类不适配、交互界面不协调等问题，导致了较差的用户体验，于是在 APP 中探索其功能的用户满意度相对于专注于课程本身的用户更低，从而降低了购买意愿。

3.2 针对城市情况的可视化分析

数据集共涉及来自 362 个城市的 136426 名用户，其中下单购买的用户共有 4639 名。

如图是用户数量排名前十的城市柱状图，重庆的用户数量远超其他城市，达到了 12411 人，约为排名第二城市成都的 3.5 倍。

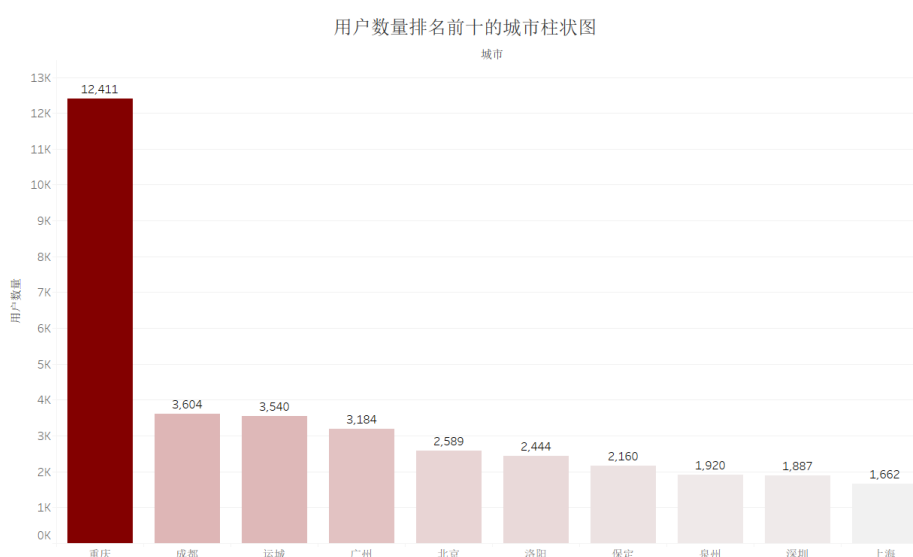


图 12 用户数量排名前十的城市柱状图

从下方的用户城市分布图可以看到，用户基本集中在以华中为主，遍布华南、华东、华中、华北的地区，以沿海、国内发达城市为核心有明显的聚集性，用户集中趋势较为明显。

用户城市分布图



图 13 用户城市分布图

推测用户的城市分布存在以上显著情况是由于该教育企业不同区域的资本化程度差异较大，受融资环境影响，企业在对应融资地段的投资也相应较多。据普华永道行业洞察报告¹表示，北上广深地区受到投资的力度较大，故企业可能在这部分地区投入较大的营销资源，与此同时，有调研表明线上教育也面临信息技术鸿沟²，发达地区的用户更能接触到线上教育资源，并持有更高的认同度与适应性。

¹ [1] 普华永道，2016 年至 2020 年中国教育行业并购活动回顾及趋势展望，
https://mp.weixin.qq.com/s?src=11×tamp=1622265718&ver=3097&signature=TL00JPZiR0-ibs0srkCxVcmVbaUr6tBli9-iFE3xwg2yj1WG8CM1H1f1nWTOWDSNX7PzduesNQvm7VzfNkQe5gx5oVLD8VqGppgY4H7F4vMobF*koa2kDQv4n-Fd2TON&new=1，
2020/05/29.

² [2]赵宏,蒋菲,汤学黎,甄志平.在线教育：数字鸿沟还是数字机遇?——基于疫情期间在线学习城乡差异分析[J].开放教育研究,2021,27(02):62-68.

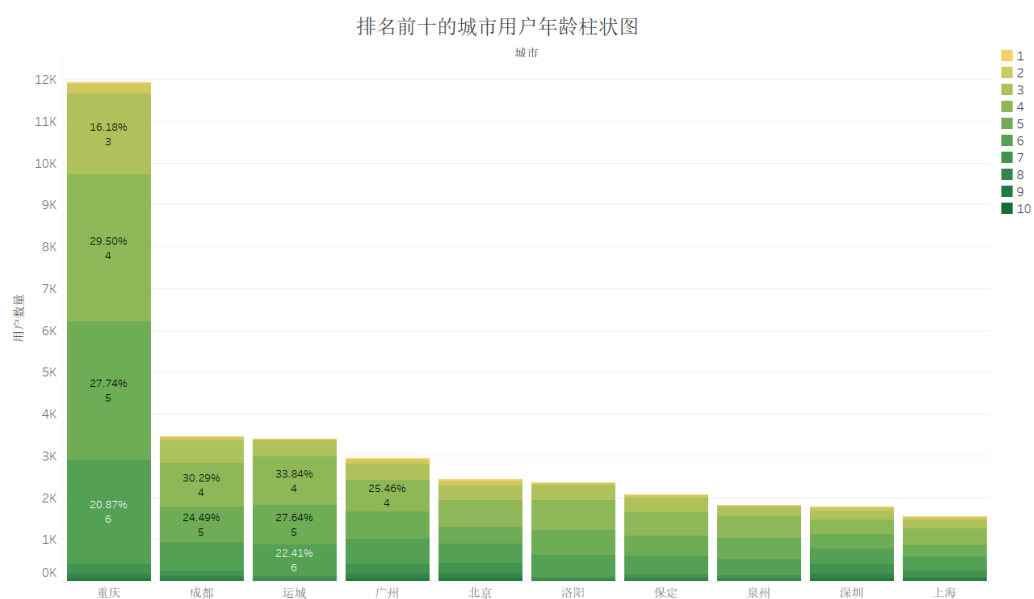


图 14 排名前十的城市用户年龄柱状图

城市用户的年龄分布情况随城市发达水平有所下沉，以重庆为首有 45%以上的用户的年龄处于 60 个月以下的年龄阶段，说明越发达的城市用户越重视教育，开启早教培训的时间会略高于其他城市用户。

1. 重庆市用户年龄平均为 66 个月，中位数为 60 个月。
2. 成都用户年龄平均为 61 个月，中位数为 59 个月。
3. 运城用户年龄平均数、中位数均为 60 个月。



