GBDT + LR理论与实践

在上一篇文章，提到了Facebook 2014年发表的一篇采用GBDT构建特征的论文：Practical lessons from predicting clicks on ads at facebook。为了深入学习GBDT，本文将重点分析这篇文章的思路，即CTR预估经典模型：GBDT+LR，一个曾风靡Kaggle、至今在工业界仍存有余温的传奇模型。同时采用scikit-learn里面的GBDT和LR来完成GBDT+LR的实验。

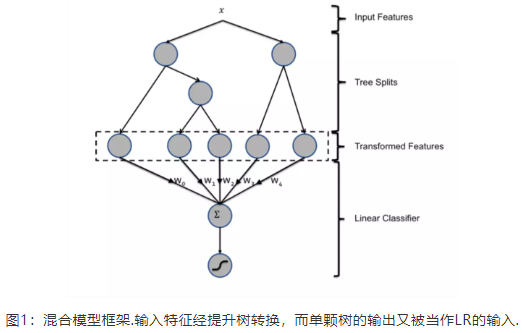
# 背景介绍

论文开篇介绍在计算广告领域，Facebook日活用户超过7.5亿，活跃广告超过1百万，这种数据规模对Facebook来说也是一大挑战。在这种情形下，Facebook是怎么做的呢？引入了一个组合决策树和LR的模型，该模型比单一的LR或GBDT的效果都要好，不仅将点击率提升了3%，还大大提升了整个系统的性能。除此之外，Facebook还在online learning、data freshness, 学习率等参数上进行了探索。

# 模型结构

Facebook论文的Section 1给出了一个重要结论：只要有正确的特征和正确的模型，其他因素对模型结果的影响就非常小。那么，正确的特征是什么呢？论文对比了两类特征，一类是用户或广告的历史信息特征(historical features)，另一类是contextual features(上下文特征)，相比之下historical features要优于contextual features。正确的模型指的boosted decision tree + LR，其中boosted decision tree又相当于对重要的特征做了feature selection。

在Section 3描述了论文的核心模型，整个hybird模型框架示意图如下：



对于线性分类器，有两种特征转换方式可以提升分类器的精度。

* 对于连续特征，可以对特征分bin，然后将bin的index作为类别特征，如此线性分类器就可以学习特征的非线性映射，这种方式里，学习有效的bin边界非常重要。
* 对于类别特征，可以采用笛卡尔积（Cartesian product）枚举出所有的二元特征组合。缺点是得到的特征会包含冗余特征。

为此，基于GBDT的特征转换方法诞生了。

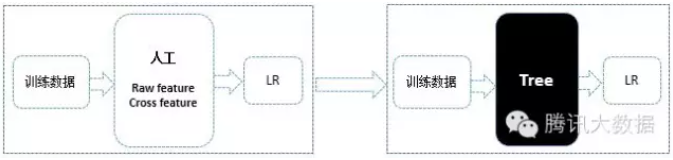
|  |
| --- |
| 基于GBDT的特征转换：将单棵决策树的结果看作是一个类别特征，取值为样本落入在决策树的叶子节点的编号。例如，图1中提升树包含两棵子树，第一棵子树包含3个叶子节点，第二棵树包含2个叶子节点。对于输入样本x（包含多个特征），采用提升决策树（GBDT）进行训练，最终对于第一棵子树上，样本分裂之后落到第二个叶子节点，对于第二棵子树，样本落到了第1个叶子节点，那么通过特征进行转化之后就是[0,1,0,1,0]。 |

==================================================================

CTR预估，广告点击率（Click-Through Rate Prediction）是互联网计算广告中的关键环节，预估准确性直接影响公司广告收入。CTR预估中用的最多的模型是LR（Logistic Regression）[1]，LR是广义线性模型，与传统线性模型相比，LR使用了Logit变换将函数值映射到0~1区间 [2]，映射后的函数值就是CTR的预估值。LR，逻辑回归模型，这种线性模型很容易并行化，处理上亿条训练样本不是问题，但线性模型学习能力有限，需要大量特征工程预先分析出有效的特征、特征组合，从而去间接增强LR 的非线性学习能力。

