Kaggle 机器人识别项目研究报告

杨秋语

2017 年 6 月 5 日

目录

一、问题定义

（一）项目概述

Facebook 于 2015 年 4 月在 Kaggle 上举行了 human or robot 拍卖网站机器人识别大赛。从该竞赛的描述可以看出，竞拍数据来源于一个在线竞拍网站，该网站为客户提供各项商品的竞拍服务。但是，由于网站用户中存在一定数量的由机器人操纵的账户，这些机器人在竞拍中的行为使得其他用户难以竞拍到自己想要的商品，继而导致了越来越多的核心用户流失。虽然该网站曾建立过一些简单模型用于鉴别机器人用户，但效果始终不佳。因此，该网站需要寻求一种更为有效的方法对机器人用户进行驱逐，从而保障拍卖公平，维护核心用户利益。

该项目提供了四个 csv 格式数据集，分别是 sampleSubmission.csv, bids.csv, test.csv, train.csv。其中，sampleSubmission.csv 为提交结果样例，test.csv 和 train.csv 为以用户 ID 开头的测试集与训练集，bids.csv 为一段时间内的拍卖流水数据。从数据构成来看，拍卖流水数据总共包含了约 760 万条流水数据，涉及用户 ID 数约六千个，其中，训练集 ID 两千个，测试集 ID 四千个。从数据结构来看，bids.csv 包含单次拍卖 ID，用户 ID，拍卖场次名称，商品种类，设备编号，拍卖时间戳，所属国，ip 地址，url 地址 9 项数据。train.csv 包含用户 ID，支付账户 ID，地址和是否为机器人用户的标注结果 4 项数据，test.csv 除未标注结果外，其余数据结构与 train.csv 一致。

从提交结果的要求来看，该项目要求参与者构建模型对测试集中各个用户是机器人用户的概率进行预测，从而达到筛选并驱逐机器人用户，保护其他用户竞拍权益的目的。

（二）问题陈述

从项目概述可以看出，这是一个尽可能精准的识别出机器人用户的问题。一方面，我们需要识别出机器人用户并进行驱逐，但另一方面，我们也不希望会误伤使用该网站的正常用户。从统计学上讲，即在保证真阳性比率（TPR）的同时，伪阳性比率（FPR）要尽可能小。

结合问题描述与业务常识，该网站中的机器人用户的目的应该有两种，作为竞拍网站中的买方，可以利用机器人随时监控拍卖进程，并在拍卖结束前进行抢拍来提高自己拍得物品的概率。作为竞拍网站中的卖方，可以利用机器人恶意哄抬竞拍商品物价，以达到尽可能抬高买方出价的目的。

从数据集来看，利用数据集分析挖掘拍卖交易流水并从中识别出机器人用户是本项目的关键。从题目所给予的信息中，我们可以推断出机器人可能拥有以下特征：

- ✧ 机器人用户可能会参与多场竞拍，涉及多个商品种类
- ✧ 机器人用户竞拍次数多，频率高
- ✧ 机器人用户可能会在多台移动设备间切换登陆
- ✧ 机器人用户可能会使用多个 ip 地址，多个 url 地址
- ✧ 机器人用户可能在使用不同 ip 时会显示不同国家
- ✧ 机器人用户在他人竞拍后可能会有相当迅速的竞拍反应
- ✧ 机器人用户在竞拍即将结束时会竞拍的更加频繁

由以上特征推断可以想到，本题中机器人竞拍的行为模式会具有一定程度的相似性，因此，应该使用 K-最近邻、决策树类算法或集成学习方法对用户进行分类，以求得测试集分类概率，为进一步判定机器人用户做准备。

（三）评价指标

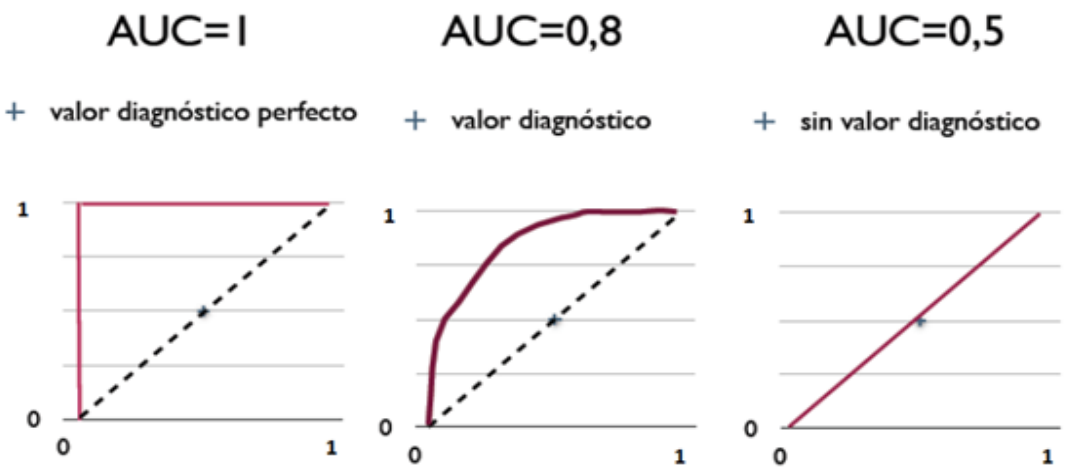
由于本项目需要同时顾及真阳性与伪阳性的特性，作者决定采用 ROC（Receiver operating characteristic）曲线作为该项目的评价指标。ROC 曲线是一种二元分析模型，在该模型中，通过设定不同的阈值并计算出模型预测的真阳性比率（TPR）和伪阳性比率（FPR）来绘制 ROC 曲线。

在统计学中，二元分析模型通常用四个指标衡量模型的准确性，分别是真阳性（TP），伪阳性（FP），真阴性（TN）和伪阴性（FN）。理想状态下，一个预测模型可以做到完全区分真阳性与真阴性，即 FP 与 FN 均为零。但现实中往往无法达到理想模型的预测准度，因此，在进行模型预测时往往需要判定预测错误的比率，即对 FP 与 FN 进行测度来评估模型。本题中，TPR 可以衡量二类错误的大小，FPR 可以衡量一类错误的大小，故结合采用 TPR 与 FPR 对模型性能进行衡量，公式如下：

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

结合 TPR 与 FPR 可以绘制 ROC 曲线，并利用计分方式相应计算对应的曲线下面积（AUC），如下图所示：



[1]Receiver operating characteristic, Wikipedia

由上图可知，理想模型 AUC = 1，表明在 TPR 逐渐提升到 1 的过程中，FPR 始终保持为 0。随机猜测模型 AUC = 0.5，表明在 TPR 提升的过程中，FPR 成线性比例随 TPR 提升，其模型准确度与随机猜测无异。预测模型 AUC 数值一般会处于二者之间，数值越大，代表 TPR 上升过程中 FPR 上升越小，模型预测效果越好。由以上特性可知，AUC 的数值可以用于衡量需兼顾 FN 与 FP 的模型，适用于本项目。

[1] https://en.wikipedia.org/wiki/Receiver_operating_characteristic#cite_note-matlab-1

二、问题分析

（一）数据初步探索

由上文可知，该项目需要分析的数据集有三个，分别是 train.csv, test.csv 和 bids.csv。

1. train.csv 与 test.csv

由于 train.csv 与 test.csv 结构相似，故将两个数据集放在一起分析。二者均记录有用户 ID，支付账户 ID 和地址三项数据，以 ID 唯一进行统计，结果如下：

训练集	用户 ID	支付账户 ID	地址
数量（个）	2013	2013	2013
测试集	用户 ID	支付账户 ID	地址
数量（个）	4700	4700	4700

可以看出，这两个数据集中每个用户 ID 均有不同的支付账户 ID 与地址，没有用户共用支付账户 ID 或地址，因此没有进一步进行数据挖掘分析的必要。

结合 bids.csv 的拍卖流水来看，拍卖流水中共有 1984 个属于训练集的账户，4630 个属于测试集的账户。计算可得，训练集中有 29 个账户没有拍卖数据，测试集中有 70 个账户没有拍卖数据。根据 Kaggle 中更新，测试集中 70 个无数据账户将在评分时移除，其数值不影响最后得分。因此在生成最后结果时可将这 70 个对应账户赋值为零。

对训练集进行账户结果分类后可得训练集的标注结果：

训练集	机器人	普通用户
数量（个）	103	1910

可以看到，机器人用户占训练集总用户数的 5%左右。

2. bids.csv

bids.csv 是拍卖网站一段时间内的拍卖流水，记录了约 760 万条拍卖数据，是本项目的重点数据挖掘对象。根据题目，各项数据含义如下：

bid_id（拍卖 ID）——每次竞拍所产生的唯一标识 ID

bidder_id（用户 ID）——每个参与竞拍的用户 ID

auction（拍卖场次）——每场拍卖的标识名

merchandise（商品种类）——每场拍卖的商品种类标识，值得注意的是，该种类标识为用户搜索进入拍卖网站的首个种类标识，与拍卖场次并非一一对应的关系

device（移动设备数）——用户所使用的移动设备标识

country（国家）——用户竞拍时所显示的国家

ip（IP 地址）——用户竞拍所显示的 IP 地址

url（URL 网址）——用户参与竞拍所使用的网址

对流水数据进行清洗后，各项数据数量如下：

流水	拍卖 ID	用户 ID	拍卖场次	商品种类	设备数
数量（个）	7656334	6614	15051	10	7351

	时间戳	国家	IP	URL	
数量（个）	776529	199	2303991	1786351	

值得注意的是，数据集中国家一项存在数据缺失现象，接下来对缺失数据进行进一步分析。

在结合 train.csv 与 test.csv 后，数据集可被划分为测试集，训练集机器人集，训练集普通用户集。对数据集进行筛查后可得，有 10 个机器人 ID，168 个普通用户 ID 和 398 个测试集 ID 存在国家缺失的情况。进一步进行抽样发现，有 9 个用户 ID 国家缺失的竞拍数超过其竞拍总数的一半，其中 2 个 ID 位于普通用户集，7 个位于测试集，其中，有 5 个 ID 仅竞拍了一次，1 个位于普通用户集，4 个位于测试集。进一步对国家缺失占比较小的用户抽样，发现国家缺失所对应的 IP 和 URL 等信息并不能进一步取得填补数据的充分理由。因此在以上调查的基础上，决定用 unknown 字段统一填补国家数据并纳入统计。在此基础上得到统计结果如下：

	人类用户				机器人用户			
	mean	std	max	min	mean	std	max	min
bid_id	1413.51	14597.54	515033	1	4004.04	16370.10	161935	1
auction	58.07	142.93	1623	1	145.04	195.10	1018	1
merch	1.00	0.02	2	1	1	0	1	1
device	73.95	184.56	2618	1	163.61	222.81	1144	1
time	1175.80	9312.84	283463	1	3670.58	13971.63	136980	1
country	12.68	22.87	164	1	26.48	31.16	179	1
ip	581.26	4140.68	109159	1	2387.80	11269.67	111918	1
url	335.19	2735.53	81376	1	544.58	1163.91	8551	1

从数据集来看，首先，人类用户与机器人用户各项数据最小值均为 1。从拍卖次数均值来看，人类用户要低于机器人用户，但结合标准差与最大值来看，人类用户中存在一定数量的用户频繁参与拍卖，其拍卖次数甚至远远高于机器人的竞拍次数。

商品种类上，少量人类用户参与了两个种类的竞拍，其余人类用户与所有机器人用户均只参与了一个种类的竞拍。

对于拍卖场次，使用设备数，时间戳，国家数，IP 数和 URL 数上，均值均反映出人类用户各项数值低于机器人用户的现象，但另一方面，除 IP 一项外各项均反映出了相似的标准差。从最大值来看，人类用户甚至在许多项，例如唯一时间戳数，设备数和 URL 数上超越了机器人用户对应的最大值。

对测试集采用同样的统计方法后发现，其数据表现出了与两个训练集相似的值，另一方面，由于测试集中也存在拍卖次数特别多，使用大量 IP 与 URL 的用户，因此，考虑保留训练集中人类用户集中的异常值进行训练以提供必要信息。