图 15 设备分布地图

事实上，地区的发达水平决定了居民的经济实力，而经济实力在一定程度上可以由设备情况反映。如图，用户数量较多的地区标号为 **13557** 的设备使用的占比更多，而这款设备相对于其他设备售价略高，可以认为经济实力越高的群体更容易使用本研究在线教育平台。

四、模型总体构建

4.1 模型评估指标选择

由于数据分布严重偏态，并且本文目的在于判断可能会购买的高质量客户，因此“将可能购买的用户识别为可能不购买”和将“可能不购买的用户识别为可能购买”这两个错误的严重程度是不一样的。仅仅使用准确率作为模型评估指标有失偏颇，因此本文加入了 **f1** 值作为模型准确率的参考。

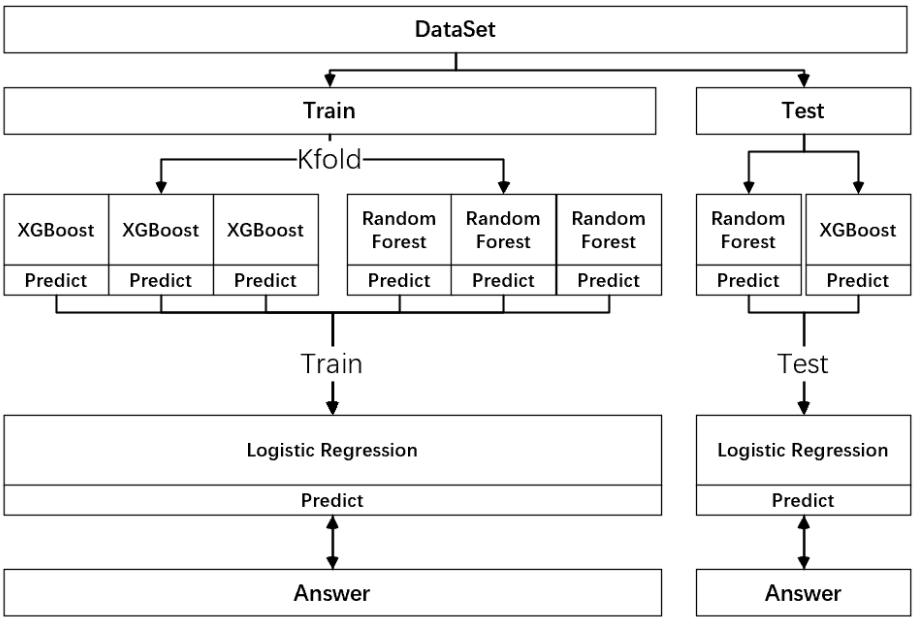
而在模型的训练集、测试集比例选择方面，由于该数据集在清洗后仍有 **1e5** 的数量级，属于大型数据集，为保证训练结果的普适性，设置训练集与测试集比例为 **9: 1**。

4.2 模型选择

模型融合是机器学习中的常用算法之一，主要分为硬投票和软投票两大类，向下细分则有 **stacking**、**averaging**、**bagging**、**boosting**、**stacking** 五类。

本文使用 **stacking** 方法进行模型融合。其主要思想是训练一个二级模型，并对多个一级模型的预测结果结果进行总结，以填补各个一级模型的缺点，获得更高的准确率。由于一般意义上的 **Stacking** 在训练时存在容易过拟合的缺点，因此在一级模型训练时采用 **3 折交叉验证**。模型总体流程图图如下所示：

图表 2 模型总体流程图



五、第一层分类器设计

5.1 方案 A：一分类

由于数据过于倾斜，故将用户尝试使用一分类模型进行异常值检测。

表 I One Class SVM

	准确率	召回率	F1值	支持度
未购买	0.97	1.00	0.98	117494
已购买	0.00	0.00	0.00	4150
准确率			0.97	121644
宏平均	0.48	0.50	0.49	121644
加权平均	0.93	0.97	0.95	121644

可以看出各维度评价值都很低，且该模型预测时间极长，为了在较短时间内得出训练结果，该模型训练集测试集之比为 9:1，综上此异常值检测模型不适用该场景。

5.2 方案 B：二分类

在数据预处理部分，因为长尾数据维度比例很高，且难以有效区别数据中的噪音和特殊值，如果盲目对其进行标准化，并使用线性模型进行分类，可能不会收到较好的准确值。以逻辑回归为例，初步训练模型结果如下。

表 II Logistics Regression

	准确率	召回率	F1值	出现次数
未购买	0.99	0.64	0.78	13071
已购买	0.08	0.89	0.14	445
准确率			0.65	13516
宏平均	0.54	0.77	0.46	13516
加权平均	0.96	0.65	0.76	13516

可以看出，虽然逻辑回归召回率高但准确率极低，属于“广撒网，多敛鱼”的策略，并不符合“小成本促销”的核心思想，因此不采用该模型。

若使用基于决策树的模型进行预测。初始选择 XGBoost、RandomForest、DicisionTree 作为备选模型。为了粗浅的比较各模型的效果，初步训练各模型结果如下。

表 III XGBoost

	准确率	召回率	F1值	出现次数
未购买	0.99	1.00	0.99	13012
已购买	0.84	0.65	0.73	503
准确率			0.64	13516
宏平均	0.92	0.82	0.86	13516
加权平均	0.98	0.98	0.98	13516

表 IV Decision Tree

	准确率	召回率	F1值	出现次数
未购买	0.99	0.99	0.99	13080
已购买	0.76	0.68	0.72	436
准确率			0.98	13516
宏平均	0.87	0.84	0.85	13516
加权平均	0.98	0.98	0.98	13516

表 V Random Forest

	准确率	召回率	F1值	出现次数
未购买	0.98	1.00	0.99	13076
已购买	0.99	0.27	0.42	440
准确率			0.98	13516
宏平均	0.98	0.63	0.70	13516
加权平均	0.98	0.98	0.97	13516

可以看出,随机森林准确率高,但召回率略低。XGBoost 在准确率和召回率方面均较优秀,普通的决策树两指标均低,在三个模型中分类效果垫底,因此不采用该模型。

综上,本文采用 XGBoost 和随机森林作为预测模型。

XGBoost 算法简介

XGBoost 模型核心公式如(1)、(2)所示

$$\hat{y}_i = \phi(\mathbf{x}_i) = \sum_{k=1}^K f_k(\mathbf{x}_i), \quad f_k \in \mathcal{F} \quad (1)$$