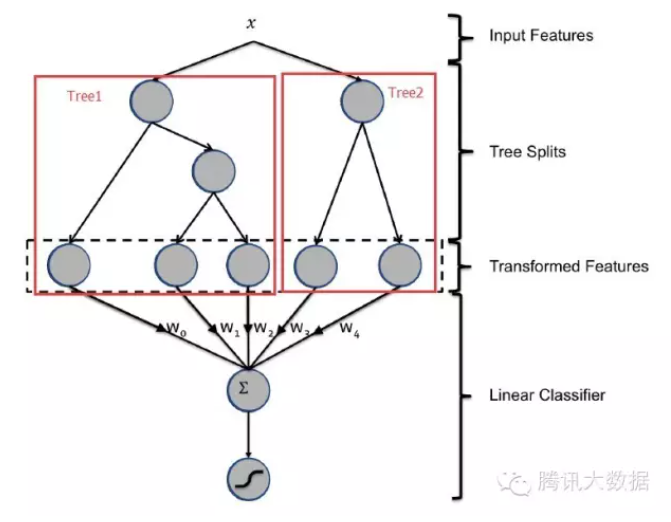
LR模型中的特征组合很关键，但又无法直接通过特征笛卡尔积 解决，只能依靠人工经验，耗时耗力同时并不一定会带来效果提升。如何自动发现有效的特征、特征组合，弥补人工经验不足，缩短LR特征实验周期，是亟需解决的问题。Facebook 2014年的文章介绍了通过GBDT （Gradient Boost Decision Tree）解决LR的特征组合问题[3]，随后Kaggle竞赛也有实践此思路[4][5]，GBDT与LR融合开始引起了业界关注。

GBDT（Gradient Boost Decision Tree）是一种常用的非线性模型[6][7][8][9]，它基于集成学习中的boosting思想[10]，每次迭代都在减少残差的梯度方向新建立一颗决策树，迭代多少次就会生成多少颗决策树。GBDT的思想使其具有天然优势，可以发现多种有区分性的特征以及特征组合，决策树的路径可以直接作为LR输入特征使用，省去了人工寻找特征、特征组合的步骤。这种通过GBDT生成LR特征的方式（GBDT+LR），业界已有实践（Facebook，Kaggle-2014），且效果不错，是非常值得尝试的思路。下图1为使用GBDT+LR前后的特征实验示意图，融合前人工寻找有区分性特征（raw feature）、特征组合（cross feature），融合后直接通过黑盒子（Tree模型GBDT）进行特征、特种组合的自动发现。



# GBDT与LR融合现状

GBDT与LR的融合方式，Facebook的paper有个例子如下图2所示，图中Tree1、Tree2为通过GBDT模型学出来的两颗树，x为一条输入样本，遍历两棵树后，x样本分别落到两颗树的叶子节点上，每个叶子节点对应LR一维特征，那么通过遍历树，就得到了该样本对应的所有LR特征。由于树的每条路径，是通过最小化均方差等方法最终分割出来的有区分性路径，根据该路径得到的特征、特征组合都相对有区分性，效果理论上不会亚于人工经验的处理方式。



GBDT模型的特点，非常适合用来挖掘有效的特征、特征组合。业界不仅GBDT+LR融合有实践，GBDT+FM也有实践，2014 Kaggle CTR竞赛冠军就是使用GBDT+FM，可见，使用GBDT融合其它模型是非常值得尝试的思路[11]。

笔者调研了Facebook、Kaggle竞赛关于GBDT建树的细节，发现两个关键点：采用ensemble决策树而非单颗树；建树采用GBDT而非RF（Random Forests）。解读如下：

1）为什么建树采用ensemble决策树？

一棵树的表达能力很弱，不足以表达多个有区分性的特征组合，多棵树的表达能力更强一些。GBDT每棵树都在学习前面棵树尚存的不足，迭代多少次就会生成多少颗树。按paper以及Kaggle竞赛中的GBDT+LR融合方式，多棵树正好满足LR每条训练样本可以通过GBDT映射成多个特征的需求。

2）为什么建树采用GBDT而非RF？

RF也是多棵树，但从效果上有实践证明不如GBDT。且GBDT前面的树，特征分裂主要体现对多数样本有区分度的特征；后面的树，主要体现的是经过前N颗树，残差仍然较大的少数样本。优先选用在整体上有区分度的特征，再选用针对少数样本有区分度的特征，思路更加合理，这应该也是用GBDT的原因。

