基于机器学习的O2O（Online to Offline）优惠券预测

林继申 2250758 软件学院 软件工程

摘要：本研究致力于解决O2O（Online to Offline）背景下的优惠券核销预测问题，通过分析用户在2016年上半年的消费行为数据，预测他们在同年7月领取优惠券后15天内的核销行为。我们采用了机器学习算法，特别是XGBoost和LightGBM，来处理这一二分类问题。通过复杂的特征工程，包括数据探索、特征提取、滑窗技术以及穿越特征的应用，我们设计了三个版本的特征集合，以增强模型的预测准确性。结果显示，引入穿越特征和使用LightGBM模型显著提升了AUC评分，有效预测了优惠券的核销概率。此外，本研究还涉及交叉验证与模型调参，进一步验证了模型的稳健性和高效性。

关键词：大数据，人工智能，O2O，特征功能，LightGBM

一、引言

随着移动设备的完善和普及，移动互联网+进入了高速发展阶段，其中以O2O（Online to Offline）消费最为吸引眼球。据不完全统计,O2O行业估值上亿的创业公司至少有十家，期中也不乏百亿巨头的身影。O2O 行业天然关联着数亿消费者，各类APP每天记录了超过百亿条的用户行为和位置记录，因而成为大数据科研和商业化运营的最佳结合点之一。

以优惠券盘活老用户或吸引新用户进店消费是O2O的一种重要营销方式，然后随即投放的优惠券会对多数用户造成无意义的干扰。对商家而言，滥发的优惠券还可能降低品牌声誉，同时难以估算营销成本。个性化投放是提高优惠券核销率的重要技术，可以让具有一定偏好的消费者得到真正的实惠，同时赋予商家更强的营销能力。

二、问题分析

（一）数据来源与目标

数据来源自用户在2016年1月1日至2016年6月30日之间真实线上线下消费行为的数据，我们需要预测用户在2016 年 7 月领取优惠券后 15 天内的使用情况，即投放的优惠券是否核销。

我们最终的目标是预测15天内用券的概率，这是一个连续值，但是因为用券只有用与不用两种情况，而且评测指标是典型的二分类评测指标AUC，所以这是一个二分类问题。

我们使用优惠券核销预测的平均 AUC，即对每个优惠券coupon_id 单独计算核销预测的 AUC 值，再对所有优惠券的AUC 值求平均作为最终的评价标准。AUC（Area Under the Curve）是ROC曲线（Receiver Operating Characteristic curve，受试者工作特征曲线）下的面积，通常用于评估分类模型的性能。在优惠券核销预测的场景中，AUC值是评估模型预测准确度的一种常用指标。

（二）数据特征

在给出数据中，训练集ccf_offline_stage1_train.csv与测试集ccf_offline_stage1_test_revised.csv都是线下的交易数据，而训练集ccf_online_stage1_train.csv是线上的交易数据。初步观察，它们有如下特点：

(1) 数据量整体不是很大，线下交易的数据才几十兆，加上线上交易的数据也才几百兆。

(2) 因为是推算未来的情况，而数据特征与时间段相关，训练集的时间周期又远远长于测试集的时间周期。因此，在特征工程的时候可以采用滑窗方法。

(3) 线下训练集 ccf_offline_stage1_train.csv 与线下测试集 ccf_offline_stage1_test_revised.csv 场景一致，而线上训练集 ccf_online_stage1_train.csv 与测试集场景不一致。我们在数据探索阶段通过比较训练集和测试集中店铺及用户的重合度来确定online训练集应该如何使用。

(4) 特征提取将重点围绕用户、店铺、优惠券及三者之间的关系展开。

（三）分析思路

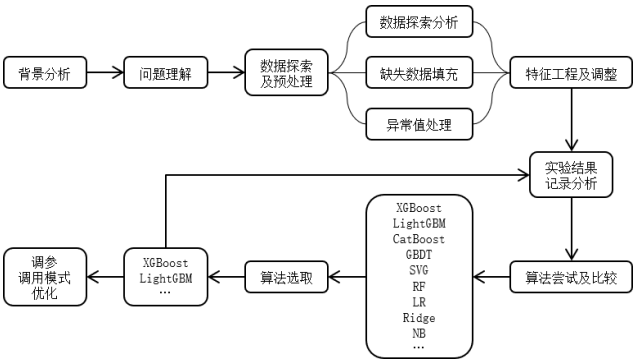


图 2.1 分析思路流程图

算法选取主要是根据数据特征来进行的。根据数据特点本题选择XGBoost和LightGBM一类算法效果会比较好。

三、数据探索

（一）数据探索与相关工具

数据探索是通过使用可视化的方式分析数据，提取数据的主要特征，包括数据量、完整性、正确性、可能的相互关系等，一般从数据质量和数据特征两方面进行数据探索分析。

数据探索的目的包括得到数据的直觉、发掘潜在的结构、提取重要的变量、删除异常值、检验潜在的假设和建立初步的模型。

我们使用 Python 中进行数据探索的主要工具有 NumPy、Pandas、Matplotlib、SciPy等。

（二）数据初步探索

通过pandas.read_csv()函数可以读取数据,在此不再赘述，可以在附件中查看详情。

使用 pandas.head()函数查看数据集的头几行数据：

	user_id	merchant_id	coupon_id	discount_rate	distance	date_received	date
0	1439408	2632	nan	NaN	0.0	NaN	20160217.0
1	1439408	4663	11002.0	150:20	1.0	20160528.0	NaN
2	1439408	2632	8591.0	20:1	0.0	20160217.0	NaN
3	1439408	2632	1078.0	20:1	0.0	20160319.0	NaN
4	1439408	2632	8591.0	20:1	0.0	20160613.0	NaN

图 3.1 Offline（线下）训练集头几行数据

	user_id	merchant_id	action	coupon_id	discount_rate	date_received	date
0	13740231	18907	2	100017492	500:50	20160513.0	NaN
1	13740231	34805	1	nan	NaN	NaN	20160321.0
2	14336199	18907	0	nan	NaN	NaN	20160618.0
3	14336199	18907	0	nan	NaN	NaN	20160618.0
4	14336199	18907	0	nan	NaN	NaN	20160618.0