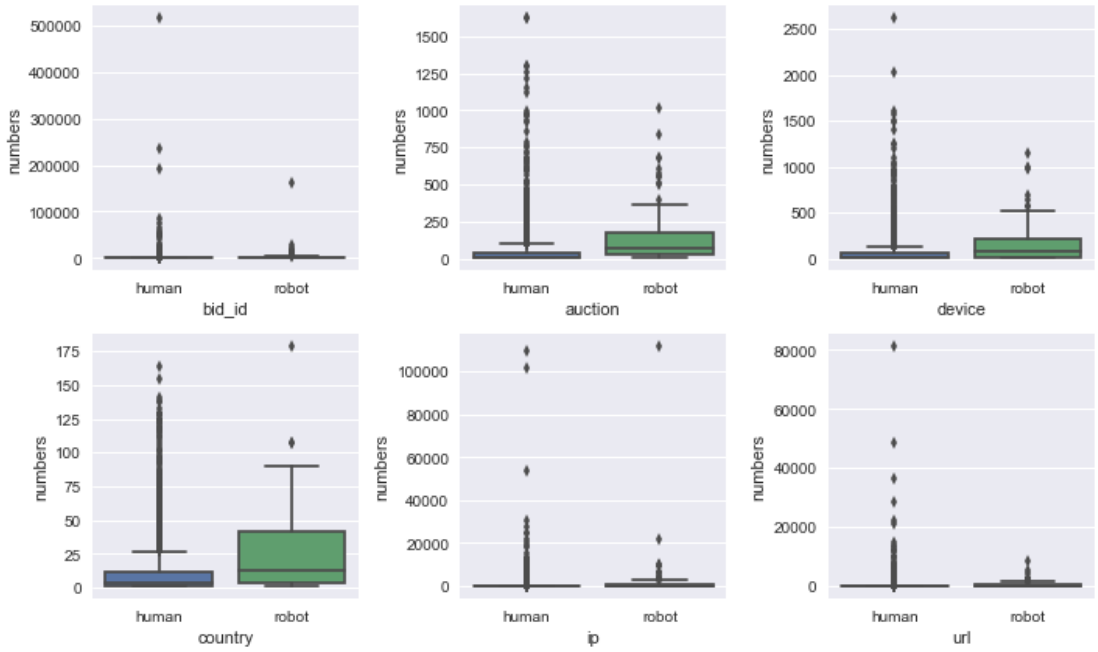
（二）数据可视化与进一步挖掘

为了更直观地体现数据集中各项数据的情况，下面将对 bids.csv 中各项进行进一步可视化分析，以训练集为主，必要时会绘制测试集图表进行对比。

1. 单项数据分析

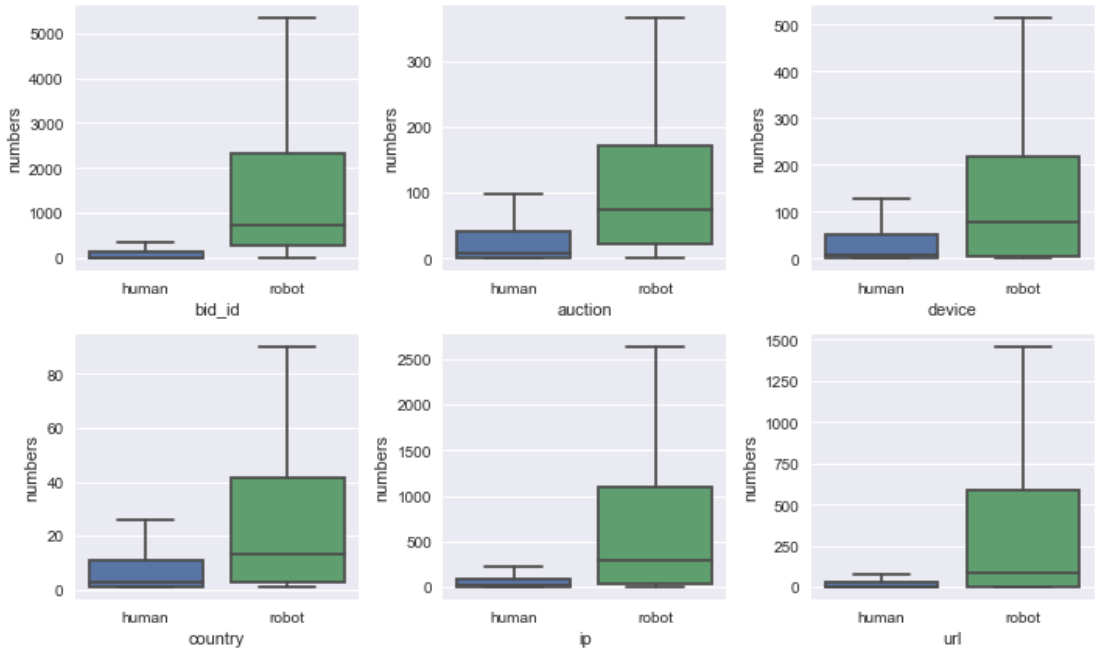
为了进一步体现人类用户与机器人用户间的差异，该部分针对 bids.csv 中

bid_id, auction, device, country, ip 和 url 各项进行箱形图绘图分析。在不对异常值进行处理的情况下，各项数据如下图所示



由图可见，虽然各项数据在统计中都出现了大量的异常值，但 auction, device 和 country 三项依然可以体现出，人类用户中至少有 75%的用户其所参与的拍卖数，使用设备数和国家数在机器人用户的均值以下。

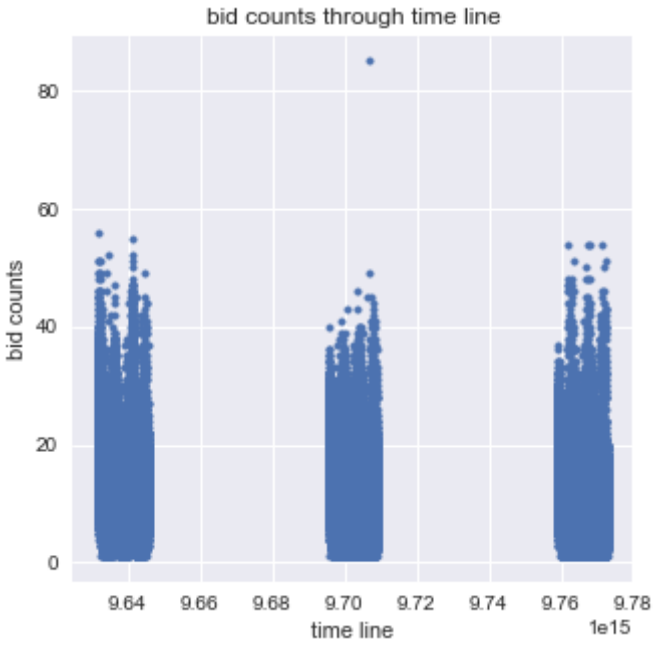
在去除异常值之后，我们可以得到下图：



由该图显示结果可得，75%以上的人类用户各项数值均在机器人用户的平均值以下，由此可推断出，简单的数据挖掘即可以将七成以上的人类用户与机器人用户区分开来，但如要获得更高的预测精度，就需要对各方异常值的行为进行更深入的挖掘以达到区分用户的目的。

2. 时间轴分析

为了进一步挖掘流水数据，下面将依照时间轴对流水数据进行分析。为了对整个拍卖有一个直观感受，首先我们基于时间轴与各时间点拍卖次数进行作图如下：

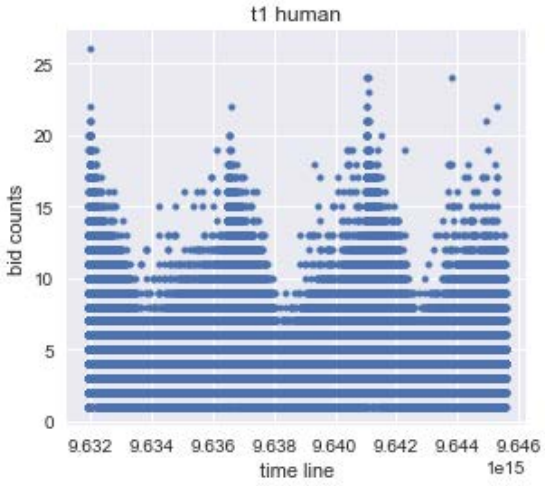


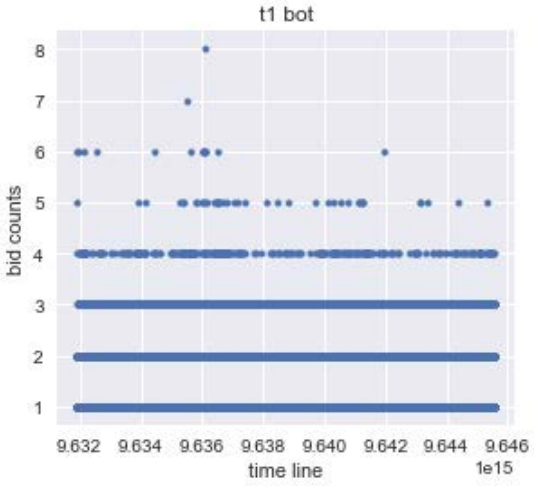
由此可见，该流水单并不是连续的，而是由三个不连续的时间块组成。由于该项目中对时间戳进行了混淆处理以保护隐私，因此我们没有办法将时间直接转换回正常的时间格式进行处理。但混淆的时间轴依然保留了其时间戳应有的顺序，因此可以直接进行运算。通过对时间间隔的计算发现，在时间间隔中有一个值相比于其他值有数量级的差异，其值为 50021105263158。通过进一步对时间轴进行处理，可以发现三个时间块的起点与终点分别为：

	时间块 1	时间块 2	时间块 3
起点	9631916842105263	9695580000000000	9759243157894736
终点	9645558894736842	9709222052631578	9772885210526315

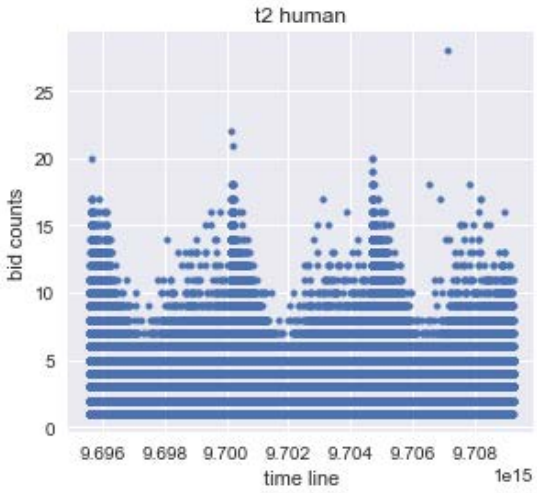
接下来进一步对各时间块进行分析，在进行时间块划分后，我们采取了人类用户与机器人用户分离的方式对各时间块内各个时点的拍卖次数进行作图，各时间块结果如下：

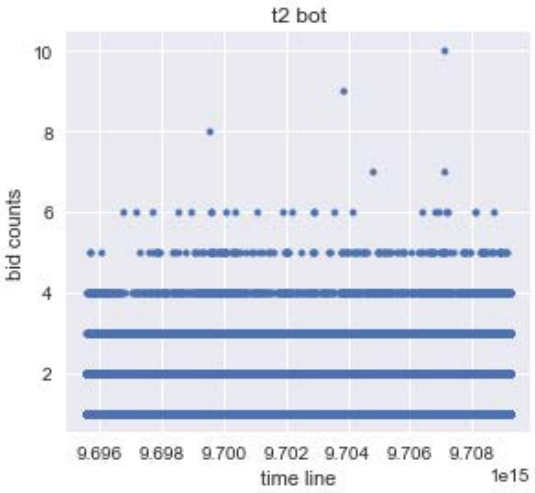
时间块 1：



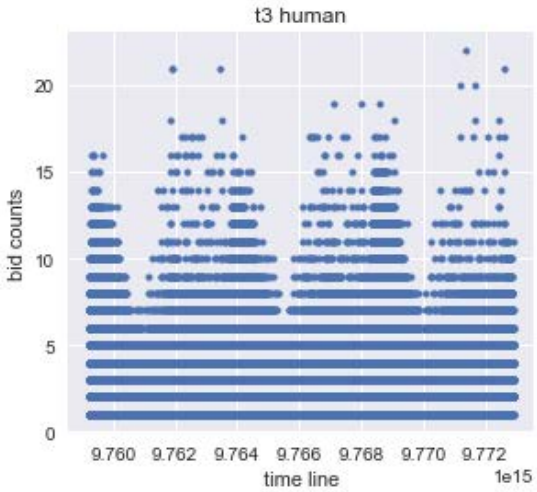


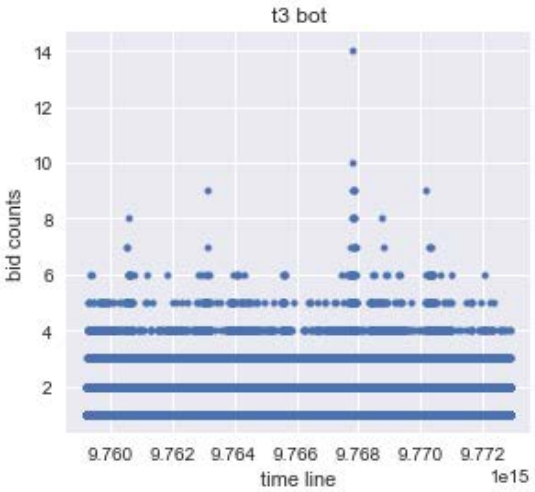
时间块 2：





时间块 3：





从各时间块的拍卖次数统计来看，人类用户的拍卖次数具有一定的周期性，推断是每天昼夜交替，人类用户拥有一定的作息时间安排所致。反观机器人用户，虽然在时间推进的过程中也存在一定的周期性，但其规律不明显，绝大部分时间点机器人用户的竞拍次数在 5 次以下。偶尔的竞拍次数上涨推断是由于竞拍即将