然而，Facebook和Kaggle竞赛的思路是否能直接满足现在CTR预估场景

呢？

按照Facebook、Kaggle竞赛的思路，不加入广告侧的AD ID特征？但是现CTR预估中，AD ID类特征是很重要的特征，故建树时需要考虑AD ID。直接将AD ID加入到建树的feature中？但是AD ID过多，直接将AD ID作为feature进行建树不可行。下面第三部分将介绍针对现有CTR预估场景GBDT+LR的融合方案。

# GBDT与LR融合方案

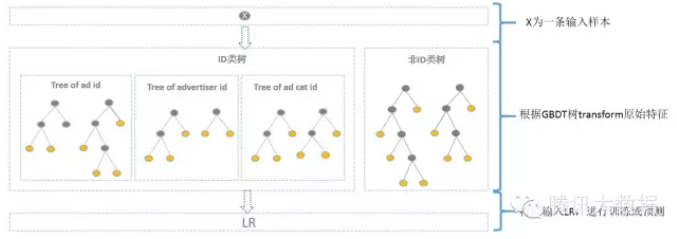
AD ID类特征在CTR预估中是非常重要的特征，直接将AD ID作为feature进行建树不可行，故考虑为每个AD ID建GBDT树。但互联网时代长尾数据现象非常显著，广告也存在长尾现象，为了提升广告整体投放效果，不得不考虑长尾广告[12]。在GBDT建树方案中，对于曝光充分训练样本充足的广告，可以单独建树，发掘对单个广告有区分度的特征，但对于曝光不充分样本不充足的长尾广告，无法单独建树，需要一种方案来解决长尾广告的问题。

综合考虑方案如下，使用GBDT建两类树，非ID建一类树，ID建一类树。

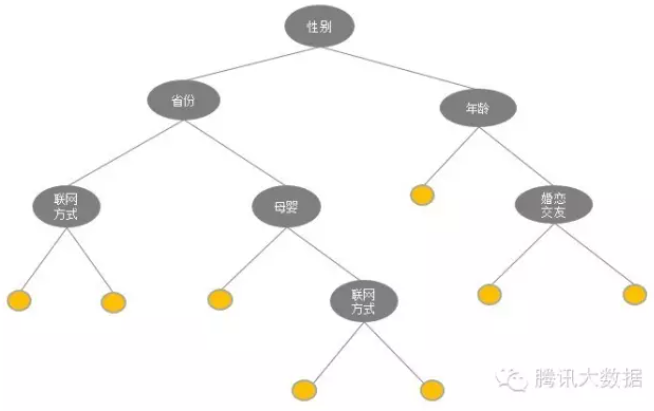
1）非ID类树：不以细粒度的ID建树，此类树作为base，即便曝光少的广告、广告主，仍可以通过此类树得到有区分性的特征、特征组合。

2）ID类树：以细粒度的ID建一类树，用于发现曝光充分的ID对应有区分性的特征、特征组合。

如何根据GBDT建的两类树，对原始特征进行映射？以如下图3为例，当一条样本x进来之后，遍历两类树到叶子节点，得到的特征作为LR的输入。当AD曝光不充分不足以训练树时，其它树恰好作为补充。

**通过GBDT 映射得到的特征空间维度如何？**

GBDT树有多少个叶子节点，通过GBDT得到的特征空间就有多大。如下图4一颗树，一个叶子节点对应一种有区分性的特征、特征组合，对应LR的一维特征。这颗树有8个叶子节点，即对应LR 的8维特征。估算一下，通过GBDT转换得到的特征空间较低，Base树、ID树各N颗，特征空间维度最高为N+N\*广告数+N\*广告主数+ N\*广告类目数。其中广告数、广告主数、广告类目数都是有限的，同时参考Kaggle竞赛中树的数目N最多为30，则估算通过GBDT 映射得到的特征空间维度并不高，且并不是每个ID训练样本都足以训练多颗树，实际上通过GBDT 映射得到的特征空间维度更低。