(1) 公式中, K 代表决策树的个数, \mathbf{x}_i 代表第 i 个样本, y_i 表示经 f_k 函数计算得到的预测值, \mathcal{F} 表示所有函数的集合。该公式将所有样本的预测值累加后得到目标值 \hat{y} 。

$$\tilde{L}(t) \cong \sum_{j=1}^T \left[\left(\sum_{i \in I_j} d_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} g_i + \lambda \right) w_j^2 \right] + \gamma N \quad (2)$$

(2) 公式对第 t 次迭代时的目标函数进行二阶泰勒展开，并移除常数项后所得。 t 为迭代次数。 $\hat{y}(t-1)_i$ 表示第 $t-1$ 次迭代时的预测值， d_i 表示函数 $\hat{y}(t-1)_i$ 关于 L 的一阶导数； g_i 表示函数 $\hat{y}(t-1)_i$ 关于 L 的二阶导数。 λ 、 γ 是为防止过拟合而设的比重系数。 I 表示第 j 个叶子节点上的样本集合， N 表示叶子节点的数量， w^2 为第 j 个叶子节点得分的 2-范数

以下是 XGBoost 的具体应用过程示例：

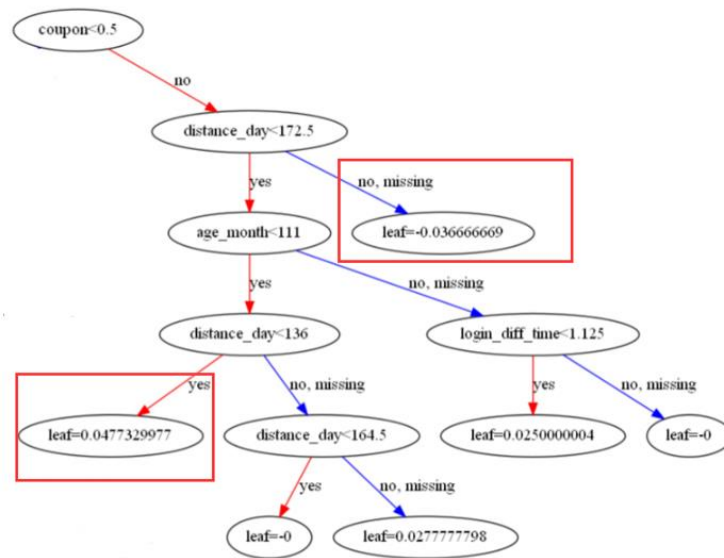


图 16-A XGBoost 应用过程

据 XGBOOST 模型其中一个预测分类树部分如上图所示，蓝色箭头表示抑制用户购买课程、红色箭头表示促进用户购买课程。例如，当领券数量 `coupon` 小于 0.5 时，用户最后登录距期末的天数（`distance_day`）若大于等于 172.5 天，计算出 `leaf` 值为负，则模型判断该用户不会倾向于购买课程；若用户最后登录距期末的天数小于 172.5 天，且用户年龄大于 136 个月，计算出 `leaf` 值为正，则该用户倾向于会购买课程。

同理，XGBOOST 完整的预测分类树如下图所示，模型分析计算所有字段后预测出完整的用户是否购买课程的分类情况。

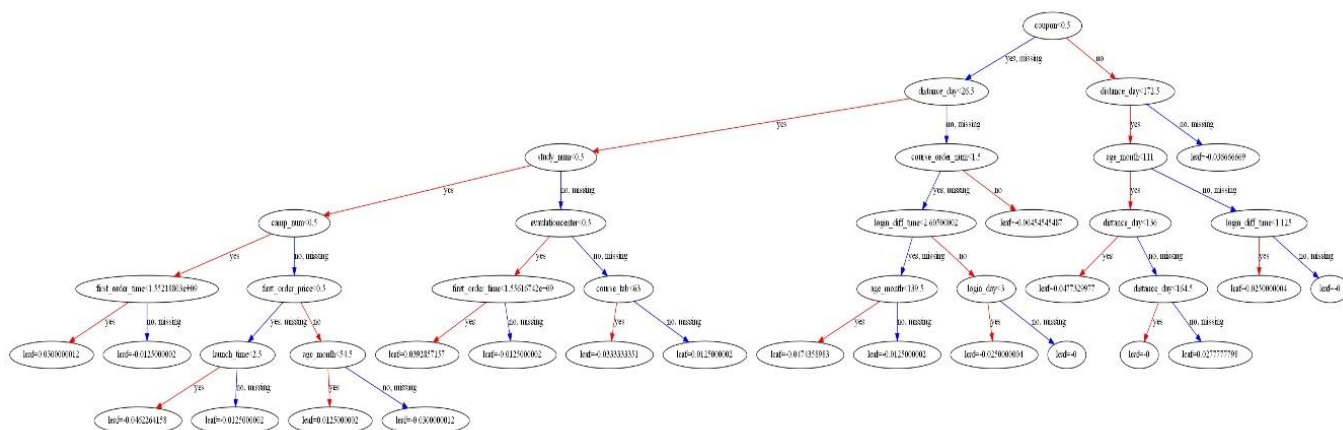


图 176-B 预测分类树

引用: Breiman L(2001)Random forests.Machine Learning 45:5–32

随机森林简介

随机森林是 Leo Breiman (2001) 提出的分类算法, 是对决策树算法的一种改进, 将多个决策树合并在一起, 每棵树的建立依赖于独立抽取的样本。其原理为从原始训练样本集中有放回地随机抽取 n 个样本, 并生成新的训练样本集合训练决策树, 再按照上述步骤生成含 m 棵决策树的随机森林, 新数据的分类结果按分类树投票多少形成的分数而定。

六、第二层分类器设计

第一层模型预测后, 需要一个总结性较好的模型进行融合, 由于第一层模型结果都较为优秀, 因此可以直接使用线性模型对预测结果总结以得出最终结果, 通过阅读文献可知第二层常用模型为逻辑回归。

由于训练结果一样是倾斜数据, 该数据集训练前也需要进行数据平衡的操作。但由于此次测试集相对第一层较大, 此处使用欠采样来平衡数据。经实验测试, 欠采样结果优于人工生成数据。

