点击率预测模型实验指南

1 数据的离散化处理

点击率预测任务的过程可分为两个步骤：特征生成和建立模型。在线广告，特别是显示广告，使用的特征信息来源于：广告特征，如广告创意、广告内容类别等信息；上下文特征，媒体网站，网址域名，网页内容主题，广告位的位置以及尺寸等；用户特征包括用户标签类别，人口特征，用户代理信息，地理位置，访问时间和历史行为等。媒体通常能够获取的信息，这些特征属性的原始数据可有两种形式：分类属性和数值属性。

表 1 原始记录示例

时间	201505180822
User-Agent	Internet Explorer 10 ， Windows 8
IP	222.197.191.*
URL	http://www.XXX.com.cn
…	…
点击行为	否

如表 1 所示，原始数据中有很多可当作离散或连续的属性值。建立模型时，由于逻辑回归模型的特性，数值（Numerical）属性可以直接作为输入，但使用例如分箱（离散化）等方法将其转为离散特征也是常见的做法。使用离散特征能够加速某些预测模型训练过程，且比直接将数值当作离散值的效果好。通常来说，分类（Categorical）属性，或类别、离散（Discrete）属性的无大小关系，并不能直接使用，应对之进行编码处理，一般是使用二分化处理，即将一个属性所有可能出现的值都作为一维来表示。以独热编码（One-Hot Encoding）为例：

假如 N 个分类属性组成的特征向量为 $x=(c_1,c_2,...,c_N)$，类别特征 c_i 由 M_i 个可能出现的不同值组成，那么对于布尔值 $b_i^j \in \{0,1\}$ ，在 $i \in \{1,2,..N\}$ ，$j \in \{1,2,...,M_i\}$ 时，有：

$$c_i=\left(b_i^1,b_i^2,...,b_i^{M_i}\right), \text{且} \sum_{j=1}^{M_i} b_i^j = 1 \tag{1}$$

可以看出二分化实质是一种将分类特征转为数值属性，若属性的可能值基数很大，经过这些初步处理后将会生成维度极大的稀疏向量，且很含有大量长尾分布的特征。可这些方式来降维：如清除非频繁的数据、筛选若干最具有代表性的特征筛选有用的特征。这些方式直观且实际效果优异，但这些处理是离线或批量（Batch）方式，需预先遍历数据集获取所有信息，建立相应字典，再利用它来处理每条记录，实际操作中开销巨大。业界通常使用哈希映射(Hash)作为大数

据预处理的二分化映射，利用其优良的查找性能，很适合广告点击率预测这种大规模的预测任务的数据预处理。

本文实验使用了如下所示的系统环境： Intel Xeon E3-1226 v3、8G ram、Ubuntu 14.04 64bit、Python3.4。且使用了多种开源的工具和库：Vowpal Wabbit、Xgboost、libfm 进行模型训练，而预处理包含 One-Hot Encoding、哈希映射、频繁值筛选、数值离散化、特征连接以及 GBDT 转换等。

Vowpal Wabbit[21]是来自 Yahoo 和 Microsoft 研究项目设计的一套强调高效、快速和高可扩展性的机器学习工具，文中简称 VW，其底层使用 C++实现，并支持并行分布式计算，适合大规模的在线算法测试。它支持特征哈希、多种损失函数和分类回归算法，是理想的广告点击率预测建模工具，本文将使用它验证大多数的线性模型和在线算法。Xgboost[22]是一个通用的梯度提升工具库，能够支持包括常见的广义线性模型、梯度提升回归（决策）树（GBRT），且支持并行计算、分布式扩展以能够处理 TB 级别的数据。LibFM[23]是一个实现了因式分解模型的工具库,可用它来验证公式 3-8 中所提到的模型。

2 实验设计

本课题设计了以下几组实验来验证工业实践中点击率预测可能出现的各种问题：

1) 特征预处理方案对离线模型的点击率效果的影响。特征工程（Feature Engineering）的优劣在实践中对结果的影响可能是决定性的，甚至高于精巧设计的数学模型对结果的贡献。本文准备设计一系列由简单到复杂的数据预处理方案，最后使用业界最成熟的逻辑回归预测模型来验证特征处理的价值。