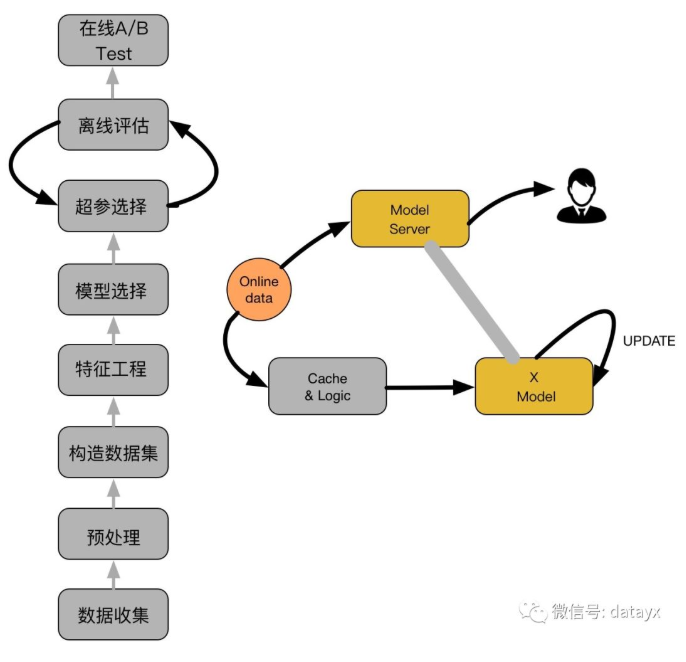


**如何使用GBDT 映射得到的特征？**

通过GBDT生成的特征，可直接作为LR的特征使用，省去人工处理分析特征的环节，LR的输入特征完全依赖于通过GBDT得到的特征。此思路已尝试，通过实验发现GBDT+LR在曝光充分的广告上确实有效果，但整体效果需要权衡优化各类树的使用。同时，也可考虑将GBDT生成特征与LR原有特征结合起来使用，待尝试。

# 4总结与展望

点击率预估模型涉及的训练样本一般是上亿级别，样本量大，模型常采用速度较快的LR。但LR是线性模型，学习能力有限，此时特征工程尤其重要。现有的特征工程实验，主要集中在寻找到有区分度的特征、特征组合，折腾一圈未必会带来效果提升。GBDT算法的特点正好可以用来发掘有区分度的特征、特征组合，减少特征工程中人力成本，且业界现在已有实践，GBDT+LR、GBDT+FM等都是值得尝试的思路。不同场景，GBDT融合LR/FM的思路可能会略有不同，可以多种角度尝试。从知乎https://zhuanlan.zhihu.com/p/29053940上看到了一个关于CTR的流程，如下图所示：



如上图，主要包括两大部分：离线部分、在线部分，其中离线部分目标主要是训练出可用模型，而在线部分则考虑模型上线后，性能可能随时间而出现下降，弱出现这种情况，可选择使用Online-Learning来在线更新模型：

2.1 离线部分

数据收集：主要收集和业务相关的数据，通常会有专门的同事在app位置进行埋点，拿到业务数据

预处理：对埋点拿到的业务数据进行去脏去重；

构造数据集：经过预处理的业务数据，构造数据集，在切分训练、测试、验证集时应该合理根据业务逻辑来进行切分；

特征工程：对原始数据进行基本的特征处理，包括去除相关性大的特征，离散变量one-hot，连续特征离散化等等;

模型选择：选择合理的机器学习模型来完成相应工作，原则是先从简入深，先找到baseline，然后逐步优化；

超参选择：利用gridsearch、randomsearch或者hyperopt来进行超参选择，选择在离线数据集中性能最好的超参组合；

在线A/B Test：选择优化过后的模型和原先模型（如baseline）进行A/B Test，若性能有提升则替换原先模型；

2.2 在线部分

Cache &amp; Logic：设定简单过滤规则，过滤异常数据；

模型更新：当Cache &amp; Logic 收集到合适大小数据时，对模型进行pretrain+finetuning，若在测试集上比原始模型性能高，则更新model server的模型参数；