结束时，机器人用户为了赢得竞拍而采取的相对激进的竞价行为所致。由此可以推断，通过对不同用户的竞拍时间段进行拆分统计，并进一步进行拍卖时间间隔分析，也许可以获得一部分用于区分机器人用户的有用信息。

（三）算法简述

通过对数据进行分析和机器人用户的行为模式推断，作者决定采用 KNN（K-最近邻）模型和集成学习模型作为该项目的预测模型。具体来讲，考虑到模型本身的特点与可解释度，将选用 KNN 模型作为预测的基础模型，并选用 random forest（随机森林）作为 bagging 方法的代表模型，gradient boosting 作为 boosting 方法的代表模型对数据进行学习与预测。下面对各模型进行简要介绍。

1. KNN 模型

KNN 模型最早源于 nearest neighbor rule，而 nearest neighbor rule 是在科学家利用贝叶斯理论对数据进行分类时形成的一套理论体系。Nearest neighber 理论指出，在一个可度量的空间内存在 n 个样本$\{x_1, x_2, x_3 \ldots x_n\}$，每个样本均存在一个$\{\Theta_1, \Theta_2, \Theta_3 \ldots \Theta_n\}$作为索引的特征集，特征集中拥有{1, 2, 3…M}个特征量。当这样的样本集存在时，如果在该特征集上引入一个新特征对（x, Θ）与度量单位 d，则新特征对的分类取决于：

$$min\mathrm{d}(x_i, x) = d(x'_n, x) \quad \mathrm{i} = 1, 2, 3 \ldots \mathrm{n}$$

也就是与样本 x 最临近的样本，与另外 n-1 个样本无关。 (COVER & HERT, 1967)后人在此基础上将 nearest neighber 模型拓展到 K 维，并确定了相应的误差计算方法，由此形成 KNN 模型。

KNN 算法是一种基于实例（instance-based）的惰性学习（lazy learning）方法。其算法特点是不用进行事先学习来做出预测，而是依照现有样本分布对放入样本集的新实例进行实时预测。一般来讲，对于一个新放入样本集的，拥有$\{a_1, a_2, a_3 \ldots a_n\}$个特征的样本 x，对于已确定的拥有$\{b_1, b_2, b_3 \ldots b_n\}$特征的样本 y，我们会采用两种方法计算两个样本间的距离：

欧氏距离法：

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

曼哈顿距离法：

$$\mathrm{d}(\mathrm{x}, \mathrm{y}) = \sqrt{\sum_{i=1}^{n}|a_i - b_i|}$$

计算距离后通过预先设定的 K 值进行加权评估，从而得到新样本的分类结果。

2. 随机森林模型

随机森林（Random Forest）模型是集成学习中的一种模型，由 Leo Breiman 与 Adele Cutler 在前人多决策树并行分类实证研究的基础上构建而成。随机森林

模型是集成学习方法中 Bagging（Bootstrap Aggregating）代表方法。

Bagging 方法的基本原理为对样本数据集采用有放回的方式进行抽样，进而通过在不同抽样样本间使用弱学习器进行学习，最终将弱学习器的学习结果进行结合，以达到学习目的。具体到随机森林算法上来看，随机森林算法会在训练集上进行 N 次数据子集划分，每一次会在每个数据子集上生成 k 个独立同分布的随机向量用于弱学习器训练，在弱学习器训练完成后，弱学习器的投票结果会结合产生一个拥有特定偏好类别的强学习器，最终由强学习器进行分类或回归完成运算。 (Breiman, 2001)

作为 Bagging 算法的代表，随机森林算法继承了 Bagging 类算法的传统优点，如离群值，噪声对模型干扰能力较弱，可通过袋外样本验证模型的泛化性等。另一方面，由于随机森林实现了 Bagging 算法中弱学习器的并行处理，因此其运行速度要远远快于 Boosting 类算法。

从统计学上看，随机森林算法通过降低弱学习器随机波动所产生的方差来达到正确学习的目的。因此对于普适性的特性学习效果较好，但对出现频率较低的特定实例分类效果较差。另一方面，随机森林算法依赖于弱学习器分类的稳定性，如果弱学习器在学习过程中均出现了相似的偏差，那么这种偏差会反映在强学习器，继而反映在最终学习结果中。

3. Gradient Boosting 模型

Gradient Boosting 模型属于集成学习中 Boosting 算法中的一种模型。与 Bagging 算法不同的是，Boosting 算法的思路源于弱学习器的线性组合，通过弱学习器的依次迭代起到减小分类误差的目的。

在介绍 Gradient Boosting 模型之前，我们有必要先看看 Adaboost 模型。Adaboost 是 Boosting 算法中较为著名的模型，其模型利用基学习器 h(x)的线性加和来实现构建强学习器的目的：

$$H(x) = \sum_{t=1}^{T} a_t h_t(x)$$

其中：

$$a_t = \frac{1}{2}\ln(\frac{1-\epsilon_t}{\epsilon_t})$$

为了实现强学习器的构建，对于训练集 D，对 D 按照 T 轮循环依次取出分布 D_t，在利用基学习器进行学习后，取该次学习的误差迭代入下一次学习分布：

$$D_{t+1}(x) = \frac{D_t(x)}{Z_t} \times \begin{cases} \exp(-a_t) & if \ \ h_t(x) = f(x) \\ \exp(a_t) & if \ \ h_t(x) \neq f(x) \end{cases} = \frac{D_t(x)\exp(-a_t f(x) h_t(x))}{Z_t}$$

由此计算每一次学习的误差，并将其权重赋予各个样本进入下一次迭代。(周志华, 2016)

Gradient Boosting 模型的迭代计算方式与 Adaboost 相同，但对于误差的界定方式不同。对于 Adaboost 模型来讲，模型的整体误差计算源于错误错误分类的样本，对于每个错误分类的样本，Adaboost 会赋予错误样本更高的权重，以便于下一个基学习器进行分类。Gradient Boosting 模型对于模型误差的估量来源于梯度，其模型会在估计过程中评估基学习器的残差，并采用梯度下降法对基学习器进行迭代，最终使得模型基于梯度衡量的残差达到设定的误差范围内。

在分类问题中，与 Bagging 类算法相比，Boosting 类算法意在通过基学习器

迭代减少整体分类所产生的偏差。在区分特征出现频率较低的特定实例时，Boosting 类算法表现要好于 Bagging 类算法，但也比 Bagging 类算法更容易产生过拟合的现象。另一方面，由于其线性迭代的特性，因此 Boosting 类算法数据较多时其处理速度往往比 Bagging 类算法要慢。

（四）基准模型设定考量

在对训练集原始数据进行分析后，根据结果可以得出以下推测：

1) 从数量上看，绝大多数机器人用户确实在行为模式上与普通人类用户有一定差别
2) 但数据集本身也包含了许多离群值（仅竞拍个位数次数的机器人用户，竞拍次数过多的人类用户）
3) 数据集存在不可移除的干扰因素，例如，不连续的时间分段必然会对时间间隔的计算产生一定程度的干扰

因此，在设定基准模型时有以下考量：

1) 基准模型可以甄别出特征普适性较强的人类用户和机器人用户
2) 模型需要有一定的抗干扰能力，不需要优秀的离群值分类能力
3) 模型可解释
4) 模型训练速度可以接受，参数调节直观简便

基于上述考量，决定采用 KNN 模型作为基准模型进行训练，并用作与其他模型进行比对。

三、 数据清洗与模型训练

（一）数据预处理

前文提到，该项目主要的数据分析对象为 bids.csv。该部分就将详细阐述作者的数据挖掘过程。首先，由前文的数据分析可知，文件中 country 一项存在缺失，在进行数据分析前，我们将用 unknown 字段对其进行填补。另一方面，在进行数据计算时发现 ip 一项在去掉 ip 各字段中间的点后，数据处理速度会略微加快，因此决定去掉 ip 中分隔点，将各字段合并为一个字段。

考虑到最后是对各个用户进行是否是机器人用户的判定，因此所有进行一次归类的特征统计将基于用户 ID 进行。另一方面，为了充分挖掘数据，也会考虑基于用户 ID，拍卖场次，IP，URL 进行二阶归类统计。考虑到数据量的大小与计算速度，数据统计方式围绕相对传统的统计量展开，如均值，标准差，最大值，最小值，中位数与四分位数。下面将进行各项的分项统计说明。

1. Bid_id(拍卖次数)

从原始数据来看，拍卖次数反映的是每个用户每次的竞拍动作。因此，作者考虑统计以用户 ID 进行一阶分类统计每个用户的总拍卖次数。另外，考虑结合用户 ID，拍卖场次，IP，URL 进行二阶分类统计每场拍卖下，每个 IP 下，每个 URL 下各用户的拍卖次数。

作者在统计训练集中各机器人的竞拍数时发现，除了 7 个机器人竞拍次数小于 10 外，其余机器人均竞拍了 100 次以上。因此，考虑对竞拍次数在 100 次以

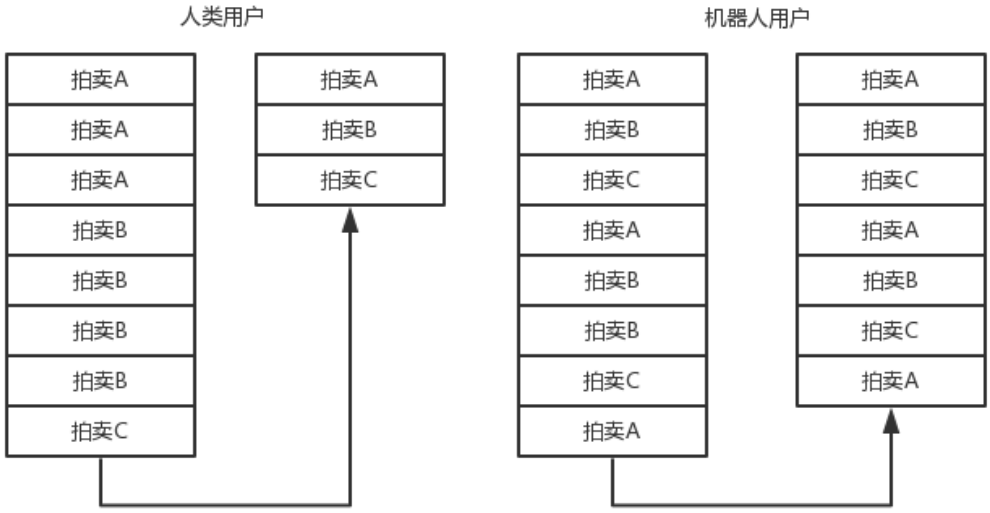
上的各拍卖场次进行筛查，并统计各用户在已经结束的拍卖场次内，可见的 100 次拍卖中所占的比例。统计该项数据需要注意两点，一方面，对于出现在最后一个时间点的拍卖场次应当予以剔除，因为不能断定这些拍卖场次是否已经结束。另一方面，由于时间断层的影响，统计出的数据可能包含一定的干扰性。

另外，根据之前的假设，机器人用户的目的往往是赢得竞拍或是哄抬物价，因此可以统计各用户在各个已经结束的竞拍内最后一次竞拍的次数和倒数第二次竞拍的次数。另外机器人的程序一般会在事先设定好，因此也可以统计第一次竞拍的次数。需要注意的是，统计时需要排除已经开拍和尚未结束的拍卖场次。

2. Auction(拍卖场次)

原始数据中每个拍卖场次有唯一的标识名称，因此可以用于统计各用户参与的拍卖场次次数和每场拍卖中竞拍次数的比重。另一方面，也可以结合 IP，URL 数据统计每个用户在每个 IP 和 URL 下参与的拍卖场次数据。

另一方面，受到 Pierre Gutierrez 所写的博客启发，在统计拍卖场次时定义了一个“顺序计数”函数。该函数用于统计各用户在拍卖场次，IP 和 URL 间切换的频繁程度。其计数方式如下图所示：



如果以代码形式表现，即为：

```
19 # order
20 def order_count(series):
21     # caution: need sort time first
22     ordered = series[series != series.shift()]
23     return ordered.count()
```

其原理为在以时间顺序排列的情况下，利用 pandas 中的 shift 函数剔除数列中的相邻重复值，从而达到顺序计数的目的。

由函数可见，该函数对于相邻的同一拍卖场次仅计数一次。对于人类用户来说，拍卖时在各拍卖场次间切换的频率应该要小于机器人用户。因此理论上可以

使用该函数对频繁切换拍卖场次、IP 和 URL 的机器人用户加以区分。

3. Merchandise（商品种类）/ Device（使用设备）

由于样本集中涉及商品种类较少，因此统计了各用户中出现次数最多的商品种类，并将其以标签形式保存，以便于进行 One-Hot Encoding 处理。在进行数据分析时，虽然发现有部分用户有使用多部设备的迹象，但并未对其进行深度挖掘。除商品种类标签外，仅对商品种类和设备进行简单统计。