6.1 模型调参

在确定模型后需要对模型进行超参数调整以提升模型准确率。本文采用随机搜索调参粗校, 手动调参精校的方式对模型进行修整。

随机搜索是目前常用的超参数优化算法, 具有随机性, 快速性的特点。该算法基于网格搜索, 即每轮计算中选取的超参数数量固定。当超参数数量较大时, 搜索时间将成指数级增长, 非常耗时。

随机搜索在每一轮优化中, 随机选取不定量的超参数, 经过若干轮计算后一定能找到最优的超参数组合。当存在大量超参数待优化时, 其效率较高。

调参完成后各模型训练结果如下:

表 V 逻辑回归调参

	准确率	召回率	F1值	出现次数
未购买	0.99	1.00	0.99	131787
已购买	0.88	0.82	0.85	4639
准确率			0.99	136426
宏平均	0.94	0.91	0.92	136426
加权平均	0.99	0.99	0.99	136426

表 VI 随机森林调参

	准确率	召回率	F1值	出现次数
未购买	0.98	1.00	0.99	13059
已购买	0.96	0.42	0.59	457
准确率			0.99	13516
宏平均	0.97	0.71	0.79	13516
加权平均	0.98	0.98	0.98	13516

表 VI XGBoost调参

	准确率	召回率	F1值	出现次数
未购买	0.99	1.00	0.99	13067
已购买	0.84	0.75	0.79	449
准确率			0.99	13516
宏平均	0.92	0.87	0.99	13516
加权平均	0.99	0.99	0.99	13516

为了模型预测的普遍性，每次运行代码之前都需要随机生成数据，因此该模型的抖动较大，经测试各参数抖动变化率一般情况下不会超过 5%。

6.2 模型效果分析与解释

在模型解释方面，本文使用 Python 的 SHAP 库进行分析。

SHAP 是基于博弈论的为所有机器学习、深度学习提供一个解释的方案库，能评价树模型中的维度特征对于结果的贡献度。与特征重要性相比，SHAP value 不仅能反映出每一个样本内，各个维度特征的影响力，而且还表现出影响的正负性，是当前主流的机器学习模型最佳解释工具

使用 SHAP 对逻辑回归模型进行分析，可以发现 XGBoost 和随机森林都对最终结果产生了贡献，其中第一层各模型结果对最终结果影响比例和总体呈现如下：

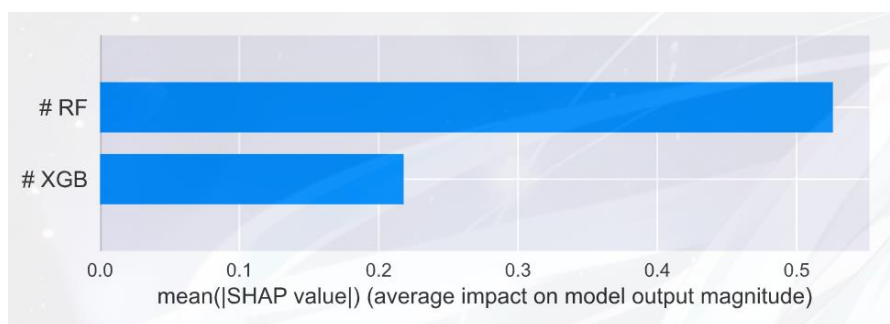


图 17-A 影响比例

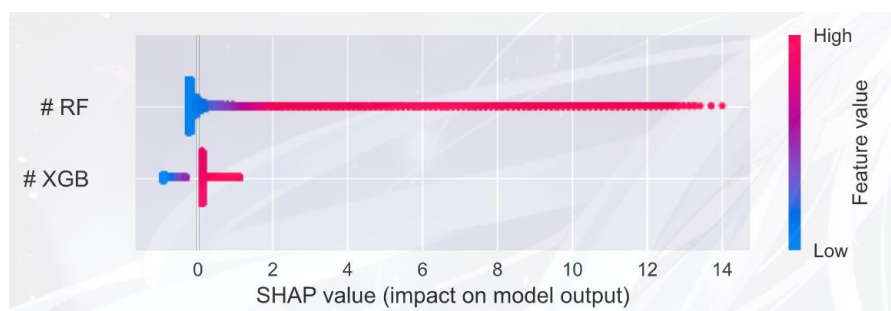


图 17-B 总体呈现

由于逻辑森林准确率高，即该模型判断为购买的用户几乎都购买了产品，结合图片可以发现随机森林模型中对这些用户的判断，在融合模型中的影响因子也很高。其他未识别出来的用户也离判别点很近，经过 XGBoost 模型贡献后也能精准补盲。

七、分析与建议

7.1 针对用户方向的建议

搭建企业数据中台，建立 CRM 部门对用户进行分类。

首先，依据 KNN 聚类算法，现将所有用户分为 0、1、2 三类，其中第 0 类表示已购买用户、第 1 类表示未购买但存在购买潜力的用户、第 2 类表示未购买且暂无购买潜力的用户。

其次，利用 SHAP 库解释模型特征：瀑布图能清楚地看到各个维度特征对于用户分类标签的贡献值，图中显示了排名前列的维度特征的贡献值；force 图列出了所有维度特征的贡献值大小，能清晰的看出维度特征的贡献比重。

注：本段分析中，因数据过小，所有的瀑布图中标记为 0 的维度特征均被四舍五入，具体值大小参见瀑布图中的瀑布的长度大小。

下面，我们选取以上三类用户的样本，分析探索针对各类用户的营销手段。

如图（）是某第 0 类用户的维度特征瀑布图，用户 id 是 2000001563151330。图中蓝色代表该维度特征抑制用户购买，红色代表该维度特征促进用户购买， $E[f(x)]$ 表示单个维度特征的平均贡献值， $f(x)$ 表示所有维度特征对该用户的贡献值。

首先,程序计算出所有维度特征的平均贡献 $E[f(x)]$ 为-0.318，然后从该值出发，从下往上依次计算、并累加出每个维度特征对于该用户具体的贡献值，算得最终的贡献值 $f(x)$ 为0.268。因此该用户购买课程的判定值为 0.268，即该用户倾向于购买课程。

