图灵联邦视频点击预测大赛

答辩说明文档

团　　队　名：　　问歌-守夜人-wbh

团 队 成 员：陆家辉、张泽晗、吴斌豪

目录

一、模型创建思路

1.1 赛题分析

1.1.1 赛题数据

本次赛题数据来源为亿刻 APP。提供的数据分为移动端行为数据和用户信息数据。移动端行为数据为 2019 年 11 月 8、9、10 日的行为日志数据，其中包括用户的设备 id、视频的 id、视频推荐位置、网络类型、经度、纬度等，用户信息数据有用户画像、用户等级、性别、安装 app 列表等。

1.1.2 赛题目标

本次赛题目的是希望参赛者根据 11 月 8、9、10 日的数据，通过构建推荐模型，预测在 11 日当天，用户在对应的视频上是否会产生点击行为。这是典型的 CTR 预估类问题，对于 CTR 问题而言，视频是否被点击的主导因素是用户，其次是视频本身的一些信息，不过本赛题对于视频信息没有提供，只提供了用户信息和点击时的上下文信息。所以其实本赛题中除了根据用户 ID 和视频 ID 来表征用户和视频的特征外，如何有效地挖掘上下文信息也格外重要。

1.2 数据预处理

1.2.1 数据清洗

经过观察，数据中其实还是存在着一些噪音和不合理数据，所以提前对数据进行预处理能够使模型更具泛化性，同时挖掘更多特征。我们的做法主要有：

（1）对重复值使用 drop 操作将其去掉，对于各个类别特征，将取值较少的字段合并成新字段以增强模型的泛化性；

（2）对 APP 版本 app_version 和操作系统版本 os_version 进行更细粒度的刻画（例如 2.1.1-->2 1 1 | 8.1.0-->8 1 0）；

（3）利用（timestamp-ts）来求得视频曝光到点击的时间差，删除时间差小于 0 的日志数据；

（4）对训练集中 7 号数据和测试集中 10 号数据进行时间调整，因为 7 号数据实际上的曝光可以认为是发生在 8 号的 0 点 0 分，测试集同理。

（5）deviceid 与 guid 的皮尔森相关系数在 0.9 以上，且 deviceid 缺失值更少，因此删除 guid 而使用 deviceid 代表用户 ID；

1.2.2 位置信息处理

原始训练文件中提供了 lng 和 lat 经纬度信息，为了更具体的表征用户的位置，我们做了以下三步操作：

（1）首先对经度取前 7 位，纬度取前 6 位，这样可以避免用户位置的细微变动带来的噪声影响，相当于做一个简单的平滑；
（2）另外我们将经纬度字符串进行拼接来表征用户的具体位置；
（3）最后我们根据用户的经纬度来对用户位置做一个区域性的划分，Geohash 是一个 Python 模块，它提供了在纬度和经度坐标之间解码和编码的 Geohashes 函数，经过测试，对经纬度分别进行 3、4、5 位编码可以大致认为划分的区域大小为省、市、区的大小。

1.3 验证集划分

因为本赛题是一个时序预测问题，所以如果使用交叉验证会出现线下分数很好但线上测试分数却不理想的情况，原因就是忽视了时间序列的影响，导致了用未来的数据去预测过去的数据。因此在切分验证集的时候，我们的做法是用前面 8、9 号的数据作为训练集，10 号的数据作为线下验证集，线下验证没有花精力在调整模型参数上，主要是为了通过线下验证集获取模型的最优迭代次数和概率阈值（当模型预测的点击概率大于阈值时，则认为用户会有点击行为）。

做预测时，使用 8、9、10 号全量数据集来训练模型。这样相比使用全量数据集进行五折交叉验证分数要更高且更稳定。

1.4 特征工程

1.4.1 统计特征

统计特征这里我们构造了各个类别特征的基本 count 计数特征，同时在此基础上使用 count、ratio、nunique 构造了不同类别特征之间的交叉特征。例如：使用 count 计数特征来表示某用户在指定 pos 位置总共的曝光次数，使用 ratio 比例特征来表示某用户在某个 pos 位置的曝光比例，使用类别变量的 nunique 特征来表示某用户共曝光了多少条视频等。