4. Time（拍卖时间戳）

从样本来看，时间戳是一项非常重要的数据。除了可以对时间戳本身进行统计外，还可以通过时间戳加工各用户自身的拍卖间隔，用户每场拍卖中自身的拍卖间隔，用户回应他人竞拍间隔，用户同一时间点竞拍次数以及可对时间进行切片统计一段时间间隔内用户拍卖数。

对于时间戳本身作者统计了每个用户的时间戳最大值最小值，并统计了时间区间，以及相同时间戳的重复次数。然后对时间戳分别依用户 ID 和拍卖场次进行差分，统计了每个用户自身的拍卖时间间隔各项数据和每个用户每场拍卖中自身竞拍的时间间隔相关统计数据。

对于跟拍时间间隔，作者在按照拍卖场次进行分类并计算出时间间隔后，采用与“顺序计数”函数类似的手法筛除了每个用户跟拍自己的竞拍记录，进而统计每个用户跟拍他人竞拍的各项统计数据。

由于在上文中的数据预测中作者发现，每个时间块内的数据均有大约 3.5 个周期，因此推测每个时间块内的时间大致为 3.5 天，进而设定以每个小时每用户竞拍数对每个时间块进行切片处理，每个时间块切片数为 84 片。以每个用户每个时间切片竞拍数建立表格，并统计其最大最小值，均值，标准差和四分位数。

5. Country（国家）/ IP / URL

对国家，IP 与 URL 的统计方法较为相似，作者均统计了各用户所使用的国家数，IP 数，URL 数。并对每个用户每场拍卖中所使用的每个国家，IP 和 URL 比重进行相应统计。与 IP，URL 不同的是，国家数相对数量较小，因此可以将每个用户所使用频率最高的国家数标签提取出来，与商品种类一起进行 One-Hot Encoding 处理。

（二）基准模型调参与实现

该部分将对已清洗出的各项数据进行缺失值填补并标准化。在数据完全处理完成后将利用 scikit-learn 库中 KNN 模型进行学习与调试。

1. 特征数据预处理

在从原始数据挖掘出各特征项总计 137 项，作者发现并不是所有的用户 ID 在每一项都有数值。经筛选后存在 NA 值的各特征项为：

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6614 entries, 0 to 6613
Data columns (total 18 columns):
bid_lst100_mean          3851 non-null float64
bid_lst100_std           3009 non-null float64
```