图 3.2 Online（线上）训练集头几行数据

	user_id	merchant_id	coupon_id	discount_rate	distance	date_received
0	4129537	450	9983	30:5	1.0	20160712
1	6949378	1300	3429	30:5	NaN	20160706
2	2166529	7113	6928	200:20	5.0	20160727
3	2166529	7113	1808	100:10	5.0	20160727
4	6172162	7605	6500	30:1	2.0	20160708

图 3.3 测试集头几行数据

同时还使用 pandas.info()函数查看数据集的基本信息，使用 pandas.describe()函数查看数据集的统计信息，使用 pandas.isnull()函数查看数据的确实情况，在此不再赘述，可以在附件中查看详情。

（三）数据边界探索

由于我们是对某一时间段内优惠券的使用进行预测，因此我们有必要查看数据集的领券日期范围和用券日期范围。通过代码我们可以得知训练集的用券日期是到 6 月 30 日，而领券日期并不是到 6 月 30 日，而是到 6 月 15 日，这一点在设计滑窗结构时需要注意。

（四）训练集与测试集的相关性

由于不同数据的价值是不一样的，即并不是所有得到的数据都是有用的，因此对测试集数据和训练集数据的重合情况进行探查都对后续特征构建的思路又很大的指导作用。

通过代码，我们依次对用户、商家和优惠券字段在训练集和测试集的重合情况进行了探索，从结果可以发现以下特点：

(1) 用户 ID 一项，测试集与 offline 训练集的重复占比达 0.999，与 online 训练集的重复占比为 0.565。

(2) 商家 ID 一项，测试集与 offline 训练集的重复占比达 0.999，与 online 训练集没有重复。

(3) 优惠券 ID 一项，测试集与 offline 训练集和 online 训练集都没有重复。

由此，我们认为 online 训练集数据价值较低，下一步的特征提取将以 offline 训练集为主，后续可视化分析将主要在 offline 训练集和测试集之间展开。

（五）对文本数据的数值化处理

通过初步观察，训练集和测试集数据分布比较一致。但是因为 X 轴的变量（折扣率）是文本类型，所以无法同济折扣率与其他变量的关系。因此，在进行进一步分析之前，需要对文本数据进行数值化处理。

在给定的数据中，除了折扣及距离，其他的数据都为文本，需要先对其进行数值化才能进行探索。我们利用 pandas.apply()函数来处理，它会自动遍历每一行 DataFrame 的数据，最后将所有结果组合成一个 Series 数据结构并返回。

对文本数据的数值化处理的代码实现在此不再赘述，可以在附件中查看详情。对数值化后的数据进行分析：

	user_id	merchant_id	coupon_id	discount_rate	distance	date_received	date	if_fd	full_value	reduction_value	day_gap	label
0	1439408	2632	null	-1.000000	0	null	20160217	0	-1	-1	-1	-1
1	1439408	4663	11002	0.866667	1	20160528	null	1	150	20	-1	0
2	1439408	2632	8591	0.950000	0	20160217	null	1	20	1	-1	0
3	1439408	2632	1078	0.950000	0	20160319	null	1	20	1	-1	0
4	1439408	2632	8591	0.950000	0	20160613	null	1	20	1	-1	0

图 3.4 训练集头几行数据

	discount_rate	distance	if_fd	full_value	reduction_value	day_gap	label
count	1.754884e+06	1.754884e+06	1.754884e+06	1.754884e+06	1.754884e+06	1.754884e+06	1.754884e+06
mean	1.069696e-01	2.158577e+00	5.812407e-01	4.665974e+01	5.807668e+00	-6.212582e-01	-3.693657e-01
std	9.061683e-01	3.470772e+00	4.933559e-01	6.830349e+01	9.037305e+00	2.414021e+00	5.534657e-01
min	-1.000000e+00	-1.000000e+00	0.000000e+00	-1.000000e+00	-1.000000e+00	-1.000000e+00	-1.000000e+00
25%	-1.000000e+00	0.000000e+00	0.000000e+00	-1.000000e+00	-1.000000e+00	-1.000000e+00	-1.000000e+00
50%	7.500000e-01	0.000000e+00	1.000000e+00	2.000000e+01	5.000000e+00	-1.000000e+00	0.000000e+00
75%	9.000000e-01	3.000000e+00	1.000000e+00	1.000000e+02	1.000000e+01	-1.000000e+00	0.000000e+00
max	9.900000e-01	1.000000e+01	1.000000e+00	3.000000e+02	1.000000e+02	9.600000e+01	1.000000e+00

图 3.5 训练集数据描述

	user_id	merchant_id	coupon_id	discount_rate	distance	date_received	if_fd	full_value	reduction_value
0	4129537	450	9983	0.833333	1	20160712	1	30	5
1	6949378	1300	3429	0.833333	-1	20160706	1	30	5
2	2166529	7113	6928	0.900000	5	20160727	1	200	20
3	2166529	7113	1808	0.900000	5	20160727	1	100	10
4	6172162	7605	6500	0.966667	2	20160708	1	30	1

图 3.6 测试集头几行数据

	discount_rate	distance	date_received	if_fd	full_value	reduction_value
count	113640.000000	113640.000000	1.136400e+05	113640.000000	113640.000000	113640.000000
mean	0.850671	1.974736	2.016072e+07	0.977420	43.396507	6.089141
std	0.063551	3.248809	9.019508e+00	0.148561	44.239815	5.640310
min	0.333333	-1.000000	2.016070e+07	0.000000	-1.000000	-1.000000
25%	0.833333	0.000000	2.016071e+07	1.000000	30.000000	5.000000
50%	0.833333	1.000000	2.016072e+07	1.000000	30.000000	5.000000
75%	0.900000	3.000000	2.016072e+07	1.000000	30.000000	5.000000
max	0.990000	10.000000	2.016073e+07	1.000000	500.000000	100.000000

图 3.7 测试集数据描述

对比发现，是否满减项在训练集和测试集上的分布相差较大，需要进一步探索。

（六）数据分布可视化

箱线图是用于显示一组数据分散情况的统计图，因其形状如箱子而得名。箱线图的绘制过程：首先，找出一组数据的上下边缘、中位数及两个四分位数；然后，连接这两个四分位数，画出箱体；最后，将上下边缘分别与箱体连接，这样中位点就位于箱体中间箱线图可以观察数据分布的特征，也可以用于对多组数据分布特征进行比较。