Model Server：接受数据请求，返回预测结果；

3. GBDT + LR 的结构

正如它的名字一样，GBDT+LR 由两部分组成，其中GBDT用来对训练集提取特征作为新的训练输入数据，LR作为新训练输入数据的分类器。

具体来讲，有以下几个步骤：

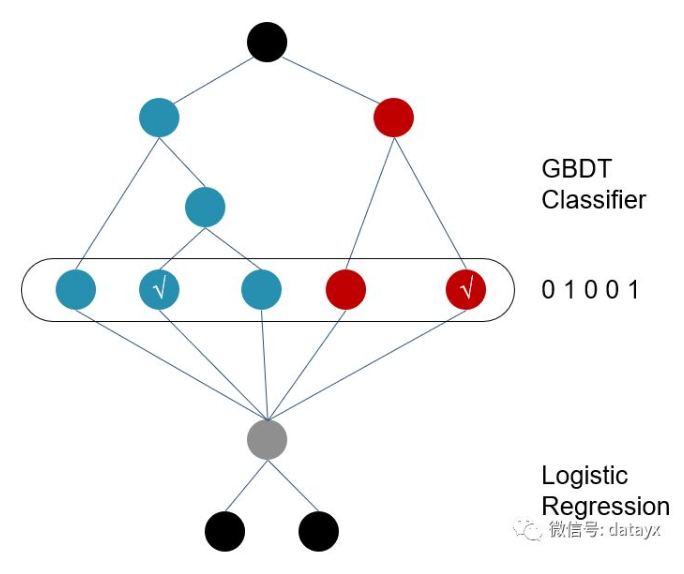
3.1 GBDT首先对原始训练数据做训练，得到一个二分类器，当然这里也需要利用网格搜索寻找最佳参数组合。

3.2 与通常做法不同的是，当GBDT训练好做预测的时候，输出的并不是最终的二分类概率值，而是要把模型中的每棵树计算得到的预测概率值所属的叶子结点位置记为1，这样，就构造出了新的训练数据。

举个例子，下图是一个GBDT+LR 模型结构，设GBDT有两个弱分类器，分别以蓝色和红色部分表示，其中蓝色弱分类器的叶子结点个数为3，红色弱分类器的叶子结点个数为2，并且蓝色弱分类器中对0-1 的预测结果落到了第二个叶子结点上，红色弱分类器中对0-1 的预测结果也落到了第二个叶子结点上。那么我们就记蓝色弱分类器的预测结果为[0 1 0]，红色弱分类器的预测结果为[0 1]，综合起来看，GBDT的输出为这些弱分类器的组合[0 1 0 0 1] ，或者一个稀疏向量（数组）。

这里的思想与One-hot独热编码类似，事实上，在用GBDT构造新的训练数据时，采用的也正是One-hot方法。并且由于每一弱分类器有且只有一个叶子节点输出预测结果，所以在一个具有n个弱分类器、共计m个叶子结点的GBDT中，每一条训练数据都会被转换为1\*m维稀疏向量，且有n个元素为1，其余m-n 个元素全为0。

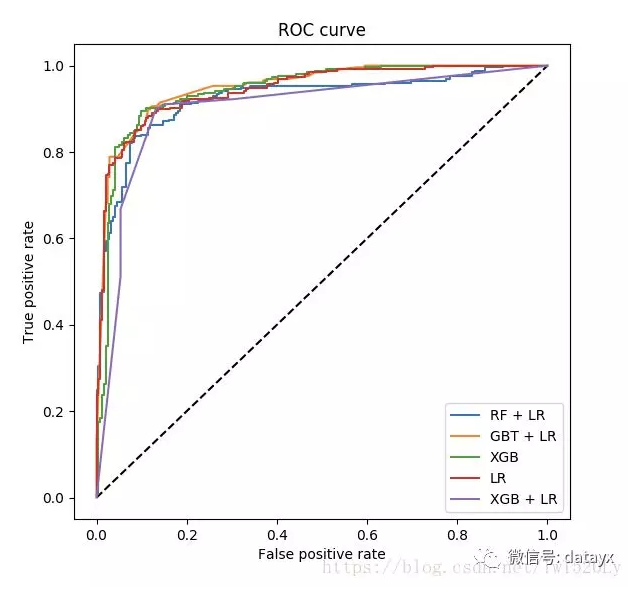
3.3 新的训练数据构造完成后，下一步就要与原始的训练数据中的label(输出)数据一并输入到Logistic Regression分类器中进行最终分类器的训练。思考一下，在对原始数据进行GBDT提取为新的数据这一操作之后，数据不仅变得稀疏，而且由于弱分类器个数，叶子结点个数的影响，可能会导致新的训练数据特征维度过大的问题，因此，在Logistic Regression这一层中，可使用正则化来减少过拟合的风险，在Facebook的论文中采用的是L1正则化。