```
bid_lst100_max          3851 non-null float64
bid_lst100_min          3851 non-null float64
first_bid_count         1469 non-null float64
last_bid_count          1426 non-null float64
last_2nd_bid_count      1398 non-null float64
t_resp_min              6610 non-null float64
t_resp_max              6610 non-null float64
t_resp_mean             6610 non-null float64
t_resp_std              6610 non-null float64
t_resp_median           6610 non-null float64
t_resp_auc_mean_mean    6610 non-null float64
t_resp_auc_mean_std     6610 non-null float64
t_resp_auc_median_mean  6610 non-null float64
t_resp_auc_median_std   6610 non-null float64
t_resp_auc_std_mean     4490 non-null float64
outcome                 1984 non-null float64
dtypes: float64(18)
memory usage: 1.3 MB
```

由此可见，各特征项中存在缺失的主要集中于最后 100 次拍卖占比统计，首次末次拍卖统计和回应他人拍卖的跟拍统计中，由于这些项的缺失均为代表在该项统计中该账户没有进行动作，因此可以将缺失值统一赋值为零。

填补缺失值后，由上文可知，在特征值中有两列特征为商品种类与国家的分类的字符串形式。因此在进一步处理前需要将数值型特征与字符串型特征分类处理。

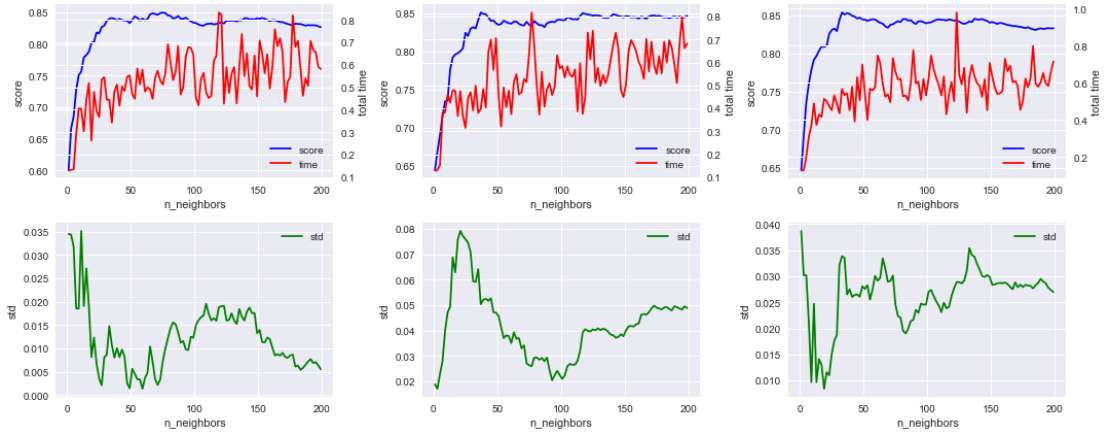
由于 KNN 模型计算的是各样本中各特征量间的距离，因此需要对不同尺度的数值特征进行标准化（集成学习类模型没有这一要求）。因此，作者引入了 sklearn 中的 StandardScaler 函数将各项特征标准化为均值为 0，方差为 1 的各标准项。

对于字符串型特征，作者引入了 MultiLabelBinarizer 函数对其进行了标签二值化处理。

2. KNN 模型学习与调参

在处理好各特征值后就可以引入机器学习模型进行学习与调参了。为了在调参过程中防止过拟合，作者采用了三个不同的 random-state 以 8：2 的比例将训练集划分为训练-交叉检验集与预测集。在调参时采用了 GridSearchCV 及其默认的三折验证法，评分方式为 ROC-AUC。

首先，对 KNN 中 n_neighbers 参数进行调整，该参数直接影响模型的验证效率。过小的参数会使得模型预测波动明显，方差较大，但过大的参数会使得模型在各分类中界限相对模糊。经三个不同的训练集检验，各训练集的交叉验证得分，训练时间和交叉验证标准差如下：



与之对应的测试集结果如下：

Best predict score for random state 15: 0.8295739348370926

Best parameter n_neighbors: 75

Best predict score for random state 233: 0.7924819319583994

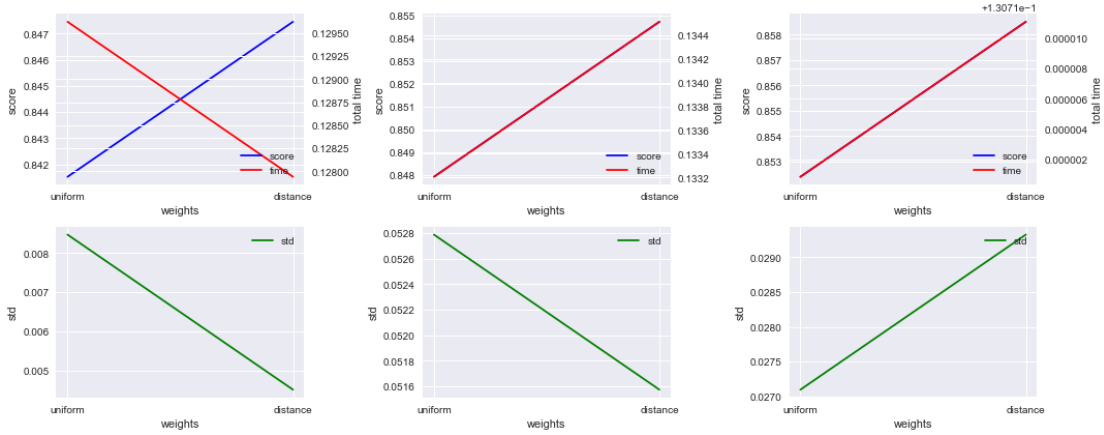
Best parameter n_neighbors: 37

Best predict score for random state 1024: 0.7707417179712694

Best parameter n_neighbors: 33

以上结果可推测，在 n_neighbers 数值为 30-50 之间时，训练集可以兼顾得分与方差，达到较好的预测效果。因此取值为 40。

接下来对 weights 参数进行调参。weights 参数表明了 KNN 中计算一个样本与其他样本间距离的方法，有 uniform 与 distance 两种参数选择。uniform 会对每个作为近邻的样本赋予相同的权重，而 distance 会进一步考量各近邻与样本间的距离，对距离较近者赋予更大的权重。在 n_neighbers 值为 40 时，调参结果如下：



预测结果如下：

Best predict score for random state 15: 0.820036201615149

Best parameter weights: distance

Best predict score for random state 233: 0.7997532169927728

Best parameter weights: distance

Best predict score for random state 1024: 0.8153034300791556

Best parameter weights: distance

可见，distance 参数可以在一定程度上提高预测精度，且对方差没有太大影响。因此选择 distance 作为 weight 参数。

相比于其他模型，KNN 的调参较为简单，除上述两个参数外，其他参数主要涉及距离计算方法等方式的调节，对模型预测影响不大，在调参完毕后对测试集进行预测并在 kaggle 上提交结果，评分如下：

可见，在数据挖掘较为充分的情况下，即使是非常简单的模型也可以保证 80%的机器人判定准确率。从结果可以说明，KNN 模型已经可以判定符合机器人行为的大部分账户，但根据 Kaggle 排行榜上的分数来看，依然有挖掘更多机器人用户的空间。

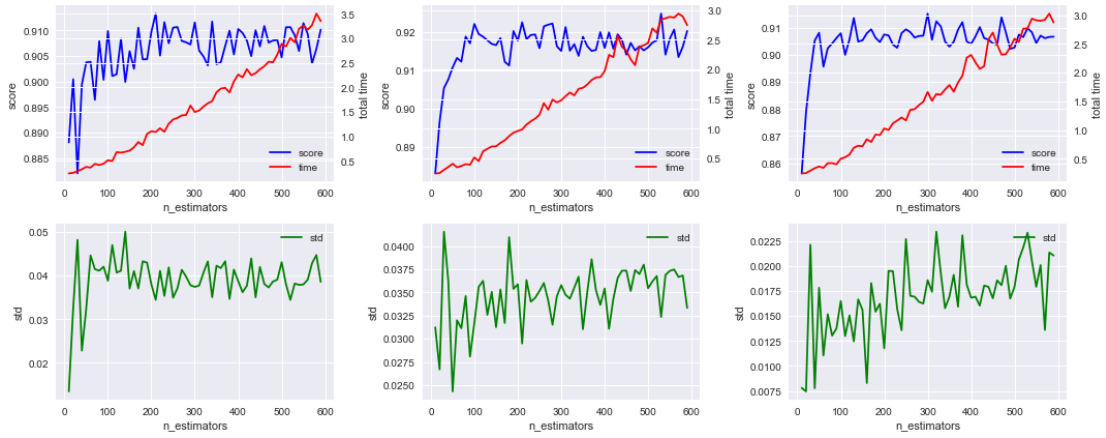
3. 进阶模型调参与训练

为了进一步提高机器人用户的判定准确率，作者引入了集成学习算法中的 random forest 算法与 lightGBM 算法。同样的，将采用相同的 random-state 将训练集拆分三次，并分割成交叉训练集合测试集进行相应的模型调参。

（1） Random Forest 训练与调参

对于 random forest（随机森林）模型（以下简称 RF 模型）来说，首先需要调整的是 n_estimators 参数。该参数决定了有多少个基学习器进行训练，如果基学习器数量不足，则 RF 模型的预测会存在较大的方差。另一方面，基学习器数量的增加除了会使得模型运行时间增加外，对模型预测的准确程度有增无减。因此在通常情况下，训练者都会倾向于在机器可以承受的范围内选用数量较多的基学习器进行学习，以控制模型的预测方差稳定在一个范围内。

由于 sklearn 中 RF 模型的默认基学习器数为 10，而我们的特征量在处理后有 200 项左右，故选择以 10 为步长，将基学习器设定为 10 至 600 进行调整。交叉训练结果如下：



预测结果如下：

Best predict score for random state 15: 0.9077554998607631

Best parameter n_estimators: 210

Best predict score for random state 233: 0.8912832716375816

Best parameter n_estimators: 530

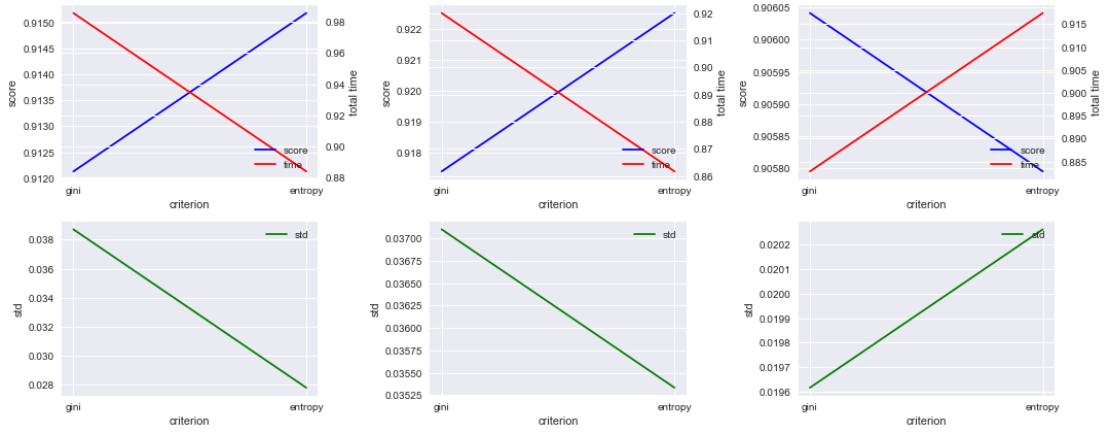
Best predict score for random state 1024: 0.9136616827909704

Best parameter n_estimators: 300

结合分数与标准差观察可知，有 200 以上的基学习器个数后，模型预测分数

与标准差波动趋于稳定。为保险起见，取 n_estimators 为 500。

接下来对 criterion 参数进行调整，该参数评估的是随机森林中决策树基学习器的分类决策方式。可选参数有 gini 和 entropy 两种。Gini 参数即基尼系数，主要由决策树分类后的分类误差率计算获得。entropy 参数即信息增益，主要由决策树分类后的信息纯度计算获得，从经验上看，entropy 比较擅长于探索未知数据集的分类情况。该参数交叉验证结果如下：



预测结果如下：

Best predict score for random state 15: 0.906641604010025

Best parameter criterion: entropy

Best predict score for random state 233: 0.9075445090780891

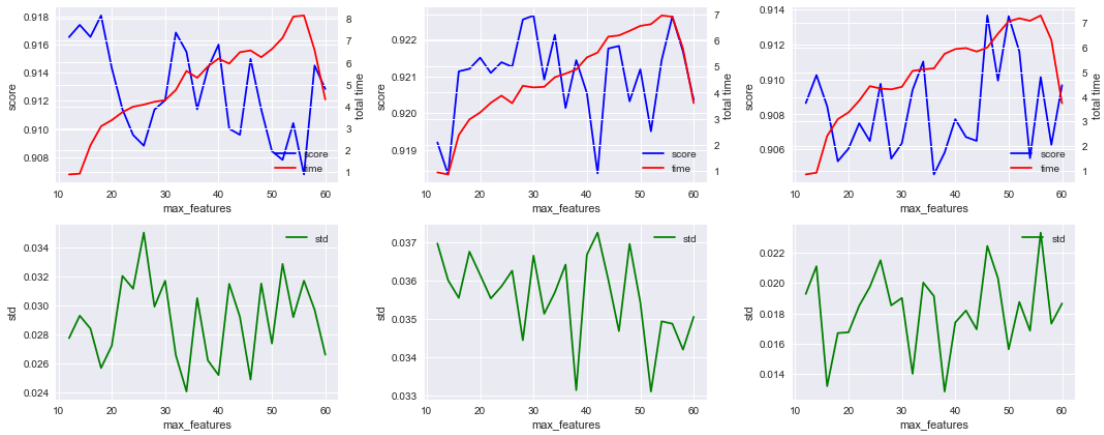
Best parameter criterion: entropy

Best predict score for random state 1024: 0.8811931984755204

Best parameter criterion: gini

可见，entropy 在模型进行预测时，预测分数有小幅提升，方差变化较小。

接下来调整 max_features 参数，max_features 参数的调节会对基学习器在进行分类时考量的最大特征数造成影响，数值越大，基学习器的学习能力越强，但过大的数值会造成基学习器的过拟合。该参数默认值为最大特征量的平方根，根据之前的特征量个数，默认 max_feature 在 15 左右。因此，作者选用 12 至 60 个特征量进行调参，结果如下：



对应的预测结果如下：

Best predict score for random state 15: 0.9160401002506267

Best parameter max_features: 18

Best predict score for random state 233: 0.8979375991538868