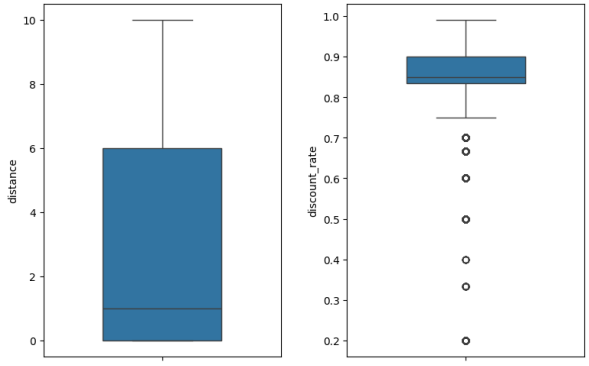


图 3.8 箱线图（训练集距离与折扣率）

直方图是用一系列高度不等的纵向条纹或线段来表示数据的分布情况，是一种统计报告图。一般横轴是数据类型，纵轴是统计特征（比如频数）。Q-Q 图以散点图的方式，通过绘制两个概率分布对应的分位数对不同分布进行比较。Q-Q 图可以用于检验数据是否为正态分布。在 Q-Q 图上，将数据的分位数作为 x 轴坐标值，同时将计算出的假定为正态分布时的数据分位数作为 y 轴坐标值。当实际数据近似为正态分布时，Q-Q 图的点会落在一条直线上。

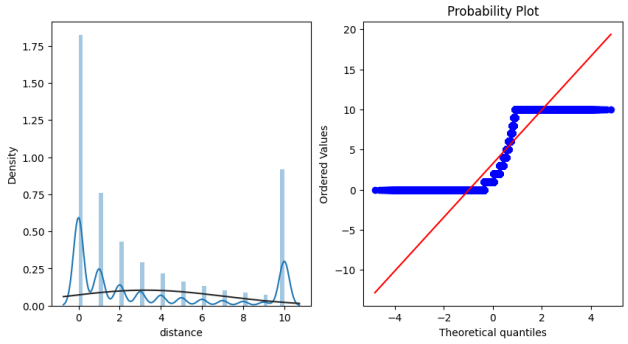


图 3.9 直方图与 Q-Q 图（训练集距离）

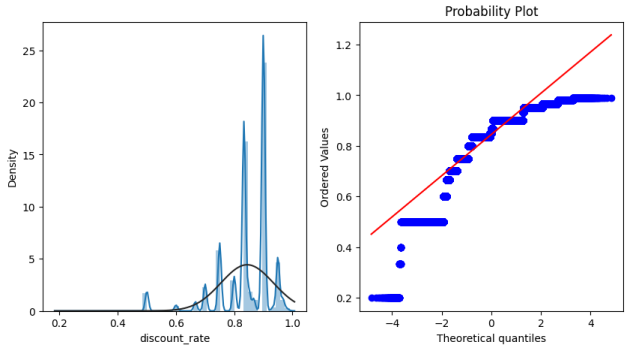


图 3.10 直方图与 Q-Q 图（训练集折扣率）

我们可以通过绘制不同特征项的核密度图来对比数据分布。由于核密度估计是通过核函数对数据点进行拟合，因此得到的数据分布图比直方图更加平滑。

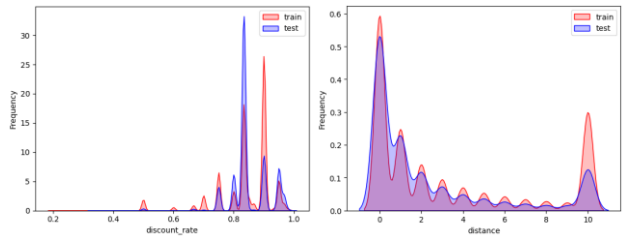


图 3.11 对比分布图（折扣率、距离）

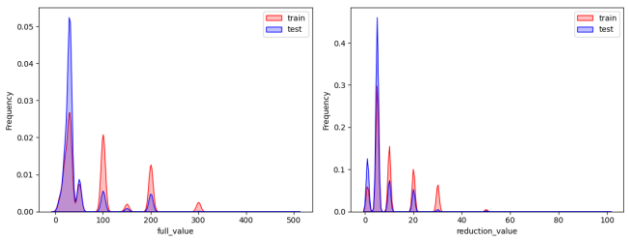


图 3.12 对比分布图（满额、减额）

使用 regplot()函数可以对两个特征项之间的线性回归关系进行可视化展示。

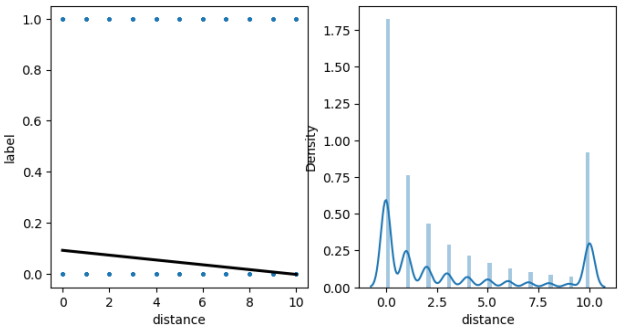


图 3.13 线性关系图（训练集距离）

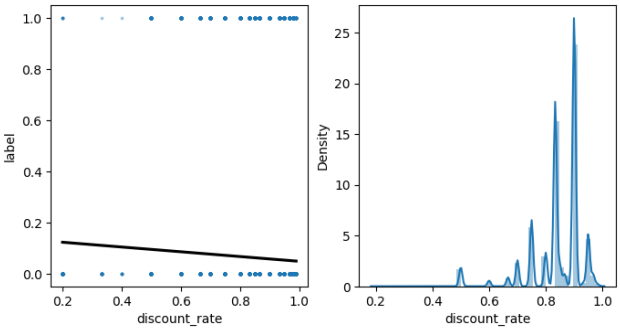


图 3.14 线性关系图（训练集折扣率）

（七）结论

初步查看发现，在 offline 训练集和测试集中，距离和折扣率的分布都相差不大，而是否为满减优惠券这一项的分布相差很大。进一步对比满减情况，发现 offline 训练集和测试集的满减占比相差较大。因此，在下一步特征工程中，可以考虑是否要对 online 训练集中非满减的数据进行抽样；或者在模型训练时，将是否为满减的数据分别进行训练和预测。