2) 逻辑回归与非线性预测模型对点击率预测效果的影响。验证使用非线性模型是否优于业界广泛使用的方案，并分析其原因。

3) 线性逻辑回归与因式分解逻辑回归对点击率预测结果的影响。理论上，因式分解模型能够解决线性和方式中存在的一些弊端，本文验证这种新式模型有多大的性能优势。

4) 在线预测模型对比离线预测模型的性能差异。本文将模拟实际中在线与离线模型的工作方式，使用仿真的情景来验证点击率预测的实际效果，并对比分析两种方式的优劣。

5) 几种在线预测模型的效果分析。横向比较若干种适合用来进行点击率预测的在线预测模型。

- **预处理对性能影响**

本实验设计了六种不同的特征预处理方案，如表 1 所示，包括以下几种：

表 1 预处理方案

	DP1	DP2	DP3	DP4	DP5	DP6
数值处理法	无	无	哈希	无	分箱	分箱
类别处理法	哈希	二分编码		点击率	频繁筛选	频繁筛选
其他						GBDT 过滤
能否在线处理	能	能	能	能（复杂度略高）	否	否

1) 方案一(DP1)，数值特征不进行处理，直接输出原始值；类别特征使用哈希建立特征空间。

2) 方案二(DP2)，数值特征不进行处理，直接输出原始值；类别特征使用One-Hot Encoding。注此方式不适应于 VW，能够用于 XGBOOST。

3) 方案三(DP3)，将所有特征都当作类别属性，即每个数值特征将用属性名和数值一起当作类别进行哈希，建立特征空间。

4) 方案四(DP4)，数值特征不进行处理，直接输出原始值，将类别特征值的历史点击率代替，参见[9]中应用 GBDT 模型的处理方案。

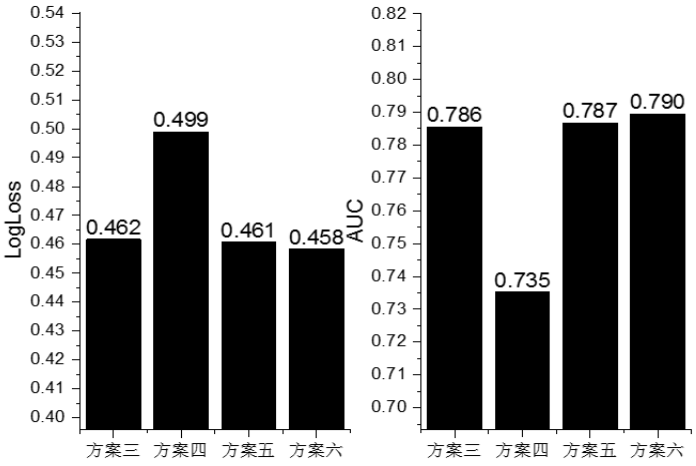
5) 方案五(DP5)，将数值特征分箱，且对分类属性进行频率筛选，这里使用$v \leftarrow \lfloor log(v)^2 \rfloor$将实数的空间压缩到整数范围，且由于数值的大小的线性关系被破坏，不能使用数值属性，且筛选出现次数大于 10 次的类别属性，最后采用方案四将处理后特征一同当作分类值进行哈希。

6) 方案六(DP6)，特征处理同方案五，但又使用了论文[10]中的思想，将初步处理过后的特征作为基本属性，然后使用方案五的输出作为 GBDT 处理的输入，使用 30 颗树，最大深度为 7，因而生成 30 个额外的增强属性，再将这 30 个特征与基本特征一起作为训练模型的输入。

在这里只验证方案三、四、五、六的预处理方案，其他几种方案（方案一、方案二）主要用作其他实验，这里使用逻辑回归并用 L-BFGS[16]求解。VW 运行参数如下：

--passes 25,--l2 25,-b 20,--bfgs,--loss_function logistic

注意这几个离线模型方案的实验中，在每次测试中都使用 10 折交叉验证以降低结果偏差，且为了充分验证算法的性能，防止数据时效性等因素的干扰，原始样本顺序被打乱。考虑到交叉验证的耗时较大，离线实验只使用数据集中前 100 万条曝光记录。默认 L-BFGS 使用 VW 工具、GBDT 使用 XGBOOST。



(a)　　（b）

图 1 预处理对性能的差异

（a）对数损失指标 （b）AUC

其结果如图 1 所示, 结果表明，采用 GBDT 的预处理能够获得更佳的性能。方案五相对方案三增加了频繁值的筛选和数值属性的离散化，但是性能提升微乎其微。而添加了 GBDT 特征的方式能够获得相对更多的性能提升，表明了此方式在如广告点击率预测的大规模线性学习应用中的有效性。而方案四的与之前决策树方式下的情形类似，都表现出了最差的性能，可能是这种方式并不太适合用于预处理广告点击率数据。