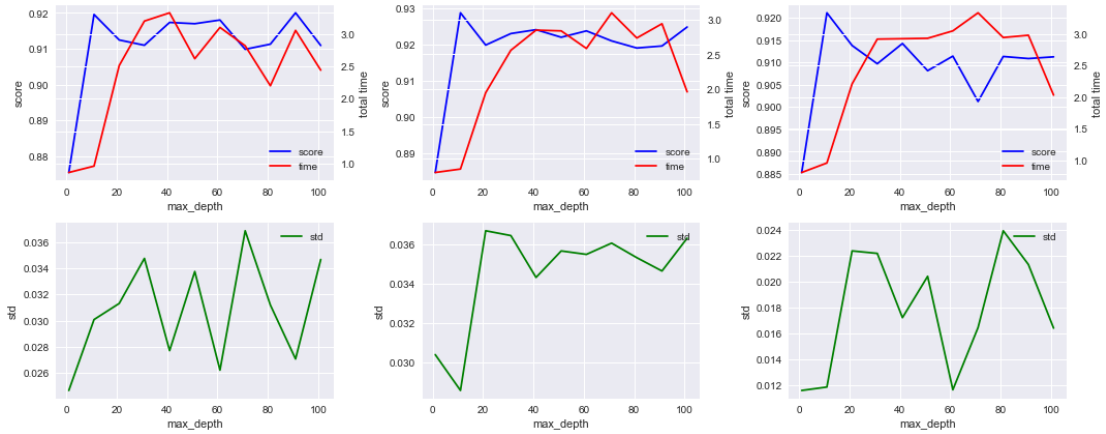
Best parameter max_features: 30

Best predict score for random state 1024: 0.9173995895631779

Best parameter max_features: 46

由图可见，模型在交叉验证时，验证分数与标准差均呈现出了波动的形态，并没有发现特别的规律。因此作者决定放弃设定该参数，保持参数的默认状态。

接下来调节 max_depth 参数。Max_depth 参数影响的是基学习器的最大深度，如果不限制最大深度，则基学习器会完全学习直到不能再进行分裂为止。作者设定最大深度为 1 至 100，步长为 10，结果如下：



预测结果如下：

Best predict score for random state 15: 0.9220272904483431

Best parameter max_depth: 91

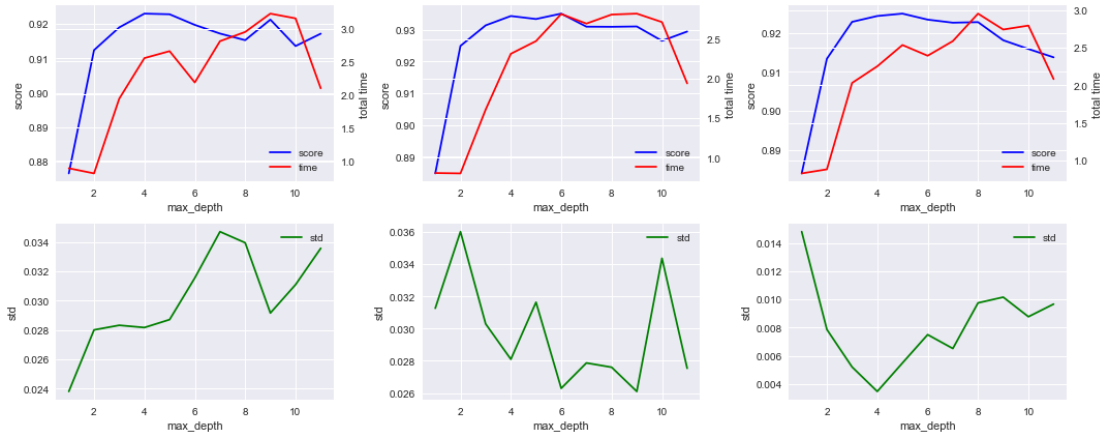
Best predict score for random state 233: 0.9015952758681474

Best parameter max_depth: 11

Best predict score for random state 1024: 0.9094107299912049

Best parameter max_depth: 11

由此可见，由于设定的基学习器数量较多，因此除了最大深度为 1 时学习效果较差外，其余深度均有较为良好的预测表现。因此作者决定将该参数进行进一步的从 1 至 10 的精细调节。精细调节结果如下：



对应预测结果如下：

Best predict score for random state 15: 0.9321915900863269

Best parameter max_depth: 4

Best predict score for random state 233: 0.901022386744227

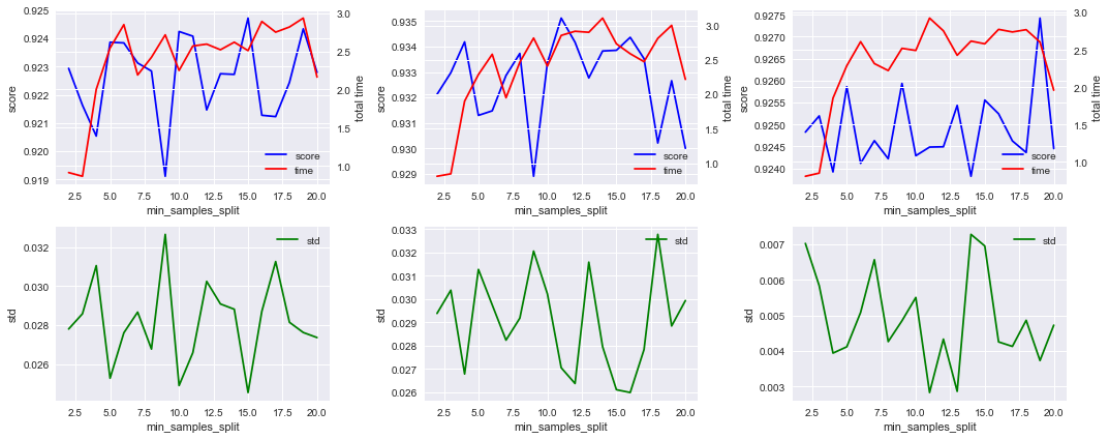
Best parameter max_depth: 6

Best predict score for random state 1024: 0.9034007622398124

Best parameter max_depth: 5

由此可见，得益于基学习器的数量，每个基学习器在最大深度为 4 时就可以达到良好的预测学习效果，因此作者决定将 max_depth 设定为 4。至此，RF 模型的参数粗调已经完成，但如果要进一步优化模型，兼顾分数与方差，可以对 min_samples_split 和 min_samples_leaf 两个参数进行调节。

接下来对 min_samples_split 参数进行调节，该参数主要衡量的是最小的样本分裂数，默认为 2。换句话讲，如果始终有样本不能归为同一类，则分类到每个节点 1 个样本为止。作者认为默认值的设定可能过小，因此决定对其进行调参。取最小分裂数 2 至 20，结果如下：



对应的预测结果如下：

Best predict score for random state 15: 0.9346978557504872

Best parameter min_samples_split: 15

Best predict score for random state 233: 0.9019918914154768

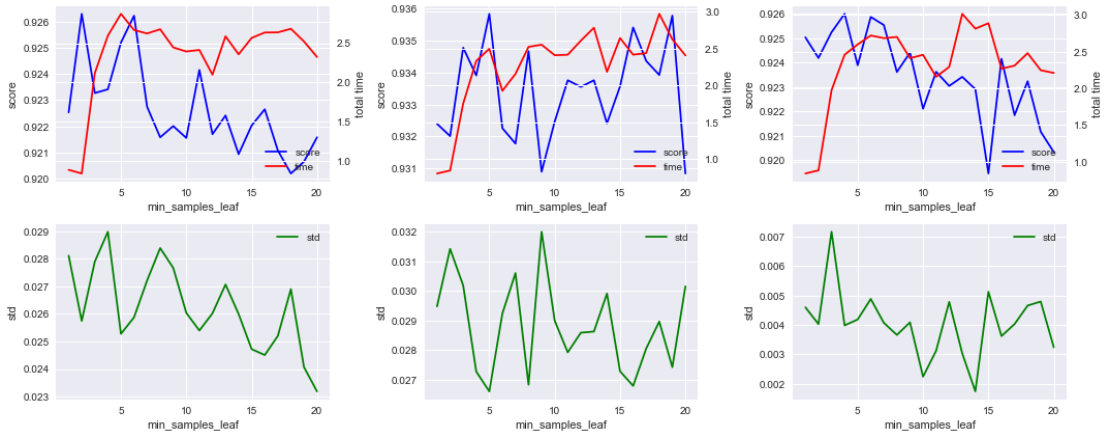
Best parameter min_samples_split: 11

Best predict score for random state 1024: 0.8994429785986513

Best parameter min_samples_split: 19

由此可见，虽然预测分数呈现出波动额趋势，但从方差来讲，在最小分裂数取 10 时可以获得较小的标准差。因此作者决定设定 min_samples_split 为 10。

接下来调节 min_samples_leaf。与 min_samples_split 相似的是，min_samples_leaf 也是为了调节基学习器的最小分类数目。不同之处在于，min_samples_split 调节的是决策树分类前的数目，而 min_samples_leaf 调节的是分类后的数目。该项默认值被设定为 1，与 min_samples_split 的“分无可分”的默认设定类似，因此作者决定同样以 1 至 20 进行调参，结果如下：



预测结果为：

Best predict score for random state 15: 0.9330270119743804

Best parameter min_samples_leaf: 2

Best predict score for random state 233: 0.9008461131676362

Best parameter min_samples_leaf: 5

Best predict score for random state 1024: 0.9034007622398124

Best parameter min_samples_leaf: 4

可以看到，在三个训练集中，有两个训练集在该参数到达 10 后模型预测分数有了较为显著的下降，结合预测结果，作者决定将该参数设定为 5。

在调节 min_samples_leaf 后，RF 模型的从粗到细的调参就完全结束了。在设定 n_estimators 为 500，criterion 为 entropy，max_depth 为 4，min_sample_split 为 10，min_sample_leaf 为 5 的情况下，对测试集进行预测，预测结果如下：

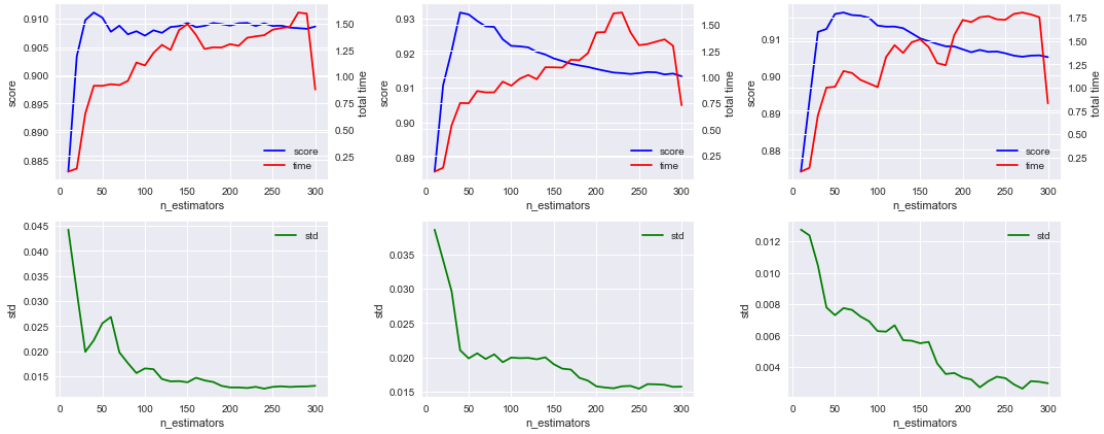
可见，在进行较为细致的调参后，相比于 KNN 模型，RF 模型可以进一步提高模型预测的 ROC 分数，使得对于该测试集的预测率达 93%的水平，在项目本身的测试集上拥有较好的预测表现。

（2）lightGBM 模型训练与调参

前文提到，由于 boosting 模型的原理使得通常情况下，boosting 模型的训练时间要远远大于 bagging 类模型，因此，作者决定采用 lightGBM 作为 boosting 模型中的代表模型对训练集进行训练以减少训练时间。

作为 boosting 训练模型的一种，lightGBM 模型首先需要关注的参数依然是 n_estimators。n_estimators 参数设定的是 lightGBM 中用于顺序分类的弱学习器的数量。简单来讲，弱学习器数量越多，分类次数越多，学习能力越强。但过多的弱学习器会使得模型学习到特定实例的罕见特征，减小模型的泛化能力，且有过拟合的风险。

lightGBM 初始学习器数量设定为 10，考虑到特征数量，决定采用 10 至 300 个基学习器，步长为 10 进行调参测试，其训练结果如下：



对应测试集结果如下：

Best predict score for random state 15: 0.9217488164856587

Best parameter n_estimators: 40

Best predict score for random state 233: 0.9035783536047948

Best parameter n_estimators: 40

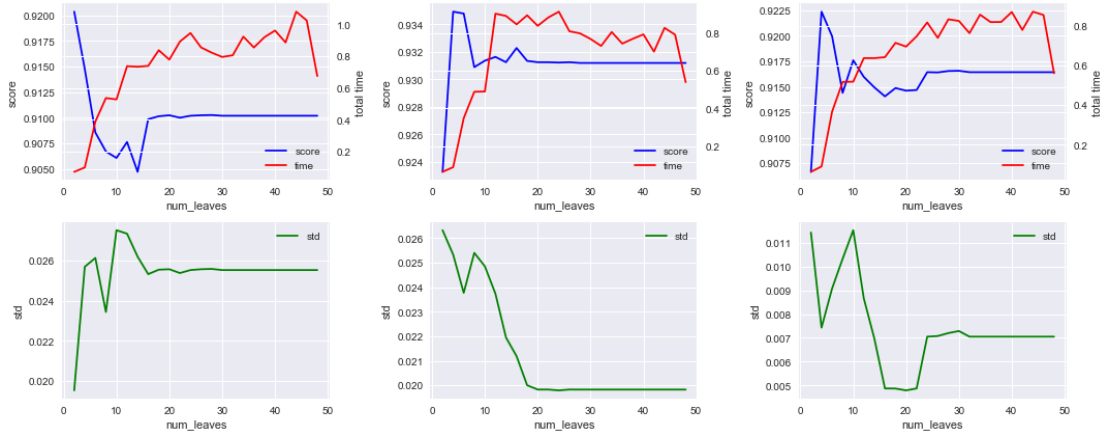
Best predict score for random state 1024: 0.8940193491644678

Best parameter n_estimators: 60

由 boosting 模型原理可知，boosting 类模型的主要学习目的是为了减少模型偏差。从训练集交叉测试的结果可以看出，lightGBM 模型在交叉测试的得分呈现出先升后降的趋势，且模型方差也随之下降。该趋势与之前的假设相吻合，在基学习器数量过多后，lightGBM 模型会学习到过多的特殊实例中的罕见特征，模型预测得分与方差均降低，模型泛化性能下降。结合图形与测试集得分，决定设定 n_estimators 为 50 进行下一步调参。

接下来需要调整 num_leaves 参数，lightGBM 在控制基学习器的决策树的最大深度时，没有采用其他 boosting 算法直接控制最大深度的做法，而是转而控制决策树的叶节点数。二者拥有转换公式：叶节点数 =2^(最大深度)，同时微软的工程师指出，在设定叶节点数时如需套用转换公式，其叶节点数的设定应小于公式计算出的叶节点数，从而减少发生过拟合的可能性。

作者设定 num_leaves 参数范围为 2 至 50，步长为 2，所得的交叉检验结果如下：



所得的对应测试集预测结果如下：

Best predict score for random state 15: 0.9221665274296853

Best parameter num_leaves: 2

Best predict score for random state 233: 0.8957782478406486

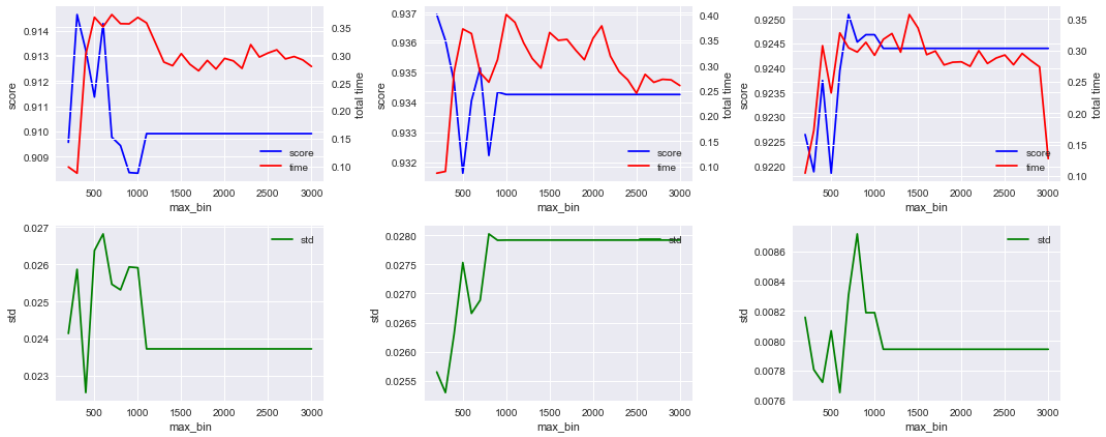
Best parameter num_leaves: 4

Best predict score for random state 1024: 0.8931398416886543

Best parameter num_leaves: 4

由此可见，num_leaves 在 6 以上时，模型的得分下降较为明显，因此作者决定取 num_leaves 为 4。

lightGBM 采用 max_bin 与 subsample_for_bin 两项联合控制模型的子采样率。其中，max_bin 用于控制纳入子采样的特征数量，subsample_for_bin 用于控制纳入子采样的样本数。由于 subsample_for_bin 默认值为 50000，对于本项目来说已经足够，因此主要对 max_bin 进行调参。将 max_bin 设定为 200 至 3000，步长 100，所得交叉检验结果如下：