四、特征工程

（一）特征工程思路

(1) 首先可以通过 offline 训练数据直接提取训练集特征并进行标记（结合“Date_received”和“Date”即可判断该优惠券是否使用），通过测试数据提取测试集特征。

(2) 在训练集中可以直接提取的特征主要与优惠券相关，包括折扣率、优惠券类型（是否为满减）、满额、减额、距离等。

(3) 进一步可以提取统计特征，包括用户统计特征，如用户领取优惠券次数（总数，核销，未核销）、用户消费次数（总数，用券，未用券）、折扣率（最大，最小，平均）等；优惠券统计特征，如核销率、领取次数、消费次数、使用距离（最大，最小，平均）等；商家统计特征，如优惠券被领取次数及使用概率（总数，核销，未核销）、使用距离（最大，最小，平均）、被使用优惠券的折扣率（最大，最小，平均）等；关联统计特征，如用户领取商家的优惠券次数、用户领取商家优惠券后的不核销次数、用户领取商家优惠券后的核销次数、用户领取商家优惠券后的核销率等。

(4) 评估穿越。评估穿越指的是由于样本划分不当，使测试集中的信息“穿越”到了训练集中，导致评估结果更偏爱

过拟合的模型，致使结果不够准确。为了防止评估穿越，在滑窗的过程中，要注意预测区间和特征区间一定不能重叠，直接使用全局数据的统计量更是不可取。

(5) 滑窗。滑动窗口就是指定单位长度的时间序列，然后计算窗口区间的统计指标。这一过程也相当于让一个指定长度的滑块在刻度尺上面滑动，每滑动一个单位即可反馈滑块内的数据。采用滑窗的方法可以得到多份训练数据集，特征的窗口区间越小，得到的训练数据集的数量越多。

(6) 穿越特征，就是在训练集上提取的特征，或者外部取得的在目标时间段内的数据穿越特征在实际应用中意义不大，因为用的是“未来”的数据，在实际过程中是得不到的。在提供的预测集中包含了同一个用户在整个 7 月的优惠券领取情况，这实际上是一种穿越。存在这种情况：某一个用户在 7 月 10 日领取了某优惠券，然后在 7 月 12 日和 7 月 15 日又领取了相同的优惠券，那么 7 月 10 日领取的优惠券被核销的可能性就很大了。在加入这部分特征后，AUC 可以提升约 10 个百分点。

(7) 特征方案。上面我们已经分析了要提取的特征，在此基础上，我们按照由浅入深、由简单到复杂的顺序提取了三个版本的特征。

① 特征 V1：直接在训练集、测试集上提取的特征，也就是将给定数据项进行数值化的结果。

② 特征 V2:在特征 V1 的基础上,增加了滑窗统计特征，包括用户统计特征、优惠券统计特征、商家统计特征、关联统计特征等。

③ 特征 V3：在特征 V2 的基础上，增加了穿越特征。

（二）工具函数实现

在分析过程中使用的一些工具函数如下。具体的函数实现过程在此不再赘述，可以在附件中查看详情。

(1) 特征处理函数：计算折扣率、将满减和折扣统一、确定是否为满减优惠、获取满减的满额条件、获取满减的优惠等。

(2) 统计特征生成函数：通过统计提取特征。在使用 Pandas 进行统计分析时，首先通过 groupby()函数按照指定的列进行分组，得到一个 DataFrameGroupBy 对象。然后选择需要分析的列，例如使用.mean()对每个分组的数据进行均值计算。从 Pandas 0.20.1 开始，引入了 agg()函数，可以更灵活地进行基于列的聚合操作,返回的结果也是一个 DataFrame。最后，使用 merge()函数将生成的统计特征整合到原始数据中，编写成可复用的函数，提高开发效率和代码整洁性。

(3) 特征群生成函数：将各个特征的提取逻辑组合成特征群是一个很好的实践。这样不仅便于理解代码、查看是否有遗漏或者重复的特征，而且也便于后续组装不同集的特征。每一个特征群生成函数，都可以理解为一个小型的特征工程，包括去重、缺失值处理、特征提取等内容。 本赛题的特征群分为四部分：商户相关特征群、用户相关特征群、用户和商户关系特征群、Leakage 特征群。

(4) 特征集成函数：将生成的特征群进行组装可以生成不同版本的特征。

① 特征版本 1：只有最基础的特征，直接在数据集上提取特征，同时对一定字段进行数值化处理后得到特征。

② 特征版本 2：在特征版本 1 的基础上，增加了商户、用户及商户和用户关系的特征。

③ 特征版本 3：在特征版本 2 的基础上，增加了 Leakage 特征。

(5) 特征输出函数：在构建特征之后，需要将特征输出为文件，在训练模型时只需调用特征文件即可，而不需要每次都从原始数据重新构建特征，这样可以大大节省时间。我们以是否包含滑窗统计特征作为区分，构建了两种特征输出类型。