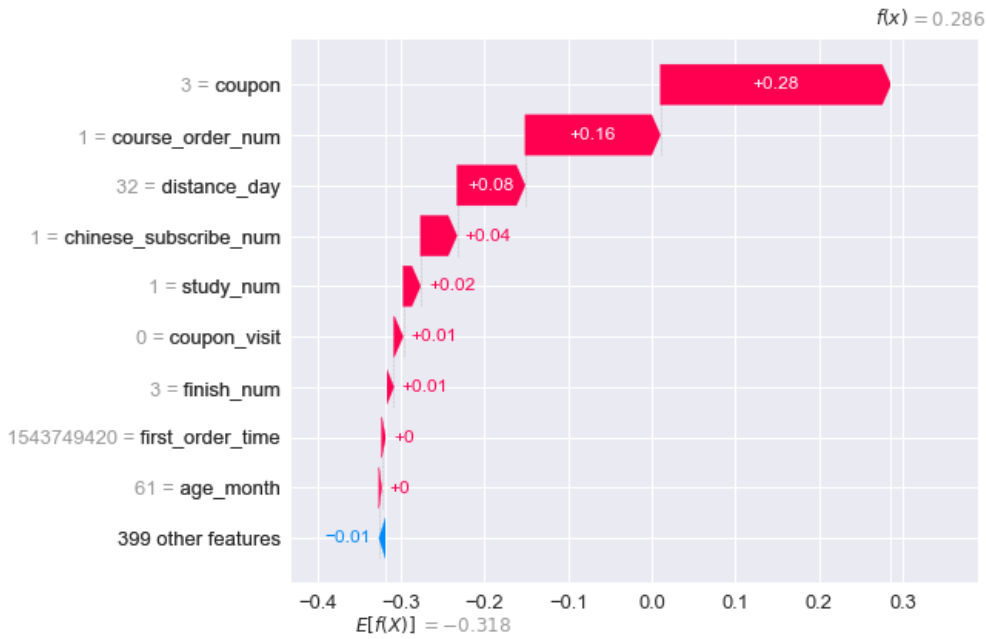


图 18 每个维度特征对模型预测结果的影响

利用 SHAP 库绘制的 force 图如图所示，该图展示了每个维度特征对模型预测结果的影响。其中领券数量（coupon）对最终结果贡献值最大，约为 0.28；有年课未完成订单（course_order_num）贡献值排第二，约为+0.16；而排名第三的最后登录距期末的天数（distance_day）对结果贡献值略小于前两个维度特征，约为 0.08。

所以，通过分析样本，得出影响第 0 类用户的维度特征为领券数量、有年课未完成订单、最后登录距期末的天数。此类用户特点为领券数量多、存在年课未完成的订单、且会在期末前登录平台。



图 19 第一类用户-典型用户示例-影响因子总览图

如图是某第 1 类用户的维度特征瀑布图，用户 id 是 2000002690191600。据瀑布图所示，首先,程序从平均贡献 $E[f(x)]$ 为-0.318 出发，从下往上依次计算、并累加出每个维度特征对于该用户具体的贡献值，算得最终的贡献值 $f(x)$ 为-0.324。因此该用户购买课程的判定值为-0.324，即该用户倾向于不购买课程。

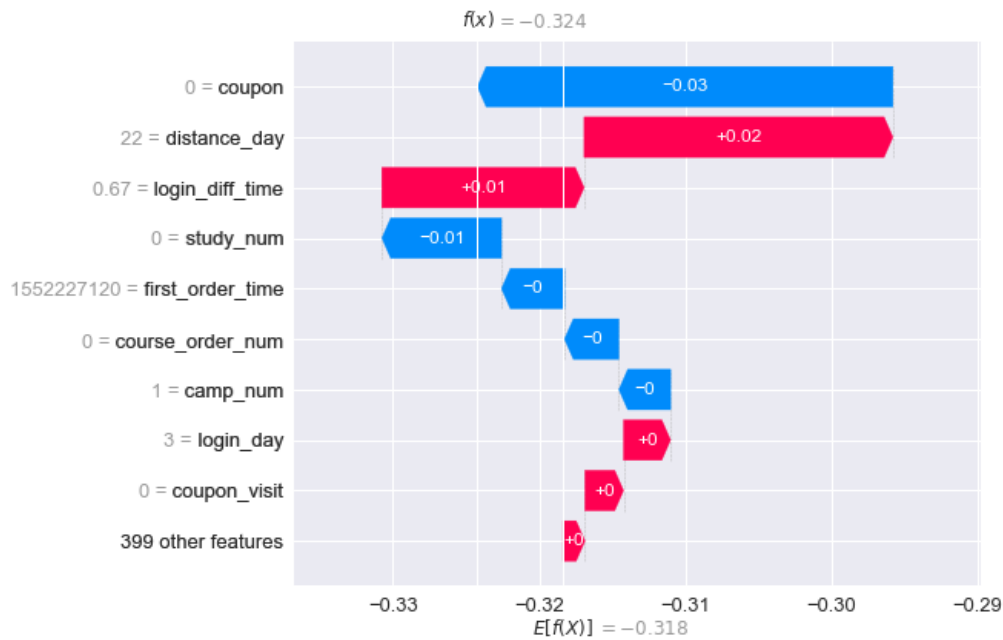


图 20 第一类用户-典型用户示例-影响因子总览图

利用 SHAP 库绘制的 force 图如图所示，该图展示了领券数量（coupon）对最终结果贡献值最大，约为-0.03；最后登录距期末的天数（distance_day）贡献值排第二，约为+0.02；而排名第三的登录间隔时间（login_diff_time）对结果贡献值略小于前两个维度特征，约为0.01。

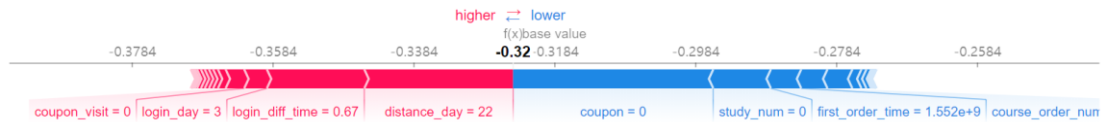


图 21 第二类用户-典型用户示例-影响因子总览图

所以，通过分析样本，得出影响第 1 类用户的维度特征为领券数量、最后登录距期末的天数、登录间隔时间。此类用户特点为领券数量少、会在期末前登录平台、且总体登录间隔较近。

如图（）是某第 2 类用户的维度特征瀑布图，用户 id 是 2000002586790540。据瀑布图所示，首先,程序从平均贡献 $E[f(x)]$ 为-0.318 出发，从下往上依次计算、并累加出每个维度特征对于该用户具体的贡献值，算得最终的贡献值 $f(x)$ 为-0.327。因此该用户购买课程的判定值为-0.327，即该用户倾向于不购买课程。