所对应的测试集结果为：

Best predict score for random state 15: 0.9342801448064606

Best parameter max_bin: 300

Best predict score for random state 233: 0.9006698395910453

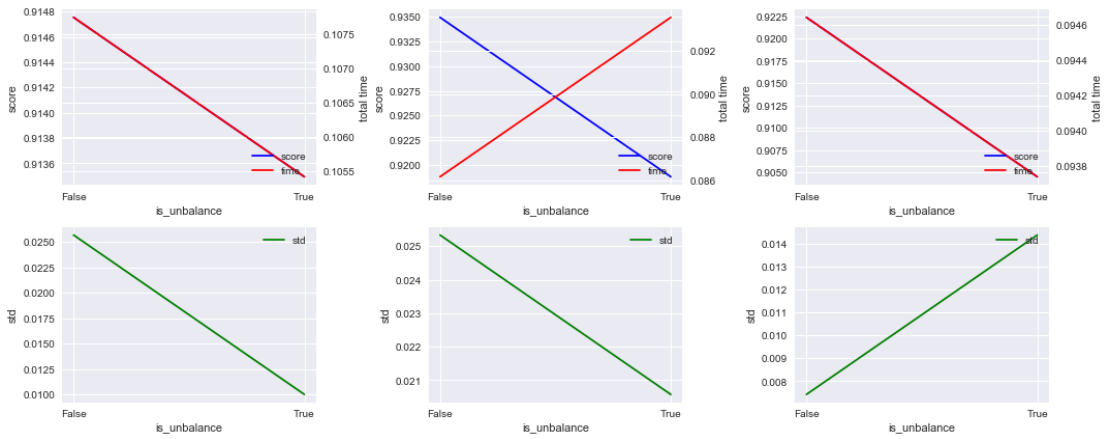
Best parameter max_bin: 200

Best predict score for random state 1024: 0.8962914101436529

Best parameter max_bin: 700

可见，在 max_bin 取值增大的过程中，模型得分趋于稳定，但三个训练集中有两个分数有所下降。因此作者决定维持默认设定 255 不变。

接下来调节 is_unbalance 参数，该参数可用于调节不平衡数据集的预测准确性，在分别取 True 和 False 后，结果如下：



测试集结果如下：

Best predict score for random state 15: 0.9360902255639099

Best parameter is_unbalance: False

Best predict score for random state 233: 0.8957782478406486

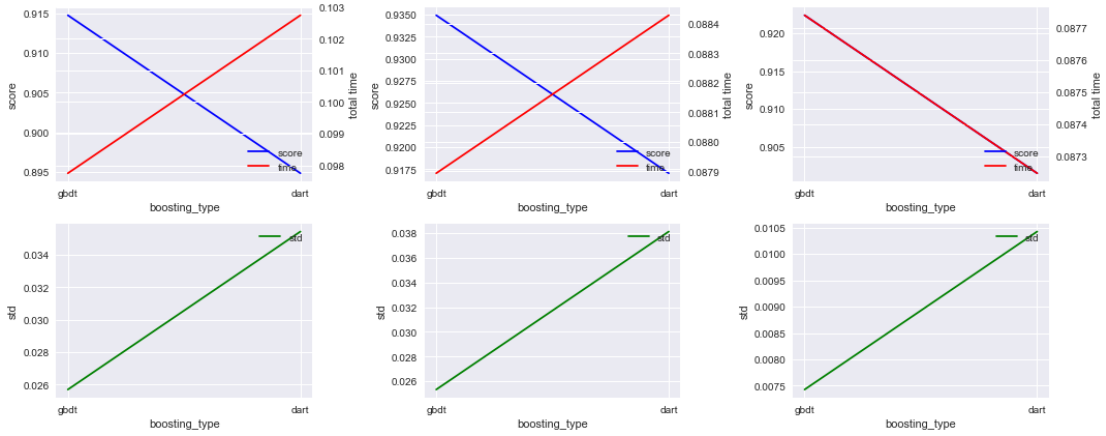
Best parameter is_unbalance: False

Best predict score for random state 1024: 0.8931398416886543

Best parameter is_unbalance: False

从训练集前期的统计数据上看，机器人用户所占的比例较小，数据集应该选择不平衡数据集。但测试结果显示，选择不平衡数据集反而使得模型的预测分数降低，且三个数据集中有两个数据集方差下降，因此推测可能有过拟合情况发生。因此作者决定维持 is_unbalance 参数为 False 不变。

接下来调节 boosting_type 参数，该参数可以选择 gbdt 和 dart 两种。gbdt 为传统的梯度下降形式，而 dart 为随机梯度下降形式。二者区别在于，gbdt 会使用训练中的所有树拟合残差，而 dart 会采取在训练中的树中进行随机抽取的形式进行残差拟合。在特定情况下，dart 会比 gbdt 有更好的模型表现。该参数交叉测试表现如下：



测试集结果如下;

Best predict score for random state 15: 0.9360902255639099

Best parameter boosting_type: gbdt

Best predict score for random state 233: 0.8957782478406486

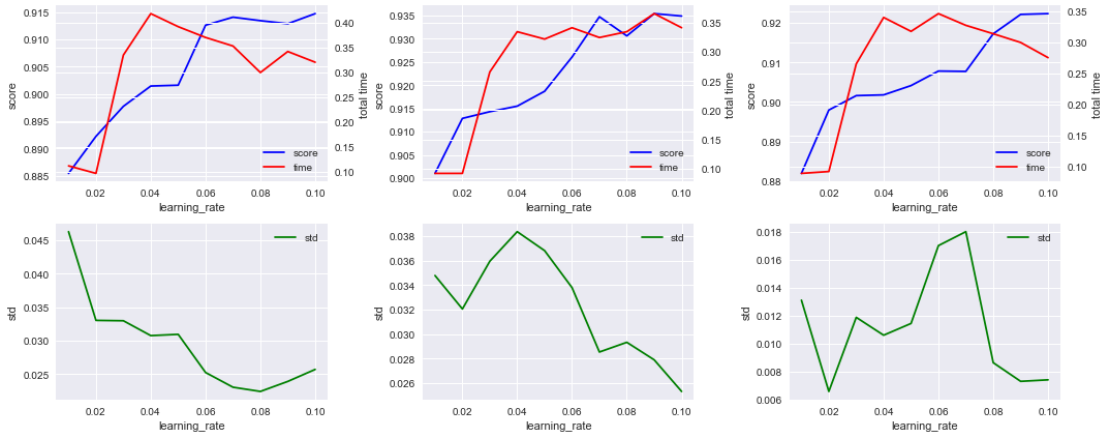
Best parameter boosting_type: gbdt

Best predict score for random state 1024: 0.8931398416886543

Best parameter boosting_type: gbdt

可见，在该项目中，dart 并没有为模型带来更好的拟合程度，反而增大了拟合方差。因此该参数保持默认 gbdt 不变。

接下来对学习率 learning_rate 进行调整，学习率调节的是模型每次分类所学习的速度大小，学习率越低，习得效率越低，计算量越大，但可能会获得更好的拟合结果。因此，作者采用 0.01 至 0.1 的学习率进行测试，结果如下：



对应测试集结果如下;

Best predict score for random state 15: 0.9360902255639099

Best parameter learning_rate: 0.09999999999999999

Best predict score for random state 233: 0.8910188612726953

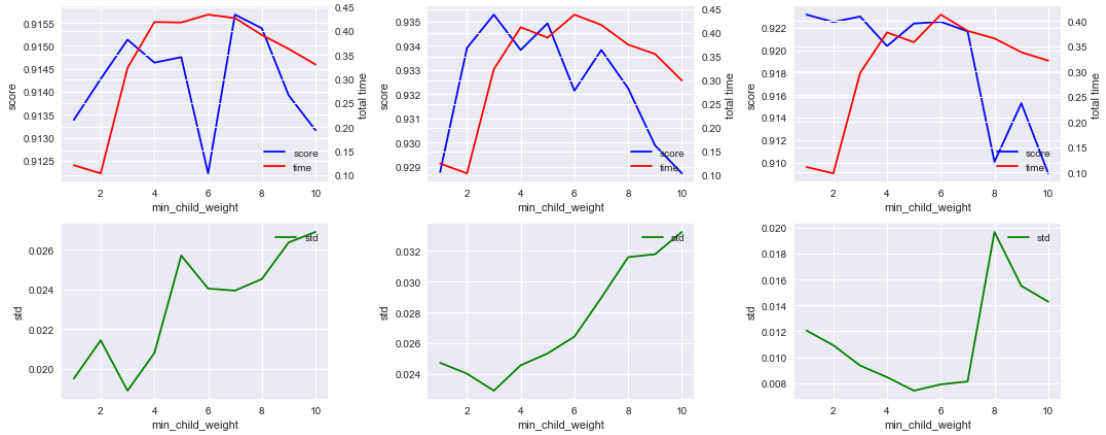
Best parameter learning_rate: 0.09

Best predict score for random state 1024: 0.8931398416886543

Best parameter learning_rate: 0.09999999999999999

可见，在基学习器数量为 50 的情况下，0.1 的学习率已经可以满足学习要求，如果需要通过降低学习率来取得更好的预测结果，需要相应的调整基学习器个数。考虑到模型的过拟合风险，作者决定维持学习率为 0.1 不变。

接下来调整 min_child_weight 参数，该参数决定了模型训练时每个分类所占有的最小比重。可以通过减小该参数调节在不均等样本中的分类拟合优度，但减小该参数的同时也应估计防止过拟合的发生。作者取值 1 至 10 交叉检验结果如下：



其对应的测试集结果如下：

Best predict score for random state 15: 0.9313561681982735

Best parameter min_child_weight: 7

Best predict score for random state 233: 0.892296844702979

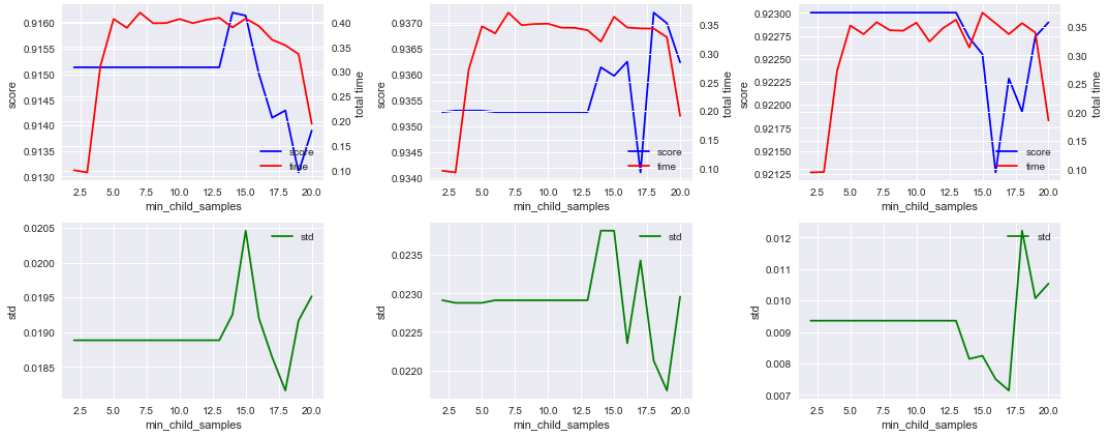
Best parameter min_child_weight: 3

Best predict score for random state 1024: 0.8968044561712105

Best parameter min_child_weight: 1

可见，在 min_child_weight 取值为 3 时，可以得到较好的预测分数，同时也可以兼顾方差的波动，故取值为 3。

与 min_child_weight 限制实例分类比重类似，min_child_sample 参数用于限制基学习器分类中的最小样本数。取该参数 2 至 20，交叉检验结果如下:



对应的测试集结果如下：

Best predict score for random state 15: 0.9358117516012253

Best parameter min_child_samples: 14

Best predict score for random state 233: 0.891195134849286

Best parameter min_child_samples: 18

Best predict score for random state 1024: 0.8915274113163295

Best parameter min_child_samples: 2

可见，由于 min_child_weight 参数的限制，min_child_samples 参数在 12 以内不会引起模型的波动。在 12 以上时模型的波动无明显规律，因此决定放弃设定该参数，维持默认值 10 不变。

在进行以上设定后，最终，作者设定了 lightGBM 中 n_estimators，num_leaves 和 min_child_weight 三个参数，其值分别为 50，4，3。最终预测结果如下：

可见，预测分数与 RF 模型预测分数差距微小，二者得分基本一致。

四、 结果分析与评价

（一）模型评价与选择

根据上文，作者采用了 KNN，random forest（RF）和 lightGBM 三个模型对项目数据集进行预测，三个模型的最终得分结果如下：

从基准模型 KNN 的结果可知，在数据挖掘较为充分的情况下，即使是简单模型也可以得出较好的预测水平，在预测结果要求不严苛的情况下，在充分的数据挖掘基础上，即使是简单的机器学习模型也能较好的完成预测任务。

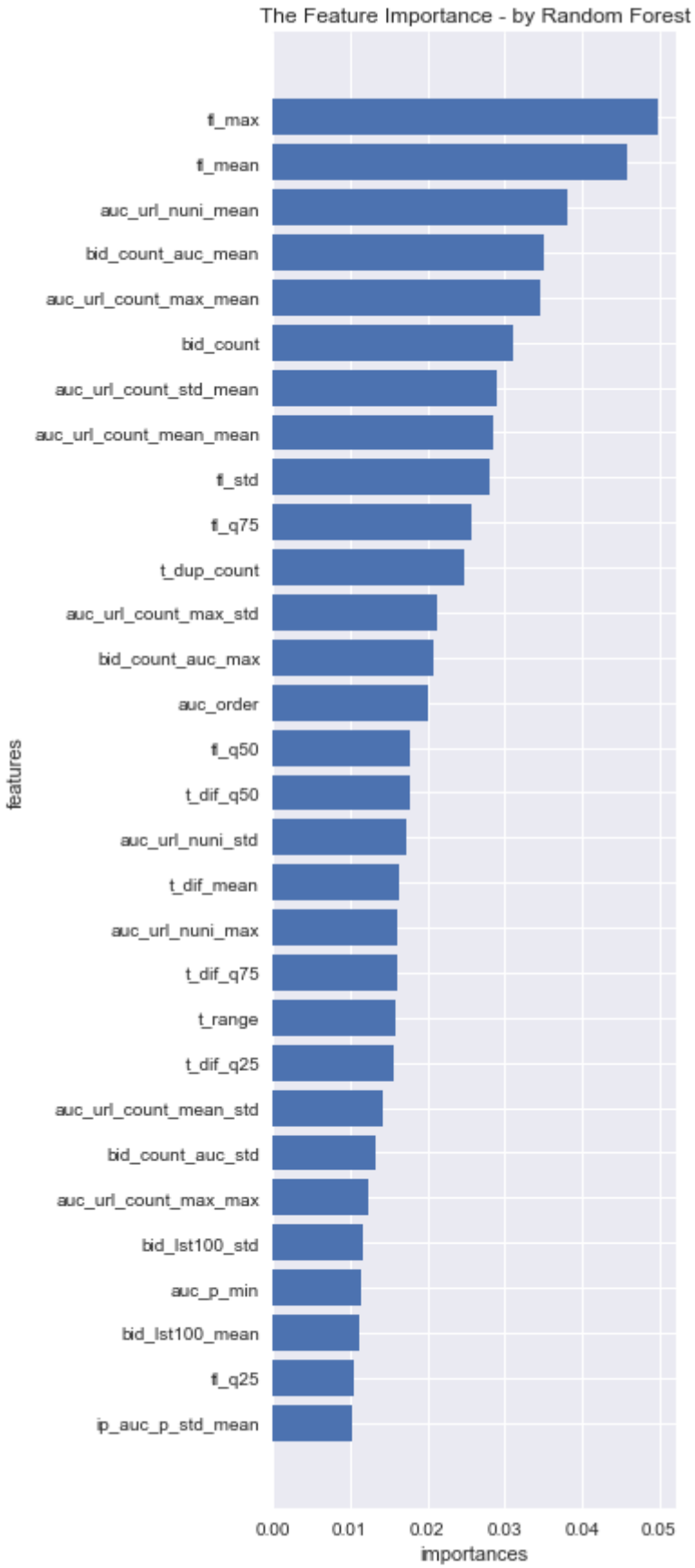
单从竞赛结果来看，RF 模型与 lightGBM 模型的 private score 得分均可以达到排行榜前 50 名左右的预测水平，说明这两个模型均可以较好的完成预测任务。但两个模型在 public score 部分得分仅有 0.91 左右，结合前人的得分经验来看，可能是由于测试集的数据分割数量不平均，且特征差异较大使得预测结果差异较大，由于 private score 评分参照了测试集 70%的数据，因此模型的优劣还应以 private score 为准。

虽然 RF 模型与 lightGBM 模型得分难分伯仲，但从两个模型的构建思路来看，作者更偏好 RF 模型。前文提到，bagging 类模型的构建思路为由大量的基学习器进行投票从而减小模型方差，而 boosting 类模型的构建思路为利用基学习器的迭代减小模型偏差。因此在实际的生产过程中，如果分类任务是为了识别泛化特征而进行分类，bagging 类模型的表现在充分的数据挖掘基础上可以有较为良好的表现，且调参较为容易，不易被实例的罕见特征所影响。而 boosting 类模型在存在罕见特征的实例分类问题中，需要时刻注意模型的过拟合性，因此调参难度更高。以本题为例，假设将前文的 lightGBM 模型 n_estimator 参数由 50 调整为 300，其他参数不变，则有如下结果：

可见，模型的预测能力有了较为显著的下降。因此可知，在有较多离群值的实例环境中，boosting 类模型的调参问题难度要高于 bagging 类模型。

（二）模型的简单统计评价

由于 sklearn 中 random forest 模型拥有特征量重要性统计功能，因此作者统计了重要性超过 0.01 的特征量，结果如下：



可见，最重要的统计量符合作者预期，为时间切片的最大值与均值统计。机

器人不分昼夜的竞拍行为确实对识别其行为模式有很大的作用。另一方面，拍卖次数与参与拍卖场数统计也对机器人的识别起到了重要作用。可能正是由于这些因子的出现，使得即使是简易模型也可以有很好的识别效果。