（三）特征分布

通过箱线图分析数据特征的分布，首先是 offline 训练集提取的特征：

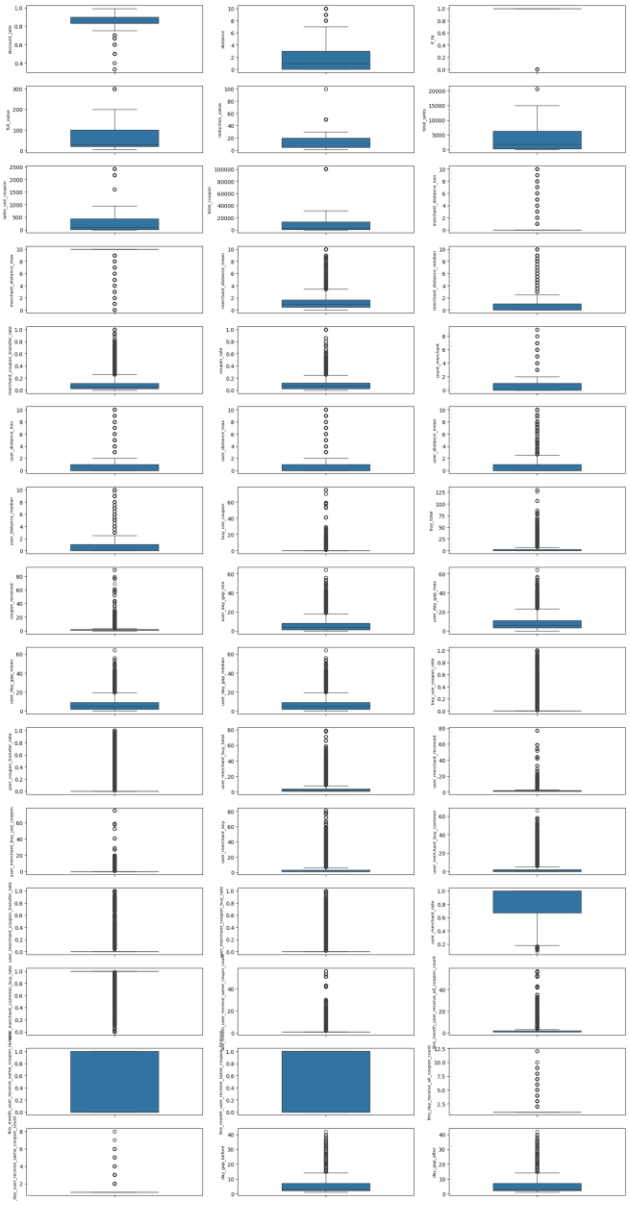


图 4.1 训练集特征分布

然后是测试集提取的特征：

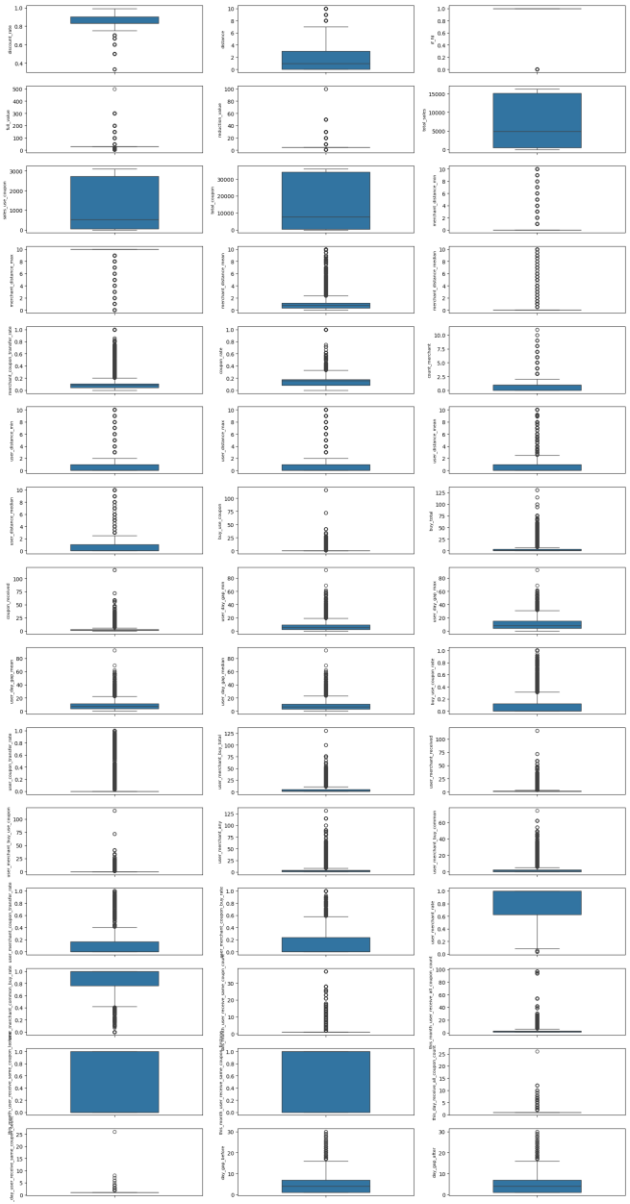


图 4.2 测试集特征分布

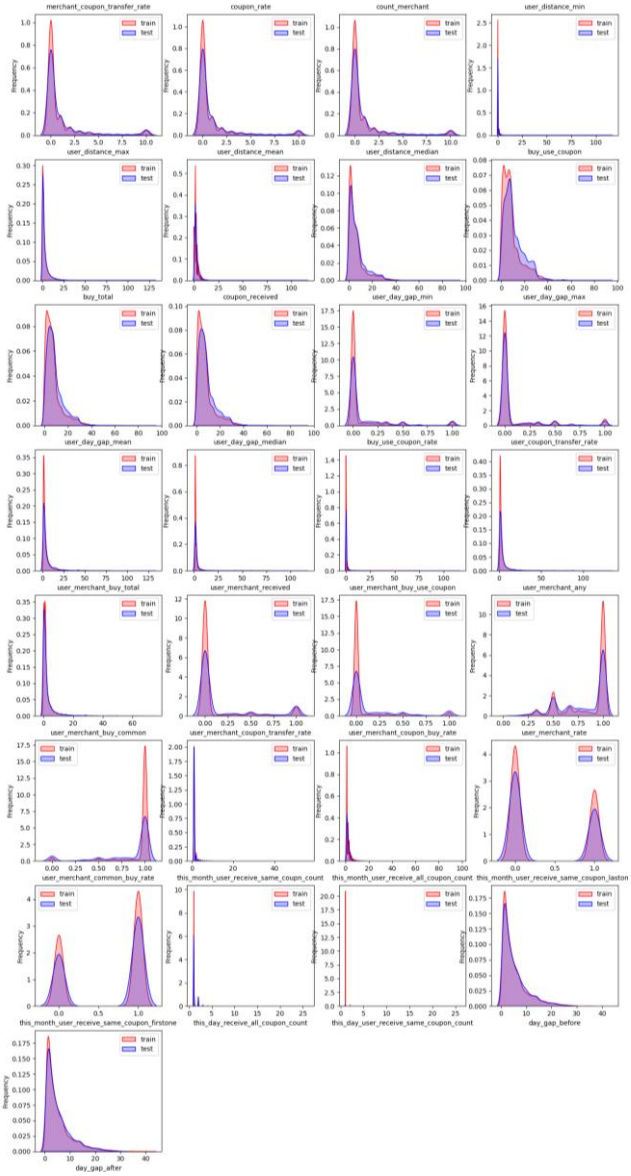
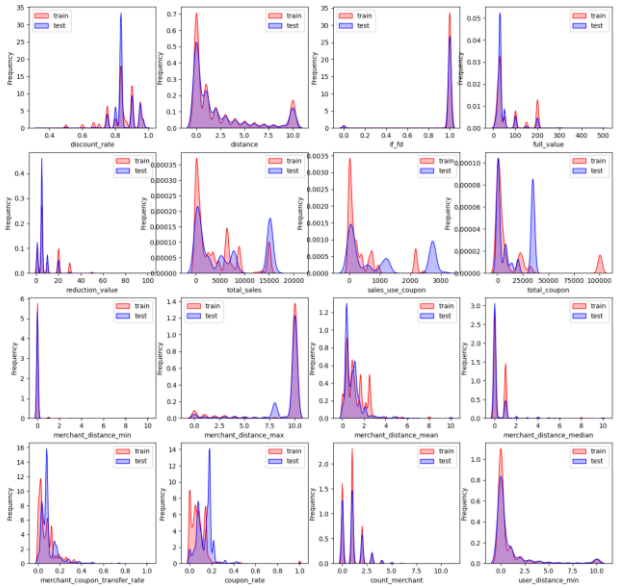