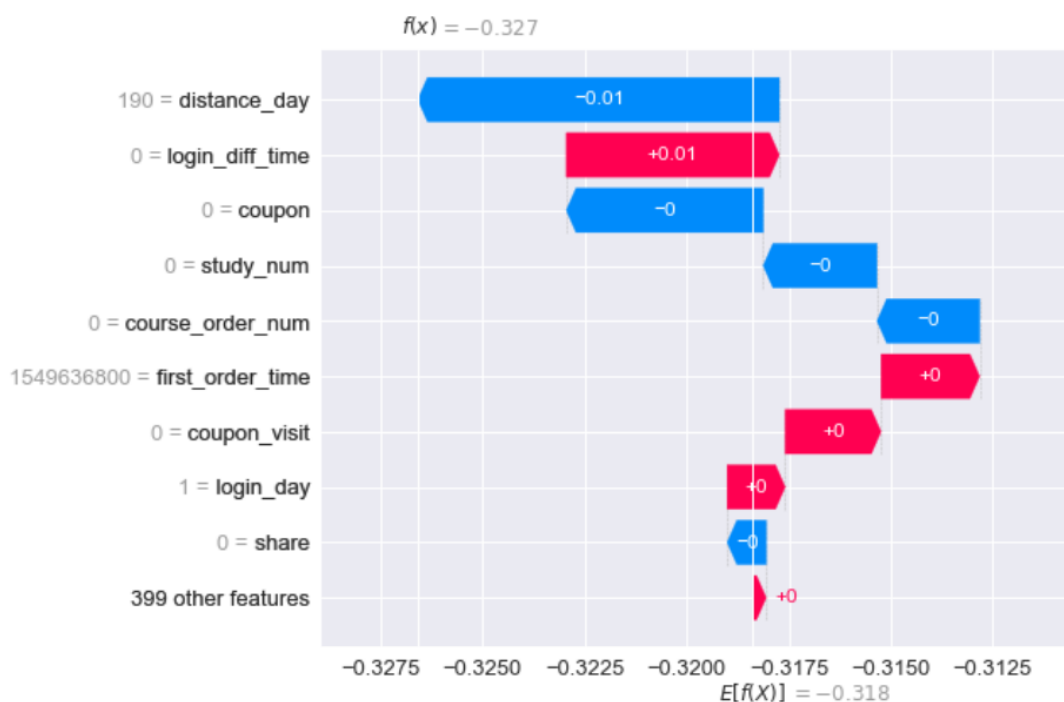


图 22 第二类用户-典型用户示例-影响因子总览图

利用 SHAP 库绘制的 force 图如图所示，该图展示了最后登录距期末的天数（distance_day）对最终结果贡献值最大，约为-0.01；登录间隔时间（login_diff_time）贡献值排第二，约为+0.01；排名第三的是领券数量（coupon），贡献比重略小于登录间隔时间。

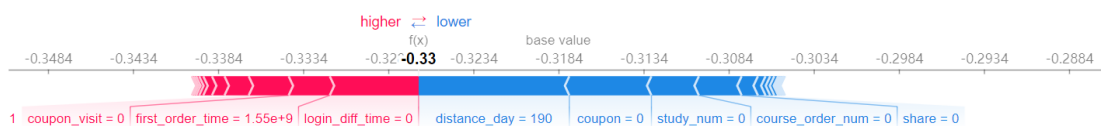


图 23 第三类用户-典型用户示例-影响因子总览图

所以，通过分析样本，得出影响第 2 类用户的维度特征为最后登录距期末的天数、登录间隔时间、领券数量。此类用户特点为期末前未登录过平台，或不常登录，领券数量很少。

根据上文的分析探讨与用户特点，我们针对这三类用户提供不同的建议方案：

针对已购买的用户：

即 KNN 聚类算法得出的第 0 类用户。APP 开发者和课程提供者可设置每日学习打卡提醒以提高用户的学习天数，同时提供订单完成奖励机制提高用户的学习积极性，例如当用户完成年课订单后，平台对其奖励适当的分数，奖励分可用来兑换礼品、学习用品等。

另一方面，还可针对此类用户开设期末短期付费课程，如速成班、提高班等进行分层教学，同时发放专属优惠券，以刺激该类用户在期末时段继续消费。

最后，确保课程老师留存该类用户的联系方式、添加微信好友，定期了解孩子学习情况，并提出学习建议，提高用户对平台的信任度，促进其持续消费。

针对未购买但有购买潜力用户

即 KNN 聚类算法得出的第 1 类用户。可在 APP 中投放更多的优惠券弹窗，针对领取了优惠券的顾客进行一定程度的电话销售行为，如采用询问孩子情况、关心购课顾虑等方式进行销售促进。

另一方面，平台可开设并销售价格较低的期末短期班，以降低消费门槛，刺激该类潜在用户的期末前消费。

同时，平台可开设课程团购活动以较低价的形式吸引潜在用户消费，同时聚集更多具有消费潜力的用户至平台。

最后，针对其登录间隔较近的特点，可对未购买用户设置奖励机制，如连续签到领取积分、大额优惠券等。

针对未购买且暂无购买潜力用户

即 KNN 聚类算法得出的第 2 类用户，则不应投入过多的营销成本。平台可设置用户回归计划，以奖励此类用户通过做回归任务增强对课程产品的了解程度、以及提高 APP 平台的使用频率。

同时，采用推荐算法向用户推荐其感兴趣的课程，提高该类用户对平台使用的兴趣程度。

7.2 针对平台的建议

附加功能属性，增强用户黏性

由于分析了解到，用户的登录时间与登录间隔等会影响到最终的购买决策，故在线教育软件应当设置一定的登录激励增强用户黏性。如企业可以优化 APP 内置的功能页，删减用户较少访问的页面，而增加类似于社区、小视频的模块以增加用户在软件内的驻留时间；同时也可以增添一定的签到奖励，体验课程也可以增加打卡领奖活动，因为用户对体验课程本身的参与程度也会间接影响到最终的购买决策；同时用户应当成立用户体验与优化部门，不断完善用户的功能交互体验，让软件内除了课程之外的内容也会吸引到用户。

