网络广告点击率预估及欺诈检测技术探究

课程名称：Python 机器学习与应用

学　　院：管理学院

姓　　名：赵哲冕

学　　号：17420182200850

提交日期：2021 年 12 月 28 日

目录

1 点击欺诈的背景与意义

1.1 网络广告点击欺诈的背景

1.1.1 网络广告模式

移动互联网蓬勃发展十几年来，不但创造了全新的社会生活方式，也影响着网民的生活方方面面。近年来互联网公司层出不穷并不断发展，他们的盈利模式逐渐分流成游戏、电商、广告等多种渠道，广告作为其中最方便快捷的途径，牢牢占据整个移动互联网行业的半壁江山。国内外各大互联网公司巨头，如脸书、谷歌、百度、阿里巴巴、腾讯及各抖音快手等小视频 APP，广告收入占其总收入比例都极高。与此同时，移动互联网的发展也为广告主提供了更多的投放渠道。随着在移动互联网上投放的广告越来越多，移动互联网广告市场规模与日俱增。

互联网广告的主要收费模式是按点击次数付费(Cost-per-click, CPC)。按点击次数付费的原理是，广告投放者按照广告浏览者通过点击广告下载应用程序的次数支付费用。

1.1.2 广告点击欺诈现状及危害

按点击次数付费模式下的无效点击会对广告计费产生较大影响，其中最需要防范和规避的是点击欺诈。一般来讲，将通过人工点击或者自动点击的方式，恶意增加广告发布者收入以及广告主支出的点击称为点击欺诈(Click Fraud)。广告主可能对自己的广告进行欺诈性点击，从而提升自身曝光率，或者在竞价环境下消耗对手广告，从而使自己得到更多曝光量，再或者向对手广告进行欺诈性点击，消费对手的广告费用，从而提高自身竞争力；媒体还存在被动的欺诈性点击行为，媒体会采纳许多引流来自各种复杂渠道的流量，并与渠道按照点击的用户数进行结算，因此这些引流方也会因此进行欺诈性点击行为。

1.2 研究意义及方法

1.2.1 研究意义

恶作剧者或竞争对手通过重复点击网站上的某个关键词，导致购买该关键词的广告客户需要为这些无效点击支付高额的广告费用。从技术角度来说，尽管可以通过后台监控部分地避免恶意点击的产生，或者减少恶意点击产生之后费用的

支付，但公司并没有办法完全避免这种现象的存在。

点击欺诈为网络生态系统的健康发展带来了挑战，不仅损害了广告主的利益，同时也破坏了广告主和广告商之间的信任。因此，行之有效的欺诈点击检测方法是至关重要的。

本文的探索基于我国移动互联网广告行业点击欺诈的背景，针对点击欺诈预测问题的准确程度进行深入探索，结合 LightGBM 算法模型，预测用户在点击移动应用广告后是否会下载应用，达到实现点击欺诈的可能性预测，为广告主节约成本，维护移动互联网广告业生态的目的。

1.2.2 研究方法

本文选取 kaggle 平台 TalkingData AdTracking Fraud Detection Challenge 比赛提供的，来自某数据服务公司的真实广告点击数据，用计算机程序建模的实验手段，对实验数据进行描述性统计、数据清洗、数据处理、特征工程，并利用 LightGBM 算法，构建适合本文所选数据的模型并调整参数，用来对欺诈点击进行预测。

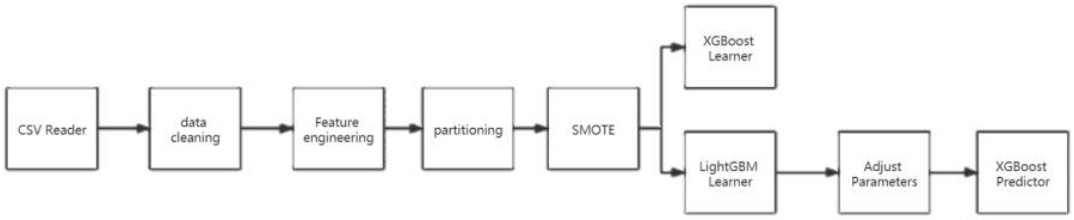

图 1-1 研究流程图

2 理论基础

2.1 核心算法

2.1.1 GBDT 算法介绍

GBDT 是一种用梯度提升的方法训练的决策树模型，是监督学习算法的一种。梯度提升决策树是对传统 Boosting 算法的一种优化，是一般可以用于处理回归和分类数据的算法。

梯度提升是一种用于回归、分类和排序任务的机器学习技术，属于 Boosting 算法族的一部分。Boosting 是一族可将弱学习器提升为强学习器的算法，属于集成学习的范畴。Boosting 方法基于这样一种思想：对于一个复杂任务来说，将多

个专家的判断进行适当的综合所得出的判断，要比其中任何一个专家单独的判断要好。梯度提升同其他 boosting 方法一样，通过集成多个弱学习器，通常是决策树，来构建最终的预测模型。

（1） 初始化弱学习器

$$f_0(x) = \arg min_c \sum_{i=1}^{N} L(y_i, c)$$

（2） 对 m = 1 , 2 , . . . , M 有

（a） 对每个样本 i = 1 , 2 , . . . , N 计算负梯度，即残差

$$\gamma_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x) = f_{m-1}(x)}$$

（b）将上步得到的残差作为样本新的真实值，并将数据(x_i , r_{im}) ， $i = 1, 2, .. N$作为下棵树的训练数据，得到一颗新的回归树 f_m (x)其对应 f_m 的叶子节点区域为 R_{jm} , $j = 1, 2, \ldots, J$ 。其中 J 为回归树 t 的叶子节点的个数。

（c）对叶子区域 j=1,2,…J 计算最佳拟合值

$$\gamma_{jm} = \underbrace{argmin}_{\gamma} \sum_{x_i \epsilon R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$

(d)更新强学习器

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} \gamma_{jm} I(x \epsilon R_{jm})$$

(3)得到最终学习器

$$f(x) = f_M(x) = f_0(x) + \sum_{m=1}^{M} \sum_{j=1}^{j} \gamma_{jm} I(x \epsilon R_{jm})$$

2.1.2 XGBoost 算法

XGBoost 算法是 GBDT 算法的进化形式，其基学习器通常选择决策树模型，通过不断迭代生成新树学习真实值与当前所有树预测值的残差，将所有树的结果累加作为最终结果，以此获取尽可能高的分类准确率。

XGBoost 算法将模型的表现与运算速度的平衡引入目标函数，在求解目标函数时对其做二阶泰勒展开，以此加快求解速度，减少模型运行时间 ；同时引入正则项控制模型复杂度，避免过拟合。XGBoost 算法的目标函数如公式所示：

$$obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \theta(f_k) = \sum_{i=1}^{n} [f_t(x_i) g_i, + \frac{1}{2}(f_t(x_i))^2 h_i] + \theta(f_t)$$

其中，g_i和h_i分别为损失函数的一阶、二阶导数，$\theta(f_t)$为第 t 棵树的结构。

XGBoost 算法在构建每棵树的过程中，通过计算所有候选特征分枝前与分枝后结构分数之差obj_{split}，从中挑选出 obj_{split}最大的特征进行分枝，保证生成的树最优，以此提高预测准确率。