图 4.3 训练集和测试集特征分布对比情况

Offline 训练集和测试集的特征分布对比情况如下：



可以发现，在均为满减优惠的数据中，offline 训练集和测试集的特征分布就比较接近了。但也有 sales_use_coupo 等特征在训练集和测试集之间相差比较大。不过因为这是生成的特征，在训练集和测试集之间可能因为不同时间的商家、用户占比不同，所以造成特征分布不同。不能因为分布不同就直接删除。要在后续通过模型来选择。

（四）特征相关性分析

列出各个特征与要预测的数据项——是否用券（label）之间的相关性。

可以发现几个穿越特征都排在了前面，因为它们都是在已经发生的“事实”基础上统计的，所以相关性一半会比正常的特征强。在正常的特征中，客户与某个商家之间的交互特征因为指向性很强，所以也都排在前面。

特征工程结束。我们生成了 3 个版本的特征：f1，sf2，sf3。后续模型的训练将主要以此为主。

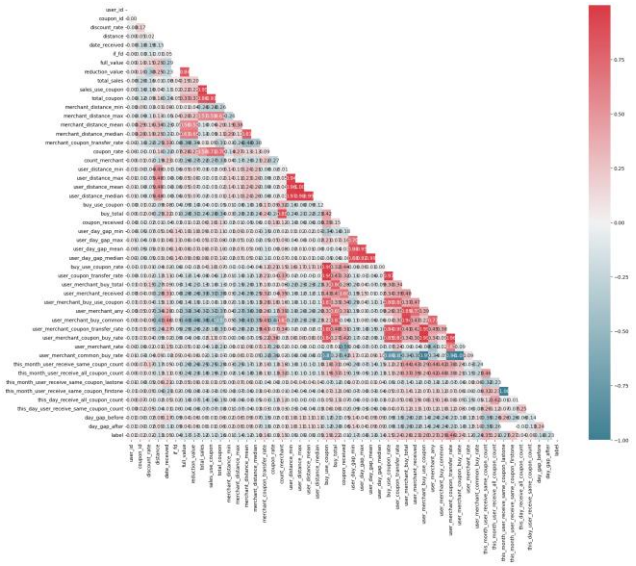


图 4.4 特征相关性

五、模型训练

（一）模型训练与评估

在应用特征和数据对模型进行训练时，需要结合评估指标来判断模型的优劣。

我们使用优惠券核销预测的平均 AUC 作为评价标准，即对每个优惠券 id（coupon_id）单独计算核销预测的 AUC 值，再对所有优惠券的 AUC 值求平均作为最终的评价标准。因此，不能直接在验证集上使用 sklearn.metrics.roc_auc_score，而要重新编写验证函数。

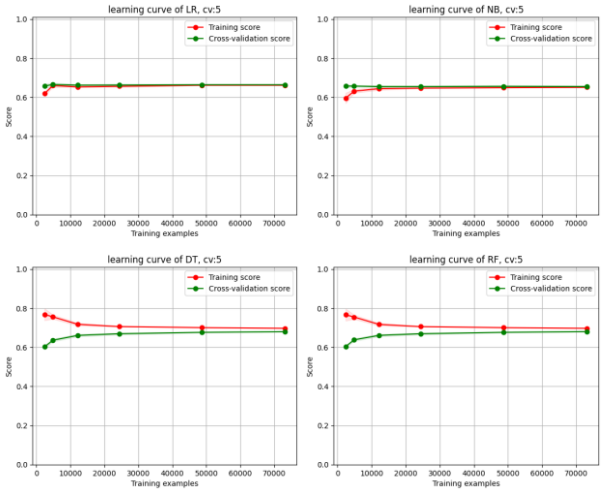
```
# 性能评价函数
#     目标是预测投放的优惠券是否核销
#     针对此任务及一些相关背景知识，使用优惠券核销预测的平均 AUC（ROC 曲线下面积）作为评价标准
#     即对每个优惠券 coupon_id 单独计算核销预测的 AUC 值，再对所有优惠券的 AUC 值求平均作为最终的评价标准
def myauc(test):
    testgroup = test.groupby(['coupon_id'])
    aucs = []
    for i in testgroup:
        coupon_df = i[1]
        if len(coupon_df['label'].unique()) < 2:
            continue
        auc = metrics.roc_auc_score(coupon_df['label'], coupon_df['pred'])
        aucs.append(auc)
    return np.average(aucs)
```