在构造统计特征的时候，为了尽可能多地细粒度刻画用户的上下文信息中所包含的特征，我们对部分类别统计特征（设备 id，视频 id，视频展示位置，网络信号，经纬度）进行了二阶到四阶的交叉统计，选择这五个类别特征是根据树模型给出的特征重要性中选择的，这五个类别特征以及其相关的交叉特征的重要性比较高，效果比较好。每构造完一组交叉特征后，都会使用后文提到的特征筛选手段来对交叉特征群进行有效性筛选。

1.4.2 历史点击、曝光特征

在时序问题中，用户历史行为的统计特征通常是比较有效的特征，所以都是基于历史行为来做了历史的点击、曝光记录相关的特征。我们统计了 8、9、10 号三天的历史点击率，来挖掘历史点击信息，同时为了防止过拟合我们使用了时间滑窗的手段只统计前一天的点击率。

另外由于数据的稀疏性，计算出来的点击率的可靠性不高，且具有较大的方差。例如当视频的曝光次数较少的时候，对其直接进行 CTR 的统计计算会导致一个偏高的结果。而当视频的曝光次数很大，但点击次数很少或几乎没有的时候，对其直接进行 CTR 的统计计算会导致一个偏低的结果。因此我们使用了贝叶斯平滑对历史点击率进行修正，贝叶斯平滑的思想是给点击率预设一个经验初始值，再通过当前的点击量和曝光量来修正这个初始值。

最后，我们也进行了一些类别特征和交叉特征（例如 pos 和 deviceid）的曝光量的细粒度统计，统计其前一天的曝光量，今天的曝光量以及每小时和每分钟的曝光量来作为点击率特征的补充。

1.4.3 曝光时间差特征

直观上分析，一般一个用户看完当前视频之后，很可能还会紧接着再看另外一个视频，那么统计这二者的时间差或许会有所帮助。时间差有两类，一类是当前点击距离下次点击的时间，另一类是当前点击距离上一次点击的时间。因此我们统计了前 1, 2, 3, 4, 5, 6, 15 次以及后几次曝光到当前的时间差，这些值是根据后面 3.1 节的特征筛选方法选择得到的。对于前后 15 次曝光的时间差这个特征有效，直观上是比较奇怪的，我们猜测这跟亿刻 app 的视频推荐策略有关。

1.4.4 Embedding 特征

Embedding 特征可以很好地描述用户和视频本身的一些特征，本次比赛我们主要使用了 Word2Vec 模型来进行 embedding 学习，核心在于如何构建训练 Word2Vec 用的文本序列，我们使用了两种构建序列的方式。

第一种构建序列的方法是以用户的曝光时间序列作为训练数据进行建模，把用户的曝光日志中 newsid 按照时间排序，也就形成了曝光序列。将曝光序列视为文档，就可以计算文档中 item（也就是 newsid）的 Embedding。这里不仅可以构造 deviceid 到 newsid 的嵌入向量，还可以是视频 newsid 到用户 deviceid，newsid 到位置 lng_lat 等。

第二种构建序列的方式就是 Deepwalk 方法，在推荐场景下，数据对象之间更多呈现的是图结构。这个时候 word2vec 就不能很好的展现这层关系，所以我们选择了 Graph Embeding 的方式，具体的使用了 DeepWalk，可以将用户的曝光记录转化为关系图。基于这些曝光视频序列构建视频关系图，采用随机游走的方式随机选择起始点，重新产生用户和视频曝光的随机游走序列。最后再将这些曝光序列输入 Word2vec 模型，生成最终的 newsid 的 Embedding 向量。

1.4.5 user-tag 特征

提供的数据中还有一类很重要的特征就是用户的画像类特征，为多值特征，每个值代表用户对某个兴趣的打分值。从推荐系统的角度思考，我们自然想到可以利用这些评分来使用矩阵分解的方法来求出用户的隐向量。有 deepwalk、lda、svd 三种候选方案，由于 deepwalk 没有采用 tag 中的权重数据，lda 训练的向量矩阵过于稀疏，最终决定使用 svd 矩阵分解算法。