另一方面，作者也对重要度为零的特征量进行了相应的统计：

merch_nuni, merch_p_max, merch_p_min, merch_p_mean, merch_p_std, t_dif_auc_min_min, cty_auc_p_max_max, ip_auc_p_max_max, am, ar, at, auto parts, az, ba, bd, bf, bg, bh, bn, books and music, br, ca, ch, cl, clothing, cm, co, computers, cz, de, dj, dk, dz, eg, es, et, eu, fi, fr, furniture, ge, gh, gm, gt, hk, ie, il, iq, ir, it, jo, jp, ke, kg, kr, kw, lk, lu, ma, me, mk, mr, mt, mw, mx, mz, na, nl, no, np, om, pa, pe, pk, pl, py, qa, ro, rs, ru, sa, sd, se, sg, sk, so, sv, sz, tn, tr, tw, tz, ua, uk, unknown, vn, ye, zm, zw

去掉二值化标签后的结果为：

merch_nuni, merch_p_max, merch_p_min, merch_p_mean, merch_p_std, t_dif_auc_min_min, cty_auc_p_max_max, ip_auc_p_max_max

可见，对于商品种类的统计基本上没有预测效果。另外，让作者比较惊讶的是，每场拍卖下 IP 所占比的最大值居然也没有预测效果，推测可能是由于人类用户中也存在频繁切换 IP 的用户导致离群值过多，进而使得模型难以挖掘出有用信息。

为了对模型的预测结果有一个直观感受，作者对三个模型对预测集的预测结果做了一个简单的数据统计：

	knn	rf	lgb
count	4700.000000	4700.000000	4700.000000
mean	0.027699	0.048776	0.051190
std	0.054629	0.091196	0.111787
min	0.000000	0.000000	0.000000
25%	0.000000	0.001804	0.005526
50%	0.000000	0.008324	0.009142
75%	0.027817	0.042140	0.019987
max	0.631316	0.643884	0.891113

从数据统计我们可以看到，由于样本分类数量的不平均，KNN 模型在分类时更偏好采用“疑罪从无”的方式把用户归为人类用户一类，但对于有显著机器人特征的用户，KNN 模型与 RF 模型对其的判定概率较为一致。

相比之下，RF 模型判定结果较为均衡，与 KNN 模型相比，RF 模型主要提高了界限较为模糊的用户的机器人判定概率。

与 KNN，RF 模型相比，从模型统计的最大值可看出，lightGBM 模型对于

机器人用户的判定要激进的多。由于样本的不均衡，作者进一步统计了 95%与99%分位的各模型预测数据：

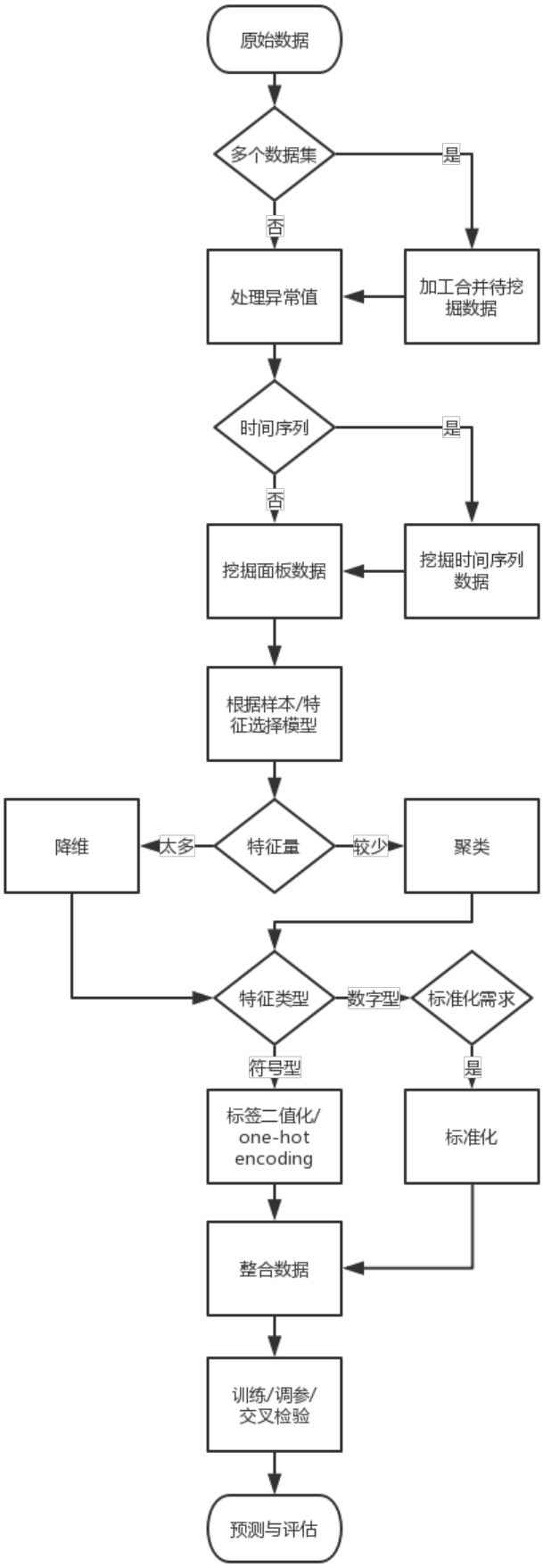
	knn	rf	lgb
q95	0.132831	0.238262	0.267393
q99	0.258214	0.439829	0.603878

上述统计结果证明了前文的假设，在面对离群值较多的数据集时，boosting模型在习得特征时有过度习得实例特征的风险，在调参是需要多加注意。

五、结论与思考

（一）工作流整理

Kaggle 的 human or robot 竞赛项目是一个典型的机器学习下的监督学习项目。该项目涉及了数据清洗，数据挖掘，模型选择与评估等监督学习的常见问题，用一个较脏的数据集迫使参与者像真正面对实际问题的数据分析师那样思考。作者根据该项目所涉及的数据处理各项流程，总结了实际工作中监督问题的工作流程如下：


原始数据
多个数据集
是
否
处理异常值
加工合并待挖掘数据
时间序列
是
否
挖掘面板数据
挖掘时间序列数据
根据样本/特征选择模型
降维
太多
特征量
较少
聚类
特征类型
数字型
标准化需求
符号型
是
标签二值化/one-hot encoding
标准化
整合数据
训练/调参/交叉检验
预测与评估

上图基本重现了一个数据分析师面对监督学习问题是所采取的必要流程。在该项目中，由于特征量的数量可以接受，且绝大部分特征量都起到了一定预测作用，因此在该项目中，作者并没有采用聚类或降维的相关技术对数据进行处理。

（二）项目总结与思考

Kaggle 的 human or robot 项目是一个很有难度的监督学习挑战项目。该项目的参与人需要在面对较脏的数据集时做出较为良好的预测，并且需要同时平衡真阳性样本和伪阳性样本的判定概率。根据题目，作者总结了参与者需要面对的以下问题：

- ✧ 经过混淆的时间戳处理问题
- ✧ 异常值替换问题
- ✧ 无法排除的时间段断层所具有的干扰问题
- ✧ 深度挖掘数据与统计量的选择
- ✧ 模型选择与调参

在该项目中，作者历时一个半月进行数据清洗二十多次，采用了多种数据挖掘方法后才成功挖掘到足够稳定的特征数据用于预测。该项目的体验说明，一个出色的数据分析师/机器学习工程师并不是只是熟识各种机器学习理论就可以完美的解决现实问题，对业务的熟悉程度，对原始数据的处理手法，都可能会大大影响现实生产环境中的数据预测质量。

另外，作者在整个项目流程中，也产生了一些思考。

首先，作者对当时竞赛中第一名所表示的“应该手工去掉训练集中仅竞拍一次的机器人用户”的说法表示质疑。虽然从通常情况来看，如果一个用户只竞拍了一次，确实很难判断这个用户是不是一个机器人用户。但在该项目的数据处理中，作者曾经在去掉训练集中仅竞拍一次的机器人用户的情况下进行过模型评估，发现数据集交叉检验的 ROC 得分会从 0.91 下降到 0.75。因此作者估计训练集中仅竞拍一次的机器人用户依然可以提供用于判定用户是否为机器人的重要信息。由于当时的竞赛第一名并未公开其代码，因此对于其去掉竞拍一次的机器人用户的模型实现过程无法进行求证。

其二，python 到目前已经更新到 3.6 版本，但基于 Cython 构建的 pandas 在进行自定义函数的计算上效率依然很低。另外，pandas 框架在计算存在 NA 值的数据时，如果计算公式中包含求和，则所有 NA 值会被当作零纳入计算。因此在计算时往往需要手工去除 NA 值，但同时这样也会导致计算效率降低。

其三，该项目中数据集的不平衡问题有进一步探究的必要。从该项目的最后得分来看，一些用户在 public score 与 private score 上的得分差距巨大，甚至可以达到 0.93 与 0.89 的差距，作者的训练结果也体现出了这一点，但 kaggle 并没有对这一问题进行更详细的说明。

最后，该项目中，排名靠前的选手得分差距都非常小，加上该项目中数据集本身的缺陷，使得作者认为该项目中选手所训练的模型都或多或少的具有一定程度的过拟合风险，离投入实际生产还有一定的距离。由于竞赛是分数导向，因此为了取得更高的分数而采用更激进的策略是可以理解的，但实际的生产过程中，应该还需要针对各方需求进行进一步的模型调整。

（三）可能的改进

虽然作者在该项目中建立了一定程度的流程化处理方案，但碍于 pandas 运行速度的限制，在数据清洗过程中依然需要耗费大量的时间。目前的部分解决方案是，对于 pandas 中一些可能运行时间较长的运算，可以选择绕过 pandas 本身的计算方式，将数据转化为 numpy 矩阵并调用相应的 numpy 函数进行运算。如果要更进一步优化运行效率，也许可以采用套用多线程的方式重构 pandas 源代码或是利用 CUDA 调用 GPU 进行运算加速。

算法上，理论上可以进一步纳入考量的算法是 SVM（支持向量机）算法，该算法抗干扰能力强，且可以通过调用不同的内核变化分类方式。由于该项目中数据集较小，因此不存在太大的训练难度。如果数据集数量巨大（训练集大于 100 万行），则该算法可能运行十分缓慢。

如果将最终模型作为新的基准，作者认为，在该项目限定的数据集下，换用模型所能带来的提升较为有限。但在模型的运行效率方面还有一定的优化空间。

引用

Breiman, L., 2001. RANDOM FORESTS. *Machine learning*, pp. 1-33.

COVER, T. & HERT, P., 1967. Nearest Neighbor Pattern Classification. *IEEE TRANSACTIONS ON INFORMATION THEORY*, 1, pp. 21-27.

Duck, S. Y., 2015. *Identifying bots in an online auction.* [联机]
Available at: http://small-yellow-duck.github.io/auction.html

Freund, Y., 1999. A Short Introduction to Boosting. *Journal of Japanese Society for Artificial Intelligence*, 9, pp. 1-14.

Gutierrez, P., 2015. *Human Or Robot, Becoming a Feature Engineer.* [Online]
Available at: https://blog.dataiku.com/2015/06/09/kaggle-feature-engineering

Robert, T., 2015. *Kaggle’s “Human Or Robot?” Competition Feedback.* [联机]
Available at: http://www.thomas-robert.fr/en/kaggles-human-or-robot-competition-feedback/

城东, 2015. *机器学习各种算法怎么调参?*. [联机]
Available at: https://www.zhihu.com/question/34470160

周志华, 2016. *机器学习*. 无出版地:清华大学出版社.