XGBoost 算法中最终目标函数的大小与树结构密切相关，最终模型效果直接与叶子节点有关，在寻找最优树结构的过程中可确定叶子节点的数量。根据 XGBoost 算法在不断迭代构建每棵树时选择最优特征和最优切分点的过程，可使用 XGBoost 算法来创造新特征组合。

2.1.3 Lightgbm 算法

LightGBM 算法是 GBDT 算法的另一种进化形式，该算法使用深度限制的叶子生长（leaf-wise）策略，从当前叶子节点中找到增益值最大的节点进行分裂，并对树的深度进行限制，防止过拟合，缩短寻找最优深度树的时间。同时保证分裂次数相同的情况下，能够降低误差，得到更高精度。在构建树的过程中，最浪费时间和计算机资源的是寻找最优分裂节点的过程，对此，LightGBM 使用直方图算法、单边梯度抽样算法和互斥特征捆绑算法来提升运行效率。

2.1.4 gridsearch

网格搜索参数寻优法是一种最基本的参数优化算法。其基本思想是让待搜索的参数在一定的空间范围内按照规定的步距划分网格并遍历网格内的所有点进行取值，并将每次取出的参数组带入系统中验证其性能，最终取使系统性能达到最优的参数组作为最佳参数。该算法实质上就是通过暴力搜索的方式，对每一组参数进行试算。因此，当规定的搜索范围比较大且网格划分比较密集时，网格搜索寻优法会非常的耗时。

由于此种方法需要逐一测试网格内所有的参数组，所以当搜索步距足够小的情况下，网格搜索法理论上能够搜索到全局最优的参数组。但也正由于这个原因，

网格搜索法相对比较耗时。

2.2 评价标准—ROC 曲线

ROC 曲线即接受者操作特性曲线，就是以假阳性概率为横轴，真阳性概率为纵轴，由被测特征在特定条件下采用相对应的判定条件得出的结果画成的坐标图。

考虑一个二分问题，即将实例分成正类或负类。对一个二分问题来说，会出现四种情况。如果一个实例是正类并且也被预测成正类，即为真正类(True positive),如果实例是负类被预测成正类，称之为假正类(False positive)。相应地，如果实例是负类被预测成负类，称之为真负类(True negative),正类被预测成负类则为假负类(false negative)。

列联表如下表所示，1 表示正类，0 表示负类。

Actual class\Predicted class	1	0	Total
1	TP	FN	P
0	FP	TN	N
Total	TP+FP	FN+TN	P+N

表 2-1 分类判断的列联表

ROC 曲线的绘制具体步骤如下：

（1）将所有样本被划分为阳性样本的概率值按照由高到低顺序排序；

（2）将概率值大于设定阈值的样本记为阳性样本，低于阈值的记为阴性样本；

（3）每次选取不同的阈值，计算得到一组 TP 概率和 FP 概率，以 FP 概率为横坐标，TP 概率为纵坐标，得到 ROC 曲线上一点；

（4）连接（3）中得到的点，即得到 ROC 曲线。

ROC 曲线是根据与对角线进行比较来判断模型的好坏，但这只是一种直觉上的定性分析，如果我需要精确一些，就要用到 AUC，也就是 ROC 曲线下与坐标轴围成的面积。AUC 值越大，则模型的拟合效果越好。一般情况下 ROC 曲线位于直线 $y=x$ 下方，因此 AUC 值的取值范围为 $AUC \in [0.5,1)$，取值越接近 1 说明模型的拟合效果越好，取值为 0.5 则说明模型真实性较低，没有实际

价值。选用 AUC 值作为模型判定的标准，能够较为直观的分析和比较模型的准确性，是一种普遍应用的检验方法。

3 数据处理

3.1 数据来源

在选取我所需要的数据之前，我先思考出了所需要做的项目类型：由于近年来网络安全问题日渐严重，所以我打算做的项目就与常见的网络点击欺诈有关，并且根据着方向进行数学查阅和搜集。

我的报告最终所用的研究数据来自 kaggle 平台的 TalkingDat a AdTracking Fraud DetectionChallenge 比赛数据，该数据集记录了某数据服务公司四天内约两亿次广告点击数据，其中每条数据都具有以下共同的属性，包括:该次点击的 IP 地址(IP)、广告所展示的手机程序(APP)、点击广告所用的设备类型(device)、点击广告的手机所用的操作系统版本(os)、移动广告发布渠道(channel)、点击广告的时间戳(click_time)、若用户点击广告后下载 APP，下载的时间戳(attr ibuted_time)以及预测的目标，用户是否下载广告内 APP(is_attributed)。其中用户是否下载广告内 APP(is_attributed)是本文需要预测的二进制目标，取值为 1 时表示此次点击导致用户下载，取值为 0 时则表示点击没有导致用户下载，IP 地址(ip)、广告所展示的手机程序(app)、点击广告所用的设备类型(device)、点击广告的手机所用的操作系统版本(os)和移动广告发布渠道(channel)都是经过编码的特征，这些特征并不是原始取值，而是将这些取值按照顺序编码;下载的时间戳(attributed_time)这一特征仅在训练集中存在。比赛所给的数据集相当大，约有 8GB，训练集和测试集共包含 2 亿个观测值，且训练集数据中观测值仅有 0.2%为欺诈性点击行为。

而在我的报告当中，无法处理如此巨大的数据集和观测值数量，所以我选取了 100000 个具有代表性的观测值组成 sample 并作为数据训练集，总数据大小也变为 15000kb 左右的数据。

3.2 描述性统计

在进行数据预处理之后，我将各个特征作为横坐标，做各个特征的唯一计数统计，并且使用统计值作为纵坐标，描绘出特征数直方分布图：

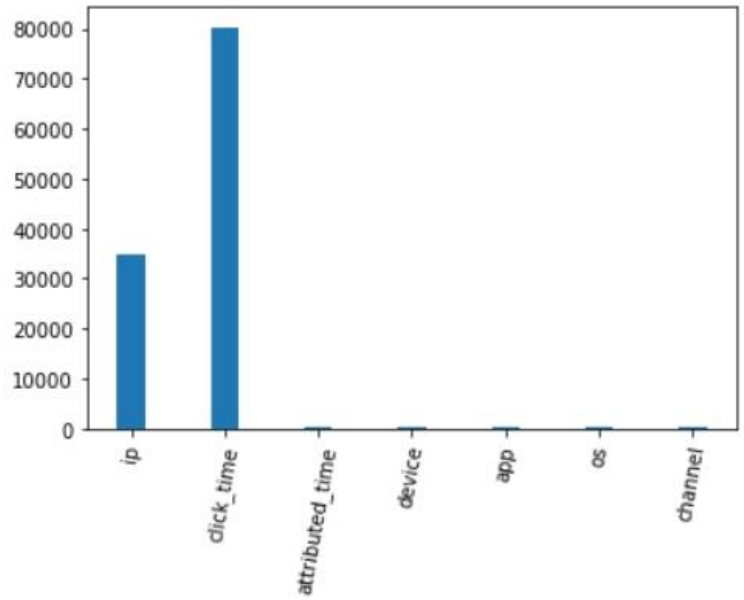

图 3-1 特征数直方分布图

可以看出，广告点击时间戳(click_time)这一特征有最多取值，其次是 IP 地址(ip)，广告所展示的手机程序(app)、点击广告所用的设备类型(device)、点击广告的手机所用的操作系统版本(os)、移动广告发布渠道(channel)唯一取值量都非常少，符合实际情况，没有发现有数据异常。点击的 IP 地址(ip)和点击广告的时间戳(click_time)作为预测点击欺诈最重要的两个特征，需要参考更多信息来对特征进行处理。