其基本步骤为首先遍历 user 表获得每个用户对其 tag 的评分值，并按照 userID,、itemsID、rating 的方式形成评分表，对其进行异常值平滑之后构建 userID 和 itemsID 的评分矩阵。评分矩阵中行代表用户 userID，列代表兴趣 tag，其中的值代表用户对 tag 的打分。基于 SVD 的优势在于用户的评分数据是稀疏矩阵，可以用 SVD 将原始数据映射到低维空间中，可以节省计算资源。我们采用线下三折交叉验证的方式来对评分矩阵进行训练，之后使用训练得到的 20 维的隐向量来表征用户的兴趣。我们使用 svd 算法进行矩阵分解，训练得到 deviceid 和 tag 的 20 维分解矩阵，根据余弦相似度，计算每个 tag 与其相似的 tag，比如“恋爱”和“结婚”两个词的余弦相似度比较大，两个词比较相似，说明模型已经从数据中学到了隐藏的关系。

二、模型说明

2.1 模型介绍

面对点击预测这类问题，通常使用两大类模型，一种是基于 gbdt 的树模型，代表有 XGBoost、LightGBM、CatBoost，另一种是基于神经网络的模型，代表有 xDeepFM 等。

本次比赛中我们主要使用了 LightGBM 和 CatBoost。

LightGBM：速度快，效果好，不过有些情况下对 GPU 支持不太友好；CatBoost：对 GPU 支持比较好，训练速度可以大幅提升，大大减少验证想法的时间，相比 LightGBM 而言，对类别特征的处理会稍微好一些。因此结合这两个模型进行训练，快速迭代，相互补充。

2.2 模型融合思路

2.2.1 模型差异化

两个典型的树模型，LightGBM 和 CatBoost，使用不同的模型相当于从不同的视角解决问题；

2.2.2 特征差异化

由于我们进行了大量的特征交叉，导致特征维度较大，并且大量特征放进一个模型后，一些冗余特征可能会稀释一些强特的作用。因此我们为不同的模型准备不同的特征群，彼此之间保留相同的原始特征，例如将三阶交叉统计特征和四阶交叉统计特征分别喂给两个模型。

另外，由于在构造特征时使用了部分时序相关的特征，所以会造成 8 号的数据没有历史点击率特征，如果不处理的话在线下验证时会有将近二分之一的点击率特征为空，因此我们分别训练了两个模型，其中一个模型保留历史点击率特征但是只使用 9、10 号数据训练，另一个模型去除历史点击率特征使用全量数据来训练。

在融合结果的时候，有两点需要注意:一是如何确定不同模型最终预测结果之间的权重，二是如何在得到融合结果之后确定计算最终 F1 Score 时的阈值。

如何确定不同模型间权重：我们选择简单的线性加权平均的方式做融合，具体做法是获取要融合的几个模型在验证集上的概率预测结果，把结果作为线性回归模型的输入，使其逼近验证集的 01 标签，得到每个模型的线性加权权重。权重用于测试集的预测概率做加权平均，即可得到测试集概率结果。然后根据阈值，获取测试集是否点击的预测结果。

如何确定阈值：在之后的相关经验和技巧处有说明。

三、相关经验技巧总结

3.1 特征筛选

对于 LightGBM 等拟合能力超强的模型来说，很多和标签完全无关的特征甚至是随机加入的噪声，都能通过海量的子树建立密切的联系，如果我们单纯根据特征重要性来删除则会删除掉很多潜在的有用的信息。我们使用了两种特征筛选的方法以供参考。

第一种是称为 Null Importance 的方法，其基本思路就是将真实特征训练得到的分数与随机假特征训练得到的分数进行对比，假如某特征的分数并不能明显超过随机假特征，那么证明这个特征是一个无用的特征。首先计算原始特征的特征重要性，然后 shuffle 标签 y 指定次数 n，每 shuffle 一次重新训练一个模型然后记录特征重要性，直到全部训练完毕，将得到的 n 个特征重要性矩阵合并之后，就可以开始进行比较了。比如对于特征 A，我们用没有 shuffle 的特征 A 的特征重要性除以多次 shuffle 之后所有的 A 的特征重要性的 75%分位数，然后结果取对数，如果结果小于 0 则说明特征 A 不如随机特征。对于每次构造的新的特征群，都会结合原始特征群后使用 Null Importance 的方法来进行特征筛选，有用的特征加入最终的特征群，明显无用的噪声特征则进行删除，以此来保证输入每个模型的最后的特征数量在 300-400 之间。