附加社交内容，实现增长裂变

平台应当鼓励用户原创内容，UGC 内容能够促进用户自发性创作行为，并极大地提升用户黏性。此外，用户原创内容后大概率会通过外部社交平台进行分享，这种自发性的分享远比邀请领奖励更有效果。为平台打造自传播属性是增长的不二手段。前期用户量较少可以暂时在站外进行传播，若后期用户形成一定的规模可以考虑增加内部好友、社交等功能进一步巩固用户忠诚度。

优化广告设置，提高广告效率

平台本身页面访问量足够，但优惠券领取却少之又少，企业应当加强广告美工设计、文案设计，针对用户访谈内容提炼用户需求，凝练产品卖点与痛点。

另一方面，广告弹窗可以适当优化其推送位置，除了访问量较多的首页外，其他高频访问场所亦可以考虑进行小窗投放。

改进课程安排，减少用户流失

用户在学习第 4-5 节课后开始大量流失，对于课程不感兴趣的群体在前两节课就已经大量流失了。同时数据表明，用户其实有一定的复看行为，体验课程内容可以适当减少节次，设定为观看次数较高的 4-5 节课为佳，等到全部课程看完以后及时在浏览界面推送“恭喜体验课结束”等字样，并推送正价课优惠券，及时进行低转高的运营从而减少用户后续的不断流失从而放弃购买行为。

附录-核心代码

注：此处仅提供核心代码，全部代码详见支撑材料

EnsembleGeneration_Full.py

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.utils import shuffle
import xgboost as XGBR
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression

from Tools.Report import MakeReport
from sklearn.metrics import f1_score

from sklearn.utils import shuffle
from imblearn.under_sampling import RandomUnderSampler
import joblib
import os

# =====
params = {
    "booster": "gbtree", # 选择每次迭代的模型
    "objective": "binary:logistic", # 定义学习任务及相应的学习目标
    "eval_metric": "auc", # 对于有效数据的度量方法。
    "max_depth": 10, # 最大深度
    "eta": 0.025, # 学习率
    "seed": 0, # 随机数种子
}
argument = {"train_size": 0.9, "test_size": 0.1, "n_splits": 3, "num_boost_round": 10}
argument.update(params)
# =====

data = pd.read_csv("./Data/Data4Predict.csv")
data = shuffle(data)

x = data.drop(["result"], axis=1)
y = data["result"]

x_train, x_test, y_train, y_test = train_test_split(
    x, y, train_size=argument["train_size"], test_size=argument["test_size"]
```

```

)
    print("x_train", np. shape(x_train))
    print("y_train", np. shape(y_train))
    print("x_test", np. shape(x_test))
    print("y_test", np. shape(y_test))

y_train_save=y_train
y_test_save=y_test

x_train=x_train. values
y_train=y_train. values
x_test=x_test. values
y_test=y_test. values

# For XGB
x_dtest = XGBR. DMatrix(x_test)

ntrain=x_train. shape[0]
ntest=x_test. shape[0]

oof_train_XGB=np. zeros((ntrain))
oof_test_XGB=np. zeros((ntest))
oof_test_skf_XGB=np. empty((5, ntest))

oof_train_RF=np. zeros((ntrain))
oof_test_RF=np. zeros((ntest))
oof_test_skf_RF=np. empty((5, ntest))

# =====

ModelList = []

kf = KFold(n_splits=argument["n_splits"])
for i, (train_index, test_index) in enumerate(kf. split(x_train)):

    print("train_index", np. shape(train_index))
    print("test_index", np. shape(test_index))

    # XGB
    print(f"XGB-{i}")
    kf_x_train=x_train[train_index]
    kf_y_train=y_train[train_index]
    kf_dtrain=XGBR. DMatrix(kf_x_train, labe l=kf_y_train)
    print(np. shape(kf_x_train))

```

```

print(np. shape(kf_y_train))

kf_x_test=x_train[test_index]
kf_x_dtest=XGBR. DMatrix(kf_x_test)
print(np. shape(kf_x_test))

watchlist=[(kf_dtrain, "train")]
bst=XGBR. train(
    params, kf_dtrain, num_boost_round=argument["num_boost_round"], evals=watchlist
)
# 输出概率
oof_train_XGB[test_index]=bst.predict(kf_x_dtest)
oof_test_skf_XGB[i, :]=bst.predict(x_dtest)
y_pred=(bst.predict(x_dtest)>=0.5)*1
(
    AUC_value,
    ACC_value,
    Recall_value,
    F1_Score_value,
    Precesion_value,
    ConfusionMatrix,
    Report
)=MakeReport("XGBoost", y_test, y_pred, {})

#RF
print(f"RF-{i}")
R_tree=RandomForestClassifier(
    n_estimators=57, min_samples_split=14, min_samples_leaf=2, max_depth=19
)
R_tree.fit(x_train[train_index], y_train[train_index])
oof_train_RF[test_index]=R_tree.predict_proba(kf_x_test)[: , 1]
oof_test_skf_RF[i, :]=R_tree.predict_proba(x_test)[: , 1]
(
    AUC_value,
    ACC_value,
    Recall_value,
    F1_Score_value,
    Precesion_value,
    ConfusionMatrix,
    Report
)=MakeReport("RandomForest", y_test, R_tree.predict(x_test), {})

ModelList.append((bst, R_tree))

```

```

# =====
# 第二层模型

oof_x_train=np.zeros((ntrain, 2))
oof_x_test=np.zeros((ntest, 2))

oof_x_train[:, 0]=oof_train_XGB
oof_x_train[:, 1]=oof_train_RF
oof_x_test[:, 0]=oof_test_skf_XGB.mean(axis=0)
oof_x_test[:, 1]=oof_test_skf_RF.mean(axis=0)

# 欠采样优于 smote
rus = RandomUnderSampler(random_state=0)
oof_x_train, oof_y_train=rus.fit_resample(oof_x_train, y_train)

clf=LogisticRegression(penalty="l2", C=0.3, max_iter=400, tol=1e-4, solver="lbfgs")
clf.fit(oof_x_train, oof_y_train)

ans=clf.predict(oof_x_test)
(
    AUC_value,
    ACC_value,
    Recall_value,
    F1_Score_value,
    Precesion_value,
    ConfusionMatrix,
    Report,
)=MakeReport("MixLR", y_test, ans, argument)