而其中的 click_time 作为点击的时间戳，本身就具有独一性，所以在进行数据特征处理的时候暂且不考虑将时间戳（click_time）纳入处理范围。

3.2.1 ip 数据特征处理

对 ip 数据特征进行处理，我分别使用 mean 函数、median 函数、mode 函数、percentile 函数、min 函数、max 函数计算出了 ip 数据特征的均值、中位数、众数、下四分位数、上四分位数、最小值以及最大值，得出的结果分别为 91255.87967、79827.0、5348、40552.0、118252.0、9、364757

在把 ip 数据特征处理完之后，我绘制了 ip 数据的直方分布图以及密度分布图，并且绘制了垂直线：

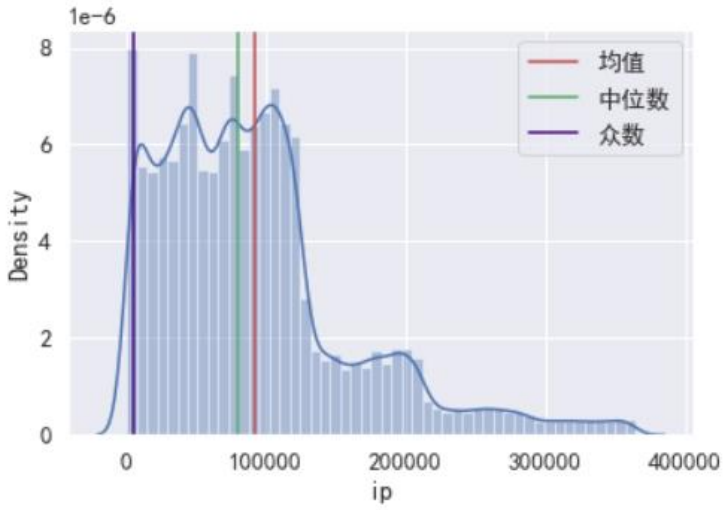

图 3-2 ip 数据的直方分布图以及密度分布图

可以看出，百分之 40 以上的点击量由 0~100000 之间的 ip 地址组成，推测这些 ip 地址可能是一些大型网络中心的 ip 地址。并且由于数据中点击的 IP 地址（ip）都是整型数据，可以看出数据集中的 IP 地址不符合我的标准 IP 地址的格式，可能是由于不同的采样或者编码加工而造成的。

3.2.2 os 数据特征处理

对 os 数据特征进行处理，我同样用 mean 函数、median 函数、mode 函数、percentile 函数、min 函数、max 函数计算出了 os 数据特征的均值、中位数、众数、下四分位数、上四分位数、最小值以及最大值，得出的结果分别为 22.81828、18.0、 19 、13.0、 19.0、 0、 866

计算出相应数据之后，我同样绘制出数据的直方分布图和密度分布图，并绘制垂直线：

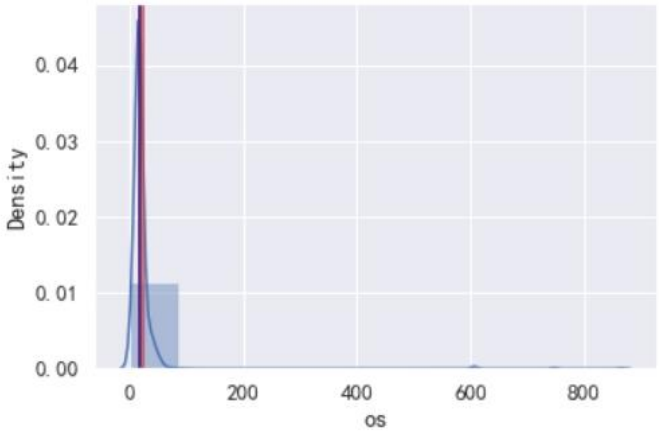

图 3-3 os 数据的直方分布图以及密度分布图

对于点击广告的手机所用的操作系统版本(os)这一项编码特征，946 个不同的系统版本从 0 到 945 编码，第一四分位数为 13.0，中位数为 18.0，均值为 21.81828，第三四分位数为 19.0，可以看出整体数据左偏，主要集中在编号较小的操作系统，可以得出结论:数据编码 10 到 20 左右的操作系统为大众常用的操作系统，其他大部分的操作系统可能极其小众，占比很小;根据 os 特征分布的直方图，使用数量排前两位的编号 19 和 13 占比超过百分之 60，排前十的操作系统编号集中在 6 到 22，与之前分析的结果基本吻合。数量最多的两个操作系统可能是流行的 Android 和 iOS 系统，也可能由于采样数据的局限性，导致大部分数据来自这两个操作系统。

3.2.3 device 数据特征处理

对 device 数据特征进行处理，我同样用 mean 函数、median 函数、mode 函数、percentile 函数、min 函数、max 函数计算出了 os 数据特征的均值、中位数、众数、下四分位数、上四分位数、最小值以及最大值，得出的结果分别为 21.77125、1.0、 1 、1.0、 1.0、 0 、3867

同样，根据数据绘制了数据直方分布图和密度分布图，并且绘制了垂直线：

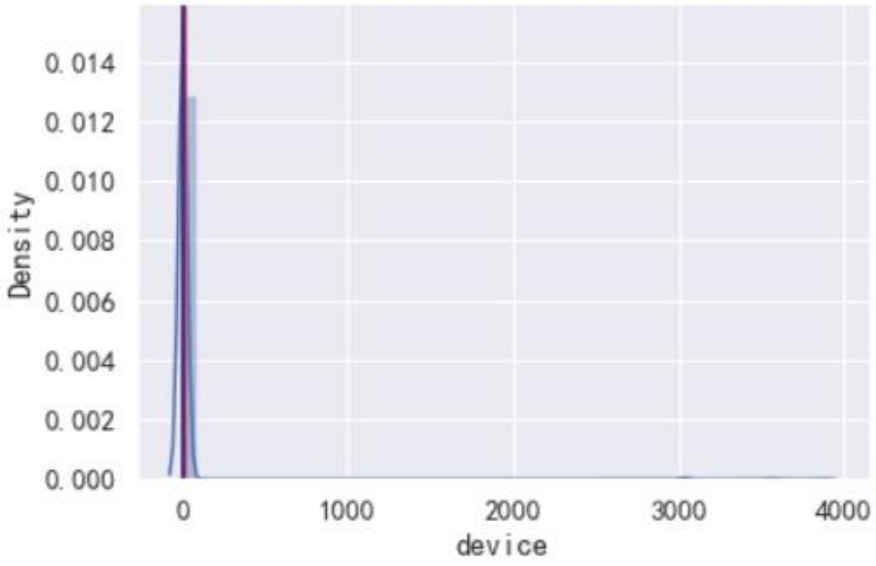

图 3-4 device 数据的直方分布图以及密度分布图

对于点击广告所用的设备类型(device)这一编码特征，数据中存在 4213 个不同的系统版本，从 O 到 4212 编码，该数据第一四分位数为 1，中位数也为 1，