第二种方法，是通过 shuffle 数据中某特征列的值，使用之前训练好的模型进行预测，观察 f1 和 auc 下降的程度进行筛选。shuffle 后对预测值的影响就越小，指标下降越小（甚至指标还会有提升），说明特征的重要程度越低，该特征可以考虑删除。具体做法如下。一、将提取好的所有特征，划分出训练集、验证集。二、使用训练集数据进行模型的训练，在验证集上进行预测得到基准的 f1 和 auc 指标。三、对每个特征，对验证集上该特征列的值进行 shuffle，再使用模型对进行预测，得到新的 f1 和 auc 指标，与基准的 f1 和 auc 指标作对比。四、可以尝试筛除掉造成指标下降最少的一些特征。

null importance 方法缺点是每次迭代都需要重新训练模型，适合对构造的特征群进行初筛，速度较慢。第二种方法只需要训练一次，速度快，适合模型训练完成后再次对特征进行精筛；

3.2 寻找最优阈值

由于 F1 Score 阈值敏感，所以我们在根据预测结果得到最终是否点击时需要对阈值做一个简单的迭代搜索，即指定一个阈值初始值和迭代步长，通过在验证集上不断增加阈值来确定 F1 Score 分数最优时的阈值，进而应用到测试集的预测结果中。

这是单模型时我们寻找阈值的方法，在多模型融合时，可以通过验证集上线性回归模型输出的概率结果，来寻找最优阈值，并将其用于测试集上（我们主要使用的是这种方式）。另一种方式是，记录每个模型在验证集上的最佳阈值，在得到最后是否点击的结果时，同样地也需要对各个模型的阈值进行同样地加权处理。

还有一种单模型、多模型通用的确定阈值的方法，即我们通过统计 8、9、10 号三天的点击数据，得到正负样本的比例大致为 0.11，因此，我们也可以对融合后的概率预测结果进行排序，取概率最大的前 11%的数据 label 置为 1，即预测会做出点击行为。这种方式比较简单，省去了搜索确定最优阈值的步骤。

四、团队介绍、方案反思

4.1 团队介绍

张泽晗，北京邮电大学研究生二年级，研究方向为异构计算。

陆家辉，中科院计算所研究生二年级。

吴斌豪，吉林大学，工作方向为推荐系统。

4.2 对目前方案的反思

（1）对 app 表利用不够，单从 app 列表中没有提取比较表现力强的的特征，没有明显提升。如果 app 表附带一些其它的信息（比如时间），可能可以更好的利用 app 表的信息。

（2）使用 xDeepfm 等神经网络类模型时效果不佳，因此后来也就没在尝试，将精力主要放在了特征工程上。

（3）到了比赛快结束时发现特征提取思路还有改进方式，我们统计特征时验证集和测试集的特征也会被统计进去，这在某种程度上可能意味着模型在训练时已经见过验证集，尽管不是直接看到验证集，但基于全量数据的统计特征已经泄露了一部分验证集信息，如果还有时间，我们可能会将统计特征的统计范围从原来的全量数据改为训练集数据重新尝试。

（4）针对类别型的特征和标签之间，我们在比赛过程中进行了五折交叉统计点击率，即目标编码，如不同 pos、不同 netmodel 点击率的平均值、最大最小值等。但是加入之后线下验证集分数上涨，线上分数降低，最终也就没有采用，推测是给的训练集天数过少导致即使采用五折交叉也会造成较严重的标签泄露。

（5）尝试根据经纬度 lat、lng 特征做聚类，得到新的类别特征：经纬度聚类后的所属类别。将该特征方法二阶交叉特征统计，以及历史统计特征的计算中，发现效果并不好，后续没有使用。还有时间的话，可以考虑再深挖一下经纬度聚类的特征。

（6）word2vec 方法中，由于 id 数据经过脱敏，不像 user-tag 的 svd 方法中，通过 tag 的 embedding 相似度可以直观地评估 embedding 训练效果，未来可以通过相似度测量、聚类、可视化等方法，定量分析 embedding 训练质量。