图 5.1 性能评价函数

其他模型训练与评估的相关代码在此不再赘述，可以在附件中查看详情。

（二）不同算法模型性能对比

在简单特征 f1 训练中，不同算法模型性能对比如下：



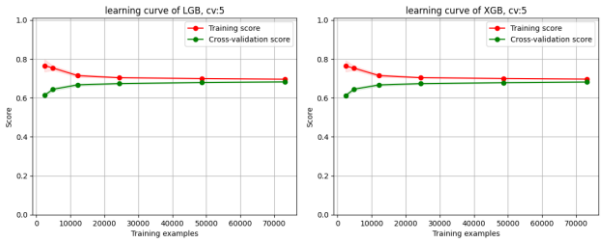


图 5.2 模型性能（逻辑回归、朴素贝叶斯、决策树、随机森林、LightGBM、XGBoost）

表 5.3 不同算法模型性能对比

特征	模型	总体 AUC	Coupon AUC
f1	逻辑回归 LR	0.6723	0.5381
sf2		0.7590	0.6020
sf3		0.8095	0.7279
f1	朴素贝叶斯 NB	0.6671	0.5381
sf2		0.7342	0.6061
sf3		0.7813	0.7119
f1	决策树 DT	0.6482	0.5303
sf2		0.5281	0.5209
sf3		0.5768	0.5708
f1	随机森林 RF	0.6481	0.5300
sf2		0.6870	0.5657
sf3		0.7725	0.7078
f1	LightGBM LGB	0.6534	0.5328
sf2		0.7741	0.5985
sf3		0.8285	0.7374
f1	XGBoost XGB	0.6535	0.5313
sf2		0.7277	0.5847
sf3		0.8049	0.7278

根据不同算法模型性能对比的总体 AUC 和 Coupon AUC 指标，我们能得到以下结论：

(1) 朴素贝叶斯算法表现还可以，但成绩不是很好，算法优化的余地不大。

(2) 逻辑回归算法表现不错，但根据逻辑回归算法的特点，它在本问题上提升的空间不大。

(3) 决策树算法的表现不好。决策树的缺点是容易过拟合，忽略属性之间的相关性。另外，决策树的结果可能是不稳定的，因为在数据中一个很小的变化就可能导致生成一个完全不同的树。这个问题可以通过集成决策树来解决，而随机森林的成绩比决策树好，就是这个原因。

(4) XGBoost 算法和 LightGBM 算法表现不错。LightGBM 具有训练效率高、使用内存低、支持并行化学习等特点。在处理不同问题时，XGBoost 算法和 LightGBM 算法的成绩各有千秋，不过 LightGBM 算法比 XGBoost 算法快得多，我们最终选择使用 LightGBM 算法。

（三）不同特征效果对比

对比上述模型的训练结果可以发现，在同一模型的情况下，采用 sf2 特征版本比采用 sf1 特征版本的结果好很多，

这是因为 sf2 特征版本的提取过程中采用了滑窗方案，增加了很多统计特征。而在同一模型的情况下，sf3 特征版本与 sf2 特征版本相比，又有了很大的提高，这是因为 sf3 特征版本中增加了穿越特征。

（四）结果预测与结果输出

通过分析发现特征 sf3 版本通过 LightGBM 分析的结果不错，我们将结果进行输出并保存在文件 sf3_lgb.csv 中。

	user_id	coupon_id	date_received	Probability
0	4129537	9983	20160712	0.007
1	6949378	3429	20160706	0.132
2	2166529	6928	20160727	0.007
3	2166529	1808	20160727	0.013
4	6172162	6500	20160708	0.036

图 5.4 输出结果头几行数据

六、模型验证

（一）评估指标

模型验证优化主要包括确定评估指标、交叉验证、模型比较、验证结果可视化、结果分析和模型调参。

我们的评估指标是 AUC。

（二）交叉验证

使用 4 种交叉验证方法：

(1) 简单交叉验证：设置切分后的训练数据为 80%，验证数据为 10%。

(2) K 折交叉验证：验证方式为 5 折交叉验证。

(3) 留一法交叉验证和留 P 法交叉验证：验证方式为 LPO-CV，p-200。

(4) StratifiedFold 和 StratifiedShuffleFold：验证方式为 5 折分层交叉验证。

通过比较发现还是 StratifiedKFold 比较适合。因为正负样本分布不均匀，而 StratifiedKFold 分层采样交叉切分，确保训练集，测试集中各类别样本的比例与原始数据集中相同。

（三）模型比较

在选定 StratifiedKFold 后，可以直接用上面的方法对不同模型进行比较，首先，定义评估模型算法性能的函数。

```
# 对算法进行分析
def classifier_df_score(train_feat, classifier, cvnum, param=None):
    clf = get_sklearn_model(classifier, param)
    train = train_feat.copy()
    target = get_target_df(train_feat).copy()
    kf = StratifiedKFold(n_splits=cvnum)

    scores = []
    score_coupons = []
    for _, (train_index, test_index) in enumerate(kf.split(train, target)):
        train_data, test_data, train_target, test_target = train.iloc[train_index], train.iloc
        [test_index], target[
            train_index], target[test_index]
        clf.fit(get_predictors_df(train_data), train_target)
        test_pred = clf.predict_proba(get_predictors_df(test_data))[:, 1]
        score_test = roc_auc_score(test_target, test_pred)
        test_data['pred'] = test_pred
        score_coupon_test = myauc(test_data)
        scores.append(score_test)
        score_coupons.append(score_coupon_test)

    print(classifier + '总体AUC:', scores)
    print(classifier + 'Coupon AUC:', score_coupons)
```

图 6.1 评估模型算法性能函数

然后，使用不同特征版本分别运行几种典型的分类算法，如朴素贝叶斯、逻辑回归、随机森林及 LightGBM，进行结果对比。

通过对比训练集上不同算法的运算结果可以发现，f1 特征版本效果最差，sf2 特征版本通过滑窗和增加统计特征提高了分数，而增加了穿越特征的 sf3 特征版本效果最好。这一点与前面的分析结论一致。

（四）验证结果可视化

将验证结果可视化主要通过绘制学习曲线和验证曲线来实现。学习曲线就是通过画出不同大小训练集和交叉验证的准确率，观察模型在新数据上的表现，是进而来判断模型是否存在方差过高或偏差过高，以及增大训练集是否可以减小过拟合等。

验证曲线和学习曲线的区别：横轴为某个超参数的一系列值，由此来看是不同参数设置下的准确率，而不是不同训练集大小下的准确率。从验证曲线上可以看到，随着超参数设置的改变，模型可能会有从欠拟合到合适再到过拟合的过程，进而可以选择一个合适的设置来提高模型的性能。