- **逻辑回归对比 GBDT 性能分析**

这里使用了离线 L-BFGS 解法与非线性模型 GBDT 在模型训练的计算时间复杂度和性能上的差别，记 LRVW:VW 逻辑回归+DP3、LRXGB：XGBOOST 逻辑回归+DP2、GDBT1:XGBOOST 的 GDBT+DP2、GDBT2: GDBT+DP4，其实验参数设置如表 2 所示：（注意在 VW 的 DP3 与 XGBOOST 的 DP2 是等价的）

表 2 参数设置

	parameters	预处理
LRVW	--passes 25,--l2 25,-b 20,--bfgs,--loss_function logistic	DP3
LRXGB	Iteration:25 ,’object’:’binary:logistic’,’booster’:’gblinear’,’lambda’:25,’eval_metric’:’logloss’	DP2
GDBT1	Iteration:25 , bst:max_depth':10, 'bst:min_child_weight':5, 'bst:eta':0.5, 'objective':'binary:logistic','eval_metric':'logloss'	DP2
GDBT2	Iteration:25 , bst:max_depth':10, 'bst:min_child_weight':5,	DP4

	'bst:eta':0.5, 'objective':'binary:logistic','eval_metric':'logloss'	

通过图 2 可发现，两种工具 VW 和 XGBOOST 在相同参数下求解线性逻辑回归时，VW 在此参数设置下性能更好，考虑到并未在 XGBOOST 下使用哈希映射，其差距较小可认为性能接近。同在使用全类别属性的 One-Hot Encoding 的预处理和相同工具 XGBOOST 下，发现线性逻辑回归与决策树模型在预测性能上差异并不大，而在本实验的参数设置下，线性模型会表现更好。而把类别属性转换图 2 逻辑回归与 GBDT 的预测精度差异。

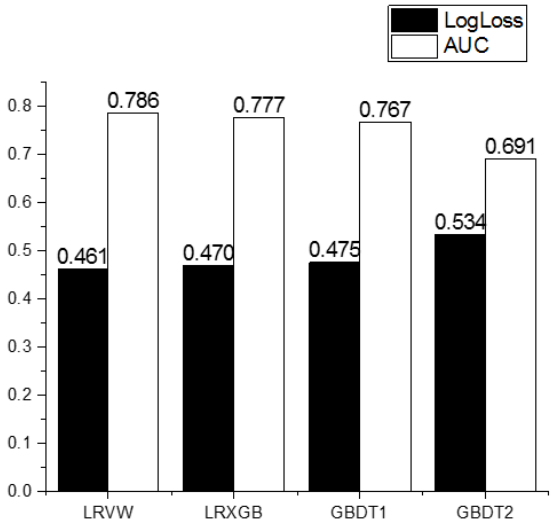


图 2

为历史统计点击率后，发现其有较大的性能损失，本实验初步分析认为：这种处理在模型训练中更易受异常值的干扰，且与原始的数值属性结合在一起导致所有数值的变化范围各异，收敛速度更受考验。

- **因式分解（FM）逻辑回归模型性能分析**

本文设计的实验四将要考察 FM 模型在性能上的优越性，每种代号对应的训练模型与预处理方案为： LR1:逻辑回归+DP5、LR2:逻辑回归+DP6、FM1:FM 逻辑回归+DP5、FM2:FM 逻辑回归+DP6。

表 2 性能对比表

提升 / 方案	$\text{LogLoss}_{提升}$	$\text{AUC}_{提升}$
$LR1 \rightarrow LR2$	0.52%	0.37%
$LR1 \rightarrow FM1$	0.36%	0.52%
$LR1 \rightarrow FM2$	1.56%	1.07%
$LR2 \rightarrow FM1$	-0.15%	0.17%
$LR2 \rightarrow FM2$	1.04%	0.72%