第三四分位数为 1，均值为 15.2，可以看出，虽然特征 device 取值很多，但是绝大部分数据来自编号为 1 的设备，可以得出结论:数据编码 1 的设备可能为大众常用的设备，是主流的手机品牌的主流机型其他绝大部分的设备可能极其小众，占比很小。再观察 device 数据的分布直方图，可以看出超过百分之九十四的数据来自同一设备类型，即编号为 1 的设备，与之前分析的结果吻合，这可能是某个爆款手机型号，但这种数据较为极端，现实情况即使有爆款手机机型也很难达到如此高的市场占比例，更有可能的原因是与数据来源有关，数据采样时可能经过筛选，抑或是在某一个手机机型的用户当中进行了较多抽样选择。但既然存在 4213 个不同类型的设备，该特征就有其分析的价值，应该继续分析，将其进行特征工程用于建模

3.2.4 channel 数据特征处理

对 channel 数据特征进行处理，我同样用 mean 函数、median 函数、mode 函数、percentile 函数、min 函数、max 函数计算出了 os 数据特征的均值、中位数、众数、下四分位数、上四分位数、最小值以及最大值，得出的结果分别为 268.83246 、258.0、 280、 145.0、 379.0、 3 、498

根据数据绘制了数据直方分布图和密度分布图，并且绘制了垂直线：

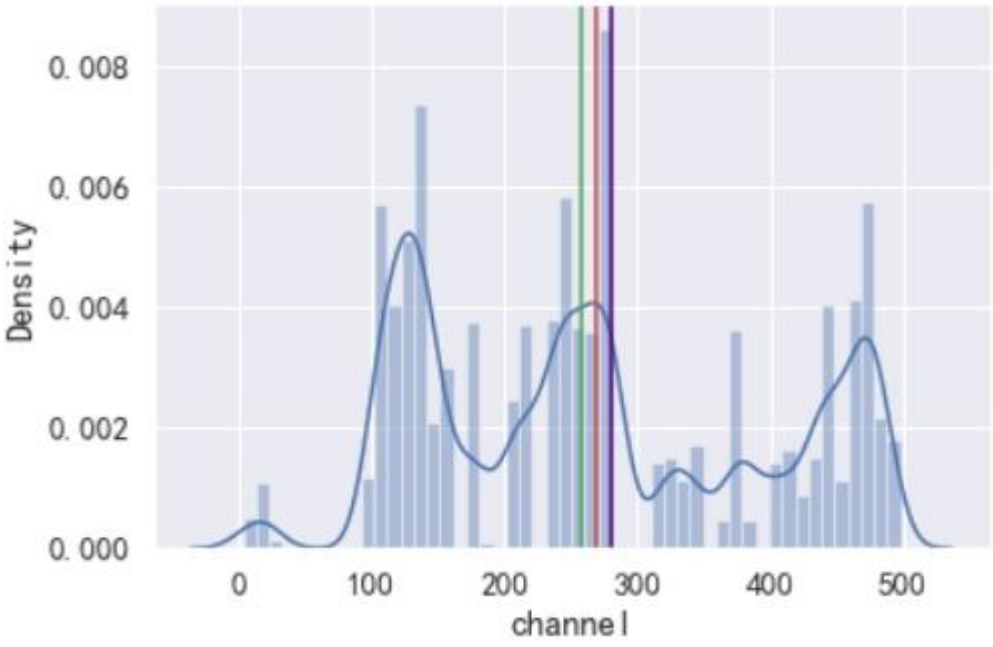

图 3-5 channel 数据的直方分布图以及密度分布图

移动广告发布渠道（channel)这一编码特征，数据中存在 499 个不同的系统

版本从 O 到 498 编码，该数据第一四分位数为 145，中位数也为 258.0，第三四分位数为 379.0,均值为 268.83246，可以看出发布渠道特征中各分类分布较为均匀，可能没有贡献突出的渠道;再观察特征 channel 的分布特征直方图，可以看出 channel 特征数量分布较为均匀，最多的为编号 258.0，仅占不到百分之二十，其他占比排名前十的渠道没有明显的分布特征，占比都在百分之十左右。可以得出结论，广告发布的各个渠道引流较为平均，这可能导致该特征在建模时贡献有限，但发布渠道作为意义很重要的特征，仍然应该将其选为建立模型的特征之一。

3.2.5 app 数据特征处理

对 channel 数据特征进行处理，我同样用 mean 函数、median 函数、mode 函数、percentile 函数、min 函数、max 函数计算出了 os 数据特征的均值、中位数、众数、下四分位数、上四分位数、最小值以及最大值，得出的结果分别为 12.04788、12.0、 3、 3.0、 15.0、 1、 551

根据数据绘制了数据直方分布图和密度分布图，并且绘制了垂直线：

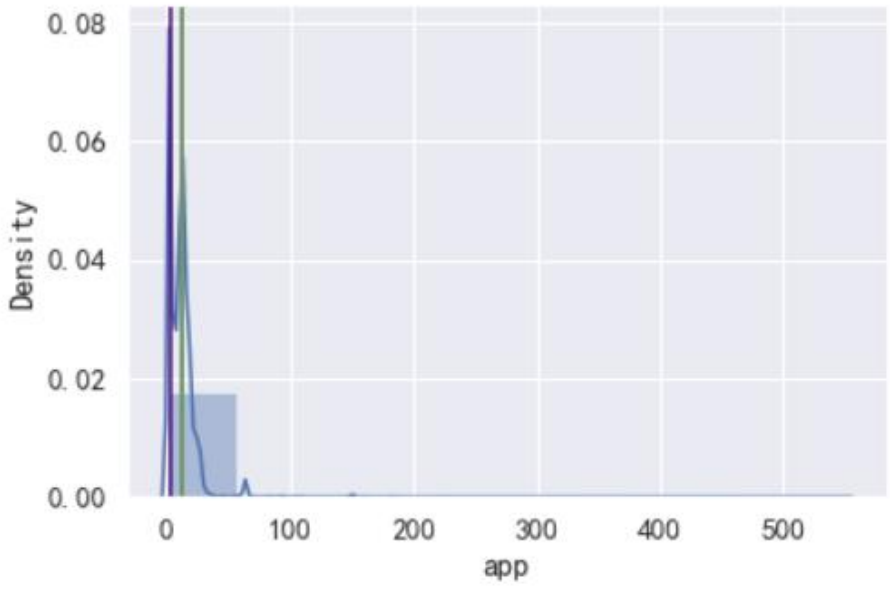

图 3-6 app 数据的直方分布图以及密度分布图

对于广告所展示的手机程序(app)，数据中存在 763 个不同的手机程序，从 0 到 762 编码，该数据第一四分位数为 3，中位数为 12，第三四分位数为 15，均值为 12.04788，可以看出整体数据左偏，主要集中在编号较小的手机程序，可以得出结论:数据编码 5 到 10 左右的手机程序下载量较大，其他大部分的手机程序