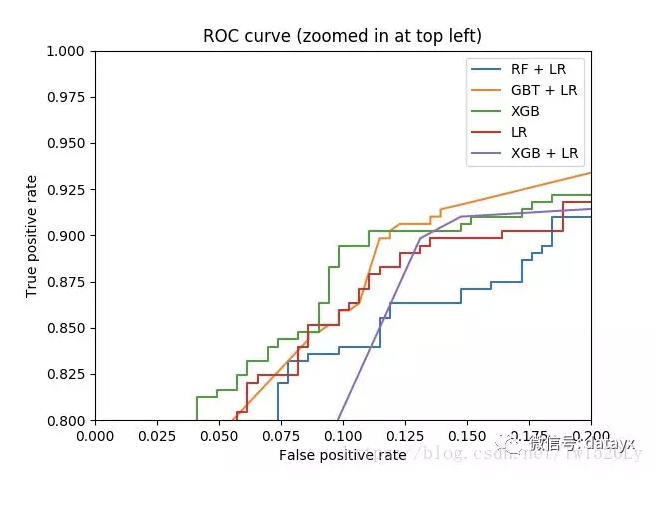


4. RF + LR ? Xgb + LR?

有心的同学应该会思考一个问题，既然GBDT可以做新训练样本的构造，那么其它基于树的模型，例如Random Forest以及Xgboost等是并不是也可以按类似的方式来构造新的训练样本呢？没错，所有这些基于树的模型都可以和Logistic Regression分类器组合。至于效果孰优孰劣，我个人觉得效果都还可以，但是之间没有可比性，因为超参数的不同会对模型评估产生较大的影响。下图是RF+LR、GBT+LR、Xgb、LR、Xgb+LR 模型效果对比图，然而这只能做个参考，因为模型超参数的值的选择这一前提条件都各不相同。

顺便来讲，RF也是多棵树，但从效果上有实践证明不如GBDT。且GBDT前面的树，特征分裂主要体现对多数样本有区分度的特征；后面的树，主要体现的是经过前N颗树，残差仍然较大的少数样本。优先选用在整体上有区分度的特征，再选用针对少数样本有区分度的特征，思路更加合理，这应该也是用GBDT的原因。





6. GBDT + LR 模型提升

现在，我们思考这样一个问题，Logistic Regression是一个线性分类器，也就是说会忽略掉特征与特征之间的关联信息，那么是否可以采用构建新的交叉特征这一特征组合方式从而提高模型的效果？

其次，我们已经在2.3小节中了解到GBDT很有可能构造出的新训练数据是高维的稀疏矩阵，而Logistic Regression使用高维稀疏矩阵进行训练，会直接导致计算量过大，特征权值更新缓慢的问题。

针对上面可能出现的问题，使用FM算法代替LR，这样就解决了Logistic Regression的模型表达效果及高维稀疏矩阵的训练开销较大的问题。然而，这样就意味着可以高枕无忧了吗？当然不是，因为采用FM对本来已经是高维稀疏矩阵做完特征交叉后，新的特征维度会更加多，并且由于元素非0即1，新的特征数据可能也会更加稀疏，那么怎么办？

所以，我们需要再次回到GBDT构造新训练数据这里。当GBDT构造完新的训练样本后，我们要做的是对每一个特征做与输出之间的特征重要度评估并筛选出重要程度较高的部分特征，这样，GBDT构造的高维的稀疏矩阵就会减少一部分特征，也就是说得到的稀疏矩阵不再那么高维了。之后，对这些筛选后得到的重要度较高的特征再做FM算法构造交叉项，进而引入非线性特征，继而完成最终分类器的训练数据的构造及模型的训练。