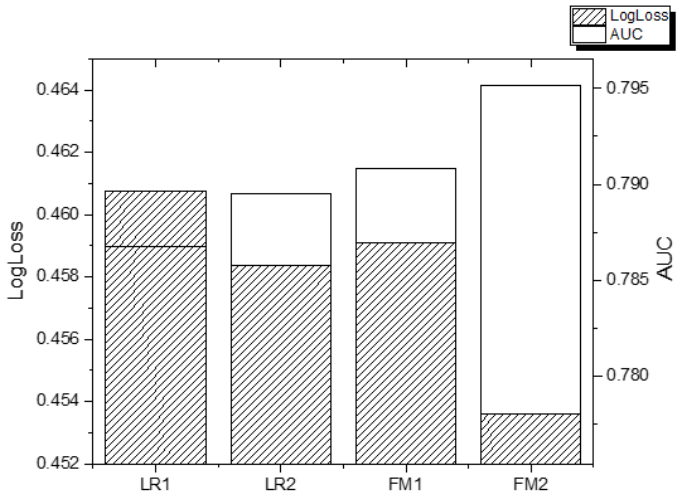


图 3 FM 与线性模型

从图 3 与表 2 中可以看出，FM 逻辑回归无论在预处理五和预处理六中都优于线性逻辑回归，而经过数值特征处理、频繁值筛选后，FM 拥有最佳的性能，此时比基本方案 LR1 有 1.56%更低的对数损失和 1.07%的预测精度提升。而经过对比发现：在基本预处理方案上使用 FM 回归和采用更佳的预处理方案并使用普通逻辑回归获得大致相近的性能的提升；通过对原始数据进行良好的数据预处理，更能够充分发挥 FM 逻辑回归的性能优势。

对于离线方式预测模型，本文还曾尝试了使用特征值的连接来改善线性逻辑回归的性能，但由于特征数平方级别的增长，导致特征向量的维度暴涨，训练时间过长，因而放弃了这种方案。

- **模拟在线算法的运行过程**

测试在线算法在实际中的性能表现、为了模拟实际情况中离线算法的效果，我们设置一个使用线性逻辑回归 L-BFGS 和预处理方案三的、采用固定间隔更新模型的对照组，来发现模型随时间间隔的时效性。每次重新计算模型时并不抛弃原有模型，将之前模型的参数作为此次更新的输入，更新所用样本为与上次训练间隔内所有样本。因数据集是由实际条件下连续七日收集来的，假设实际中每天更新一次模型，并用这个模型预测下一天的曝光，然后利用当天的数据去更新前一天的模型，依次这样循环。这里离线方式使用了之前性能与开销俱佳的预处理方案三与线性逻辑回归的 L-BFGS 求解的组合，而在线算法使用了预处理方案三下的随机梯度下降。为了进行离线测试，我们粗略地将整个数据集平均分成七天，每份 6550000 记录，依次用前一天的训练模型并预测下一天的点击率，每次模型的训练开始时，使用之前的模型作参数初始化。训练时参数的依然与离线方式的一致：迭代 25 次、20 位地址空间以及 25 的 L2 正则化参数。

结果如图 4 所示，很显然离线方式的平均性能要强于在线梯度下降，但却难以看出离线模型的时效性，这很可能是在此数据集中每一天的特征只有较少的变化，即模型时效性难以在短短一天中充分体现。但是需要注意的是，由于每次性能的验证都是基于当天的数据，在每天的开始，会由于参与检验的样本数据较少，性能会有较大波动，而随着样本数据量的增多，性能也趋于稳定。除此之外，还能发现随着天数的增加，其平均性能是略微下降的，这不难理解：随着时间的增加，发现的内在规律越多，其模型的适用度的也就增加，对专一数据的（某一天）的拟合就会变差。

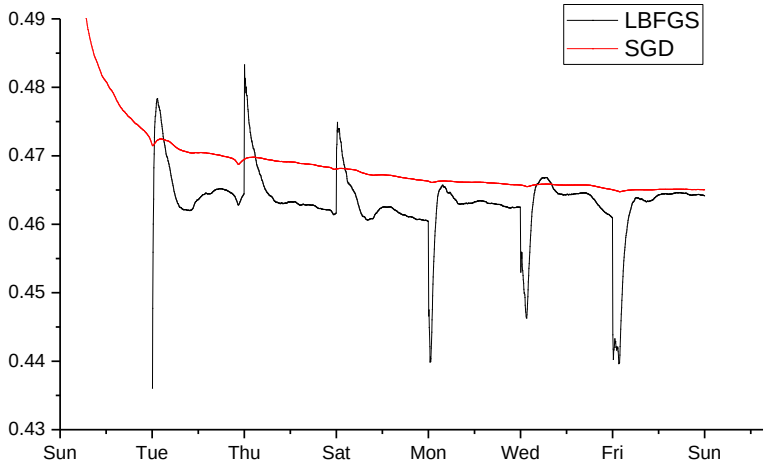


图 4 在线算法与离线算法，考察性能随样本数的关系