可能较为小众，下载量很小;根据 app 特征分布的直方图，下载数量排前十位的手机程序呈阶梯状分布，其中下载最多的 app 编号为 3，占比近百分之二十，排名前五的广告数量达到总量的百分之七十五，这五个手机程序可能是较为流行的手机应用程序，他们的广告准化率会相对较高，这个特征基本符合资本市场的行为特征。广告所展示的手机程序这一特征分布特点鲜明，且符合市场规律，有比较大的分析价值，可能对预测模型有较大的贡献，因此将手机程序特征列入建模特征。

至此，我将数据的各个特征总体处理和分析完毕，之后将进行数据的进一步预测处理。

4 模型建立

4.1 特征工程

特征工程是本文的核心部分，而构建特征的主要思路是：该特征是否能够作为区分点击是否是欺诈点击的特征。这里的欺诈者被定义为机器人或是人工反复点击广告而无意下载广告中的应用程序。例如，给定一个特定的 IP 地址，如果该 IP 地址的两次点击间隔时间很短，则表明来自该 IP 地址的点击可能是欺诈性的。基于类似于这样的思路，我在原有特征的基础上构建了如下特征：

原数据集数据特征		
序号	变量名	解释
1	ip	该次点击的IP地址
2	App	用于广告所展示的aap
3	device	点击广告所用手机的设备型号(e.g., iphone 6 plus, iphone 7, huawei mate 7, etc.)
4	os	点击广告的手机所用的操作系统版本
5	channel	移动广告商发布渠道
6	click_time	点击广告的时间点
7	is_attributed	用户是否下载广告内的app

表 4-1 原数据集数据特征

按原数据集构建的数据特征		
序号	**变量名**	**解释**
1	freq_ip	统计每个ip总点击次数之和
2	date	点击广告的时间点
3	month	点击广告的时间点(月份)
4	weekday	点击广告的时间点(周几)
5	day	点击广告的时间点(日期)
6	hour	点击广告的时间点(小时)
7	minute	点击广告的时间点(分钟)
8	second	点击广告的时间点(秒)
9	tm_hour	按小数记的点击广告时间所在的小时
10	ip_app_channel_device_os_next_time_delta	按ip,app,channel,device,os分类后两次点击的时间间隔
11	ip_os_device_next_time_delta	按ip,app,device分类后两次点击的时间间隔
12	ip_os_device_app_next_time_delta	按ip,os,app,device分类后两次点击的时间间隔
13	last_click	是否是该IP的最后一次点击
14	clicks_by_hour	某个小时内该广告被点击的次数
15	clicks_by_os	某种操作系统点击该广告的次数
16	clicks_by_app	某个app的广告被点击的次数
17	clicks_by_device	某种设备点击该广告的次数
18	dl_rate_app	按不同app分类后的下载率
19	dl_rate_os	按不同操作系统分类后的app下载率
20	dl_rate_device	按不同设备分类后的app下载率
21	DL_by_hour	某时内的下载率
22	channel_by_ip	某一IP所点击的发布渠道数量
23	app_by_use	在固定环境下某一设备所点击的app数量
24	app_by_ip	某一IP所点击的app数量
25	ip_app_count	按ip和app分类后的数量
26	ip_app_os_count	按ip、app、os分类后的数量

表 4-2 特征工程表

（1）统计每个 ip 总点击次数之和（freq_ip）

按 IP 地址（ip）对数据进行分类，分类后统计频数，得到该新特征。由于欺诈点击可能来自于某一或者某几个特定的 IP 地址，因此按照 IP 地址点击的次数多少可以为有效区分是否为虚假点击提供信息。

（2）点击广告的时间点（date、month、weekday、day、hour、minute、second）

由于点击广告的时间点这一特征是连续性特征，取值数量多，而本项目的目的从根本上说是解决一个二分类问题，这就要求用于建模的特征需为离散数据特征。故将点击广告的时间点(click_time)特征拆分使用并删除其原本的特征，主要拆分为月(month)、星期几(weekday)、日(day)、小时(hour)、分钟(minute)、秒(second)。

（3）按小数记的点击广告时间所在的小时(tm_hour)

在区分是否为欺诈点击时，我主要考虑了人工或者机器人在短时间或固定时间的密集操作，因此所构建的特征需要体现两次点击的时间间隔以及是否在特定时间内所进行的点击。根据之前划分出的小时和分钟构建了以小数来表示分钟的小时特征。

(4) 按 ip,app,channel,device,os 分类后两次点击的时间间隔(ip_app_channel_device_os_next_time_delta)

计算点击的时间间隔的原因仍然为判断是否在短时间内进行密集点击，若是在短时间内的密集点击，则该点击有可能是欺诈点击。因此分类主要是根据区分度来划分，用于分类的特征越多，区分度越高，类别越多，分类后每一类别的数据越少。

（5）按 ip,app,device 分类后两次点击的时间间隔
(ip_os_device_next_time_delta)

按照不同的区分度依次划分，划分原因和方式同上。

（6）按 ip,app,device 分类后两次点击的时间间隔
(ip_os_device_next_time_delta)

按照不同的区分度依次划分，划分原因和方式同上。

（7）是否是该 IP 地址的最后一次点击(last_click)

通常进行欺诈性点击的 IP 地址不会只使用依次，而是会进行多次重复性点击，因此该次点击若为该 IP 地址的最后依次点击，那该 IP 地址可能不是进行欺诈性点击的 IP 地址。

（8）某个小时内该广告被点击的次数(clicks_by_hour)

通过计算单位小时内某一广告被点击的次数，可以区分出可能容易被欺诈点击的广告，而点击这一广告的人被认为是欺诈点击者的概率也会增加，这将有助于后续对欺诈点击的划分。

（9）某种操作系统点击该广告的次数(clicks_by_os)

通常欺诈性点击来自于同一台或者同一批次的设备，因此通过计算某种操作系统点击该广告的次数，可以区分出是否为欺诈点击。

(10）某个 app 的广告被点击的次数(clicks_by_app)

一些互联网公司为了获取关注度，可能会尝试进行恶意的欺诈性点击来刷取点击量，从而获得更多的曝光，这一类 app 的点击量往往会较高。因此通过计算某个 app 的广告被点击的次数可以帮助区分是否为欺诈点击

（11）某种设备点击该广告的次数(clicks_by_os)

通常欺诈性点击来自于同一台或者同一批次的设备，因此通过计算某种设备点击该广告的次数，可以区分出是否为欺诈点击。

（12）按不同 app 分类后的下载率(dl_rate_app)

进行欺诈性点击的广告往往只刷点击量但不真正下载广告内的 app，因此通过计算不同 app 的下载率可以有效区分出是否为欺诈点击，通常欺诈点击的 app 的下载率会较低。

（13）按不同操作系统分类后的 app 下载率(dl_rate_os)

通常欺诈性点击来自于同一台或者同一批次的设备, 进行欺诈性点击的设备往往不会下载广告内的 app，因此通过计算不同操作系统的下载率可以有效区分出是否为欺诈点击，通常欺诈点击的手机操作系统的下载率会较低。