使用三种特征版本，比较朴素贝叶斯、逻辑回归、决策树、随机森林及 LightGBM 算法。

图 6.2 模型验证结果（从上至下依次为朴素贝叶斯、逻辑回归、决策树、随机森林、LightGBM 算法，从左到右依次为 f1、sf2、sf3 特征版本）

可以发现，使用 f1 特征版本的评测分普遍比较低，是欠拟合。而使用 sf3 特征版本，决策树和随机森林都表现出过拟合，LightGBM 表现比较好。

通过对比训练集上不同算法的运算结果可以发现：

(1) f1 特征版本因为特征比较少，有严重的欠拟合，所以所有算法的分数都比较低。

(2) sf2 特征版本通过滑窗和增加统计特征，比 f1 特征版本的性能有了飞跃性的提高。

我们选取 sf3 特征版本和 LightGBM 算法作为最后方案。

七、模型改进与优化

（一）模型调参方法

(1) 网格搜索：网格搜索是在所有候选的参数选项中进行循环遍历，尝试每一种可能性，表现最好的参数就是最终的结果（暴力搜索）。原理：在一定的区间内，通过循环遍历，尝试每一种可能性，并计算其约束函数和目标函数的值。对满足约束条件的点，逐个比较其目标函数的值，抛弃坏的点，保留好的点，最后得到最优解的近似解。为了避免初始数据的划分对结果的影响，一般情况下网格搜索需要和交叉验证结合使用。

(2) 随机搜索：随机搜索是利用随机数去求函数近似最优解的方法，区别于网格搜索的暴力搜索。原理：在一定的区间内，不断地、随机地而不是有倾向性地产生随机点，并计算其约束函数和目标函数的值。对满足约束条件的点，逐个比较其目标函数的值，抛弃坏的点，保留好的点，最后得到最优解的近似解。随机搜索建立在概率论的基础上，所取随机点越多，得到最优解的概率也就越大。这种方法存在精度较差的问题，但是找到近似最优解的效率高于网格搜索。随机搜索一般用于粗选或普查。

(3) 启发式搜索：启发式搜索又称有信息搜索，是利用问题拥有的启发信息来引导搜索，以达到减少搜索范围、降低问题复杂度的目的。原理：在状态空间中，对每一个搜索的位置进行评估得到最好的位置，再从这个位置进行搜索，直到目标。这样可以省略大量无谓的搜索路径，提高了搜索效率。在启发式搜索中，对位置的估价是十分重要的。

（二）模型调参方案

我们使用 sf3 特征集和 LightGBM 算法进行网格调参。调参次序为：

(1) 学习率和迭代次数

(2) max_depth 和 num_leaves

(3) min_data_in_leaf 和 max_bin in

(4) feature_fraction、bagging_fraction、bagging_freq

(5) lambda_l1 和 lambda_l2

(6) 确定 min_split_gain

(7) 降低学习率，增加迭代次数，验证模型

```
model = LGBMClassifier(boosting_type='gbdt',
                       objective='binary',
                       metrics='auc',
                       n_estimators=200,
                       max_depth=6,
                       num_leaves=40,
                       max_bin=400,
                       min_data_in_leaf=120,
                       learning_rate=0.05,
                       lambda_l1=1e-05,
                       lambda_l2=1e-05,
                       min_split_gain=0.0,
                       bagging_freq=4,
                       bagging_fraction=0.9,
                       feature_fraction = 0.6)
```

图 7.1 最优参数

具体的代码实现过程在此不再赘述，可以在附件中查看详情。

将模型优化后预测的 AUC 与用模型默认参数预测的 AUC 进行对比，比较运行结果，可以发现：调参后的结果比默认参数的有所提高，但不是高很多，比不上特征对结果的影响。而且因为调参只能在测试集上操作，有时候还会造成过拟合，所以调参一般都是在后期再做，前期主要是做好特征和模型的选择。

（三）可视化调参过程

通过绘制验证曲线可视化调参过程。

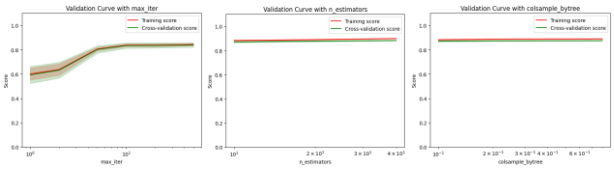


图 7.2 调参过程可视化曲线

八、总结

在本问题中，我们主要关注了 O2O 优惠券核销预测的问题，采用了机器学习方法，特别是 XGBoost 和 LightGBM 两种算法。通过详尽的数据探索和特征工程，我们从原始数据中提取了多维度的特征，包括用户行为、优惠券属性和商户信息等，并通过滑窗等方法进一步丰富了数据特征，以适应模型的需求。

我们的目标是提高优惠券核销的预测准确度，为此我们采用了 AUC 作为模型评估的标准。在实验过程中，我们对不同的特征组合和算法进行了广泛的测试和对比，最终发现通过引入穿越特征和利用 LightGBM 模型可以显著提高预测的准确性。

通过本项目，我们实现了以下几个预期目标：

(1) 成功应用机器学习技术对 O2O 优惠券的核销概率进行预测，特别是在处理大规模数据集时，展示了数据挖掘和特征工程的重要性。

(2) 通过不断迭代的特征工程和模型调优，显著提升了预测的 AUC 值，从而增强了模型的实用性和商业价值。

(3) 探索了特征之间的关系和对模型影响的深入分析，增进了对 O2O 行业用户行为和优惠券使用模式的理解。

此外，在解决本问题的过程中也收获了一些宝贵的经验：

(1) 特征工程是提升模型性能的关键，合理的特征提取和优化能够有效提高模型的预测准确度。

(2) 交叉验证和适当的模型调参对于防止过拟合和提升模型泛化能力至关重要。

(3) 在实际应用中，对算法的选择和优化应结合具体业务场景和数据特点，以实现最优的预测效果。

通过对本问题的研究，我不仅加深了对机器学习在大数据时代下的应用理解，也为 O2O 行业的营销策略提供了科学的数据支持和决策工具。

九、参考文献

[1] 天池平台,《阿里云天池大赛赛题解析——机器学习篇》,电子工业出版社,2020 年 9 月.