# =====
# 保存模型

model_path=f"./Predict/Model/EnsembleGeneration/{F1_Score_value}"
os.mkdir(model_path)
joblib.dump(clf, f"{model_path}/LR.pkl")
for i, (j, k) in enumerate(ModelList):
    joblib.dump(j, f"{model_path}/XGB-{i}.pkl")
    joblib.dump(k, f"{model_path}/RF-{i}.pkl")

data_path=f"./Data/EnsembleGeneration/{F1_Score_value}"
os.mkdir(data_path)

np.savetxt(
    f"./Data/EnsembleGeneration/{F1_Score_value}/oof_test_XGB.csv",

```

```

oof_test_XGB,
encoding="utf-8-sig",
delimiter=", ",
fmt="%. 5f",
header="XGB",
)
np.savetxt(
f"./Data/EnsembleGeneration/{F1_Score_value}/oof_test_RF.csv",
oof_test_RF,
encoding="utf-8-sig",
delimiter=", ",
fmt="%. 5f",
header="RF",
)
np.savetxt(
f"./Data/EnsembleGeneration/{F1_Score_value}/oof_train_XGB.csv",
oof_train_XGB,
encoding="utf-8-sig",
delimiter=", ",
fmt="%. 5f",
header="XGB",
)
np.savetxt(
f"./Data/EnsembleGeneration/{F1_Score_value}/oof_train_RF.csv",
oof_train_RF,
encoding="utf-8-sig",
delimiter=", ",
fmt="%. 5f",
header="RF",
)

y_train_save.to_csv(
f"./Data/EnsembleGeneration/{F1_Score_value}/train.csv", encoding="utf-8-
sig", index=False, sep=", "
)
y_test_save.to_csv(
f"./Data/EnsembleGeneration/{F1_Score_value}/test.csv", encoding="utf-8-
sig", index=False, sep=", "
)

```

DataWash.py

```
import pandas as pd
```

```

import numpy as np

import time

from sklearn.utils import shuffle

def TimeTransfer(x):
    if not pd.isna(x):
        timeArray = time.strptime(x, "%Y/%m/%d %H:%M")
        timeStamp = int(time.mktime(timeArray))
        return timeStamp
    else:
        return np.NaN

data = pd.read_csv("./Data/data_merge.csv")
data = shuffle(data)
data["city_num"].fillna("未知", inplace=True)
data.dropna(inplace=True)

# city 转 one-hot
city_num = pd.get_dummies(data["city_num"])
data = data.join(city_num)
data.drop(["city_num"], inplace=True, axis=1)
data.drop(["user_id"], inplace=True, axis=1)
data.drop(["app_num"], inplace=True, axis=1)
data.rename(columns={"error": "错误"}, inplace=True)

# 时间转时间戳
data["first_order_time"] = data["first_order_time"].apply(TimeTransfer)

# pd.set_option('display.max_columns', None)

```

```
# print(data.isnull().any)
```

```
data.to_csv("./Data/Data4Predict.csv", encoding="utf-8-sig", index=False, sep=",")
```

DataMerge.py

```
import pandas_profiling as pp
```

```
import pandas as pd
```

```
import os
```

```
data_all = []
```

```
data_merge = pd.DataFrame()
```

```
#遍历文件夹合并数据
```

```
for filename in os.listdir("./Data/RawData"):
```

```
    head, tail = filename.split(".")
```

```
    print(filename)
```

```
    data = pd.read_csv(f"./Data/RawData/{filename}")
```

```
    data_all.append(data)
```

```
for i in data_all:
```

```
    if data_merge.empty:
```

```
        data_merge = i
```

```
    else:
```

```
        data_merge = pd.merge(data_merge, i, on="user_id", how="outer")
```

```
data_merge["result"] = data_merge["result"].apply(lambda x: 1 if x == 1 else 0)
```

```
data_merge.to_csv(
```

```
    "./Data/data_merge.csv", encoding="utf-8-sig", index=False, sep=",",
```

```
)
```


HtmlOverview_all.py

```
import pandas_profiling as pp
import pandas as pd
import os

# 绘制各表预览
for filename in os.listdir("./Data/RawData"):
    head,tail=filename.split(".")
    print(filename)
    data = pd.read_csv(f"./Data/RawData/{filename}")
    report = pp.ProfileReport(data,title=filename)
    report.to_file(f"./Overview/OutPut/{head}.html")

# 绘制总表预览
data = pd.read_csv(f"./Data/data_merge.csv")
report = pp.ProfileReport(data,title="data_merge")
report.to_file(f"./Overview/OutPut/Html/data_merge.html")
```

Report.py

```
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import f1_score
from sklearn.metrics import precision_recall_fscore_support

def MakeReport(type, y_test, y_pred, argument, *ypred):
    if len(ypred) == 1:
        AUC_value = str(metrics.roc_auc_score(y_test, ypred[0]))[:7]
    elif len(ypred) == 0:
```

```

    AUC_value = "NULL"

else:
    AUC_value = "False"

ACC_value = str(metrics.accuracy_score(y_test, y_pred)[:7])
Recall_value = str(metrics.recall_score(y_test, y_pred)[:7])
F1_Score_value = str(metrics.f1_score(y_test, y_pred)[:7])
Precesion_value = str(metrics.precision_score(y_test, y_pred)[:7])
ConfusionMatrix = str(metrics.confusion_matrix(y_test, y_pred))
Report = classification_report(y_test, y_pred)
str_argument = ""
for key, value in argument.items():
    str_argument += f"{key} : {value}\n"

Report_txt = f"{' '*50}\n type : {type}\n {str_argument}{' '*50}\n AUC : {AUC_value}\n ACC_value : {ACC_value}\n Recall_value : {Recall_value}\n F1_Score_value : {F1_Score_value}\n Precesion_value : {Precesion_value}\n ConfusionMatrix : \n {ConfusionMatrix}\n {' '*50}\n Report : \n {Report}\n {' '*50}\n"

print(Report_txt)

print("f1 is :", f1_score(y_test, y_pred, average="binary"))
# None: 返回每一类各自的f1_score, 得到一个array。
# "binary":返回由 pos_label 指定的类的f1_score。
with open("./Predict/Log/Log.txt", "a+") as f:
    f.write(Report_txt)

return (
    AUC_value,
    ACC_value,
    Recall_value,

```

```
F1_Score_value,  
Precesion_value,  
ConfusionMatrix,  
Report_txt  
)
```