（14）按不同设备分类后的 app 下载率(dl_rate_device)

通常欺诈性点击来自于同一台或者同一批次的设备, 进行欺诈性点击的设备往往不会下载广告内的 app，因此通过计算不同设备的下载率可以有效区分出是否为欺诈点击，通常欺诈点击的移动通信设备的下载率会较低。

（15）某时内的下载率(DL_by_hour)

通常欺诈性点击是在短时间内的集中操作，因此单位时间内的广告点击次数越多，其欺诈点击的可能性越高。

（16）某一 IP 所点击的发布渠道数量(channel_by_ip)

通常同一 app 广告的发布渠道众多，因此欺诈点击的 IP 地址可能会点击不同发布渠道的同一 app，通过统计某一 IP 所点击的发布渠道数量可以区分出是否为欺诈点击。

（17）在固定环境下某一设备所点击的 app 数量(app_by_use)

通常专门进行欺诈性点击的组织或个人来自于同一台或者同一批次的设备、同一地点和网络环境，同时这一欺诈点击组织或个人可能会点击多个 app，因此统计在固定环境下某一设备所点击的 app 数量能区分出是否为欺诈点击行为。

（18）某一 IP 所点击的 app 数量(app_by_ip)

通常专门进行欺诈性点击的组织或个人来自于同一 IP 地址，同时这一欺诈点击组织或个人可能会点击多个 app，因此通过统计某一 IP 所点击的 app 数量能区分出是否为欺诈点击行为。

（19）按 ip 和 app 分类后的数量(ip_app_count)

专门进行欺诈点击的组织或个人通常来自同一 IP 地址，而该来自 IP 地址的设备多次点击同一 app 则可以被认为该 IP 地址有欺诈点击的嫌疑。

（20）按 ip、app、os 分类后的数量(ip_app_os_count)

专门进行欺诈点击的组织或个人通常来自同一 IP 地址和使用同一操作系统，而该来自 IP 地址的设备多次点击同一 app 则可以被认为该 IP 地址有欺诈点击的嫌疑。

4.2 不平衡数据集处理

由于点击欺诈发生的概率较低，其样本只占总样本的极小部分，因此该问题属于不平衡分类问题，要处理该问题，我首先要对不平衡数据集进行处理。

标签	训练集样本数量	占比
0	69841	99.77%
1	159	0.23%

表 4-3 不平衡数据集处理前的样本

SMOTE（Synthetic Minority Oversampling Technique）算法，全称为合成少数类过采样技术．它是基于随机过采样算法的一种改进方案，由于随机过采样采取

简单复制样本的策略来增加少数类样本，这样容易产生模型过拟合的问题，即使得模型学习到的信息过于特别(Specific)而不够泛化(General)，SMOTE 算法的基本思想是对少数类样本进行分析并根据少数类样本人工合成新样本添加到数据集中，具体如下图所示，算法流程如下。

Algorithm *SMOTE*(T, N, k)
Input: Number of minority class samples T; Amount of SMOTE N%; Number of nearest neighbors k
Output: $(N/100) * T$ synthetic minority class samples
1. (* *If N is less than 100%, randomize the minority class samples as only a random percent of them will be SMOTEd.* *)
2. **if** $N < 100$
3. **then** Randomize the T minority class samples
4. $T = (N/100) * T$
5. $N = 100$
6. **endif**
7. $N = (int)(N/100)$ (* *The amount of SMOTE is assumed to be in integral multiples of 100.* *)
8. k = Number of nearest neighbors
9. $numattrs$ = Number of attributes
10. $Sample[\,][\,]$: array for original minority class samples
11. $newindex$: keeps a count of number of synthetic samples generated, initialized to 0
12. $Synthetic[\,][\,]$: array for synthetic samples
(* *Compute k nearest neighbors for each minority class sample only.* *)
13. **for** $i \leftarrow 1$ **to** T
14. Compute k nearest neighbors for i, and save the indices in the $nnarray$
15. Populate(N, i, $nnarray$)
16. **endfor**

Populate(N, i, $nnarray$) (* *Function to generate the synthetic samples.* *)
17. **while** $N \neq 0$
18. Choose a random number between 1 and k, call it nn. This step chooses one of the k nearest neighbors of i.
19. **for** $attr \leftarrow 1$ **to** $numattrs$
20. Compute: $dif = Sample[nnarray[nn]][attr] - Sample[i][attr]$
21. Compute: gap = random number between 0 and 1
22. $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$
23. **endfor**
24. $newindex$++
25. $N = N - 1$
26. **endwhile**
27. **return** (* *End of Populate.* *)
End of Pseudo-Code.

图 4-1 SMOTE 算法

在实际应用本算法的过程中，我主要平衡的是训练集样本数量。这主要是因为基于 k-最近邻理论的 SMOTE 算法会生成大量高度趋同的新样本，由于这个原因，在对给定数据集应用 SMOTE 后进行拆分会导致信息从测试集泄漏到训练集，从而导致分类器或机器学习模型高估其准确性和其他性能指标。盲目将 SMOTE

使用到整个数据集中会使得测试集失去其意义。基于这个原则，我在训练集中使用 SMOTE 后得到的样本数量如下：

标签	训练集样本数量	占比
0	69841	50%
1	69841	50%

表 4-4 不平衡数据集处理后的样本

4.3 模型选择

首先对测试集与训练集中的数据分别进行 Z-score 标准化,然后调用 SelectKBest 包对现有的 14 个特征进行选择。

Select K Best 为分类提供了三种评价特征的方式：1.卡方检验，计算非负特征和类之间的卡方统计；2.样本方差 F 值；3.离散类别交互信息。我将目标特征的个数设为 10，得到的所选择的特征如下所示：

"clicks_by_os","clicks_by_app","clicks_by_device","dl_rate_app",
"dl_rate_os","dl_rate_device","DL_by_hour","channel_by_ip",
"app_by_use","app_by_ip"

4.3.1 XGBoost 算法

实例化一个默认参数下的 xgboost 模型，导出其文本报告。在报告中显示每个类别的精确度，召回率，F1 值等信息。结果如下所示：

	precision	recall	f1-score	support
0	1.00	1.00	1.00	69841
1	1.00	1.00	1.00	69841
accuracy			1.00	139682
macro avg	1.00	1.00	1.00	139682
weighted avg	1.00	1.00	1.00	139682

图 4-2 默认参数下 XGBoost 模型对于训练集的分类指标的文本报告

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29932
1	0.38	0.51	0.44	68
accuracy			1.00	30000
macro avg	0.69	0.76	0.72	30000
weighted avg	1.00	1.00	1.00	30000

图 4-3 默认参数下 XGBoost 模型对于测试集的分类指标的文本报告

通过比较可以发现对于测试集该模型识别诈骗点击的精确度为 0.38，召回率为 0.51，而在对训练集的结果中表现良好。说明目前该模型存在过拟合的情况。

4.3.2 LightGBM 算法

实例化一个默认参数下的 LightGBM 模型，导出其文本报告。在报告中显示每个类别的精确度，召回率，F1 值等信息。结果如下所示：

	precision	recall	f1-score	support
0	1.00	1.00	1.00	69841
1	1.00	1.00	1.00	69841
accuracy			1.00	139682
macro avg	1.00	1.00	1.00	139682
weighted avg	1.00	1.00	1.00	139682

图 4-4 默认参数下 LightGBM 模型对于训练集的分类指标的文本报告

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29932
1	0.29	0.59	0.39	68
accuracy			1.00	30000
macro avg	0.65	0.79	0.70	30000
weighted avg	1.00	1.00	1.00	30000

图 4-5 默认参数下 LightGBM 模型对于测试集的分类指标的文本报告

通过与 XGBoost 模型进行对比，我可以发现其召回率由 0.51 上升为 0.59，而精确率有所下降。精确率高意味着虚警少,能保证机器检测为阳性时,事件真正发生的概率高,但不能保证机器检测为阴性时,事件不发生。相反,召回率高意味着漏报少,能保证机器检测为阴性时,事件不发生的概率高,但不能保证机器检测结果为阳性时,事件就一定发生。因此也可以说,精确率衡量机器做出阳性判断的准确度,召回率衡量机器做出阴性判断的准确度。由于本问题中，我的研究目的在于尽可能高的识别出广告欺诈行为，因此我更多地关注的指标为召回率，因此我接下来将在 LightGBM 的模型下进行进一步调优。

4.4 模型优化

在模型优化过程中，我分别以 recall 以及 roc_auc 两个指标为目标利用网格

搜索法通过遍历一定范围内的参数取值进行参数调优。

当我以 recall 为目标进行调参时，分类指标的文本报告如下所示：

	precision	recall	f1-score	support
0	1.00	0.99	1.00	29932
1	0.16	0.74	0.26	68
accuracy			0.99	30000
macro avg	0.58	0.86	0.63	30000
weighted avg	1.00	0.99	0.99	30000

图 4-6 以 recall 为目标调参后 LightGBM 模型的分类指标的文本报告

可见召回率从 0.59 上升为 0.74，有了明显的提升。但准确率以及 F1-score 的结果有所下降。

当我以 roc-auc 为目标进行调参时，分类指标的文本报告如下所示：

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29932
1	0.23	0.62	0.34	68
accuracy			0.99	30000
macro avg	0.62	0.81	0.67	30000
weighted avg	1.00	0.99	1.00	30000

图 4-7 以 roc-auc 为目标调参后 LightGBM 模型的分类指标的文本报告

可见召回率从 0.59 上升为 0.62，上升的幅度较小，并且准确率以及 F1-score 的提升也并不明显，因此我最终选择以召回率为目标进行调参。

4.5 特征重要度直方图

特征重要性是一种为预测模型的输入特征评分的方法，该方法揭示了进行预测时每个特征的相对重要性。可以为回归问题和分类问题上计算特征重要性得分。

4.5.1 调参前

通过 plot_importance 函数做出特征重要性直方图，具体情况如下图所示。从图中可以看出，’某时内的下载率’‘ 不同 app 的下载率’ ‘某个 app 的广告被点击的次数’以及‘不同操作系统的 app 下载率’对模型贡献最大，这也和实际相符合，当某小时的下载率、某个 app 广告点击数超出一定值，越有可能发生点击欺诈行为。同时，点击欺诈更多地集中在某几个特定的 app 和操作系统。在我的描述性统计中也可以看出，大部分数据来源于特定几个 app 和操作系统。

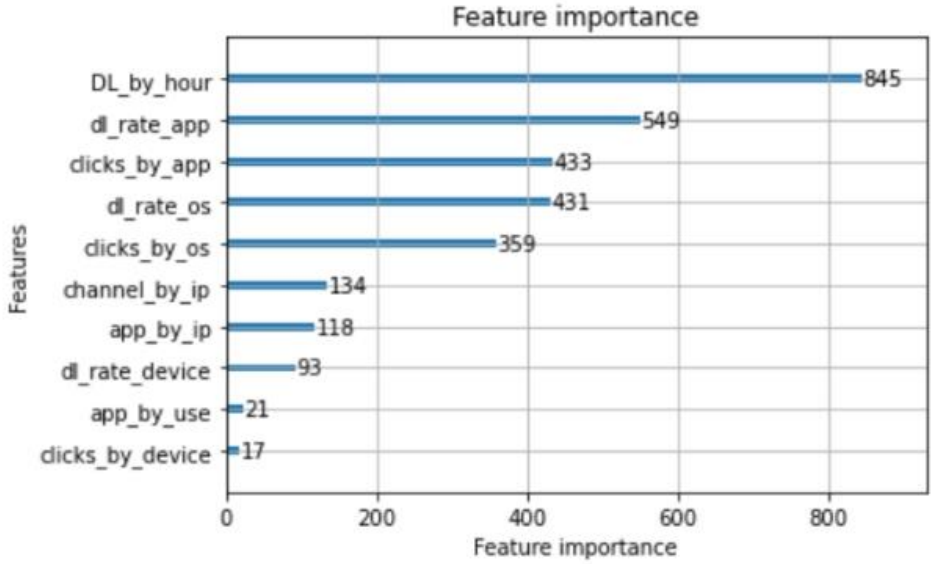

图 4-8 特征重要度直方图——调参前

4.4.2 调参后

可以看出，调参前后对模型贡献大的几个特征值基本没有改变，这也反映出了模型的稳定性以及分析的可靠性。

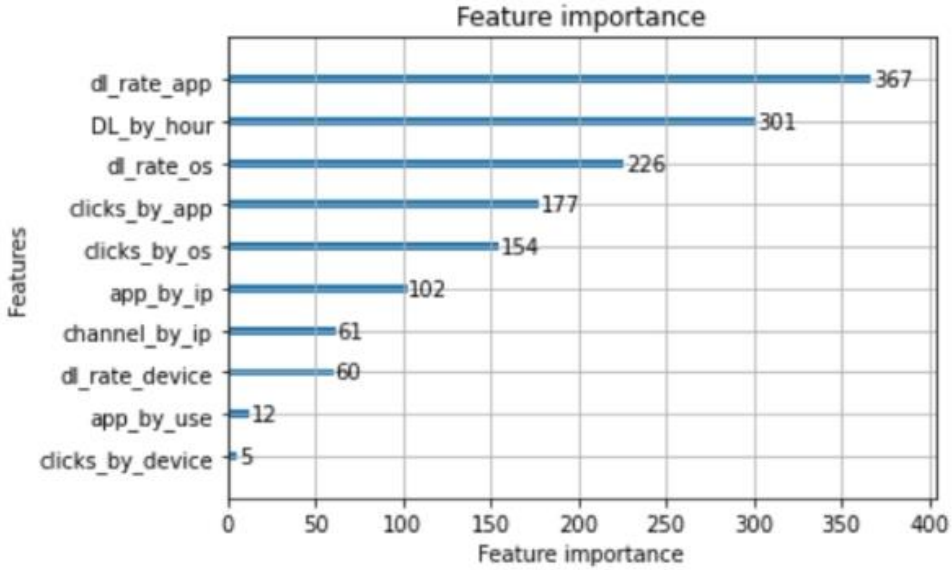

图 4-9 特征重要度直方图——调参后

4.6 ROC 曲线

4.6.1 调参前

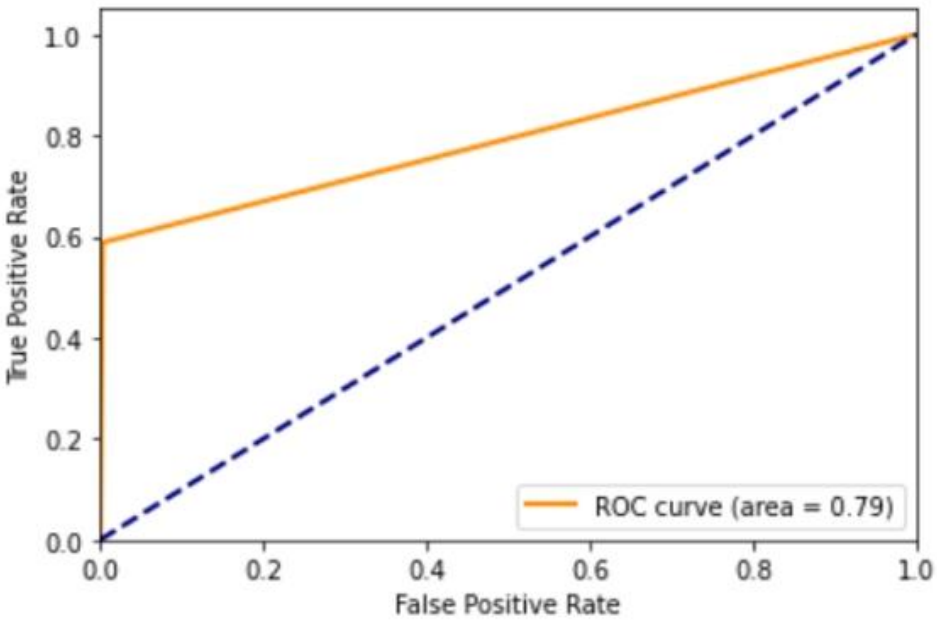

图 4-10 ROC 曲线——调参前

建立使用默认参数下的 lightgbm 预测模型，本文选择以 ROC 曲线，以及曲线下面积指数 AUC 为评价标准，根据建立预测模型的结果，绘制模型的 ROC 曲线。

在图中，对角线的实际含义是:随机判断响应与不响应，正负样本覆盖率应该都是 50%,表示随机效果。ROC 曲线越陡越好，所以理想值就是 1,一个正方形，而最差的随机判断都有 0.5,所以一般 AUC 的值是介于 0.5 到 1 之间的。可以看出此时 AUC=0.79，预测效果中等。

4.6.2 调参后

由下图明显看出，调参后的 ROC 曲线表现较好，AUC 指数也从 0.79 上升到了 0.86，这也说明我的预测精度达到了更好的水平。

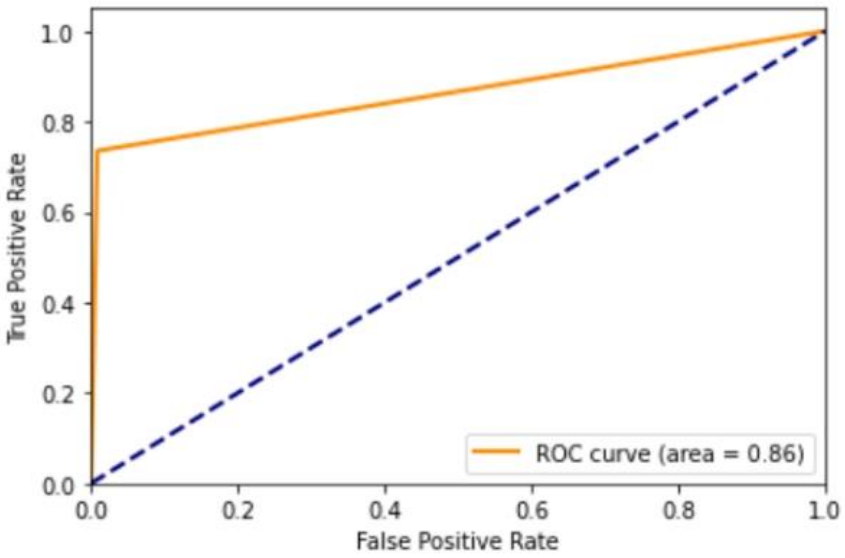

图 4-11 ROC 曲线——调参后

5 结论

在全球实体经济由于疫情的原因而低迷的情况下网络经济由于其特性仍然可以欣欣向荣。所以广告主们对于网络广告的信赖程度依然很高。但是点击欺诈却一直被认为是网络广告的噩梦。

点击欺诈，是指对网络广告进行恶意反复点击的活动。点击欺诈大致出于两个目的：一是为了打击竞争对手，一些企业使用点击欺诈，以增加竞争对手的广告成本；二是不排除网络广告公司利用软件反复点击自家的广告产品的可能性，从而为自己带来更多网络广告收入。“点击欺诈背后主要隐藏有两种人，一种是只要点击广告就能赚钱的人；一种是恨那些广告主的人，即广告主的竞争对手，为了增加广告主的成本压力，不断点击广告主的广告。第一种人包括，给客户做广告的人、搜索引擎公司本身，还有搜索引擎广告代理公司、搜索引擎的联盟网站等。第二种人则包括广告主的竞争对手以及竞争对手所雇佣的水军。

所以，点击欺诈的目的，说到底是为了以不正当的方式，直接或者间接地获得利益。使用软件、使用脚本抑或是雇佣水军来进行无效形式的点击，成本十分低廉，但却可以给自己带来高额的收益，或者使得竞争对手花费更多的成本，是一种有效但不正当的竞争方式。某些企业会使用这样的方式，来增大自己的市场份额或者收入。

而点击欺诈之所以能够成功，是因为传统的判断“有效点击”的判断依据只是

“是否点击”，也就是我报告分析当中的 is_attributed 为 1，则视为有效点击。

而点击欺诈的预测，则是以多个特征数据作为判断依据，来划分有效点击亦或是点击欺诈，这样则可以剔除点击欺诈的点击，从而判断真实有效的点击的次数以及比例。

对点击欺诈进行预测，最大的受益方就是网络广告的广告主。预测出点击欺诈并以此计算出真实的点击量之后，广告主可以对投放广告的准确情况进行统计以及分析，更好地调整广告投放的各项指标，并且也能够减少广告投放的成本。据统计，网络广告中约有 16%~30%的点击欺诈，那么点击欺诈预测最少可以帮广告主减少每次广告投放 16%的成本，甚至更多。

其次，点击欺诈的预测还可以有效减少点击欺诈的发生。当点击欺诈的预测成熟之后，点击欺诈者意识到点击欺诈并没有办法给自己带来潜在的利益，那么就会放弃或者减少点击欺诈，以降低不必要的成本。减少点击欺诈的发生频率还可以有效地营造出一个公平的网络竞争环境。

不仅如此，点击欺诈的预测对于网络广告的其他方面，比如数据处理、反馈等问题都有着巨大的作用，甚至可以从点击欺诈来延申到其他方面，比如购买欺诈、活跃度欺诈等等，有着深远的研究意义。